



Terms and Conditions of Use of Digitised Theses from Trinity College Library Dublin

Copyright statement

All material supplied by Trinity College Library is protected by copyright (under the Copyright and Related Rights Act, 2000 as amended) and other relevant Intellectual Property Rights. By accessing and using a Digitised Thesis from Trinity College Library you acknowledge that all Intellectual Property Rights in any Works supplied are the sole and exclusive property of the copyright and/or other IPR holder. Specific copyright holders may not be explicitly identified. Use of materials from other sources within a thesis should not be construed as a claim over them.

A non-exclusive, non-transferable licence is hereby granted to those using or reproducing, in whole or in part, the material for valid purposes, providing the copyright owners are acknowledged using the normal conventions. Where specific permission to use material is required, this is identified and such permission must be sought from the copyright holder or agency cited.

Liability statement

By using a Digitised Thesis, I accept that Trinity College Dublin bears no legal responsibility for the accuracy, legality or comprehensiveness of materials contained within the thesis, and that Trinity College Dublin accepts no liability for indirect, consequential, or incidental, damages or losses arising from use of the thesis for whatever reason. Information located in a thesis may be subject to specific use constraints, details of which may not be explicitly described. It is the responsibility of potential and actual users to be aware of such constraints and to abide by them. By making use of material from a digitised thesis, you accept these copyright and disclaimer provisions. Where it is brought to the attention of Trinity College Library that there may be a breach of copyright or other restraint, it is the policy to withdraw or take down access to a thesis while the issue is being resolved.

Access Agreement

By using a Digitised Thesis from Trinity College Library you are bound by the following Terms & Conditions. Please read them carefully.

I have read and I understand the following statement: All material supplied via a Digitised Thesis from Trinity College Library is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of a thesis is not permitted, except that material may be duplicated by you for your research use or for educational purposes in electronic or print form providing the copyright owners are acknowledged using the normal conventions. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone. This copy has been supplied on the understanding that it is copyright material and that no quotation from the thesis may be published without proper acknowledgement.

A stochastic model of damage accumulation in acrylic bone cement

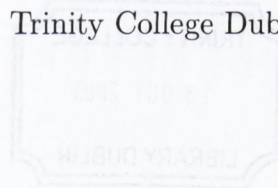
Application to failure modelling of cemented hip replacement

Alexander Benedict Lennon, B.E. (N.U.I.)

A thesis submitted to the University of Dublin in partial fulfilment of the requirements for the degree of

Doctor in Philosophy

Trinity College Dublin



Supervisors

Prof. P.J. Prendergast (1998–2002)

Dr. B.A.O. McCormack (1995–1998)*

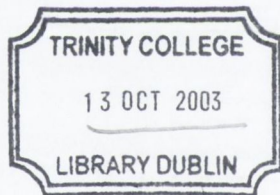
(*Presently at the Institute of Technology, Sligo)

External examiner

Prof. J.-L. Chaboche

Internal examiner

Prof. A. A. Torrance



TH68V8
7491
7367

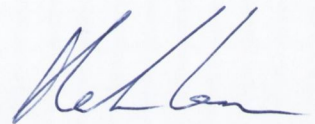
Handwritten text in three lines, likely a library call number or accession number.

In memory of
my uncle John,
Ray, Jim, and Mary Jane

Declaration

I declare that I am the sole author of this thesis and that all the work presented in it, unless otherwise referenced, is my own. I also declare that this work has not been submitted, in whole or in part, to any other university or college for any degree or other qualification.

I give permission for the Library to lend or copy this thesis upon request.

A handwritten signature in blue ink, appearing to read 'Alexander Lennon', with a stylized, cursive script.

Alexander Lennon

October, 2002

Acknowledgements

Many people have assisted me in the years leading to this thesis. Firstly, I would like to thank both of my supervisors, Prof. Patrick Prendergast and Dr. Brendan McCormack, for their advice, patience, trust, and encouragement.

My years in U.C.D. under Brendan have left me with many fond memories and lasting friendships. Dr. Alun Carr deserves particular credit for his inexhaustible knowledge and frequent guidance, not to mention his good company. I am also indebted to both the staff and students during my time in 205A, especially those connected with the old 'B.R.C.': Brian, David, Donnachadha, Donal, Goodwin, Joe, Kevin, Luke, Suzanne, Tadgh, and Victor. The Bioengineering Research Centre and the Department of Mechanical Engineering are also to be thanked for allowing me the use of their wet room and optical comparator after my move to Trinity College.

Since coming to Trinity under Paddy, I have been lucky enough to work in a vibrant research group that has matched zeal for research with an enthusiasm for the more social side of study. A few familiar faces in the form of Suzanne, Victor, and Kevin also made for an easy transition. The combined research groups of Paddy, David, Kevin, Gary, and Clive have provided a large social circle and frequently useful pool of diverse knowledge. My thanks go to Adrielle, Bruce, Ciaran, Damien, Danny, David, Fergal, Jan, John, Kevin, Laoise, Linda, Matteo, Matthew, Peter, Seosamh, Triona, and the newer faces of Conor, James, the Johns, Laura, and Paul. Outside the 'Bio' groups, thanks also go to Richard, Mark, Mary, and the many more postgrads, past and present, who have contributed to the friendly and lively atmosphere in the department.

I would also like to thank Prof. John Monaghan, head of department, and the rest of the staff, technicians, and secretaries for their help during my time here, especially during my period of Paddy impersonation. Special thanks go to Gabriel Nicholson for bringing my drawings to life, Peter O'Reilly for frequent assistance with the

Instrons, and John Gaynor and Toman McGinley for their help with hardware and software problems when they arose.

I have been lucky enough to have spent some time at the Joint Research Centre in Ispra, Italy to learn a little of the art of optical strain measurement. Thanks to Maurice, Colin, Robert, and the many students and staff I had the pleasure of working with there.

Much of the work undertaken in this thesis has been part of a wider European study of hip replacement failure. I would like to thank the other partners of the 'SMT Pre-clin' project for the exchange of information, ideas, and data that has improved the understanding of all those involved, and would not have been possible working in isolation from one another. Within our own group, Bruce has been the source of much experimental data and in depth discussions of bone cement. Our visits to Nijmegen, also, have always been both welcoming and stimulating. Sulzer Medica, Winterthur Switzerland, are also thanked for stem polishing.

I would also like to thank all those friends, at home and abroad, who have offered moral support over the last few years. Finally, I would like to thank my family, both immediate and extended. Their contribution cannot be overstated.

I acknowledge the funding of the Standards Measurements and Testing Program of the European Commission, Contract No. SMT4-CT96-2076, the MediLink Project of the Higher Education Authority PRTLII Programme, DePuy, a Johnson & Johnson Company, and MediSolve Ltd.

Publications and presentations resulting from this study

A.B. Lennon and B.A.O. McCormack, 1996. A finite element investigation of the stress distribution in an experimental model of a cemented hip reconstruction. Presented at Bioengineering in Ireland, Tulfarris, Co. Wicklow, Ireland.

A.B. Lennon and B.A.O. McCormack, 1996. A finite element and experimental investigation of the stress distribution in a cemented hip replacement. T.U.B.E.S. 11th Annual Spring Meeting, Cookstown, Co. Tyrone, Northern Ireland.

A.B. Lennon, B.A.O. McCormack, and P.J. Prendergast, 1997. Structural analysis of physical models of THRs using analytical and finite element methods. In J. Middleton, M.L. Jones, and G.N. Pande, eds., *Computer Methods in Biomechanics and Biomedical Engineering-2*, pp. 87-94, Gordon and Breach.

A.B. Lennon, B.A.O. McCormack, and P.J. Prendergast, 1998. Calculation of maximum transient stresses and 'locked-in' residual stresses in bone cement in THR. *Irish Journal of Medical Science*, pp. 272. Presented at Bioengineering in Ireland/TUBES Joint Conference. Dundalk, Co. Louth, Ireland.

A.B. Lennon, B.A.O. McCormack, and P.J. Prendergast, 1998. Issues in modelling mechanical failure of cemented orthopaedic joint replacements. *Irish Society for Scientific and Engineering Computation Annual Symposium*, University College Dublin, Ireland.

A.B. Lennon, B.A.O. McCormack, and P.J. Prendergast, 1998. Development of a physical model of a cemented hip replacement for investigation of cement damage accumulation. In 11th Conference of the European Society of Biomechanics, *Journal of Biomechanics*, 31(Suppl. 1):129.

A.B. Lennon, P.J. Prendergast, M.P. Whelan, and C. Forno, 1999. Use of grating interferometry for validation of finite element models and to investigate residual strain in polymethylmethacrylate. In J. Middleton, M. L., Jones, N. G. Shrive, and G.N. Pande eds., *Computer Methods in Biomechanics and Biomedical Engineering*–3, pp. 75–80, Gordon and Breach Science Publishers.

A.B. Lennon, P.J. Prendergast, M.P. Whelan, R.P. Kenny, and C. Cavalli, 2000. Modelling of temperature history and residual stress generation due to curing in polymethylmethacrylate. In P.J. Prendergast, T.C. Lee, and A. J. Carr, eds., *Proceedings of the 12th Conference of the European Society of Biomechanics*, Dublin, Ireland, pp. 253, Royal Academy of Medicine in Ireland.

A.B. Lennon, P.J. Prendergast, D. Lacroix, I. Revie, and M. Pfeiderer, 2000. Stress analysis of cement mantle of a polished femoral hip replacement. In P.J. Prendergast, T.C. Lee, and A. J. Carr, eds., *Proceedings of the 12th Conference of the European Society of Biomechanics*, Dublin, Ireland, pp. 189, Royal Academy of Medicine in Ireland.

A.B. Lennon and P.J. Prendergast, 2001. Evaluation of cement stresses in finite element analyses of cemented orthopaedic implants. *Journal of Biomechanical Engineering*, *Transactions of the ASME*, 123:623–628.

A.B. Lennon and P.J. Prendergast, 2001. Finite element modelling of the formation of pre-load damage in cement mantles of orthopaedic joint replacements, *ASME 2001 Summer Bioengineering Conference*, pp. 41–42, Snowbird, Utah.

A.B. Lennon and B.P. Murphy and P.J. Prendergast, 2001. Modelling fatigue failure of PMMA specimens as a stochastic process. Presented at 5th International Symposium of Computer Methods in Biomechanics and Biomedical Engineering, Rome, Italy.

A. B. Lennon and P.J. Prendergast, 2002. Residual stress due to curing can initiate damage in porous bone cement: experimental and theoretical evidence. *Journal of Biomechanics*, 35:311-321.

Contents

Declaration	i
Acknowledgements	ii
Publications and presentations resulting from this study	iv
Contents	vi
Abstract	viii
List of Figures	ix
List of Tables	xiv
Nomenclature	xv

Chapter 1 Introduction 1

A brief history of failure modelling is presented to illustrate the application driven nature of predictive failure theories. Some issues relevant to the failure of cemented joint replacements are introduced. Variability of bone cement failure is identified as a key element in failure of cemented joint replacement prostheses. It is argued that current practice of using purely deterministic failure models may lead to unrealistic conclusions regarding cemented joint replacement failure.

Chapter 2 Literature review 9

Salient issues in damage accumulation of bone cement are identified. Modelling techniques are surveyed as background to methods developed later. Previous simulation methods for bone cement failure are reviewed and areas for further development are identified.

Chapter 3 Methods 50

A modelling scheme for bone cement failure is presented. It is stochastic in the sense that it introduces random spatial distributions of pores and uses Monte Carlo simulations to predict trends in damage accumulation. Separate tests were devised

to confirm the model under different loading and boundary conditions: (a) fatigue data from the literature for uniaxial tension specimens and (b) a purposely developed experimental model of a cemented hip replacement.

Chapter 4 Results 81

Predicted porosity, damage accumulation, and fatigue life for uniaxial specimens are illustrated and compared to experimental data. Features of damage accumulation in the experimental model of a hip replacement are presented and it is shown that differences can occur due to small changes in structural features. Predictions of the stochastic models for these structural changes are compared with predictions of deterministic models. Factors influencing damage accumulation are also analysed.

Chapter 5 Discussion 119

Limitations of the study are listed and the main results are discussed in the context of experimental confirmation, comparison of stochastic versus deterministic modelling assumptions, and implications for cemented fixation of hip replacements.

Chapter 6 Conclusions 143

The main conclusions drawn from this study are listed and possibilities for future work are recommended.

Bibliography 146

Appendix A Relevant published articles 166

Appendix B Appended results 191

Appendix C Finite element code 201

Appendix D Drawings 252

Abstract

Acrylic bone cement, used to fixate several forms of orthopaedic joint replacement prosthesis, has a variable fatigue resistance arising mainly from porosity. It is argued that deterministic modelling assumptions of homogeneous cement with constant fatigue strength lead to unrealistic conclusions regarding bone cement failure and that physical sources of variability should be included to better understand cement damage accumulation.

A computational modelling scheme was developed to predict damage accumulation in bone cement. A nonlinear fatigue damage rule was extended to predict anisotropic damage accumulation and the effect of cracks on cement constitutive properties. Stochastic influences were incorporated by introducing random distributions of porosity and performing Monte Carlo simulations. Two tests were devised, incorporating fatigue damage accumulation under different loading and boundary conditions: (i) comparison with existing data from uniaxial tension specimens and (ii) a cement layer subjected to similar constraints and loading as occurs in cemented hip replacement.

Simulations of uniaxial fatigue failure show that inclusion of pores can account for much of the variability observed in fatigue tests of bone cement. Furthermore, changes in fatigue behaviour for different cement mixing methods could also be simulated by altering the average volume fraction and average radius of the pore distributions. Stochastic models predicted similar distributed cracking to that observed in experimental testing of the hip replacement model. Deterministic models predicted much more localised damage accumulation, which was not found experimentally.

In conclusion, deterministic modelling assumptions led to the prediction of unrealistic failure modes. Realistic predictions are better modelled by incorporating physical sources of variability. Stochastic models are thus recommended to increase the probability of predicting realistic early failures in cemented hip replacement.

List of Figures

1.1	Illustration of scales applicable to damage mechanics and fracture mechanics analysis.	4
1.2	Schematic of a total hip replacement.	5
1.3	Failure scenarios in total hip replacement.	6
2.1	Survival data for two implants.	11
2.2	Examples of microcracks from an autopsy retrieved cement mantle. .	12
2.3	Physical model of McCormack and Prendergast (1999) and damage growth observed in their specimens.	13
2.4	Bone cement microstructure	15
2.5	Cracks occurring around pores.	18
2.6	Theoretical and experimental evidence of interaction between pores .	18
2.7	Survival of cemented hip replacements grouped by time periods in which different cementing techniques dominated	20
2.8	Relationship between damage and net section and illustration of strain equivalence	24
2.9	Stress vs. cycles-to-failure curve illustrating typical quantities provided from fatigue tests.	25
2.10	Illustration of load-sequence effects for High-Low (H-L) sequence and Low-High (L-H) sequence.	27
2.11	Examples of varying degrees of nonlinearity in damage accumulation .	28
2.12	Nonlinear evolution vs. nonlinear accumulation.	29
2.13	Nonlinear evolution and nonlinear accumulation.	30
2.14	Relationship between distribution of rupture strengths and damage tolerance	35

2.15	Total damage accumulated in cement around a hip prosthesis predicted by Verdonschot and Huiskes (1997b) for three interfacial conditions.	43
2.16	Mean damage accumulated in cement layer predicted by Colombi (2002b) for <i>elasto-brittle</i> and <i>continuous damage</i> algorithms.	45
2.17	Total damage predicted by Stolk et al. (2003) in the cement layer of two different prosthesis designs.	46
3.1	Nonlinear evolution and accumulation of chosen damage rule.	52
3.2	Illustration of problem with using actual elapsed number of cycles for a multi-step test and definition of equivalent number of elapsed cycles.	53
3.3	Nonlinear accumulation for model of Murphy (2001) for a two-level test.	54
3.4	Motivation for uncoupling of elastic properties from damage growth until rupture.	56
3.5	Illustration of overprediction by Newton-Raphson scheme for curve of continuously increasing slope.	61
3.6	Algorithm for calculating cycle increment.	62
3.7	2D analogue of search through surrounding integration points when pore too large for current integration point was found.	63
3.8	Algorithm used to assign porosity to cement integration points.	64
3.9	Summary of finite element algorithm for simulating damage accumulation in bone cement.	68
3.10	The experimental model used for studying damage accumulation in cemented femoral replacements.	71
3.11	Photograph of surface finishes used in the study.	71
3.12	Photograph of polyethylene inserts	72
3.13	Photographs of equipment used to count cracks.	73
3.14	Loading configuration of the experimental model.	74
3.15	Illustration of generation of estimated maximum life curve for a pore-free cement.	76

3.16	Finite element mesh of uniaxial specimen	77
3.17	Finite element mesh of experimental model of femoral replacement . .	78
4.1	Simulated porosity distributions of hand-mixed specimens.	82
4.2	Simulated porosity distributions of vacuum-mixed specimens.	83
4.3	Photograph of typical porosity distribution for a hand-mixed and vacuum-mixed specimen.	84
4.4	Spatial damage distributions for hand-mixed specimens.	85
4.5	Spatial damage distributions for vacuum-mixed specimens.	86
4.6	S-N data from experimental study of Murphy and Prendergast (2000a) and simulated data of present study.	88
4.7	Comparison of (a) hand-mixed and (b) vacuum-mixed regression lines of Murphy and Prendergast (2000a) with regression lines fitted to vacuum-mixed data of simulated tests.	89
4.8	Minimum and maximum failure life at each stress level from experi- mental study of Murphy and Prendergast (2000a) and simulations of present study.	91
4.9	Average failure life at each stress level and resulting regression lines for hand-mixed and vacuum-mixed data of Murphy and Prendergast (2000a) and simulations of present study.	91
4.10	Spatial crack distributions for each of the specimens before and after testing.	92
4.10	cntd. Spatial crack distributions for each of the specimens before and after testing.	93
4.10	cntd. Spatial crack distributions for each of the specimens before and after testing.	94
4.10	cntd. Spatial crack distributions for each of the specimens before and after testing.	95
4.10	cntd. Spatial crack distributions for each of the specimens before and after testing.	96
4.11	Regions used to quantify damage.	97

4.12 Mean pre-load damage according to region. Error bars indicate one standard deviation.	98
4.13 Mean damage accumulated during testing according to region.	99
4.14 Relationship between pre-load damage and damage accumulated during testing for each region.	100
4.15 Migration of actuator displacement at peak load during testing of six specimens.	101
4.16 Inducible displacement of actuator during testing of eight specimens.	101
4.17 Inducible displacement of actuator vs total damage at end of testing.	102
4.18 Stress, porosity, and damage distributions for bonded specimen 1. . .	103
4.19 Stress, porosity, and damage distributions for debonded specimen 1. .	104
4.20 Tensile maximum principal stress for shrinkage of a pore-free cement and due to loading for the deterministic models.	105
4.21 Mean number of integration point cracks (i.e. completely damaged directions) accumulated during loading according to region.	107
4.22 Mean damage accumulated during loading according to region for (a) bonded and (b) debonded specimens for the stochastic simulations compared with predictions of a deterministic model.	109
4.23 Evolution of (a) number of cracks during testing and (b) total damage.	110
4.24 Relationship between damage accumulated during testing and pre-load damage.	111
4.25 Relationship between damage accumulated during testing and porosity.	112
4.26 Stressed volume and damage growth patterns.	114
4.27 Damage growth vs. volumes stressed above 9 MPa.	115
4.28 Migration over duration of test for each specimen for (a) medial and (b) distal directions.	116
4.29 Dependence of resultant migration at end of test on damage accumulation.	117

4.30	Dependence of resultant migration at end of test on damage accumulation in the cement surrounding the distal tip and in the most proximal regions.	117
5.1	Comparison of experimental pre-load cracking with simulated pre-load cracks.	130
5.2	Comparison of damage accumulated during testing for the experimental model with the number of cracks initiated during loading predicted by the stochastic simulations.	132
5.3	Comparison of maximum and minimum failure life for experimental and simulated uniaxial fatigue specimens.	133
5.4	Comparison of damage accumulated during testing for the experimental model with the number of cracks initiated during loading predicted by the deterministic simulations.	134
5.5	Change in stressed volumes during testing for region MD of the debonded specimens.	135
B.1	Stress, porosity, and damage distributions for bonded specimen 2. . .	192
B.2	Stress, porosity, and damage distributions for bonded specimen 3. . .	193
B.3	Stress, porosity, and damage distributions for bonded specimen 4. . .	194
B.4	Stress, porosity, and damage distributions for bonded specimen 5. . .	195
B.5	Stress, porosity, and damage distributions for debonded specimen 2. .	196
B.6	Stress, porosity, and damage distributions for debonded specimen 3. .	197
B.7	Stress, porosity, and damage distributions for debonded specimen 4. .	198
B.8	Stress, porosity, and damage distributions for debonded specimen 5. .	199

List of Tables

2.1	Manufacturer's Composition for Surgical Simplex P bone cement . . .	15
2.2	Mechanical properties of Plexiglas and three commercial brands of bone cement.	16
2.3	Comparison of mechanical properties of Simplex P for three mixing techniques.	19
2.4	Examples of varying degrees of anisotropy proposed and/or used in several studies.	26
3.1	Elastic properties used in damage accumulation simulations.	79
3.2	Summary of damage accumulation simulations	80
4.1	Mean and standard deviation (S.D.) of porosity and pore radius used as input for each mixing method.	84
4.2	Mean and standard deviation (S.D.) of total specimen volume fraction of pores for each mixing method achieved in the simulations.	84
4.3	Mean and standard deviation (S.D.) of failure life for each data set and significance values (p) for a Student's t-test	90
4.4	Mean and standard deviation of pre-load damage and growth in dam- age during testing.	98
4.5	Mean and standard deviation of pre-load damage and growth in dam- age during testing.	106
5.1	Comparison of mean porosity, mean pore radius (\bar{r}), and maximum pore radius (r_M) for hand-mixed and vacuum-mixed cements from several studies.	129

Nomenclature

Roman letters

- \mathcal{Y} Thermodynamic force associated with damage
- D Uniaxial damage variable
- n Number of elapsed cycles at a given stress
- n_1^{eq} Hypothetical number of elapsed cycles from the first block of a two level fatigue test to cause the *equivalent* damage at the stress for the second block
- N_f Number of cycles to failure
- s Specific entropy
- v_p Volume of a pore
- v_{ip} Volume of an integration point
- E Young's modulus

Greek letters

- α Exponent of nonlinear fatigue damage rule
- ν Poisson's ratio
- Φ Total thermodynamic dissipation
- ϕ^* Dual thermodynamic dissipation potential
- Φ_{con} Thermodynamic dissipation attributable to heat conduction
- Φ_{loc} Local thermodynamic dissipation attributable to mechanical processes
- Π Potential energy

ψ Helmholtz free energy

ρ Density

σ Stress

σ_{uts} Ultimate tensile stress

τ_r Time constant for stress relaxation

θ Temperature

Matrices

$[\mathbf{r}^\sigma]$ Rotation matrix between global reference coordinate system and principal stress coordinate system

$[\mathbf{R}]$ Rotation matrix for stiffness between global reference coordinate system and principal damage coordinate system

$[\mathbf{r}]$ Rotation matrix between global reference coordinate system and principal damage coordinate system

$\{\boldsymbol{\gamma}\}$ Engineering strain

Vectors

$\hat{\mathbf{e}}_i^\sigma$ i th normalised eigenvector of stress tensor

$\hat{\mathbf{e}}_i^d$ i th normalised eigenvector of the damage tensor

$\hat{\mathbf{e}}_j$ j th base vector of a global coordinate system

Second order tensors (lowercase bold face)

$\boldsymbol{\delta}$ Kronecker delta

$\boldsymbol{\epsilon}$ Small strain tensor

$\boldsymbol{\sigma}$ Cauchy stress tensor

\mathbf{d} Second order damage tensor

$\tilde{\epsilon}$ Effective strain tensor

$\tilde{\sigma}$ Effective stress tensor

Fourth order tensors (uppercase bold face)

C Compliance tensor

D Fourth order damage tensor

E Stiffness tensor

H Crack compliance tensor

I Identity tensor

M Damage effect tensor

P_i Fourth order projection tensor for i th principal damage vector

\tilde{E} Effective stiffness tensor of the damaged material

\tilde{E}^d Effective stiffness of the damaged material in the principal damage coordinate system

Chapter 1

INTRODUCTION

Contents

1.1 Modelling mechanical failure: a brief history	1
1.2 Mechanical failure in orthopaedic implants	4
1.3 Aim of this thesis	7

1.1 Modelling mechanical failure: a brief history

Ancient civilisations were undoubtedly aware of limits imposed by the materials used in their many impressive constructions and mechanical devices, and can be attributed with development of static analysis of structures. However, the development of analytical methods to predict *failure* under mechanical loading can be said to have started with Galileo Galilei (1564–1642), who studied breaking loads of rods (Timoshenko, 1983). His identification of the dependence of breaking load on cross-sectional area was an early form of the concept of stress. The statement of the laws of motion by Isaac Newton (1642–1727), in particular his third law of action and reaction, gave rise to the concept of bodies reacting to applied loads to maintain equilibrium. The concept of linear elasticity has been attributed to Robert Hooke (1635–1703), who observed that deflections of bodies were proportional to applied load. James Bernoulli (1654–1705) first proposed the concepts of stress and strain for the description of loading and deformation of materials. A linear relationship between stress and strain was then proposed by Leonhard Euler (1707–1783) but it was Thomas Young (1773–1829) who first showed that this formulation of Hooke’s Law resulted in a constant-of-proportionality for different materials that was independent of the specimen geometry. A general theory of three dimensional elasticity was then developed by Augustin Louis Cauchy (1789–1857) in 1822; this represented

the first complete theory to state the equations of motion of a continuum, calculate stress, describe its properties under transformations of frames of reference, and to describe strain in terms of displacement gradients. Further rigour was provided by George Green (1793–1841) who placed a limit on the number of elastic constants using the concept of an elastic strain energy. William Thomson (1824–1907), better known as Lord Kelvin, provided support for Green’s concept by showing that a strain energy function must exist for isothermal and isentropic processes. Many solutions to specific three dimensional problems were developed by mathematicians in the following century, e.g. Saint-Venant, Hertz, Kelvin, and Boussinesq. However, the relationship between predicted stress states and actual failure processes was not well understood; thus, engineers had to rely on empirically developed codes of practice to prevent failure of structures. [Malvern (1969), Gordon (1976), Timoshenko (1983), Rice (1993), Lemaitre (2001)].

Failure by plastic flow was an early focus of analysis due to the predominance of compressive loading in structures. Charles Augustin de Coulomb (1736–1806) pioneered the description of frictional yielding of soils. Observation of plastic flow during experimental tests prompted the study of yielding in metals; Henri Edouard Tresca (1814–1885) developed the first continuum theory of metal plasticity in 1864. Barré de Saint Venant (1797–1886) was the first to set up the fundamental equations and apply them to practical problems. The use of steel in both civil structures and machinery meant that this field of continuum mechanics was to receive significant attention for the best part of a century. [Malvern (1969), Timoshenko (1983), Rice (1993)].

In the same period, it became recognised that materials subjected to repeated loading well below their static strength could fail after a large number of applied cycles. Jean-Victor Poncelet (1788–1867) was perhaps the first to discuss the resistance of materials to repetitive loads and may have introduced the term ‘fatigue’. Observation of fatigue failures in railway axles led William John Macquorn Rankine (1820–1872) to develop these ideas further, identifying the influence of cracks and changes in the section of shafts. August Wöhler (1819–1914) performed extensive

experimental studies of the same problem and was the first to propose a curve relating applied stress to the number of cycles to failure. The analytical study of failure related to cracks and fissures was further motivated by the occasional failure of ships and frequent observation of cracking around openings in their hulls and decks. Gury Vasilyevich Kolosov (1867–1936) and Charles Edward Inglis (1875–1952) independently derived the stress state around elliptical holes and showed that large stress concentrations occurred for sharp elongated holes. Alan Arnold Griffith (1893–1963) used Inglis' solution in combination with an analysis of the energy of a cracked body to develop a theory of brittle fracture. Extension of Griffith's theory to metal fracture was initiated by George Rankin Irwin (1907–1998). Fracture mechanics, as the theory came to be known, has been applied to a wide range of problems, including rupture and fatigue of aerospace components, pressure vessels, and propagation of fissures in the earth's crust during earthquakes. [Gordon (1976), Timoshenko (1983), Hertzberg (1996), Rice (1993), Lemaitre (2001)].

Fracture mechanics is well suited to the analysis of propagation of individual and well defined flaws. However, the increasing application of metals in high temperature operating environments gave rise to new failure modes—intergranular decohesions in highly stressed regions resulted in the gradual accumulation of large numbers of microscopic cracks distributed throughout the region. Such distributed damage was not well suited to analysis using fracture mechanics due to difficulties in analysing the large number of small and closely spaced defects. Kachanov introduced the concept of a continuum description of damage. Instead of trying to analyse individual cracks, damage was considered in an average sense over a region of interest. These concepts were further developed by Hult, Lemaitre, Chaboche, and Krajcinovic and were shown to have foundations in the thermodynamics of irreversible processes. Much of the development effort has focussed on providing a tool to bridge the gap between crack initiation and propagation (Fig. 1.1); significant research has also been directed at describing local fracture processes ahead of propagating cracks. Applications to materials containing random microstructural defects, e.g. concrete and rock, have highlighted the need for the incorporation of stochastic modelling in

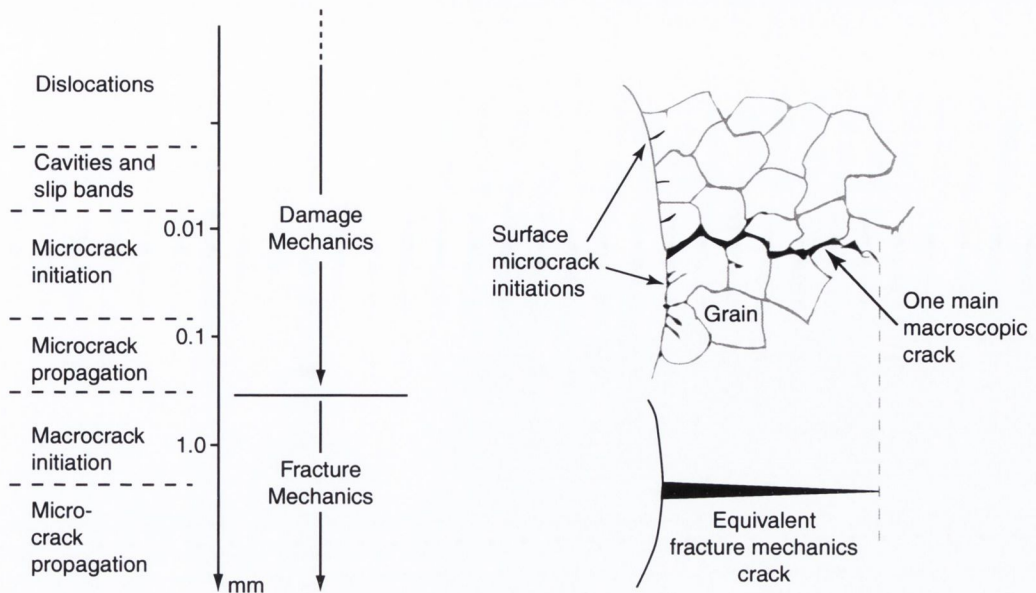


Figure 1.1. *Illustration of scales applicable to damage mechanics and fracture mechanics analysis. Adapted from Chaboche (1988)*

predicting the transition from distributed damage accumulation to localisation of damage to form a macrocrack or shear band. Developments of cohesive atomistic models have also tackled material failure but, due to present limitations in computational power, are of limited use to the engineer. [Chaboche (1988), Krajcinovic (1996), Lemaitre (2001)].

1.2 Mechanical failure in orthopaedic implants

Orthopaedic joint replacements are used to replace the parts of a degenerated human joint. Joint replacements typically consist of one or more components; these may be fixated to the bone using an acrylic polymer (polymethylmethacrylate; PMMA) known as ‘bone cement’. The most common example is total hip replacement (Fig. 1.2). A prosthesis, usually metallic, is inserted into the medullary cavity of the femur to replace the ‘ball’ portion of the joint, while an artificial cup, typically ultra high molecular weight polyethylene (UHMWPE), forms the new socket. Fixation of both components is normally achieved with acrylic bone cement that has been prepared in the operating theatre shortly before the introduction of the articulating components into the femur and pelvis. The cement does not bond chemically with either metal or bone. Instead, fixation is achieved by mechanical interlocking arising

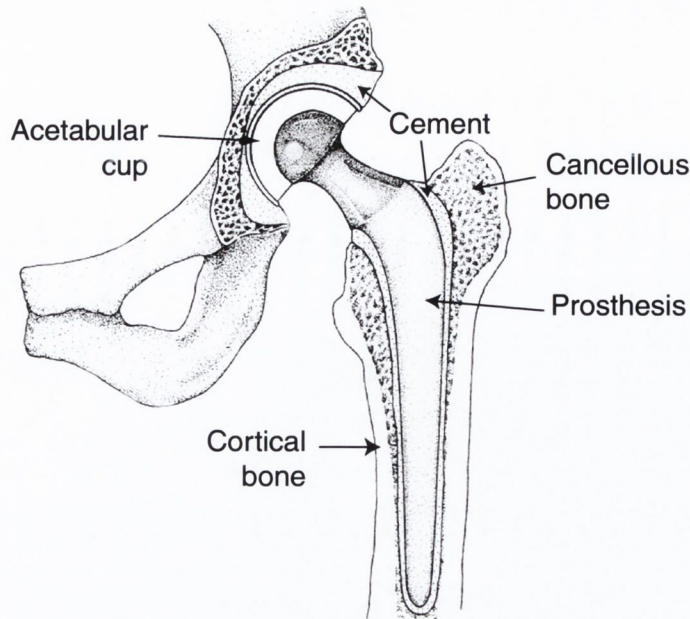


Figure 1.2. *Schematic of a total hip replacement. Adapted from Hardinge (1983).*

from surface roughness of the implant and interdigitation with cancellous bone—i.e. the cement performs the function of a ‘grout’.

Failure of cemented hips can be attributed to either infection or to mechanical (aseptic) loosening of the components. Infection has been virtually eliminated with improvements in surgical conditions leaving aseptic loosening as the dominant failure mode (Malchau et al., 2000). This mode of loosening exhibits substantial variability so that, although the average survival rate may be deemed acceptable, many patients experience early failure of their joint replacement.

Mechanical loosening arises through deterioration of both living tissues and implanted materials. Huiskes (1993) has proposed two interacting failure scenarios to account for loosening: (i) the particulate reaction scenario and (ii) the damage accumulation scenario. In the “particulate reaction scenario”, biological reactions to particulate wear debris cause deterioration of the bone until it no longer supports the implant. In the “damage accumulation scenario”, debonding of the prosthesis from the cement, along with microcracking in the cement, proceeds until there is no longer sufficient fixation for the prosthesis. Interactions between failure scenarios can occur from either increased loading of the cement as the living tissues become unable to support the load, or from increased wear particle production as

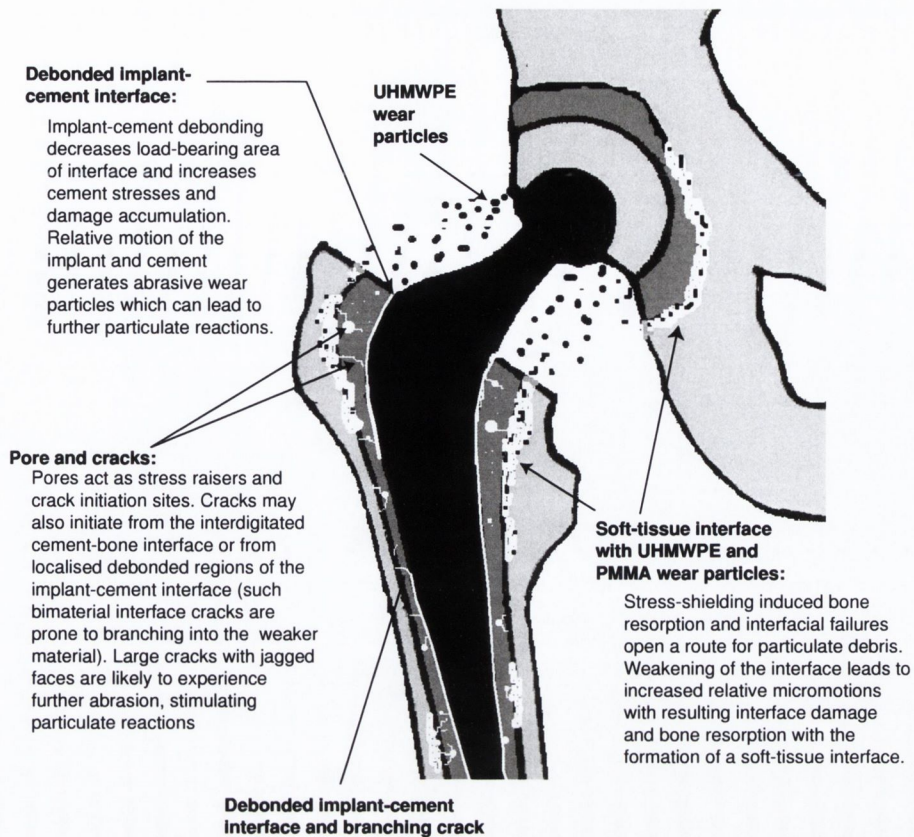


Figure 1.3. *Illustration of instances and interaction of damage accumulation and particulate reaction failure scenarios*

the debonded prosthesis abrades the deteriorating cement layer (see Fig. 1.3 for schematic illustration of interactions). Improvements in the bearing materials of the acetabular cup and prosthesis head have improved resistance to the particulate reaction failure scenario. However, reduction of damage accumulation within the cement and its interfaces has proved more intractable.

Bone cement has similar composition to Perspex/Plexiglas but its method of preparation causes it to have a much lower fatigue resistance compared with its industrial cousin. The main difference is the introduction of significant porosity from the entrapment of air during handling, mixing, and implant insertion. Evidence of a link between damage accumulation in cemented hip replacements and porosity in the cement has been demonstrated in several retrieval studies (Topoleski et al., 1990, Jasty et al., 1991, Culleton et al., 1993). Such mixing induced porosity can vary considerably with alterations in the mixing method. Improvements in mixing methods (mixing under vacuum to prevent air entrapment and centrifuging) have increased average survival of cemented components but significant variability remains. This is

because improved mixing methods have not completely eliminated porosity (Wang et al., 1996). Furthermore, the pores that remain can be large, in the case of vacuum mixing, or heterogeneously distributed, in the case of centrifuging, compared to traditional manual mixing.

Factors other than the variability in bone cement fatigue could be responsible for variable performance of hip prostheses. For example, prosthesis design, precision of implantation achieved, condition of the surrounding bone, and abnormal activity patterns of the patient. However, the variability in fatigue of bone cement under controlled laboratory conditions is large enough so that variability from this source alone might dominate. It will also interact with other variables; e.g. a poorly positioned prosthesis or more compliant surrounding tissues may expose a greater volume of cement to high stresses, thereby increasing the probability of finding a pore in a highly stressed region.

A suitable replacement for PMMA bone cement has yet to be found. The only possible alternative eliminates the role of the cement by using bioactive coatings on the prosthesis to form a bond directly between prosthesis and bone. However, these 'cementless' designs have yet to achieve equivalent performance to cemented replacements (Malchau et al., 2000). Thus, significant effort has been directed towards measuring and predicting the fatigue behaviour of bone cement. However, modelling has been mainly deterministic, based on regression lines from cycles-to-failure data, and a predictive model has yet to incorporate variability due to porosity in the cement. Such a deterministic design methodology is unsuitable for predicting realistic minimum lifetimes or the sensitivity of particular prostheses when fixated with bone cement.

1.3 Aim of this thesis

The key to understanding the inadequate survival rates of joint replacement prostheses appears to lie with the variability exhibited by the cement used for prosthesis fixation. If the variability in strength of bone cement is a factor, it should be possible to demonstrate it in computational and experimental models.

In this thesis, the author aims to test the hypothesis that the variable nature of damage accumulation in bone cement leads to different conclusions about the failure process compared to if deterministic models are used. In other words, it will be attempted to prove that the implicit simplification of homogeneity in fatigue strength oversimplifies the process of damage accumulation in cemented joint replacement. Damage accumulation is path dependent and is likely to be sensitive to a variety of phenomena so that interactions with other processes could lead to complex failure modes. To increase the probability of capturing the more critical early failures, it is necessary to include sources of variability that are likely to cause such complexity.

Chapter 2

LITERATURE REVIEW

Contents

2.1 Purpose of this chapter	10
2.2 Cemented total hip replacement	10
2.3 Bone cement and failure of hip replacements	11
2.3.1 Cement chemistry and physical properties	14
2.3.2 Mixing methods, porosity, and fatigue behaviour	17
2.3.3 Viscoelasticity of bone cement	20
2.3.4 Residual stress and the initiation of damage	22
2.4 Fatigue damage accumulation	23
2.4.1 Description of the damaged state	24
2.4.2 Evolution of the damage variable for complex cyclic loading histories	26
2.4.3 Coupling of elastic properties with damage	36
2.5 Simulation of damage accumulation in hip replacement	42
2.5.1 Differences in published models	47
2.5.2 Deficiencies in previous models	47
2.6 Concluding remarks	48

2.1 Purpose of this chapter

Issues relevant to artificial joint replacement, and in particular the use of bone cement, are reviewed first. Next, a survey of techniques used to model fatigue failure of materials is presented as background to the modelling methods to be developed later. Finally, an assessment of simulation methods reported in the literature for failure prediction of cemented hip replacements is presented, and areas for further development are identified.

2.2 Cemented total hip replacement

Total hip replacement has become a widespread procedure, with up to one million procedures performed annually (Huiskes and Verdonschot, 1997). ‘Total’ refers to replacement of both femoral and acetabular sides. With the ageing of the world population¹, the incidence and accompanying economic burden of joint disease is likely to increase.

Hip replacement is considered as one of the most successful surgical interventions, with survival rates in excess of 90% at 10 years (Huiskes and Verdonschot, 1997). However, for many thousands of patients the procedure is a failure. One of the most comprehensive studies to date has been the Swedish National Hip Registry. Regular reports from the registry (e.g. Malchau et al., 2000) are used worldwide as a means of monitoring trends in hip replacement technology. Between 1979–1991, the registry collected data regarding the interventions per year by clinic as well as interventions categorised by implant type. More recently, the registry can quite clearly demonstrate the variability inherent in hip replacement—e.g. two implants from the registry can show both different survival curves and different amounts of scatter in survival, see Fig. 2.1. Even surgical factors have been followed. Pulsatile lavage, which serves to clean the inside cavity and aid cement-bone interdigitation, and plugging and sealing of the femoral cavity, which improves interdigitation by increasing pressurisation of the cement when the prosthesis is inserted, have both

¹The number of individuals over the age of 50 is expected to double between 1990 and 2020 http://www.bonejointdecade.org/background/background_consensus.html

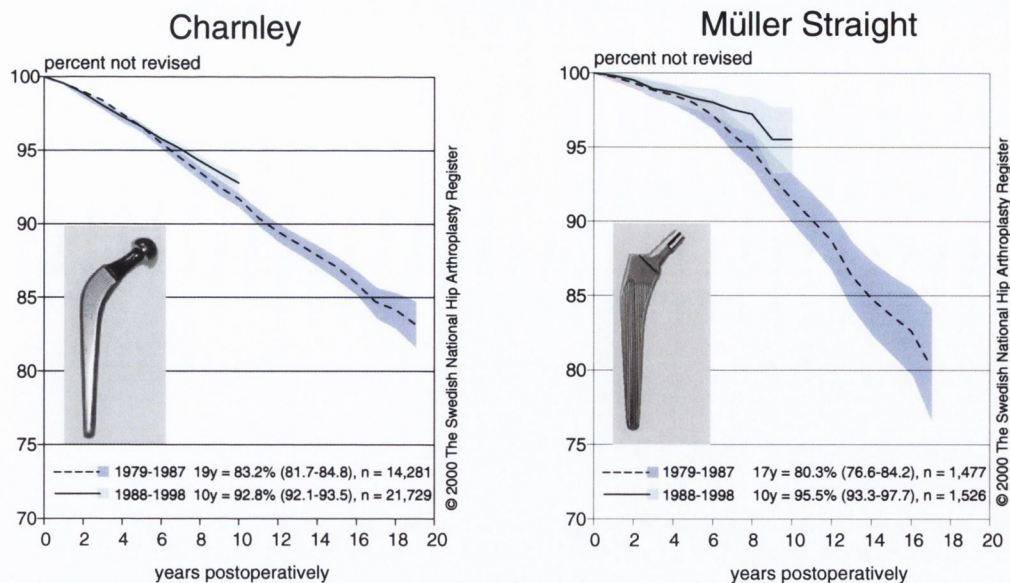


Figure 2.1. *Survival data for two implants (Charnley and Müller Straight) from the Swedish National Hip Registry (adapted from Malchau et al. (2000)). Notice that 10% of failures occur for both implants at approximately 11 years. However, the failure rate of the Muller Straight is greater in this time frame, causing it to perform more poorly in the longer term. Also, the increased variability for the Muller Straight compared with the Charnley implies that the Muller design is less reliable.*

been shown to reduce the risk of the need for revision operations.

2.3 Bone cement and failure of hip replacements

The long term results of the earliest designs identified both infection and mechanical loosening as failure modes (Charnley, 1972). Infections are no longer a major problem; the cumulative revision rate for deep infection after 10 years is only 0.3% (Malchau et al., 2000). Charnley's study also noted the existence of abnormal radiological appearances in the bone adjacent to the cement, even in cases that were clinically successful. A radiolucent zone on an x-ray implies a loss (resorption) of bone tissue. Although not perceived as a problem for a long time, e.g. Reckling et al. (1977), these regions are now commonly identified with final mechanical loosening of the implant (Harris, 1991, Malchau et al., 2000). Early microscopic studies of the tissues around the implant demonstrated the existence of cement particles in a soft surrounding tissue, cracks in the cement, and initial bone necrosis followed by bone repair (Willert et al., 1974). Later, Freeman et al. (1982) concluded that changes in the tissues were a response to the cement. They further hypothesised that mi-

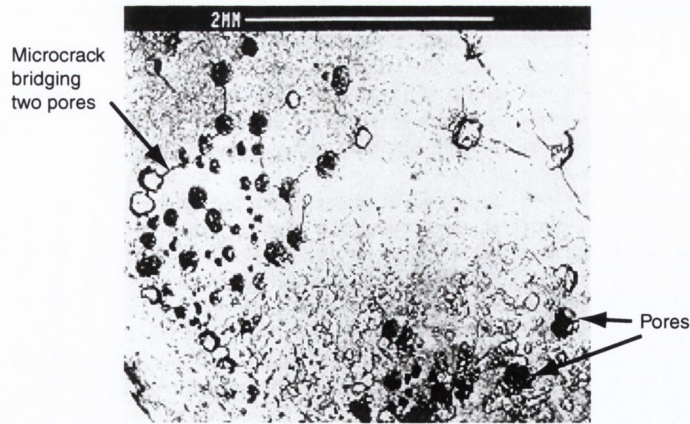


Figure 2.2. *Examples of microcracks from an autopsy retrieved cement mantle illustrating that damage accumulation has occurred in vivo. Note that all cracks occur around pores. Adapted from Jasty et al. (1991)*

cromotion at the cement-bone interface, resulting in cell death, in combination with particulate debris, (e.g. PMMA, UHMWPE, and metal) produced by wear of the materials, would stimulate these tissue reactions. In the same year, the formation of such a soft tissue layer was shown to reduce torsional stiffness between the implant and femoral cortex (Radin et al., 1982). Over time the link between the soft tissue layer and loosening became reinforced, e.g. Goldring et al. (1986), Spector et al. (1990), Fornasier et al. (1991). In conclusion, the formation of a circumferential soft tissue layer between implant and bone was accepted as signifying final loosening.

The initiation and evolution of failure remained, however, to be fully elucidated. Fornasier and Cameron (1976) found in an autopsy retrieval study that the implant frequently debonds from the cement early in the lifetime of the implant, often resulting in a thin fibrous tissue film between the implant and cement. Subsequent mechanical testing of cement-metal interfaces showed that both static (Ahmed et al., 1984) and fatigue (Raab et al., 1981) strengths of this interface were substantially less than the bulk cement material. Jasty et al. (1991) hypothesised, based on results from a time-series autopsy retrieval study, that prosthesis debonding occurred early and was followed by distributed, slowly developing, fractures in the cement initiated by stress concentrations at the implant-cement interface as well as from pores in the cement (Fig. 2.2). By fractographic examination of a retrieved cement mantle, Culleton et al. (1993) further confirmed these findings. Finally, a process of distributed damage accumulation, initiating mainly from pores under bending

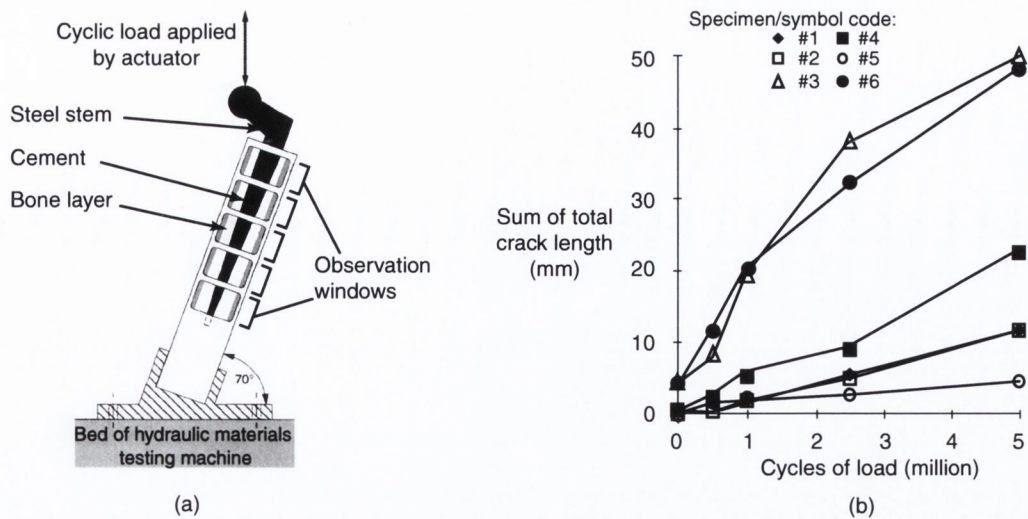


Figure 2.3. (a) Schematic of the physical model used by McCormack and Prendergast (1999) to study damage accumulation on the femoral side of a total hip replacement under flexural loading. (b) Total damage, expressed as summed crack lengths, for each specimen tested in the same study. Note the increased rate of damage growth for the specimens containing initial damage. Adapted from McCormack and Prendergast (1999)

loads and from the interface under torsional loads, was demonstrated in two physical models of the femoral side of total hip replacement, one under bending (see Fig. 2.3 and McCormack and Prendergast, 1999), and the other under torsion (McCormack et al., 1999). Statistical analysis of these results also showed that the damage accumulation rate in the cement was correlated with the amount of pre-load damage (McCormack et al., 1998). The variable damage accumulation rate observed could be attributed to the performance of the cement.

Recalling the failure scenarios proposed by Huiskes (1993), it can be envisaged that the role of bone cement in aseptic loosening is not limited to the damage accumulation scenario. Particulate debris will also form during damage accumulation and lead to the particulate reaction failure scenario. However, damage accumulation is likely to be the more dominant process on the femoral side because the prosthesis-cement interface is more highly stressed there and the cement experiences greater tensile loading. For the acetabular side, wear of the bearing surfaces is the more dominant mechanism.

2.3.1 Cement chemistry and physical properties

Bone cement is primarily composed of poly(methylmethacrylate) (PMMA), also known as Plexiglas or Perspex when manufactured for industrial purposes. It is an amorphous, glassy polymer at both room and body temperature (glass transition for PMMA is approximately 105°C). Because of the need for the cement to fill and conform to the cavity inside the bone of a replaced joint, it is normally prepared during surgery as a self-curing resin several minutes prior to insertion of the prosthesis. The dough-like resin must flow freely enough to achieve interdigitation with cancellous bone and contact with the implant materials. Once the prosthesis has been inserted the resin is allowed to cure *in situ*. It becomes hard within 10–15 minutes of initial preparation. The resin is prepared by mixing powdered polymer with liquid monomer in an approximate powder:liquid ratio of 2 g:1 ml; the actual ratios can be varied to alter viscosity, during the working phase, and setting time. Polymerisation proceeds as a free radical addition reaction, initiated by benzoyl peroxide contained in the powder. Addition of *N,N*-dimethyl-*p*-toluidine to the monomer liquid is used to activate the free radical decomposition of the benzoyl peroxide initiator. Propagation of the reaction proceeds as additions of individual monomer molecules to the free radical side of the growing polymer chain. An auto-acceleration effect, known as the Trommsdorf or Gel effect, occurring at approximately 20–50% of conversion, causes the reaction to become highly exothermic—homogenous cement masses can reach temperatures in the range 50–90°C. Tissue damage thresholds have been reported in the range of 50–60°C but sufficiently lower temperatures often occur when the cement cures in contact with a metal implant and circulating blood to prevent thermal necrosis of the bone. Also, polymerisation is inhibited by oxygen; this has the effect of decreasing the rate of monomer reaction, exotherm, chain length, and molecular weight. Relatively large proportions of peroxides, in comparison with industrial PMMA, also result in greater quantities of low molecular weight polymer. Reductions in molecular weight for a polymer are known to reduce elastic modulus, due to decreased resistance to relative motion between chains, and fracture toughness, because of the decreased craze resistance of shorter chains. In addition to

Table 2.1. *Manufacturer's Composition for Surgical Simplex P bone cement (taken from Kusy, 1978)*

Powder (40 g)		Liquid (20 cc)	
6.0 g	Polymethylmethacrylate	97.4%	Methylmethacrylate
30.0 g	Methylmethacrylate-styrene copolymer	2.6 %	N, N-dimethyl-p-toluidine
4.0 g	BaSO ₄	75 ±15 ppm	Hydroquinone

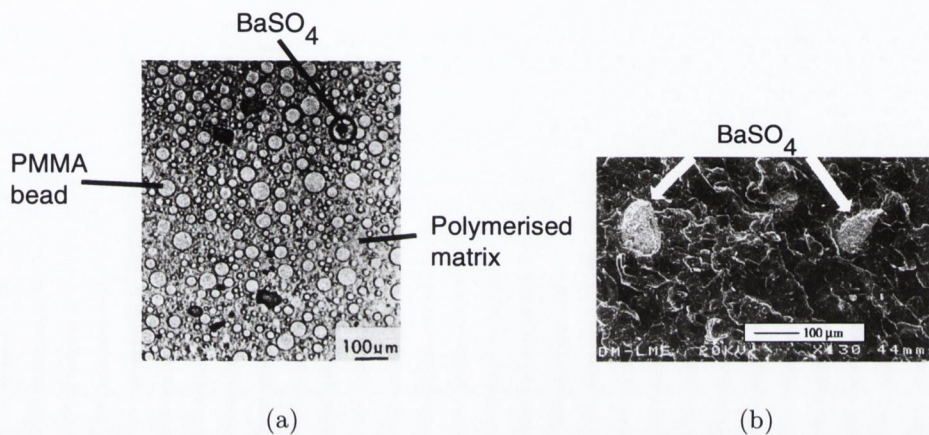


Figure 2.4. (a) *Photomicrograph of two-phase bone cement (Simplex P) showing radiopaque filler BaSO₄; adapted from Kusy (1978).* (b) *Scanning electron micrograph of fracture surface of a tensile test bone cement specimen showing BaSO₄ particles; adapted from Ginebra et al. (2002)*

reaction related additives, radiopacifiers, e.g. barium sulfate (BaSO₄), to render the cement visible under X-rays, and antibiotics, e.g. gentamycin, to minimise risk of infection, are also often added. The composition of a commercial cement is listed in Table 2.1. [Kusy (1978), Radin et al. (1982), Saha and Pal (1984), Kine and Novak (1987), McCrum et al. (1988), Sandler and Karo (1992), Pascual et al. (1996), Starke et al. (1997), Pascual et al. (1999), Kühn (2000)].

Morphology of the cured cement takes the form of the pre-polymerised beads embedded in a matrix of the polymerised monomer with interspersed inclusions of radiopaque filler (Fig. 2.4). Many studies have reported that fatigue crack initiation and propagation tends to dominate in the inter-bead matrix, while fast fracture occurs through the beads. This may be because molecular weight tends to be higher in the pre-polymerised beads than the matrix, due to the inherent imperfections of

Table 2.2. *Mechanical properties of Plexiglas and three commercial brands of bone cement. E = Young's modulus, UTS = ultimate tensile strength, UCS = ultimate compressive strength, and γ = fracture energy. E, UTS, and γ taken from Kusy (1978); UCS taken from Saha and Pal (1984) and manufacturer's ISO5833 data (Sulzer, Inc. and Biomet, Inc.)*

Brand	E (GPa)	UTS (MPa)	UCS (MPa)	γ (J/cm ²)
Plexiglas (Rohm and Haas Co.)	2.8	73.8	103	0.013–0.039
Sulfix-6 (Sulzer, Inc.)	2.4	48.3	93.3	0.034
Simplex P (Stryker Howmedica Osteonics, Inc.)	2.6	33.8	77	0.026
Palacos R (Biomet, Inc.)	2.6	46.2	87.8	0.029

the operational environment during polymerisation. Inclusions, such as radiopaque fillers, can affect fracture properties both positively and negatively since particles may act as crack arrestors or initiation sites. Copolymers, e.g. polystyrene which copolymerises readily with methylmethacrylate, are also sometimes added to bone cement to improve processing characteristics, radiation resistance, or reduced polymerisation exotherm—high concentrations of 1-hydroxypropyl methacrylate have been shown to cause bead detachment from the interstitial matrix, while polystyrene has been proposed as a potential weak link because its mean inherent flaw size is an order of magnitude greater than that for PMMA. [Kusy (1978), Wright and Robinson (1982), Kine and Novak (1987), Gilbert et al. (1990), Topoleski et al. (1993), Pascual et al. (1999), Murphy and Prendergast (2000a), Ginebra et al. (2002)].

Biological inclusions, e.g. blood and fat, reduce mechanical strength. PMMA is also hydrophilic, absorbing up to several weight percent water. Absorbed water acts as a plasticising agent and has been shown to increase fatigue life. Mechanical properties of different cement formulations can therefore be expected to vary, especially considering the application environment of surgery (Table 2.2) [Freitag and Cannon (1977), Kusy (1978), Saha and Pal (1984), McCrum et al. (1988)].

Further to the effects of additives, bone cement is prone to ageing as complete

conversion of monomer is difficult to achieve under the relatively uncontrolled reaction conditions; between 2–5% monomer have been reported. Residual monomer tends to either polymerise over time, causing ageing phenomena due to increasing molecular weight, or to diffuse into the surrounding tissues where it can lead to necrosis because of its cytotoxicity. [Kusy (1978), Davy and Braden (1991), Willert et al. (1974)].

2.3.2 Mixing methods, porosity, and fatigue behaviour

In the early days, mixing of the cement was done in a bowl with a spatula; this is often referred to as the “first generation” mixing technique. However, high porosity, caused by air entrapment during mixing, resulted in inferior mechanical properties compared with industrial PMMA for this mixing method (Table 2.2). Heating of the cement as monomer boils can also form pores. The third major cause of porosity is related to the rheological behaviour of the cement as it comes in contact with the inserted prosthesis—large numbers of pores have been observed forming at this interface during implant insertion and the phenomenon has been related to the shear rate experienced by the doughy cement as the prosthesis is inserted. [Charnley (1979), Kusy (1978), Saha and Pal (1984), Jasty et al. (1990), James et al. (1993), Pascual et al. (1996), Spiegelberg and McKinley (1999), Dunne and Orr (2001)].

Importance of the role of porosity in initiating fatigue failure of bone cement has been demonstrated in many fractographic studies. In the already raised stress state occurring around a pore, a further stress concentration occurs between the bead-matrix microstructure, making these primary sites for crack initiation (Fig. 2.5). In addition to this, pores often cluster together so that interactions are likely to occur. Tsukrov and Kachanov (1997) have shown that complex interaction effects can occur between pores in a brittle solid, even under remote uniaxial tension, such that cracks may not initiate or propagate normal to the remote stress (Fig. 2.6a). Evidence of interaction between propagating cracks and pores can be observed in fractographic results (Fig. 2.6b). In a time-lapse study of damage accumulation in uniaxial specimens, Murphy and Prendergast (1999) noted that microcracks initiated from

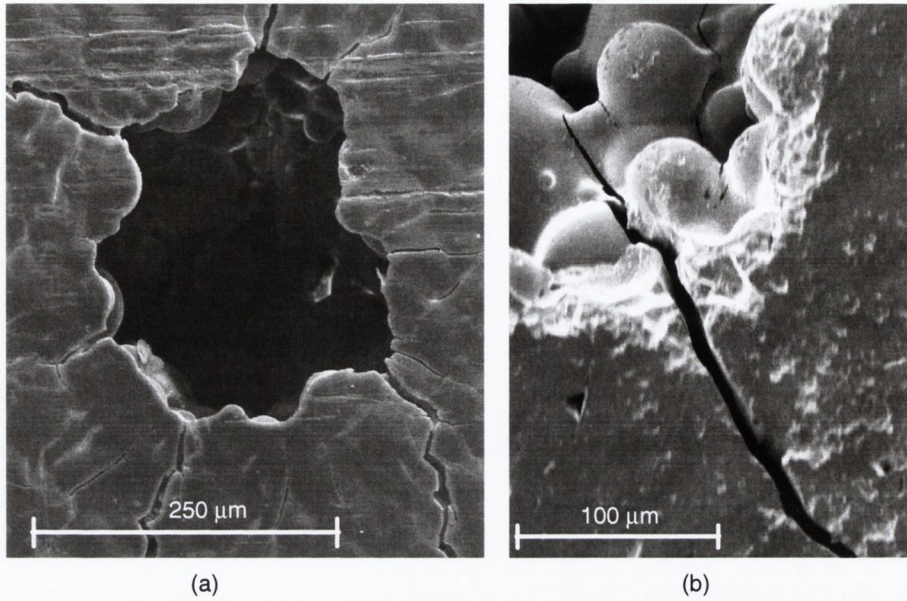


Figure 2.5. (a) Example of crack initiation around a pore. (b) Stress concentrations between beads initiate cracks in the weaker matrix phase. Scanning electron micrographs courtesy of Dr. Bruce P. Murphy (Murphy (2001)).

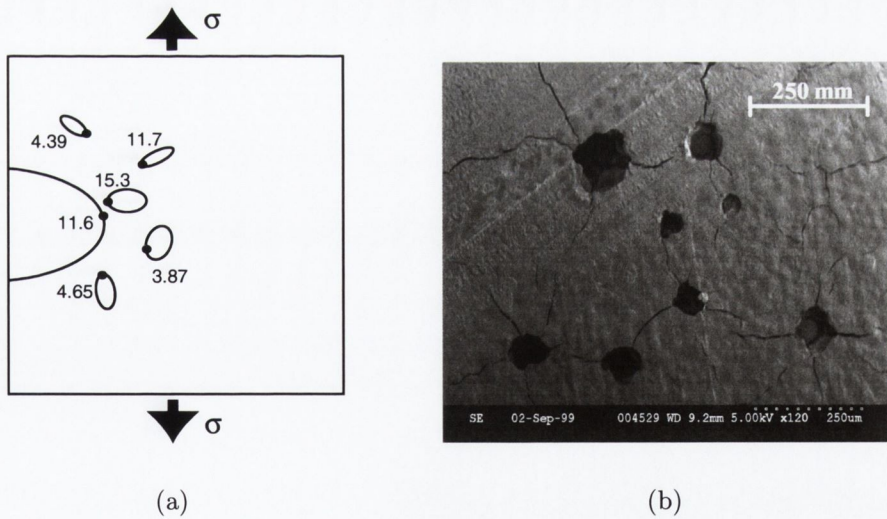


Figure 2.6. (a) Illustration of the effect of interactions between a large pore on stress intensity factors (SIF; ●) for a set of smaller pores—one of the moderately elliptical pores near the larger one has the highest SIF rather than one of the more elongated pores or the large pore. Peak stress for a given pore may cause a crack to initiate in a direction that is not perpendicular to applied stress (e.g. pore with 4.65 SIF). Adapted from Tsukrov and Kachanov (1997). (b) Evidence of cracks emanating from pores and propagating towards other pores; courtesy of Dr. Bruce P. Murphy (private communication).

Table 2.3. Comparison of mechanical properties of Simplex P for three mixing techniques. Porosity values do not correspond to failure lives as they are taken from a separate study; instead they are shown as representative values.

Property	Manual	Centrifuge	Vacuum
Porosity (vol%) [†]	10.0 ±2.3	5.0 ±2.7	0.5 ±0.5
Weibull fatigue lives at 15 MPa (cycles) [‡]	20,000 ±9,000	85,000 ±45,000	150,000 ±45,000
Fracture Toughness (MPa(m ^{1/2})) [§]	0.95 ±0.01	1.05 ±0.01	1.14 ±0.02

[†] Linden and Gillquist (1989); [‡] Wixson et al. (1987); [§] Lautenschlager et al. (1986)

pores. Further development of this work showed positive correlation of damage accumulation per pore per cycle with applied stress [Freitag and Cannon (1977), Gilbert et al. (1990), Krause and Mathis (1984), Topoleski et al. (1993), Murphy and Prendergast (2000ab)].

Efforts to increase cement fatigue life by reduction of porosity have focussed on two methods: mixing (either mechanically or manually) in a sealed container under partial vacuum and centrifugation. Both methods have been shown to reduce porosity (Table 2.3). [Demarest et al. (1983), Eyerer and Jin (1986), Rimnac et al. (1986), Wixson et al. (1987), Linden and Gillquist (1989), Jasty et al. (1990), Lewis et al. (1997), Smeds et al. (1997), Dunne and Orr (2001)].

Numerous studies of fatigue behaviour have also demonstrated improved fatigue life for vacuum mixing and centrifugation. However, the fatigue lives for such specimens are highly variable—centrifuged specimens still contain small pores and are prone to density variation while vacuum mixed samples are susceptible to occasional large pores. Davies and Harris (1990) speculated that vacuum mixing would not eliminate early failures, in spite of increases in average strength, because of the presence of such large pores. [Burke et al. (1984), Lautenschlager et al. (1986), Davies et al. (1987), Wixson et al. (1987), Linden and Gillquist (1989), Linden (1989), Jasty et al. (1990), Lewis and Austin (1994), Fritsch et al. (1996), Wang et al. (1996), Lewis et al. (1997), Smeds et al. (1997), Lewis (1999), Murphy and Prendergast (2000a)].

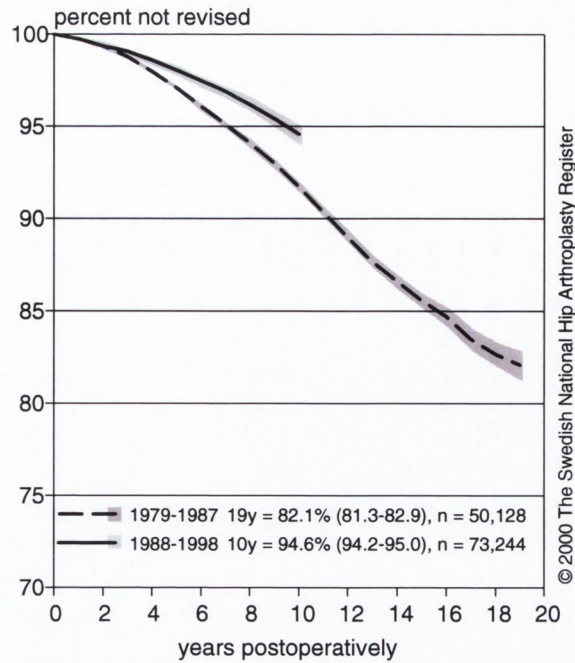


Figure 2.7. *Survival of cemented hip replacements grouped by time periods in which different cementing techniques dominated. Adapted from Malchau et al. (2000)*

Data on bone cement strength has been obtained from laboratory studies which may or may not translate to clinical applications. Controversially, Ling and Lee (1998) have suggested that clinical evidence does *not* support the goal of porosity reduction in hip replacement. On the other hand, results from the Swedish Hip Registry clearly demonstrate that improvements in cementing technique have led to improved survival of cemented hip replacements (see Malchau et al., 2000, and Fig. 2.7).

2.3.3 Viscoelasticity of bone cement

Although fatigue is a significant factor in failure of hip replacements, it is not the only possible failure mode. Migration of the implant relative to the bone over time have been observed on X-rays. This signified the possibility of creep induced loosening. Like other amorphous polymers, PMMA is viscoelastic—it experiences two relaxation process which are due to:

1. Motion and rotation of the molecular backbone of the polymer chain (α -relaxation) and
2. Rotation of the COOCH_3 side group (β -relaxation).

The glass transition corresponds to the α -relaxation and occurs at approximately 105–110°C while β -relaxation occurs at approximately 50°C; these relaxation processes are thus significantly retarded at the typical service temperatures of bone cement (i.e. body temperature). Nevertheless, both creep and stress relaxation have been demonstrated for bone cement. [Lee et al. (1977), Pal and Saha (1982), Ebrahimzadeh et al. (1983), Kine and Novak (1987), McCrum et al. (1988), Yetkinler and Litsky (1998), Murphy (2001)]

It has been argued that, because of radial and hoop creep of the cement layer surrounding a femoral prosthesis, implants should be designed to accommodate the inevitable effects of creep by facilitating prosthesis subsidence within the cement (Fowler et al., 1988, Ling, 1992, Lee, 1994). An example of such a femoral prosthesis is the Exeter prosthesis (Stryker Howmedica, Inc.). It has a polished surface and therefore debonds easily from the cement, leaving its tapered geometry to generate support through a wedging action; the prosthesis can thus achieve continuous stabilisation as cement creep leads to further subsidence. As mentioned previously, early migration has been proposed as an indicator of clinical loosening—this has caused an apparent paradox for the Exeter prosthesis since it migrates significantly more than other prostheses while its clinical results are in the range of the best performing implants (Malchau et al., 2000).

Finite element studies, using viscoelastic constitutive models, have not supported the hypothesis that clinically-observed prosthesis migration is due to creep only (Lu and McKellop, 1997). On the contrary, Verdonschot and Huiskes (1997a) have predicted that the cement and its interfaces can experience beneficial stress relaxation because of load redistribution as the prosthesis subsides. They further proposed that such reductions would decelerate damage accumulation. The hypothesis of Verdonschot and Huiskes (1997a) for creep-fatigue interaction in bone cement is different than the type of interaction observed in metals. Fatigue in metals is characterised mainly by transcrystalline microcracks initiating from surface flaws while creep occurs due to cavity formation at grain boundaries (Chaboche, 1999)—simultaneous creep and fatigue damage could therefore be expected to accelerate total damage

accumulation. In contrast, bone cement creep is due to molecular rearrangements and, according to Verdonschot and Huiskes' hypothesis, may decelerate damage by relaxing stresses. However, this hypothesis only considers one direction of interaction, i.e. the effect of creep on fatigue damage accumulation. Damage accumulation is likely to affect creep also by altering local stresses in damaging regions. Decisive investigation of the overall effect therefore requires fully coupled creep-damage models.

2.3.4 Residual stress and the initiation of damage

Residual stress is a common problem in manufacturing PMMA components (Kine and Novak, 1987) and early evidence of its existence in bone cement was reported by Kusy (1978). If residual stresses are large enough, in particular for a porous cement, they may initiate microcracks.

There are two main sources of shrinkage in curing bone cement. Firstly, a volume change of approximately 20% occurs as the liquid polymerises to a solid. However, as bone cement is a two phase mixture of MMA and PMMA beads this volume change is not as large as might be expected. It depends on the ratio of liquid monomer to polymer beads—shrinkage for typical liquid-powder ratios is approximately 7%. Secondly, due to the exothermic nature of the polymerisation reaction, there is a significant thermal shrinkage during cooling to ambient temperature. These can be considered as separate phenomena since the first is associated with the formation of bonds while the second is associated mainly with thermal deformation of already formed polymer chains. The thermal expansion coefficient for a cooling cement mass has been measured to be in the range $7\text{--}9\%^\circ\text{C}^{-1}$. However, at what time the cement becomes capable of supporting stress is not certain because of the rapid molecular changes occurring during the reaction. [Kine and Novak (1987), Ahmed et al. (1982a)].

Initial attempts to model residual stress generation around hip prostheses were based on thermal shrinkage from peak temperature (Huiskes, 1980). This analysis predicted peak tensile hoop stresses of approximately 3.5 MPa. However, Ahmed

et al. (1982a) proposed that the cement became capable of supporting stress at the onset of the rapid temperature rise. In the second part of their study, Ahmed et al. (1982b) used a constitutive model for a biphasic hardening material to simulate stress generation during both thermal expansion and contraction. They predicted peak tensile hoop stresses of approximately 2 MPa. Mann et al. (1991) predicted higher stresses but assumed only thermal shrinkage from a uniformly distributed peak temperature. In a physical model of the femoral side of a hip replacement, McCormack and Prendergast (1999) measured significant levels of microcracking around pores before any external loads had been applied and hypothesised that shrinkage stress was the cause.

Using fibre-optic Bragg sensors embedded in curing cement, Whelan et al. (2000) did not measure any shift in grating wavelength attributable to strain until after the peak temperature had been reached. This supports the hypothesis of stress-locking at the time of the temperature peak. Using a similar heat generation model to Huiskes (1980) and a bilinear thermal expansion response, the strain of the grating was simulated in a finite element model of the same test (Lennon et al., 2000)². Relatively large shrinkage strains ($> 4 \text{ m}\epsilon$) were measured and predicted in this homogeneous sample. Because of the constraints imposed by bonding with the interfaces in a femoral replacement, significant shrinkage stress could therefore be expected. Application of a later development of this numerical model to a physical model of a femoral replacement, in which damage prior to external loading had been measured, showed that shrinkage stress could interact with porosity to initiate damage for the case of shrinkage from peak temperature (Lennon and Prendergast, 2002)³.

2.4 Fatigue damage accumulation

From the description given in Section 2.3 above, it is evident that bone cement undergoes fatigue damage accumulation and that this is one of the reasons for failure of hip replacements. Damage is initiated around pores by polymerisation-induced

²This paper is included in Appendix A, pp. 167

³Included in Appendix A, pp. 168

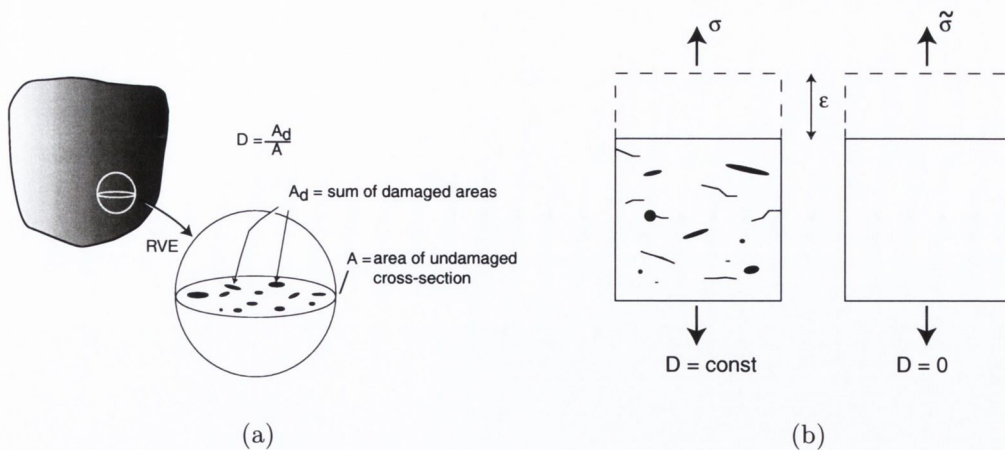


Figure 2.8. (a) Definition of damage as a reduction in cross-sectional area for a given RVE. (b) Strain equivalence: effective stress, $\tilde{\sigma}$, is the stress required to produce the equivalent strain, ϵ , in an identical but undamaged specimen as is observed in the damaged specimen.

shrinkage stress. It accumulates under mechanical load to cause failure of the fixation.

2.4.1 Description of the damaged state

Two questions immediately arise in describing a damaged state:

1. What measurable physical property best describes the damaged state?
2. To what extent does this measurement depend on direction of observation?

Chaboche (1987) lists measures of the damaged state as follows: microcrack area or void volume of a representative volume element (RVE), remaining life, density change, resistivity change, acoustic emission and/or changes in sound velocity, and changes in elastic properties (e.g. stiffness). Although direct measurement of cracks or voids is often only possible through destructive testing, the concept of reduction of net cross-sectional area or crack/void density, see Fig. 2.8(a), is both common and intuitive; see for example the papers of Davison and Stevens (1973), Budiansky and O'Connell (1976), Krajcinovic and Fonseka (1981), Murakami (1983), Onat and Leckie (1988), Singh and Digby (1989), Kachanov (1992), and Radayev (1996). The damage variable may also be introduced as an operator in a mapping between the response of the undamaged and damaged materials. A common form is an effective

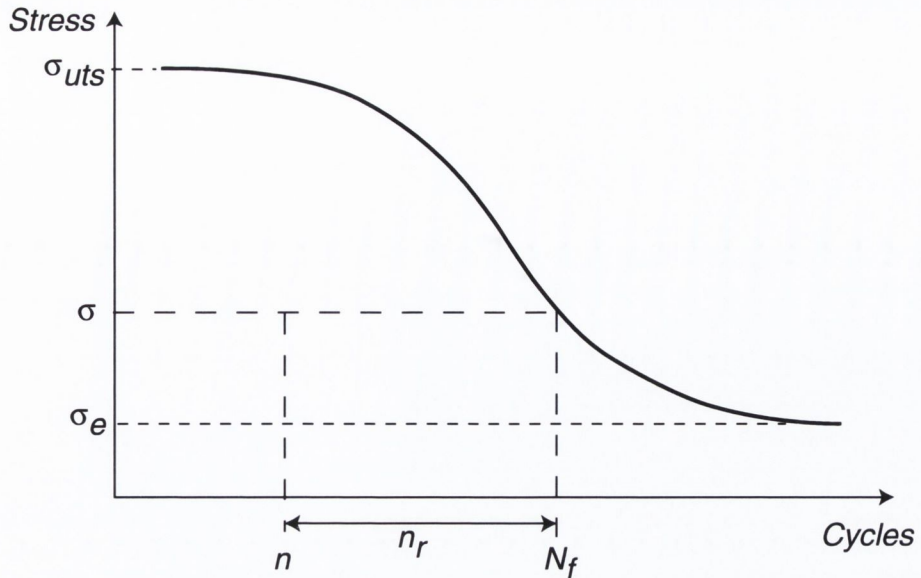


Figure 2.9. Stress-cycles to failure curve illustrating typical quantities provided from fatigue tests: N_f is no. cycles to failure for an applied stress σ , σ_{uts} is the ultimate tensile strength of the undamaged specimen, n is the number of cycles elapsed at the applied stress without having reached failure, σ_e is the endurance limit, if one exists, at which the specimen can undergo cycling indefinitely, and n_r is the remaining life (i.e. $n_r = N_f - n$).

stress corresponding to a fictional stress required to cause either an equivalent strain or strain energy density for an undamaged material (see Fig. 2.8(b) and Cordebois and Sidoroff, 1983, Simo and Ju, 1987, Voyiadjis and Kattan, 1992). These measures are useful when actual crack distributions or recorded strain data exist but this is often not the case for fatigue.

Fatigue data is often presented as a relationship between applied stress, σ , and average lifetime at that stress, N_f . Therefore, fatigue damage can be expressed as a function of remaining life, n_r , or life fractions at a particular stress level, n/N_f , deterioration in ultimate tensile strength, σ_{uts} , or decrease in endurance limit σ_e , see Fig. 2.9 for illustration of nomenclature, and Miner (1945), Chaboche (1977), Manson (1979), Fatemi and Yang (1998) for examples.

To illustrate the potential effect of direction of observation on measured damage, consider the case of spherical versus ellipsoidal voids. Spherical voids have the same projection regardless of the section taken through the RVE whereas ellipsoidal voids will not. This, and the requirement of invariance under changes in frames of reference in constitutive theories, has led to damage variables ranging from simple

Table 2.4. *Examples of varying degrees of anisotropy proposed and/or used in several studies.*

Tensor rank	Authors	Description
0 (scalar)	Chaboche (1977)	Uniaxial fatigue
	Lemaitre (1985)	Ductile void growth
1 (vector)	Krajcinovic and Fonseka (1981)	Oriented penny-shaped cracks
2	Murakami (1983)	Intergranular creep cavity growth
	Kachanov (1992)	Micromechanical analysis of effective properties of cracked elastic media
4	Chaboche (1983)	Effective stress transformation using strain equivalence hypothesis
	Simo and Ju (1987)	Damage as operator for both effective stress and effective strain transformations
8	Chaboche (1983)	Discussed a formal requirement for an eighth order tensor to transform the fourth order elasticity tensor from undamaged to damaged state

scalars to tensors of varying rank, see Table 2.4. A further development of tensorial descriptions of anisotropy has been the use of series expansions of even ordered tensors (Onat and Leckie, 1988, Krajcinovic, 1996); although in theory these are infinite series, practical considerations usually limit the expansion to the fourth order tensor term.

2.4.2 Evolution of the damage variable for complex cyclic loading histories

Microcracks are not usually measured during fatigue tests so models for fatigue damage growth have had to rely predominantly on time-to-failure data. The first fatigue damage rule has been attributed to Palmgren (1924). Miner (1945) developed

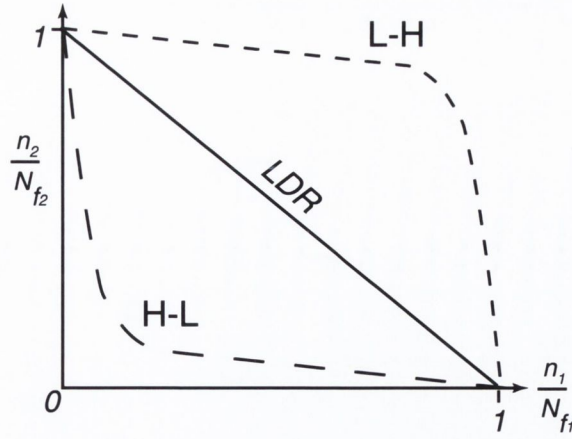


Figure 2.10. Illustration of load-sequence effects for High-Low (H-L) sequence and Low-High (L-H) sequence (LDR = Linear Damage Rule).

this into a linear damage rule (LDR) and tested it in experiments on an aluminium alloy. Briefly, the linear rule (2.1) defines damage, D_i , at a particular stress level, σ_i , as the ratio of cycles accumulated at that stress level, n_i , relative to the total number of cycles required to cause failure at the same stress, N_{f_i} :

$$D_i = \frac{n_i}{N_{f_i}} . \quad (2.1)$$

Furthermore, it states that N loading blocks of different stress levels for a given specimen can be linearly combined and should equal unity at failure (2.2):

$$\sum_{i=1}^N D_i = \sum_{i=1}^N \frac{n_i}{N_{f_i}} = 1 . \quad (2.2)$$

However, for specimens subjected to alternating sequences of high stress (H) and low stress (L) loading blocks, large deviations from linearity occur (Manson, 1979, Halford, 1997, Fatemi and Yang, 1998). For metals, H-L sequences frequently result in $\sum n_i/N_{f_i} < 1$ (i.e. the LDR over-predicts the lifetime) while L-H sequences often result in $\sum n_i/N_{f_i} > 1$ (Fig. 2.10).

Many developments in fatigue damage modelling have focussed on addressing this problem. Fatemi and Yang (1998), in a comprehensive review, grouped departures from the linear damage rule into the following categories:

1. Two-stage (double) linear damage rule (DLDR) and nonlinear damage curve approaches (DCA). Both of the models presume that a separation of damage accumulation into crack initiation and propagation phases can be made,

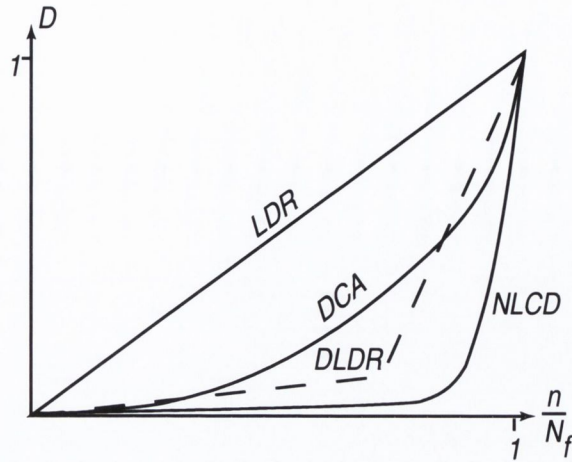


Figure 2.11. Examples of varying degrees of nonlinearity—LDR = Linear Damage Rule, DLDR = Double Linear Damage Rule, DCA = Damage Curve Approach, NLCD = Nonlinear Continuous Damage rule.

2. Life-curve modification methods—endurance limit reduction and/or rotation of S-N curve between loading blocks,
3. Approaches based on crack growth concepts—application of fracture mechanics concepts to short crack and macrocrack propagation,
4. Continuum damage models—relate current damage and loading to damage growth as a nonlinear continuous damage (NLCD) model, and
5. Energy-based theories—relate hysteresis observed in strain energy to fatigue behaviour.

Fig. 2.11 shows some of these graphically.

Many of these models, although derived from different assumptions, can be expressed in terms of life fractions, n_i/N_{fi} , and hence can be represented as individual, or combinations of, damage curves with different exponents, α ,

$$D_i = \left(\frac{n_i}{N_{fi}} \right)^\alpha . \quad (2.3)$$

For example, (Chaboche and Lesne, 1988) proposed a differential formulation,

$$dD = D^{\alpha(\sigma_M, \bar{\sigma})} \left[\frac{\sigma_M - \bar{\sigma}}{M(\bar{\sigma})} \right]^\beta dn , \quad (2.4)$$

where α is an exponent used to make damage and stress inseparable variables, σ_M and $\bar{\sigma}$ are maximum and mean stress, respectively, for a cycle. $M(\bar{\sigma})$ is a function

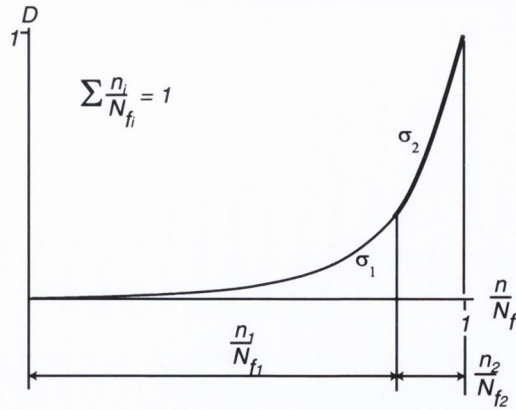


Figure 2.12. Schematic representation of nonlinear evolution and linear accumulation. The heavier line represents damage accumulated at the second stress. Notice that, in spite of the nonlinear evolution, the life fractions sum linearly to unity

used to describe the dependence of fatigue limit on mean stress and β is a material coefficient. Integration of (2.4), between $D = 0$ to $D = 1$, enabled them to write an expression similar to (2.3); i.e.

$$D = \left(\frac{n}{N_f} \right)^{\frac{1}{1-\alpha}} . \quad (2.5)$$

They then illustrated, by suitable choice of the exponent function, α , an equivalence between this formulation and several others.

Although nonlinear evolution is a common feature of most of these models, it is not sufficient in itself to result in nonlinear accumulation of damage between different loading blocks, i.e. not sufficient to invalidate eqn. 2.2 (Chaboche and Lesne, 1988). Dependence on damage only, i.e. when the exponent does not contain any dependence on loading, will result in the same damage curve with respect to life-fractions and always result in summation to unity, just as for the LDR (see Fig. 2.12). Nonlinear accumulation requires a dependence on stress levels in addition to existing damage in the function for damage evolution (Chaboche and Lesne, 1988). For a damage curve representation, this implies that the exponent, α , is a function of applied stress, i.e. damage growth is then a function of both existing damage and applied stress—this results in distinct curves for each loading (see Fig. 2.13). When accumulating damage from an earlier loading it is then necessary to do so from the point of equal damage on the curve for the current loading, resulting in a non-unity summation of life-fractions.

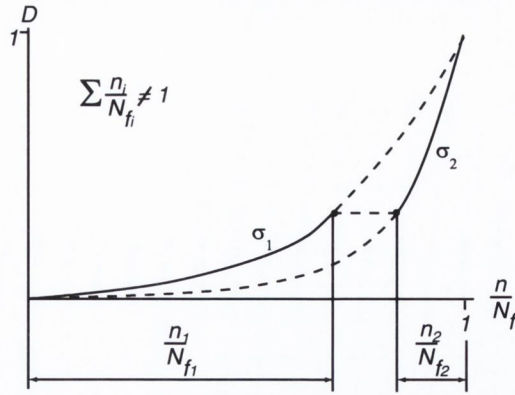


Figure 2.13. Schematic representation of nonlinear evolution and accumulation; solid curves represent periods of application of each stress level.

Most fatigue damage accumulation models have been developed to describe uniaxial test data and so represent the evolution of a scalar damage variable. Extensions to multiaxial stress states have comprised two main approaches:

1. Functions of the invariants of the stress or strain tensor (reviewed in Lemaitre and Chaboche, 1994) to take account of dependence on equivalent shear stress amplitudes and mean hydrostatic stress,
2. Critical plane criteria, e.g. De-guang and De-jun (1998), formulated in terms of the amplitude of shear stress or strain occurring in the plane that experiences maximum shear during a cycle and incorporating a dependence on hydrostatic stress.

Another approach uses estimates of the elastic and plastic portions of the strain energy density for a cycle to relate the total mechanical cyclic strain energy density to fatigue life, e.g. Ellyin and Golos (1988). Since strain energy density can be formulated in terms of invariants of the stress and strain tensor this approach also incorporates sensitivity to shear components and mean or maximum hydrostatic stress. Lemaitre and Chaboche (1994) extended an earlier uniaxial fatigue model (Chaboche, 1977) to propose a framework for multiaxial fatigue using criteria based on the hydrostatic and von Mises stress to give a scalar damage growth equation; multiaxial damage was recovered from the scalar evolution equation using a fourth order tensor accounting for the predominant orientation of damage.

2.4.2.1 *Thermodynamic considerations for damage evolution*

In selecting a suitable form for a damage rule, the branch of damage modelling commonly referred to as Continuum Damage Mechanics (CDM) applies principles of irreversible thermodynamics to develop admissible constitutive theories. A fundamental assumption of these models is that damage⁴, \mathcal{D} , corresponds to an internal variable capable of describing the thermodynamic state of a material, e.g. Krajcinovic and Fonseka (1981), Cordebois and Sidoroff (1983), Murakami (1983), Lemaitre (1985), Simo and Ju (1987), Onat and Leckie (1988), Chaboche (1992), Bhattacharya and Ellingwood (1999). The Second Law of Thermodynamics, in the form of the Clausius-Duhem inequality, can be written

$$\Phi = \Phi_{loc} + \Phi_{con} \geq 0, \quad (2.6)$$

where Φ is the total dissipation, Φ_{loc} is the local dissipation attributable to mechanical processes, and Φ_{con} is the dissipation attributable to heat conduction. Furthermore, the strong form of this inequality states that these dissipations must separately satisfy the inequality (Malvern, 1969); i.e.

$$\Phi_{loc} \geq 0 \quad \text{and} \quad \Phi_{con} \geq 0. \quad (2.7)$$

The next assumption is that two functions can be used to describe the thermodynamic state and complementary evolution of the internal state variables. Firstly, the Helmholtz free energy, ψ , of the system is used as the thermodynamic potential for an isothermal process and is assumed to be a function of the observable and internal state variables, e.g. Malvern (1969), Lemaitre and Chaboche (1994). For a material with damage this corresponds to

$$\psi = \psi(\epsilon, \theta, \mathcal{D}), \quad (2.8)$$

where ϵ , small strain tensor⁵, and θ , temperature, are the observable state variables and the only internal variable corresponding to an irreversible process is damage, \mathcal{D} .

⁴The symbol \mathcal{D} is used to signify that no tensorial nature has been assumed and to differentiate it from subsequent notation

⁵The following notation is adopted for tensor quantities: scalars are upper- or lowercase normal typeface (a, A, α), vectors are normal typeface with an overhead arrow (\vec{v}), second order tensors are lowercase bold ($\mathbf{t}, \boldsymbol{\tau}$), and fourth order tensors are uppercase bold (\mathbf{T})

A more general form is often used to account for other irreversible processes, such as plasticity, in addition to damage, e.g. Cordebois and Sidoroff (1983), Murakami (1983), Krajcinovic (1983), Lemaitre (1985), Chaboche (1987), Simo and Ju (1987), Onat and Leckie (1988), Bhattacharya and Ellingwood (1999). The thermodynamic forces associated with changes in the internal variables can be written

$$\boldsymbol{\sigma} = \rho \frac{\partial \psi}{\partial \boldsymbol{\epsilon}}, \quad s = -\frac{\partial \psi}{\partial \theta}, \quad \text{and} \quad \mathcal{Y} = -\rho \frac{\partial \psi}{\partial \mathcal{D}}, \quad (2.9)$$

where $\boldsymbol{\sigma}$ is the Cauchy stress, ρ is the mass density, s is the specific entropy, and \mathcal{Y} is the associated thermodynamic force for damage (Lemaitre and Chaboche, 1994); the minus sign is arbitrary and corresponds to the intuitive assumption that damage releases energy from the system.

Having identified the thermodynamic force associated with damage, a function to describe damage evolution is required. Two approaches are commonly found in the literature: (i) postulating a dissipation potential and (ii) construction of a damage surface. In the first approach, dissipation is assumed to be governed by evolution of the internal variables so that a complementary potential of dissipation, ϕ , can be hypothesised as a function of the flux of the irreversible variables. The internal variables themselves may also be included as parameters, denoted by quantities following the semicolon: (Murakami, 1983, Krajcinovic, 1983, Lemaitre, 1985):

$$\phi = \phi \left(\dot{\mathcal{D}} ; \boldsymbol{\epsilon}, \theta, \mathcal{D} \right). \quad (2.10)$$

As the evolution of the internal variables are of interest it is preferable to obtain the dissipation potential in terms of the thermodynamic forces. Such a dual dissipation potential, ϕ^* , can be constructed by means of a Legendre-Fenchel transformation (Germain et al., 1983, Lemaitre and Chaboche, 1994)

$$\phi^* = \phi^* \left(\mathcal{Y} ; \boldsymbol{\epsilon}, \theta, \mathcal{D} \right), \quad (2.11)$$

where only local dissipation has been considered. Damage evolution is then expressed as

$$\dot{\mathcal{D}} = -\frac{\partial \phi^*}{\partial \mathcal{Y}}. \quad (2.12)$$

In the context of generalised forces and fluxes (Malvern, 1969, Germain et al., 1983, Krajcinovic, 1983) the contribution to dissipation from damage evolution can be

considered as $\mathcal{Y}\dot{\mathcal{D}}$ so that, to satisfy the strong form of the Clausius Duhem inequality (2.7), $\dot{\mathcal{D}}$ must be non-negative. An example of an explicit postulation of a dissipation potential for fatigue with subsequent derivation of the evolution equation can be found in Cheng and Plumtree (1998).

In the second approach, a damage criterion, f , is proposed as

$$f(\mathcal{Y}, \mathcal{D}) = \mathcal{Y} - K(\mathcal{D}) \leq 0, \quad (2.13)$$

where $K(\mathcal{D})$ is the critical value that \mathcal{Y} must reach before damage growth can occur and additionally is a function of the damage state (Simo and Ju, 1987, Chaboche, 1992); if \mathcal{D} is tensorial of rank > 0 , then this criterion function implies a surface. A function of the local dissipation and the damage criterion can be constructed as

$$F = \mathcal{Y}\dot{\mathcal{D}} - \dot{\lambda}f, \quad (2.14)$$

where $\dot{\lambda}$ is a Lagrange multiplier (Voyiadjis and Kattan, 1992). A hypothesis of maximum dissipation (Simo and Ju, 1987) implies that $\partial F/\partial \mathcal{Y} = 0$ to give

$$\begin{aligned} \frac{\partial F}{\partial \mathcal{Y}} &= \dot{\mathcal{D}} - \dot{\lambda} \frac{\partial f}{\partial \mathcal{Y}} = 0 \\ \Rightarrow \dot{\mathcal{D}} &= \dot{\lambda} \frac{\partial f}{\partial \mathcal{Y}}. \end{aligned} \quad (2.15)$$

From (2.13) $\partial f/\partial \mathcal{Y} = 1$ so that the Lagrange multiplier is identified as $\dot{\mathcal{D}}$. To prevent damage growth during unloading the following conditions,

$$\dot{\mathcal{D}} \geq 0, \quad f(\mathcal{Y}, \mathcal{D}) \leq 0, \quad \text{and} \quad \dot{\mathcal{D}}f(\mathcal{Y}, \mathcal{D}) = 0, \quad (2.16)$$

known as the Kuhn-Tucker relations, are imposed, see e.g. Simo and Ju (1987) and Chaboche (1992). This framework thus implies that $\dot{\mathcal{D}}$ is always positive so that the second law is satisfied as long as \mathcal{Y} is positive. Other dissipative processes can also be introduced into this framework using extra criterion functions, i.e. f_i , and Lagrange multipliers, $\dot{\lambda}_i$, for each dissipative process, e.g. Voyiadjis and Kattan (1992). An example of a fatigue evolution equation developed in similar fashion to this approach can be found in Xiao et al. (1998).

Development of a constitutive model thus involves construction of these functions in a suitable form such that the First and Second Laws are satisfied. Additionally

the principles of Determinism of Stress, Locality, and Material Frame Indifference (Objectivity) along with material symmetry constraints must be satisfied (Malvern, 1969, Germain et al., 1983, Gummert, 1999). A benefit of these continuum damage models is that they provide a natural framework for incorporating other damage processes by the introduction of extra internal variables, such as creep damage (Chaboche, 1999).

2.4.2.2 *Interactions (nonlocality) in damage evolution*

For dilute concentrations of damage little or no interaction between the stress fields around microcracks occurs and the damage can be considered local (Krajcinovic, 2000). However, the aim of a damage theory is to predict the onset of localisation of damage to a specific region, initiating a critical flaw in the specimen, and to describe the subsequent failure process. At high density the interactions of stress fields between cracks can no longer be neglected. This can lead to stress shielding or amplification relative to the dilute case (Chudnovsky et al., 1987, Kachanov, 1987 1992, Krajcinovic, 2000). This has led to difficulties with numerical implementations of damage models. Instances of both mesh sensitivity and numerical instability have been reported in finite element models (Benallal et al., 1988, Bažant and Pijaudier-Cabot, 1988). A solution to this is to impose localisation limiters, such as a lower bound on the finite element size or the introduction of nonlocal descriptions of the state variables (Bažant and Pijaudier-Cabot, 1988, Belytschko and Lasry, 1988). However, possibilities can generally be classified as (Belytschko and Lasry, 1988):

1. Integral limiters—state variables include an integral over a finite surrounding domain within the discretisation (mesh in finite element analysis),
2. Differential limiters—gradients of state variables are included in the local definition of the variables,
3. Rate limiters—time dependence is built into equations.

Bažant and Pijaudier-Cabot (1988) proposed an integral limiter for damage only with local elastic behaviour, having concluded from an earlier study that this was

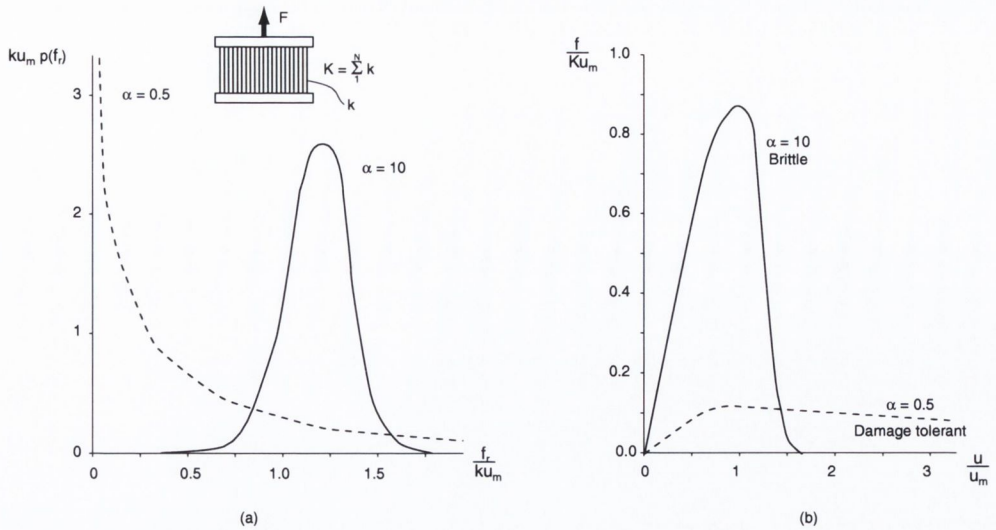


Figure 2.14. Relationship between distribution of rupture strengths and damage tolerance for a parallel bar model. (a) Weibull distribution of link rupture strengths for two values of the Weibull shape parameter α ; k = individual link stiffness, u_m is the displacement at maximum force, and f_r is the rupture force of a given link. Increasing values of α imply decreasing bandwidth of rupture strengths. (b) Corresponding force displacement curves; F = total force of the bundle and K = stiffness of bundle. Adapted from Krajcinovic (1996).

sufficient. Saanouni et al. (1989) also used an integral limiter, of exponential type, to show that equivalent results to macroscopic fracture approaches could be attained for brittle fracture. They also achieved better comparison with experimental results of ductile failure than for a global approach. Belytschko and Lasry (1988) have implemented a gradient limiter for strain while Costa Mattos and Sampaio (1995) have used the gradient of a scalar cohesion variable, representing damage, to formulate a thermodynamic model.

2.4.2.3 Stochastic effects in damage accumulation

Krajcinovic (2000) has postulated that damage can be driven by the interaction of stress concentrations with randomly distributed regions of poor cohesive strength in the material. He further hypothesised that the type of distribution can influence the damage process—materials with a wide bandwidth of barrier strengths exhibit damage tolerance while materials with narrow bandwidths of barrier strengths are liable to unstable damage growth in the presence of long range stress. For example, in a parallel bar model Krajcinovic (1996) showed that bundles of nearly identical rupture strengths failed in a characteristically brittle manner (Fig. 2.14). This

was because most of the bars reach their failure value almost simultaneously and so could not sustain much more deformation after the peak force was reached. On the other hand, bundles with a wide bandwidth of rupture strengths accumulated damage much earlier and over a much wider region, causing a ductile-like response. This behaviour was defined as ‘damage-tolerant’ as specimens sustained larger and more stable deformation beyond the point of peak force (Fig. 2.14). Noting that the statistics of the microstructure is almost never considered, Krajcinovic identified the inclusion of the statistical nature of damage as an area still in need of significant research effort.

Methods to include stochastic effects have used mainly probability functions and Monte Carlo simulations. Probability density functions have been used to describe the distributions of rupture strengths (Krajcinovic, 1982) and probability of failure at a given strain (Breysse, 1990) in parallel bar/spring models for elastic brittle and elasto-plastic brittle failure. A similar procedure has been applied to the rupture of fibers in composites using random number generators to assign the distribution (Diao et al., 1997, Xia and Curtin, 2001). Zavattieri and Espinosa (2001) used Veronoi grids to generate finite element models of different grain microstructures. They also used a Weibull distribution for both interfacial failure strengths and fracture toughness of cohesive interface elements between the grains. The model was applied to brittle fracture during plate impact. Laz and Hillberry (1998) used Monte Carlo simulations to generate random initiating flaw sizes and predicted failure lives using a deterministic fatigue crack growth model. Lassen and Sørensen (2002) have used both Monte Carlo simulations and Markov chains to simulate stochastic fatigue fracture in welded joints. Muc and Kędziora (2001) have used a fuzzy set analysis in combination with finite element analysis to analyse the variability in energy release rates due to geometrical and mechanical parameters for cross-ply laminates.

2.4.3 Coupling of elastic properties with damage

For damage accumulation involving the nucleation and propagation of cracks and voids, the change in response can often be observed as nonlinear load-deflection

behaviour attributable to softening of the material as the effective load carrying capacity reduces. Since practical analysis of engineering components most often necessitates a relationship between loads and deformations, the constitutive modelling of stress-strain behaviour of damaging materials has been a subject of much interest. Two approaches dominate the literature:

- Micromechanical models—direct consideration of microcracks followed by integration (homogenisation) over the representative volume element
- Effective continuum models—consideration of the effective (fictitious) loads and/or deformations required to achieve an equivalent state in an undamaged body

2.4.3.1 *Micromechanical models*

Micromechanical models decompose the potential energy of a body, Π , often expressed as complementary energy, into energy of the elastic matrix, Π^0 , and the change in energy attributed to the introduction of microcracks, $\Delta\Pi^{cr}$, where the superscript cr denotes cracking, see Budiansky and O'Connell (1976), Kachanov (1992), Lawn and Marshall (1998). This can be expressed as

$$\Pi = \Pi^0 + \Delta\Pi^{cr} = \frac{1}{2}\boldsymbol{\sigma} : \mathbf{C}^0 : \boldsymbol{\sigma} + \Delta\Pi^{cr}, \quad (2.17)$$

where $\boldsymbol{\sigma}$ represents a uniform (average) stress acting on the RVE, \mathbf{C}^0 is the compliance tensor for the elastic matrix without cracks, and ':' represents contraction of tensors over two indices.

In addition to decomposition of energy, average strain, $\boldsymbol{\epsilon}$, can also be split into matrix and crack strains (e.g. Horii and Nemat-Nasser (1983)):

$$\begin{aligned} \boldsymbol{\epsilon} &= \boldsymbol{\epsilon}^0 + \boldsymbol{\epsilon}^{cr}, \\ \text{where } \boldsymbol{\epsilon}^{cr} &= \frac{1}{V^{cr}} \int_{S^{cr}} \frac{1}{2} (\vec{u} \otimes \vec{n} + \vec{n} \otimes \vec{u}) dS, \end{aligned} \quad (2.18)$$

and V^{cr} is the total volume of cracks, S^{cr} is the combined surface of all cracks, \vec{u} is the displacement field, \vec{n} are crack normals, and \otimes represents the tensor (dyadic)

product. Differences in particular approaches lie in their estimation of $\Delta\Pi^{cr}$, which can be represented as the following:

$$\Delta\Pi^{cr} = \frac{1}{2}\boldsymbol{\sigma} : \boldsymbol{\epsilon}^{cr} = \frac{1}{2}\boldsymbol{\sigma} : \mathbf{H} : \boldsymbol{\sigma} , \quad (2.19)$$

where \mathbf{H} represents a compliance tensor that incorporates the integrated effect of all cracks within the RVE. For example, Horii and Nemat-Nasser (1983) solved for crack compliance upon substitution of (2.18) into the strain-stress relationship for the crack:

$$\frac{1}{V^{cr}} \int_{S^{cr}} \frac{1}{2} (\vec{u} \otimes \vec{n} + \vec{n} \otimes \vec{u}) dS = \mathbf{H} : \boldsymbol{\sigma} . \quad (2.20)$$

An alternative proposal by Kachanov (1992) takes the form

$$\mathbf{H} = \frac{1}{2V} \sum S^i (\vec{n} \otimes \mathbf{b} \otimes \vec{n})^i , \quad (2.21)$$

where i implies individual cracks, and \mathbf{b} is a second order tensor relating a uniform traction vector on the crack surface to average crack opening displacement (explicitly derived from crack geometry using elliptic integrals). Substitution of (2.19) into (2.17) implies that the effective compliance of the material containing microcracks can be obtained by addition:

$$\mathbf{C} = \mathbf{C}^0 + \mathbf{H} . \quad (2.22)$$

These relationships were proposed for the case of non-interacting cracks and hence are applicable for dilute crack concentrations (Kachanov, 1992, Horii and Nemat-Nasser, 1998). Kachanov (1992) proposed that the dilute case could be used at higher crack densities, even with significant interactions, provided there was no bias towards an *amplifying* arrangement of closely spaced collinear cracks or a *shielding* arrangement of widely separated rows of stacked cracks. Crack interactions have been introduced into the effective moduli by approximation techniques such as the Self-consistent Scheme (SCS) and Differential Scheme (DS). Both of these involve the insertion of a representative crack into a matrix with the effective moduli calculated from the non-interacting case (Kachanov, 1992, Horii and Nemat-Nasser, 1998). Interactions are thus incorporated by simulating a reduced stiffness for the surrounding material.

2.4.3.2 *Effective continuum models*

Approaches based on the concept of an effective (fictitious) undamaged medium exhibiting some form of equivalent response have been introduced in section 2.4.1 but their implications for elastic properties of the material were not reviewed. Two forms that have been proposed are: (i) strain equivalence (Chaboche, 1983, Lemaitre, 1985, Simo and Ju, 1987) and (ii) strain-energy equivalence (Cordebois and Sidoroff, 1983, Voyiadjis and Kattan, 1992). Strain equivalence implies that the effective stress is the stress required to produce the equivalent strain in an identical but undamaged specimen as is observed in the damaged specimen. Mathematically this can be expressed as (Chaboche, 1983)

$$\boldsymbol{\sigma} = \tilde{\mathbf{E}} : \boldsymbol{\epsilon} \quad (2.23a)$$

$$\text{and } \tilde{\boldsymbol{\sigma}} = \mathbf{E} : \boldsymbol{\epsilon} , \quad (2.23b)$$

where \mathbf{E} is the elasticity tensor of stiffness coefficients, $\tilde{\mathbf{E}}$ is the effective elasticity of the damaged material, and $\tilde{\boldsymbol{\sigma}}$ is the effective stress. Writing (2.23a) in terms of the equivalent strain and substituting into (2.23b) leads to the concept of a damage effect tensor, \mathbf{M} :

$$\tilde{\boldsymbol{\sigma}} = \mathbf{E} : \tilde{\mathbf{E}}^{-1} : \boldsymbol{\sigma} = \mathbf{M} : \boldsymbol{\sigma} . \quad (2.24)$$

Chaboche (1983) used a homogenisation solution for systems of parallel cracks to find an expression for $\tilde{\mathbf{E}}$ and defined a damage tensor, \mathbf{D} , and the resulting damage effect tensor as

$$\mathbf{D} = \mathbf{I} - \tilde{\mathbf{E}} : \mathbf{E}^{-1} \longrightarrow \mathbf{M} = (\mathbf{I} - \mathbf{D})^{-1} , \quad (2.25)$$

where \mathbf{I} is the fourth order identity tensor. This gives

$$\tilde{\mathbf{E}} = (\mathbf{I} - \mathbf{D}) : \mathbf{E} . \quad (2.26)$$

In general $\tilde{\mathbf{E}}$ may not be symmetric so that a symmetrisation scheme is needed, see Chaboche (1999). A fourth order damage tensor thus represents the lowest order tensor that can directly operate on the stiffness tensor (Chaboche, 1999). However, many strain equivalence models have been used for isotropic or uniaxial damage and have assumed a scalar damage variable, e.g. Lemaitre (1985) and Bhattacharya

and Ellingwood (1998). Ju (1990) showed that such an assumption resulted in a constant Poisson's ratio for a damaging material, which he noted was not always the case. Through a micromechanical analysis, Ju found that the general form could be recovered using an isotropic form of \mathbf{D} instead of a scalar.

Strain energy equivalence, a concept that has been attributed to Sidoroff (Voyiadjis and Kattan, 1992), implies both an effective stress and effective strain of the undamaged medium that gives rise to an equivalent strain energy to that of the damaged medium:

$$\frac{1}{2}\boldsymbol{\sigma} : \boldsymbol{\epsilon} = \frac{1}{2}\tilde{\boldsymbol{\sigma}} : \tilde{\boldsymbol{\epsilon}}, \quad (2.27)$$

where

$$\boldsymbol{\sigma} = \tilde{\mathbf{E}} : \boldsymbol{\epsilon} \quad \text{and} \quad \tilde{\boldsymbol{\sigma}} = \mathbf{E} : \tilde{\boldsymbol{\epsilon}}, \quad (2.28)$$

which allows equivalent forms to be expressed in terms of either stiffness or compliance:

$$\tilde{\mathbf{E}} : \boldsymbol{\epsilon} : \boldsymbol{\epsilon} = \mathbf{E} : \tilde{\boldsymbol{\epsilon}} : \tilde{\boldsymbol{\epsilon}} = \mathbf{C} : \tilde{\boldsymbol{\sigma}} : \tilde{\boldsymbol{\sigma}} = \tilde{\mathbf{C}} : \boldsymbol{\sigma} : \boldsymbol{\sigma}. \quad (2.29)$$

Consideration of the equivalence in terms of stiffness gives

$$\tilde{\mathbf{E}}^{1/2} : \boldsymbol{\epsilon} = \mathbf{E}^{1/2} : \tilde{\boldsymbol{\epsilon}} \quad (2.30a)$$

$$\Rightarrow \tilde{\boldsymbol{\epsilon}} = \tilde{\mathbf{E}}^{1/2} : \mathbf{E}^{-1/2} : \boldsymbol{\epsilon}. \quad (2.30b)$$

This suggests a fourth order damage tensor in terms of effective stiffness of the form:

$$\mathbf{D} = \mathbf{I} - \tilde{\mathbf{E}}^{1/2} : \mathbf{E}^{-1/2}. \quad (2.31)$$

If a damage effect operator is assumed to transform the Cauchy stress to its effective counterpart as

$$\tilde{\boldsymbol{\sigma}} = \mathbf{M} : \boldsymbol{\sigma} \quad (2.32)$$

then suitable manipulation yields

$$\tilde{\mathbf{C}} = \mathbf{M}^T : \mathbf{C} : \mathbf{M} \quad \text{and} \quad \tilde{\mathbf{E}} = \mathbf{M}^{-1} : \mathbf{E} : \mathbf{M}^{-T}. \quad (2.33)$$

[Cordebois and Sidoroff (1983), Voyiadjis and Kattan (1992), Skrzypek (1999), Chaboche (1999)].

Unlike the strain equivalence theory developed by Chaboche (1983), Cordebois and Sidoroff (1983) did not define a fourth order damage tensor directly in terms of the stiffness change but instead chose a second order tensor and introduced damage into the constitutive relations through a symmetrised effective stress:

$$\tilde{\boldsymbol{\sigma}} = (\mathbf{1} - \mathbf{d})^{-1/2} \cdot \boldsymbol{\sigma} \cdot (\mathbf{1} - \mathbf{d})^{-1/2} , \quad (2.34)$$

where $\mathbf{1}$ is the second order identity tensor and \mathbf{d} is a symmetric *second* order damage tensor. This can be represented using a damage effect tensor as (Voyiadjis and Park, 1997)

$$M_{ijkl} = (\delta_{ik} - d_{ik})^{-1/2} (\delta_{jl} - d_{jl})^{-1/2} , \quad (2.35)$$

where δ_{ij} is the Kronecker delta. Several other symmetrisation schemes have been developed for the effective stress and their resulting representation as damage effect tensors; see Voyiadjis and Park (1997) for a review and explicit expressions. A limitation of such an approach is that orthotropic damage is the highest degree of anisotropy that can be represented. However, Kachanov (1992), in a micromechanical investigation, has shown that deviation from orthotropy can often be neglected, in particular for dilute crack concentrations.

Other equivalence principles that have been proposed are stress equivalence (Simo and Ju, 1987) and total energy equivalence (reviewed by Skrzypek, 1999). Stress equivalence implies an effective strain to produce the nominal stress measured in the damaged specimen. Total energy equivalence defines energy in terms of the work done by external tractions. This allows energy from inelastic deformation to be included in addition to the elastic energy.

2.4.3.3 *Active/passive unilateral condition*

A major difficulty in modelling behaviour of damaged materials is accounting for the change in response that occurs when cracks close due to changes in the directions of loading. Several approaches have tried to account for this in constitutive models. Krajcinovic and Fonseka (1981) included explicit description of the microcrack distribution using vectors to track crack normal strain. Murakami (1988) used decomposition of the stress tensor into tensile and compressive components in the

principal stress coordinate system. Simo and Ju (1987) used spectral decomposition of the strain tensor for tensile principal strains.

Chaboche (1992) reviewed several models of these types and found that each exhibited either discontinuities in stress response upon closure or loss of symmetry in the stiffness tensor. Subsequently, Chaboche (1993) proposed a model using a spectral decomposition of the strain and stiffness tensors. Using the principal planes of the damage tensor to define a fourth order projection tensor for each principal damage, the normal strain and stiffness for each damage plane could be decomposed; closure of cracks could thus be monitored through each decomposition of the strain and used to activate the stiffness for that plane. This model was shown to retain symmetry as well as avoiding the discontinuous stress response.

2.5 Simulation of damage accumulation in hip replacement

Several damage accumulation models have been introduced into finite element analysis of cemented hip replacement. A review of them is presented in this section. The purpose is to identify shortcomings in the work published to date and thereby focus on factors that have yet to be included in such investigations.

Verdonschot and Huiskes (1995) developed an anisotropic damage algorithm for bone cement and applied it in an axisymmetric finite element model of a prosthesis surrounded by a cement mantle. The algorithm was based on a second order damage tensor with damage growth governed by the Palmgren-Miner linear damage rule. Elastic coupling was introduced using a vendor-supplied crack option in the finite element code (MARC, MSC Software, USA). A crack closure option was also available. This algorithm was subsequently applied to a realistic bone geometry to estimate the distribution of cement damage around hip prostheses with normal bone properties and with a layer of degraded bone around the cement (Verdonschot and Huiskes, 1997b). This was done for prostheses that stay bonded to the cement and those that debond. The practical result of this study was that a debonded prosthesis was very sensitive to the properties of the surrounding tissue (Fig. 2.15).

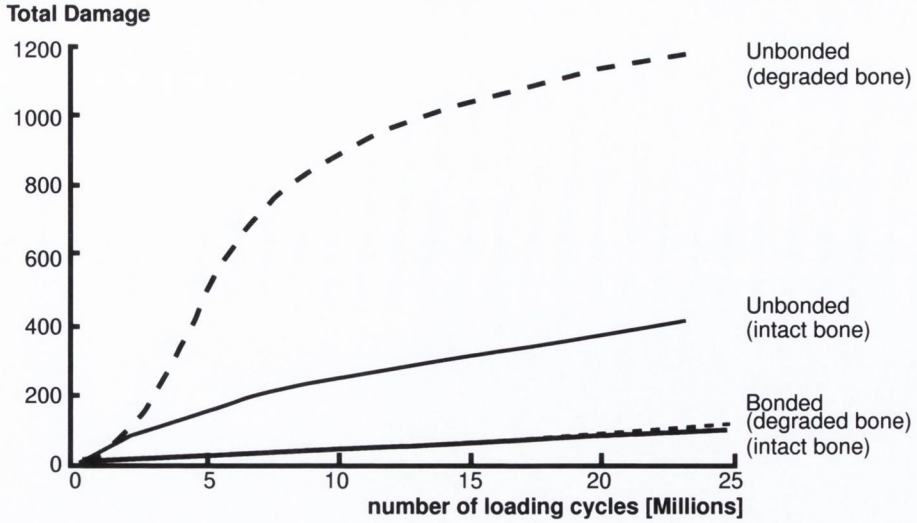


Figure 2.15. Total damage accumulated in cement around a hip prosthesis predicted by Verdonschot and Huiskes (1997b) for four conditions: bonded to cement, bonded to cement with a degraded bone layer surrounding the cement, debonded from cement, and debonded from cement with a degraded bone layer surrounding the cement. Damage was calculated as the sum of the principal values of the damage tensor for each integration point and summed over all cement integration points.

Colombi (2002ab) developed a similar algorithm and applied to a 2D finite element model of an implanted femur. In addition to the Palmgren-Miner rule, a nonlinear damage evolution equation was used:

$$D = 1 - \left((1 - D_0)^{1+m} - \frac{n \sigma^m}{c} \right)^{\frac{1}{1+m}}, \quad (2.36)$$

where D_0 is damage at the beginning of a loading block, n is the number of cycles for the block, and m and c are constants from an S-N curve of the form:

$$\log N_f = -m \log \sigma + \log c \longrightarrow N_f = c / \sigma^m. \quad (2.37)$$

Although eqn. (2.36) describes nonlinear evolution, it can be seen by replacing σ^m/c with $1/N_f$, i.e.

$$D = 1 - \left((1 - D_0)^{1+m} - \frac{n}{N_f} \right)^{\frac{1}{1+m}}, \quad (2.38)$$

that there is no stress dependence when the equation is expressed in terms of life-fractions. For a two-step test, damage accumulated from the first block, $0 \rightarrow D_1$, will be

$$D_1 = 1 - \left(1 - \frac{n_1}{N_{f1}} \right)^{\frac{1}{1+m}}. \quad (2.39)$$

The second block, $D_1 \rightarrow 1$, is then

$$1 = 1 - \left((1 - D_1)^{1+m} - \frac{n_2}{N_{f_2}} \right)^{\frac{1}{1+m}}.$$

Cancelling and raising to the power $1 + m$ enables the term inside the large brackets to be equated as

$$D_1 = 1 - \left(\frac{n_2}{N_{f_2}} \right)^{\frac{1}{1+m}}.$$

Substituting (2.39) for D_1 gives

$$1 - \left(1 - \frac{n_1}{N_{f_1}} \right)^{\frac{1}{1+m}} = 1 - \left(\frac{n_2}{N_{f_2}} \right)^{\frac{1}{1+m}}.$$

Cancelling and raising to the power $1 + m$ again shows that this results in linear accumulation, as expected:

$$1 - \frac{n_1}{N_{f_1}} = \frac{n_2}{N_{f_2}}.$$

Colombi (2002ab) introduced effective elastic properties using an elastic energy equivalence assumption. For the linear damage rule coupling was introduced only when damage reached completion for an integration point (called the *elasto-brittle* algorithm in the study). For the nonlinear model coupling was implemented for partially damaged points at every timestep in addition to the point for which the timestep was computed to cause failure (denoted the *continuous damage* algorithm). Crack closure was reported to be a feature of the coupling but details of the mechanism were not included. The *elasto-brittle* algorithm was found to predict much higher damage growth than the *continuous damage* algorithm (Fig. 2.16). Both algorithms gave unrealistically early predictions of failure of hoop support in the cement—2.7 million and 6.9 million cycles, respectively. This corresponds to approximately 1 and 2 years according to Colombi's hypothesis, taken from Sedhom and Wallbridge (1985), of 3 million cycles per year.

Stolk et al. (2003) developed a nonlinear damage growth equation in combination with a Maxwell creep model and then confirmed the model against data obtained for uniaxial specimens cycled in tension. The nonlinear damage evolution for a single level test was described by

$$D = \left(\frac{n}{N_f} \right)^{3.92}, \quad (2.40)$$

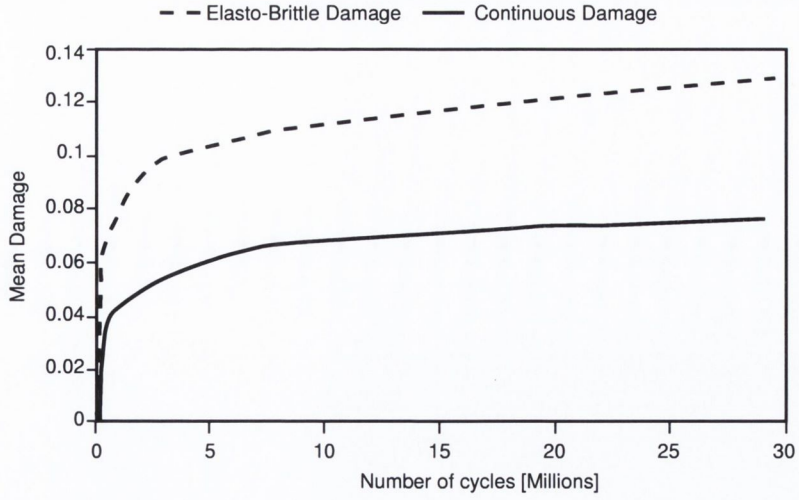


Figure 2.16. Mean damage accumulated in cement layer predicted by Colombi (2002b) for elasto-brittle and continuous damage algorithms. Principal damages were added for each integration point and summed over all integration points. This was averaged by the number of integration points and the number of dimensions.

and for variable loading history was

$$\Delta D_i = \left(\frac{n_{i-1} + \Delta n_i}{N_{f_i}} \right)^{3.92} - D_{i-1}. \quad (2.41)$$

Substituting the values for a two-step test, $D_0 = 0 \rightarrow D_1 \rightarrow D_2 = 1$, into (2.41) shows that this also represents linear accumulation; this was to be expected since there is no dependence on stress level in the damage curve (2.40). They applied the algorithm to predict damage evolution around two prosthesis designs, Lubinus SPII and Müller Curved, as an example of a pre-clinical test. The main result of the study was that their creep-damage accumulation algorithm could differentiate between different prosthesis designs to give the same survival ranking as found in the Swedish Hip Registry (Fig. 2.17).

Because of the complexity of Stolk et al's simulations (simultaneous incorporation of creep, damage and implant-cement frictional contact) a number of simplifications were made to accelerate convergence and minimise the number of timesteps.

1. Elastic coupling was only introduced when a critical rupture value was reached.

This was motivated by experimental evidence that very little change of Young's modulus occurs in bone cement during fatigue tests up to fracture. Upon attaining the rupture value, a full loss of stiffness was introduced by increasing the relevant coefficients of the compliance matrix to very large values. Some

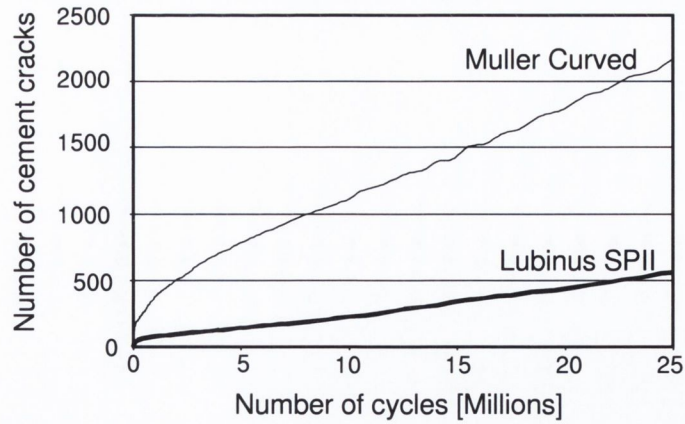


Figure 2.17. Total damage predicted by Stolk et al. (2003) in the cement layer of two different prosthesis designs, Lubinus SPII (Lub) and Müller Curved (MC); Lub = thick line and MC = thin line.

stiffness was retained to avoid convergence difficulties which meant that separate checks had to be enforced to ensure that cracked integration points were not allowed to sustain stress.

2. Crack closure was not incorporated. Based on previous studies (Verdonschot and Huiskes, 1995 1997b) they assumed that little or no crack closure would occur for constant loading conditions. However, they noted that crack closure may become important because the cement layer of a hip prosthesis is subject to a wide variety of loadings. It should also be noted that the presence of initial damage from some source other than the applied loading, e.g. shrinkage, may also result in cracks located in regions of compression.
3. No initial damage was included in their model. This also meant that cracks were unlikely to be located in regions of compressive stress upon loading.
4. The rupture criterion was reduced from a value of $D_c = 0.95$, used in earlier studies (Verdonschot and Huiskes, 1995 1997b), to $D_c = 0.75$. This avoided the need to search for rupture on the most nonlinear portion of the damage curve and thus eliminated very small timesteps. As this value represented 0.93 of the life-fraction according to their scheme, it was felt to be an acceptable compromise between computational efficiency and accuracy of modelling the damage history.

2.5.1 Differences in published models

Since each of the above models accumulate damage linearly from separate loading blocks, the main differences lie in the modelling of elastic properties of the damaged cement and the degree of nonlinearity of the evolution. Although no variability was implemented for the cement damage mechanism, some sensitivity to other factors was considered in each study. The following conclusions can be drawn from the parameter studies of previous work:

1. Verdonshot and Huiskes (1997b) concluded that a debonded prosthesis increased the damage accumulation rate in the cement, in particular when the surrounding bone support was reduced.
2. Colombi (2002a) performed a sensitivity analysis for cement modulus, interface friction coefficient between prosthesis and cement, and prosthesis modulus. Results indicated that fatigue lifetime was most sensitive to prosthesis subsidence. Any parameter variation that increased subsidence tended to decrease the lifetime, in particular the prosthesis cement friction coefficient. However, the inverse relationship did not hold, i.e. subsidence was not found to be sensitive to damage. This led Colombi to conclude that clinically observed subsidence could not result from cement damage alone.
3. Stolk et al. (2003) found that increased damage was predicted for the implant that was shown to perform inferiorly in the Swedish Hip Registry. In a parallel study they found that four different prostheses could be ranked according to revision rates found in the Swedish Hip Registry (Stolk et al., 2001).

2.5.2 Deficiencies in previous models

None of the published models capture the variability inherent in hip prosthesis performance. Therefore, the ability to predict outlying behaviour that leads to the most critical early failures has not been achieved with these methods. Also, the use of linear damage accumulation rules, even when nonlinear evolution is included, is likely to overestimate lifetime for specific load-sequences occurring as the stress

distribution changes within the damaging cement. The presence of initial damage is not included in any of the above studies and is likely to affect both lifetime estimation as well as potential crack opening/closing. Furthermore, not modelling crack closure of the damaged cement is justifiable only in the presence of constant loading conditions and in the absence of pre-load damage—neither of these assumptions hold true in reality. Realistic estimation of lifetimes will require the introduction of variable loading histories (e.g. walking, stair climbing, occasional stumbling) which is likely to invoke the unilateral condition of crack opening/closure.

2.6 Concluding remarks

Mechanical degradation of the cement has been implicated in several failure scenarios for total hip replacements. Fatigue damage has been shown to accumulate as distributed regions of microcracking around stress concentrations such as pores and inclusions. The initiation of damage has been observed prior to external loading of the cement and is most likely related to thermal shrinkage and porosity. Clinical preparations of bone cement are subject to significant variability, caused mainly by air entrapment during preparation. This variability is already high in laboratory fatigue studies and is likely to be at least similar *in vivo*.

Sophisticated models exist to predict failure of components that are prone to distributed microcracking. Important issues in constructing such models are decisions concerning the form of the damage growth rule, the relationship of damage to the elastic properties of the material, inclusion of stochastic features, and limitations caused by numerical implementations. In particular, incorporation of stochastic effects has been highlighted as one feature that may be instrumental in achieving more realistic simulations of damage accumulation in cemented hip replacement. A further issue for describing fatigue of such materials is the possibility of load-sequence effects, which may arise when separate loading periods of widely varying amplitude occur.

A number of studies have attempted to model fatigue damage accumulation in bone cement. All have used damage rules that result in linear accumulation, i.e.

no load-sequence effects. Porosity and initial damage have yet to be included in such models—their predictions are thus deterministic, and may not account for the most serious early failures seen in clinical studies. Only one of the models has been compared with experimental failure lives (Stolk et al., 2003). Quantitative comparison of spatial damage distributions has not yet been attempted. Results of these previous studies are nonetheless encouraging as they show that damage accumulation is sensitive to factors such as creep, quality of surrounding tissue, and prosthesis design. This implies that incorporation of damage accumulation in simulations may be useful in comparing the performance of different prosthesis designs during pre-clinical testing.

In this thesis, the author presents the results of several years of research directed towards inclusion of physical features into models of damage accumulation in cemented hip replacement. These include porosity, pre-load damage, and crack closure effects. Addition of these features extends the modelling capabilities of engineers to more realistic predictions of hip prosthesis failure and will also be applicable to other cemented joint replacements.

Chapter 3

METHODS

Contents

3.1	Overview	51
3.2	Theoretical development	51
3.2.1	Nonlinear damage growth	51
3.2.2	Coupling of elastic properties with damage	56
3.2.3	Damage activation/deactivation	58
3.2.4	Timestep prediction	61
3.2.5	Generation of porosity distributions	63
3.2.6	Residual stress, damage initiation, and stress relaxation	65
3.2.7	Summary of computational scheme	67
3.3	Development of the experimental model	70
3.3.1	Features of model design	70
3.3.2	Model preparation	72
3.3.3	Crack counting procedure	73
3.3.4	Loading and test set-up	74
3.4	Damage accumulation simulations	75
3.4.1	Uniaxial tension specimen	75
3.4.2	Experimental model of femoral replacement	77

3.1 Overview

The approach taken was to develop a computational scheme to include those neglected features of cement damage accumulation that are likely to have a bearing on lifetime, and to test that computational scheme against experimental results. For these tests, an experimental model was devised to create conditions similar to those encountered in the femoral side of total hip replacements.

3.2 Theoretical development

The following features were included in the scheme: (i) load-sequence effects, (ii) variable porosity distributions, (iii) existence of pre-load damage, (iv) crack closure capability, due to possibility of pre-load cracks occurring in regions of load-induced compression, and (v) residual stresses and their relaxation over time.

3.2.1 Nonlinear damage growth

Damage is assumed to grow according to a damage curve determined experimentally by Murphy (2001). The damage curve is expressed in terms of a life fraction, n/N_F , and a maximum stress dependency is introduced through an exponent, α :

$$D = \left(\frac{n}{N_F} \right)^{\alpha(\sigma)}, \quad (3.1)$$

where

$$\alpha(\sigma) = \frac{\sigma - \beta}{\gamma}, \quad (3.2)$$

$\beta = 5.6$, and $\gamma = 2.73$. It should be noted that Murphy obtained these curves from direct microscopic measurements of crack growth in flat uniaxial dog-bone specimens cycled at three stress levels (0–9.76, 0–11.11, and 0–15 MPa), see Murphy and Prendergast (2002). Damage, D , represents the total length of all cracks at a given time normalised by the length of cracks at failure. Examination of (3.2) shows that a linear curve is predicted for a stress level of 8.33 MPa while the highest stress level from the study (15 MPa) shows considerable nonlinearity with respect to the linear case (Fig. 3.1). However, below 8.33 MPa damage grows almost instantaneously to a significant level (Fig. 3.1). Furthermore, at a value of $\sigma = \beta$ the exponent is

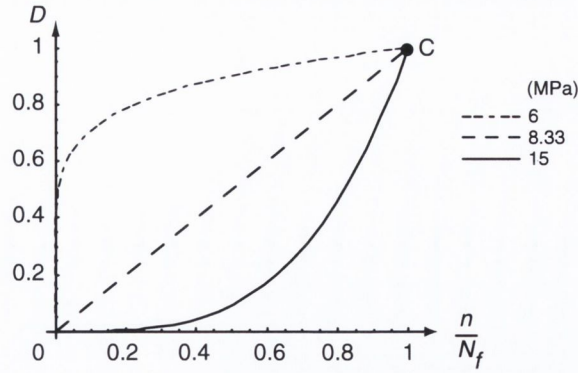


Figure 3.1. Nonlinear damage curves according to the damage curve model proposed by Murphy (2001). Note the very rapid rise in damage for very low stresses—as no data existed for this region to support such behaviour, the minimum exponent was limited to the linear case.

zero, implying instantaneous rupture. Clearly this unphysical situation could not be allowed in the computational model. Therefore, the exponent, $\alpha(\sigma)$, was not allowed to be less than one in the computational implementation of this model.

$$\alpha(\sigma) = \frac{\sigma - \beta}{\gamma} ; \quad \sigma > \beta + \gamma \text{ and} \quad (3.3a)$$

$$\alpha(\sigma) = 1 ; \quad 0 < \sigma \leq \beta + \gamma . \quad (3.3b)$$

The minimum value, $\alpha = 1$, was chosen as it corresponds to the well known Palmgren-Miner Rule, which works well for high cycle fatigue of many materials.

Nonlinear accumulation can be shown by first casting the expression in differential form and then integrating over the loading blocks of a two level test. Differentiating with respect to the number of cycles gives the damage growth rate as

$$\dot{D} = \alpha \frac{n^{\alpha-1}}{N_F^\alpha} . \quad (3.4)$$

For the first block, (3.4) can either be rewritten in integral form and integrated between the limits $0 \rightarrow D_1$ and $0 \rightarrow n_1$ to give the damage accumulated for the first level, or obtained directly from (3.1).

$$D_1 = \left(\frac{n_1}{N_{F_1}} \right)^{\alpha_1} , \quad (3.5)$$

where $\alpha_1 = \alpha(\sigma_1)$. In applying the limits for the second integration, the elapsed cycles for the first step cannot simply be chosen. If the low stress comes first, then the elapsed cycles might be higher than the failure life at the new stress. This

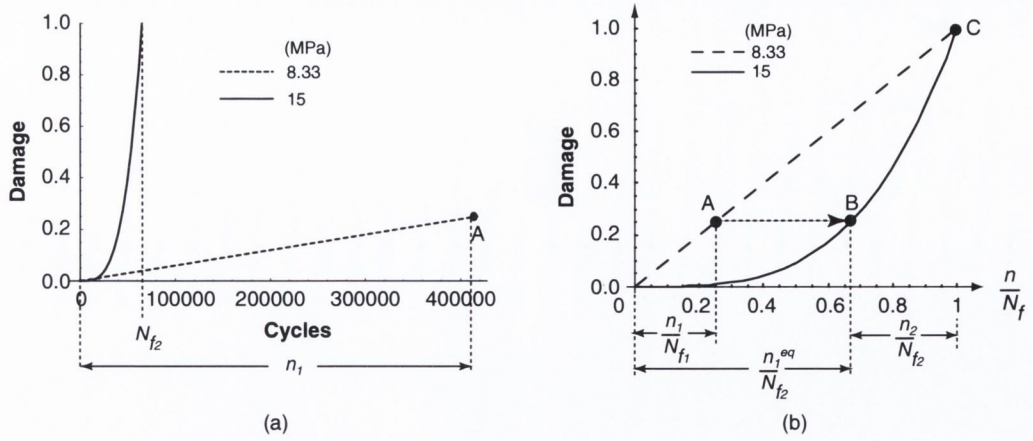


Figure 3.2. (a) Illustration of damage curves for two stress levels expressed in terms of actual elapsed cycles rather than life-fractions. Notice that use of the elapsed cycles at point A would cause instantaneous failure when transferring to the curve of the new stress level. (b) This can be overcome by moving to a point of equivalent damage, B, on the second curve and defining the equivalent elapsed number of cycles to cause this damage. Point C corresponds to failure of the specimen.

implies that damage jumps instantaneously to a value in excess of failure, see Fig. 3.2a. This unsatisfactory situation can be overcome by assuming that, in moving from one curve to another, a point of equivalent damage must be chosen at each stress level (i.e. moving from A to B in Fig. 3.2b)—it is this procedure that leads to the load sequence effect. Hence,

$$D_1 = \left(\frac{n_1}{N_{F_1}} \right)^{\alpha_1} = \left(\frac{n_1^{eq}}{N_{F_2}} \right)^{\alpha_2}, \quad (3.6)$$

where n_1^{eq} is the hypothetical number of elapsed cycles at the second stress to cause the equivalent damage on the second damage curve.

Damage accumulation for the second level must then start from this equivalent time, represented by the following integration:

$$\int_{D_1}^{D_2} dD = \int_{n_1^{eq}}^{n_1^{eq} + \Delta n} \alpha_2 \frac{n^{\alpha_2 - 1}}{N_{F_2}^{\alpha_2}} dn \quad (3.7)$$

$$\Rightarrow D_2 = \left(\frac{n_1^{eq} + \Delta n}{N_{F_2}} \right)^{\alpha_2}. \quad (3.8)$$

For cycling to failure at the second stress the accumulation,

$$1 = \frac{n_1^{eq}}{N_{F_2}} + \frac{\Delta n}{N_{F_2}}, \quad (3.9)$$

can be seen to be linear only in terms of the *equivalent* life fraction from the previous step. Using (3.6) to substitute the actual life fraction from the first step into (3.9),

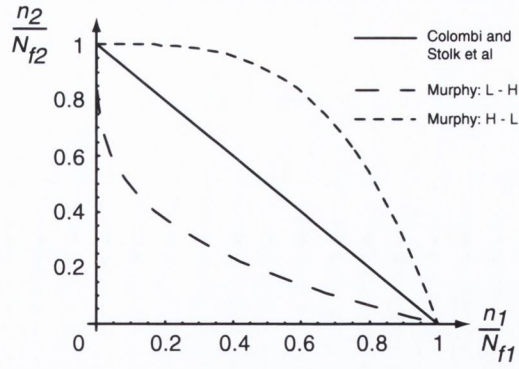


Figure 3.3. *Nonlinear accumulation for a two-level test compared with linear accumulation used by Colombi (2002b) and Stolk et al. (2003).*

the life fraction for the second stress level can be expressed as

$$\frac{\Delta n}{N_{F_2}} = 1 - \left(\frac{n_1}{N_{F_1}} \right)^{\frac{\alpha_1}{\alpha_2}}. \quad (3.10)$$

A graph of this function for alternate sequences of the stresses used to produce the damage curves of Fig. 3.1 shows that, in contrast to the case for metal fatigue, it is a low-high (L-H) stress sequence that causes the more detrimental accumulation of life fractions to less than unity (Fig. 3.3). This can also be observed by following the load sequence through 0ABC in Fig. 3.2b.

3.2.1.1 *Multiaxial damage growth*

The damage curve from the previous section only accounts for uniaxial damage. A direct extrapolation to three dimensions can be made by applying the uniaxial model to each tensile principal stress direction and assuming that damage growth is a function of both stress, and existing damage normal to that stress. This assumes that the damage tensor is rotated to a coordinate system coaxial with the principal stresses.

For a given state of damage, \mathbf{d} , the normalised eigenvectors¹, $\hat{\mathbf{e}}_i^\sigma$, of the stress tensor, $\boldsymbol{\sigma}$, were used to generate a rotation matrix of direction cosines from

$$r_{ij}^\sigma = \hat{\mathbf{e}}_i^\sigma \cdot \hat{\mathbf{e}}_j, \quad (3.11)$$

where the superscript ‘ σ ’ implies the principal stress coordinate system and $\hat{\mathbf{e}}_j$ is the j th base vector of a global coordinate system used for the finite element analysis

¹symbol ‘ $\hat{\cdot}$ ’ denotes a unit vector

(Bathe, 1995). Rotation of the damage tensor to the principal stress axes was achieved using the matrix transformation

$$[\mathbf{d}^\sigma] = [\mathbf{r}^\sigma] [\mathbf{d}] [\mathbf{r}^\sigma]^T . \quad (3.12)$$

In the general case of non-proportional loading or rotating principal stress during a cycle, a critical plane approach was used. Damage was allowed to grow in the tensor components normal to tensile principal stresses in principal stress planes defined at the moment of maximum principal stress for the cycle, i.e.

$$d_{ii}^{\sigma_{1\max}} = \left(\frac{n_i^{eq} + \Delta n}{N_{F_i}} \right)^{\alpha_i} , \quad (3.13)$$

where Δn is the number of cycles applied for the current loading block and the superscript ' $\sigma_{1\max}$ ' implies referral to a principal stress coordinate system occurring for the maximum principal stress of the cycle. Off-diagonal components were therefore assumed not to increase. For the general case of rotating principal stress directions during a loading block, as opposed to a cycle, account must be taken of the rotation of principal stresses in applying the damage growth equation. Skrzypek (1999) proposed the following form for an objective damage rate tensor:

$$\begin{aligned} \frac{\partial \mathbf{d}}{\partial n} &= \sum_{i=1}^3 \left(\dot{d}_i \hat{\mathbf{e}}_i^\sigma \otimes \hat{\mathbf{e}}_i^\sigma + d_i \dot{\hat{\mathbf{e}}}_i^\sigma \otimes \hat{\mathbf{e}}_i^\sigma + d_i \hat{\mathbf{e}}_i^\sigma \otimes \dot{\hat{\mathbf{e}}}_i^\sigma \right) \\ &= \dot{\mathbf{d}} - \mathbf{d}^T \cdot \mathbf{s} - \mathbf{s}^T \cdot \mathbf{d} , \end{aligned} \quad (3.14)$$

where $\dot{\mathbf{d}}$ is the multiaxial form of (3.4), $\hat{\mathbf{e}}_i^\sigma$ is the base vector with respect to the i th principal stress, and \mathbf{s} is a second order skew symmetric spin tensor due to rotation of principal stress directions. The total damage at the end of a cycle increment, Δn , is then

$$\mathbf{d}(n + \Delta n) = \mathbf{d}(n) + \frac{\partial \mathbf{d}}{\partial n} \Delta n . \quad (3.15)$$

Finally, the resulting damage was rotated back to the global coordinate system using the following matrix transformation:

$$[\mathbf{d}] = [\mathbf{r}^\sigma]^T [\mathbf{d}^\sigma] [\mathbf{r}^\sigma] . \quad (3.16)$$

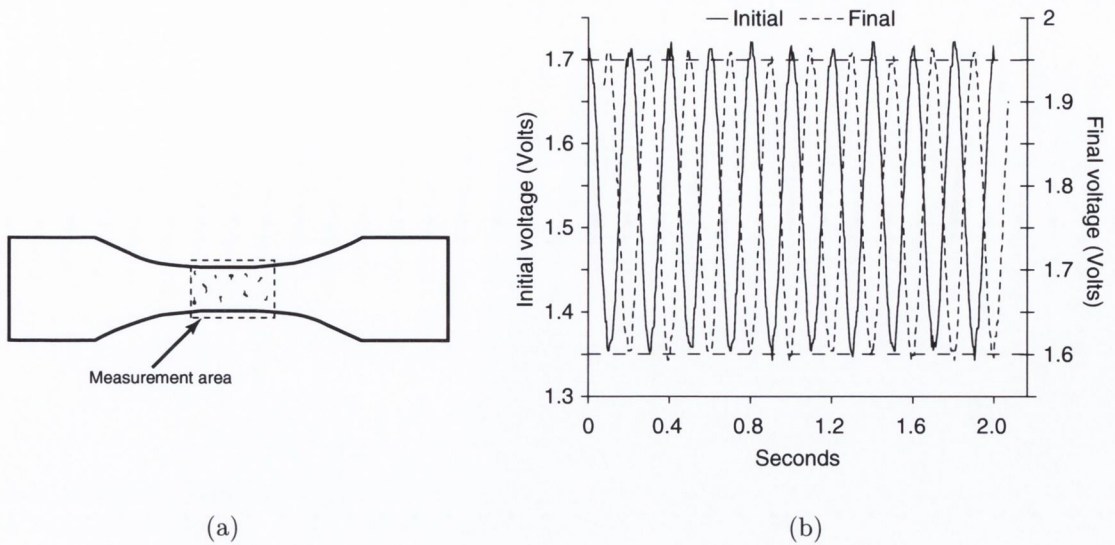


Figure 3.4. (a) Example of distributed nature of crack measurement—because only surface damage can be measured, the damage parameter could not be directly related to net section loss. (b) Signal from displacement transducer for first and last data samples of a uniaxial test. As specimen is in load control, the similar cyclic range implies little change in stiffness during testing. Change in scales between beginning and end of test is due to creep. Data provided courtesy of Dr. Bruce P. Murphy.

3.2.2 Coupling of elastic properties with damage

Many continuum damage models couple the damage variable with elastic properties of the material. The damage variable used in this study was based on a normalisation of summed crack length at a given time with respect to the summed crack length at failure. Also, the crack measurement of Murphy (2001) was performed over an exposed surface rather than a load resisting cross-section, see Fig. 3.4a, so it is not a direct representation of net section loss. Therefore, it is not strictly a crack density measurement as used in micromechanical models. Furthermore, cyclic displacement data for a given specimen exhibited little change between the beginning of testing and just prior to rupture (Fig. 3.4b). Therefore, *continuous* coupling through strain or energy equivalence was not necessary and it was decided to couple damage with stiffness loss only on reaching the rupture state, i.e. when a given principal damage reached unity. Nonetheless, this still requires a constitutive model based on one of the procedures described in section 2.4.3.

Coupling of elastic properties with damage was introduced using an assumption of elastic energy equivalence. This had the advantage of producing a symmetric

effective stiffness tensor with a straightforward transformation from a second order damage tensor. The 4th order damage effect tensor, \mathbf{M} , was constructed from the 2nd order *principal* damage tensor, \mathbf{d}^d , where superscript d implies referral to a coordinate system corresponding to the principal damage axes, as the following tensor product (Voyiadjis and Park, 1997)

$$M_{ijkl} = (\delta_{ik} - d_{ik}^d)^{-1/2} (\delta_{jl} - d_{jl}^d)^{-1/2}, \quad (3.17)$$

which has the matrix form²

$$[\mathbf{M}] = \begin{pmatrix} \frac{1}{1-d_{11}^d} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{1-d_{22}^d} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{1-d_{33}^d} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{\sqrt{1-d_{11}^d}\sqrt{1-d_{22}^d}} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{\sqrt{1-d_{22}^d}\sqrt{1-d_{33}^d}} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{\sqrt{1-d_{33}^d}\sqrt{1-d_{11}^d}} \end{pmatrix}. \quad (3.18)$$

Because the principal damage tensor is used to generate this matrix, there are no non-zero off-diagonal components. Recalling (2.33), stiffness of the damaged material in the principal damage coordinate system was expressed as

$$\tilde{\mathbf{E}}^d = \mathbf{M}^{-1} : \mathbf{E}^d : \mathbf{M}^{-T}. \quad (3.19)$$

Assuming the undamaged material is isotropic, the matrix form of (3.19) can be expressed as

$$[\tilde{\mathbf{E}}^d] = \begin{pmatrix} Q_1 & 0 \\ 0 & Q_2 \end{pmatrix}, \quad (3.20)$$

where

$$[Q_1] = \begin{pmatrix} (1-d_{11}^d)^2 E_{11} & (1-d_{11}^d)(1-d_{22}^d) E_{12} & (1-d_{11}^d)(1-d_{33}^d) E_{12} \\ (1-d_{11}^d)(1-d_{22}^d) E_{12} & (1-d_{22}^d)^2 E_{11} & (1-d_{22}^d)(1-d_{33}^d) E_{12} \\ (1-d_{11}^d)(1-d_{33}^d) E_{12} & (1-d_{22}^d)(1-d_{33}^d) E_{12} & (1-d_{33}^d)^2 E_{11} \end{pmatrix},$$

$$[Q_2] = \begin{pmatrix} (1-d_{11}^d)(1-d_{22}^d) E_{44} & 0 & 0 \\ 0 & (1-d_{22}^d)(1-d_{33}^d) E_{44} & 0 \\ 0 & 0 & (1-d_{33}^d)(1-d_{11}^d) E_{44} \end{pmatrix},$$

²A column vector storage of a symmetric second order tensor with indices transforming according to 11 → 1; 22 → 2; 33 → 3; 12, 21 → 4; 23, 32 → 5; 13, 31 → 6 was used in this study.

$$E_{11} = \frac{E(1 - \nu)}{(1 + \nu)(1 - 2\nu)}, \quad E_{12} = \frac{E\nu}{(1 + \nu)(1 - 2\nu)}, \quad E_{44} = \frac{E}{2(1 + \nu)},$$

E = Young's modulus, and ν = Poisson's ratio.

Having introduced the stiffness loss due to a crack in the principal damage plane, it was then necessary to rotate the stiffness back to the global coordinate system in which the finite element calculations were carried out. A rotation operator can be constructed from the eigenvectors of the damage tensor, $\hat{\mathbf{e}}_i^d$, as

$$r_{ij} = \hat{\mathbf{e}}_i^d \cdot \hat{\mathbf{e}}_j. \quad (3.21)$$

A larger rotation matrix can then be constructed to transform the stiffness matrix back to the global coordinate system, see e.g. Cook et al. (1989)

$$[\tilde{\mathbf{E}}] = [\mathbf{R}]^T [\tilde{\mathbf{E}}^d] [\mathbf{R}], \quad (3.22)$$

where

$$[\mathbf{R}] = \begin{pmatrix} \mathcal{R}_{11} & \mathcal{R}_{12} \\ \mathcal{R}_{21} & \mathcal{R}_{22} \end{pmatrix}, \quad (3.23)$$

$$[\mathcal{R}_{11}] = \begin{pmatrix} r_{11}^2 & r_{12}^2 & r_{13}^2 \\ r_{21}^2 & r_{22}^2 & r_{23}^2 \\ r_{31}^2 & r_{32}^2 & r_{33}^2 \end{pmatrix}, \quad [\mathcal{R}_{12}] = \begin{pmatrix} r_{11}r_{12} & r_{12}r_{13} & r_{11}r_{13} \\ r_{21}r_{22} & r_{22}r_{23} & r_{21}r_{23} \\ r_{31}r_{32} & r_{32}r_{33} & r_{31}r_{33} \end{pmatrix},$$

$$[\mathcal{R}_{21}] = \begin{pmatrix} 2r_{11}r_{21} & 2r_{12}r_{22} & 2r_{13}r_{23} \\ 2r_{21}r_{31} & 2r_{22}r_{32} & 2r_{23}r_{33} \\ 2r_{11}r_{31} & 2r_{12}r_{32} & 2r_{13}r_{33} \end{pmatrix}, \quad \text{and}$$

$$[\mathcal{R}_{22}] = \begin{pmatrix} r_{12}r_{21} + r_{11}r_{22} & r_{13}r_{22} + r_{12}r_{23} & r_{13}r_{21} + r_{11}r_{23} \\ r_{22}r_{31} + r_{21}r_{32} & r_{23}r_{32} + r_{22}r_{33} & r_{23}r_{31} + r_{21}r_{33} \\ r_{12}r_{31} + r_{11}r_{32} & r_{13}r_{32} + r_{12}r_{33} & r_{13}r_{31} + r_{11}r_{33} \end{pmatrix}.$$

3.2.3 Damage activation/deactivation

To allow restoration of stiffness normal to a damage plane on crack closure, the spectral decomposition of strain and stiffness proposed by Chaboche (1993) was

implemented. First, for a given principal damage vector, $\hat{\mathbf{e}}_i^d$, a 4th order projection tensor can be constructed as

$$\mathbf{P}_i = \hat{\mathbf{e}}_i^d \otimes \hat{\mathbf{e}}_i^d \otimes \hat{\mathbf{e}}_i^d \otimes \hat{\mathbf{e}}_i^d. \quad (3.24)$$

The spectral decomposition of the strain tensor with respect to a given principal damage plane can be represented as

$$\epsilon_{d_i} = \mathbf{P}_i : \boldsymbol{\epsilon}. \quad (3.25)$$

In writing the matrix form of (3.25) the strain tensor is written in the form of engineering strains, i.e. $\{\gamma\}^T = \{\epsilon_{11}, \epsilon_{22}, \epsilon_{33}, 2\epsilon_{12}, 2\epsilon_{23}, 2\epsilon_{31}\}$, which gives the following form for the projection matrix:

$$\{\gamma_i\} = [\mathbf{P}_i]\{\gamma\}, \quad (3.26)$$

where $\{\gamma_i\}$ is the projection of engineering strain for the i th principal damage direction and

$$[\mathbf{P}_i] = \begin{pmatrix} a^4 & a^2b^2 & a^2c^2 & a^3b & a^2bc & a^3c \\ a^2b^2 & b^4 & b^2c^2 & ab^3 & b^3c & ab^2c \\ a^2c^2 & b^2c^2 & c^4 & abc^2 & bc^3 & ac^3 \\ 2a^3b & 2ab^3 & 2abc^2 & 2a^2b^2 & 2ab^2c & 2a^2bc \\ 2a^2bc & 2b^3c & 2bc^3 & 2ab^2c & 2b^2c^2 & 2abc^2 \\ 2a^3c & 2ab^2c & 2ac^3 & 2a^2bc & 2abc^2 & 2a^2c^2 \end{pmatrix}, \quad (3.27)$$

where $\{a, b, c\}$ are the coefficients of $\hat{\mathbf{e}}_i^d$. This needs to be evaluated for each principal direction that has ruptured.

The effect of this transformation is best understood by consideration of the simplest case of a principal damage direction that is coaxial with one of the global reference axes. For example, consider the case $\hat{\mathbf{e}}_1^d = \{1, 0, 0\}$. Evaluation of (3.27) leaves only the a^4 term and substitution into (3.26) gives

$$\{\gamma_1\}^T = \{\epsilon_{11}, 0, 0, 0, 0, 0\}. \quad (3.28)$$

Thus, it can be seen that the projection operator can be used to extract the strain normal to a crack once the crack normal is known. In general, this operation is

used for crack directions not aligned with a global axis. In this case one can use the invariance of the trace operator to determine the normal strain, i.e.

$$\epsilon_{nn}^{d_i} = Tr(\mathbf{P}_i : \boldsymbol{\epsilon}) .$$

Chaboche (1993) proposed that the unilateral condition could be modelled as

$$\tilde{\mathbf{E}} = \tilde{\mathbf{E}} + \sum_{i=1}^3 H(-Tr(\mathbf{P}_i : \boldsymbol{\epsilon})) \mathbf{P}_i : (\mathbf{E} - \tilde{\mathbf{E}}) : \mathbf{P}_i , \quad (3.29)$$

where \mathbf{E} is the original undamaged stiffness tensor referred to the global coordinate system and $H(\cdot)$ is the Heaviside function (equal to zero for positive normal strain and one for negative strain). The corresponding matrix form can be expressed as

$$[\tilde{\mathbf{E}}] = [\mathbf{E}] + \sum_{i=1}^3 H(-Tr([\mathbf{P}_i]\{\boldsymbol{\gamma}\})) [\mathbf{P}_i]^T \left([\mathbf{E}] - [\tilde{\mathbf{E}}] \right) [\mathbf{P}_i] . \quad (3.30)$$

Returning to the previous example of $\hat{\boldsymbol{\epsilon}}_1^d = \{1, 0, 0\}$ and substituting $d_{11}^d = 1$ into (3.20) gives

$$[\mathbf{E}] - [\tilde{\mathbf{E}}] = \begin{pmatrix} E_{11} & E_{12} & E_{12} & 0 & 0 & 0 \\ E_{12} & 0 & 0 & 0 & 0 & 0 \\ E_{12} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & E_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & E_{44} \end{pmatrix} . \quad (3.31)$$

The projection contained in (3.30) is thus

$$[\mathbf{P}_i]^T \left([\mathbf{E}] - [\tilde{\mathbf{E}}] \right) [\mathbf{P}_i] = \begin{pmatrix} E_{11} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} . \quad (3.32)$$

Therefore, the effect of the projection operation is to restore only the stiffness normal to the crack plane when closure occurs.

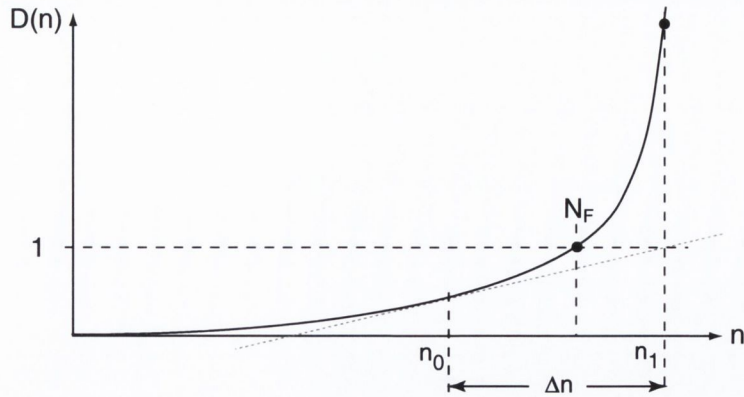


Figure 3.5. *Illustration of overprediction by Newton-Raphson scheme for curve of continuously increasing slope. Intersection of the tangent line with the desired damage, i.e. $D = 1$, provides the cycle increment. This cycle increment corresponds to $D \gg 1$. At higher stresses this overprediction could become severe enough to cause numerical instability.*

3.2.4 Timestep prediction

As elastic coupling was introduced only at rupture, the stress tensor was independent of the damage state until rupture occurred, i.e. principal stresses could not rotate during damage accumulation. This meant that a timestep to cause failure in at least one integration point could be implemented. However, one complication remained. The principal damage axes may rotate under conditions of constant loading when existing principal damages are not coaxial with principal stresses. Furthermore, the stress dependent nonlinear evolution will cause some components to accelerate relative to one another. When added to the existing damage this results in a change in the orientation of the principal damage axes. As time to failure was based on what coordinate system it was calculated in, the rotation of principal directions required that rupture had to be checked in a new coordinate system. Because of the stress dependent nonlinearity, the rate of this rotation was not constant and would be difficult to predict *a priori*. This made it necessary to use an iterative scheme to find the timestep. The choice of iteration scheme was governed by the type of nonlinearity exhibited by the model. As the slope is continuously increasing, analogous to a hardening force-displacement behaviour, a Newton-Raphson scheme can severely overpredict the timestep, see Fig. 3.5. For this reason a bisection algorithm was employed (Fig. 3.6) which offers a robust solution scheme once an interval containing the solution has been found.

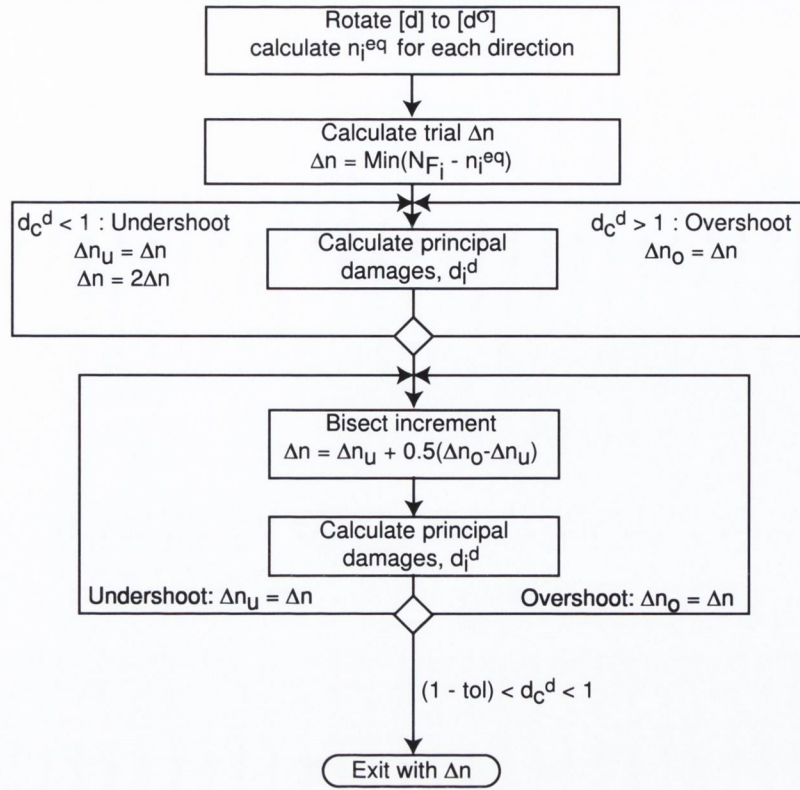


Figure 3.6. Algorithm for calculating cycle increment, Δn , to cause failure for an integration point.

Upon rotation of the existing damage to the coordinate system defined by the principal stresses, the number of cycles required to create the equivalent damage at the current stress level is calculated for each of the normal damages (i.e. diagonal components of matrix). A trial timestep is estimated from the minimum of the remaining lives for the three directions. In order to use bisection, an interval containing the root must first be found. If the trial step does not cause a damage greater than unity (undershoot) the timestep is increased until a value in excess of one (overshoot) is achieved. Steps not causing overshoot are used to narrow the interval. Once the interval is identified, it is bisected until the target principal damage falls within a specified tolerance of the desired value, causing an exit from the subroutine. This process is repeated for every integration point that belongs to a cement element. On completing a pass of all the cement integration points, the minimum timestep is chosen.

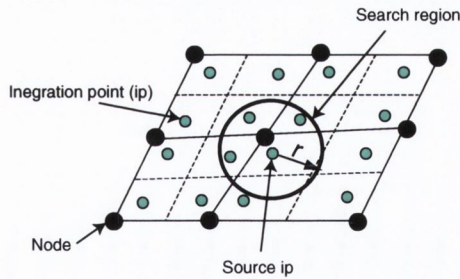


Figure 3.7. *Illustration in 2D of a pore that had larger size than the region within an element immediately surrounding an integration point. Excess pore volume was divided amongst neighbouring points that lay inside the hypothetical pore radius.*

3.2.5 Generation of porosity distributions

All aspects of the damage accumulation scheme presented above are deterministic. It was decided to incorporate porosity because most of the evidence from fatigue studies of bone cement strongly suggest that porosity is the major source of variability. Monte Carlo simulation was used to attribute porosity to the cement in a given finite element model using an algorithm based on random number generation.

Two parameters were used to control the type of porosity generated: mean porosity (% volume) and pore radii. A random number generator was used to produce a standard normal distribution for each parameter with the same number of data points as there were cement integration points. Each of the generated distributions were scaled and offset to match standard deviations and mean values specified by the calling program. The mean porosity distribution was used to define whether or not a pore existed at an integration point by specifying a tolerance about the mean value and checking whether the value for that point lay within that tolerance. For points that were assigned a pore, the pore volume was calculated by retrieving its radius. To allow pores to occupy volumes incorporating more than one integration point, pores were allowed to populate regions occupied by neighbouring integration points (this required the setting up of volumetric search buckets before this subroutine was called). When a pore was generated that was larger than the portion of an element occupied by an integration point (integration point volume was taken as the determinant of the Jacobian at that point for a linear element), the excess volume was divided amongst the neighbouring integration points (Fig. 3.7). These

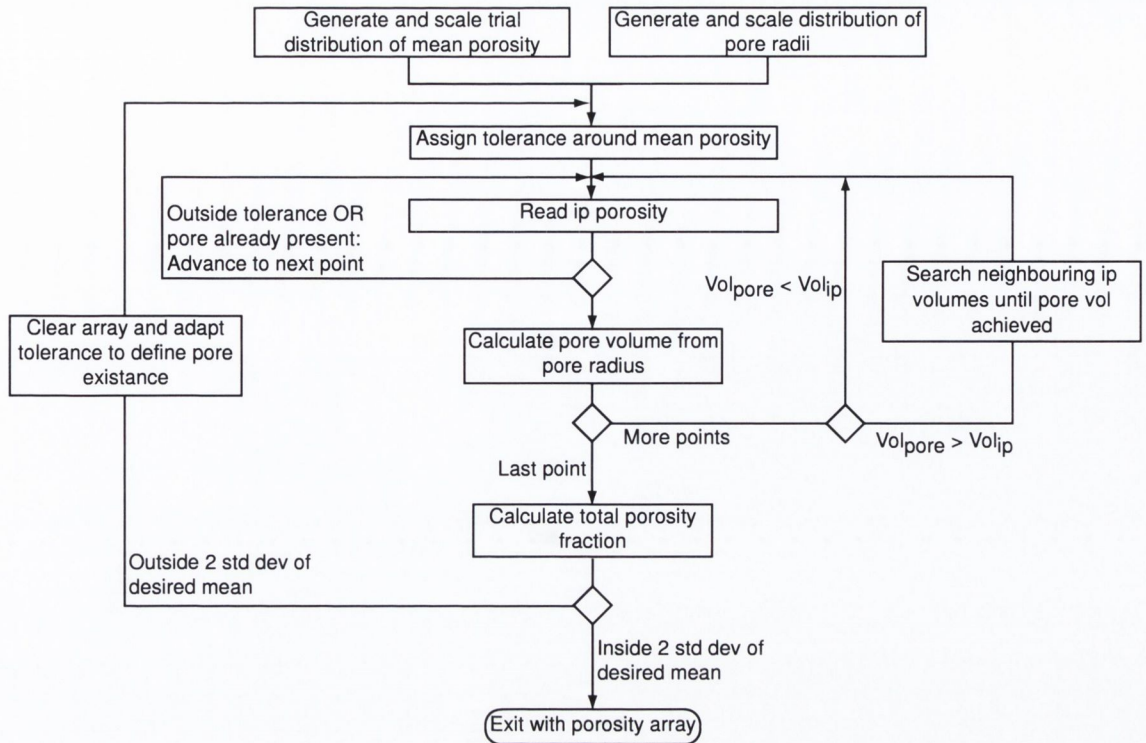


Figure 3.8. Algorithm used to assign porosity to cement integration points.

points were then excluded from further searching. On completing a pass through the cement, the total volume fraction of pores with respect to the volume of cement was calculated. If this value fell within two standard deviations of the desired mean porosity, the volume fractions for each individual point were stored and the subroutine returned control to the calling program. A mean total pore fraction outside the desired range caused the tolerance used to define whether or not a pore existed at a given point to be either widened or narrowed, depending on whether the mean fell below or above the range. This algorithm is summarised in Fig. 3.8.

In order to cause interaction between porosity and damage accumulation, it was necessary to couple porosity with elastic properties. In this way a stress raising effect could be achieved to accelerate damage, in addition to accounting for the stiffness reduction that pores would have on the loaded cement. This coupling was achieved by considering pores as isotropic damage tensors, and removing the capability of crack closure. Porosity tensors were generated for cement integration points by setting the normal (diagonal) components equal to the volumetric fraction of a pore

for each point:

$$p_{ii} = \frac{v_p}{v_{ip}}, \quad (3.33)$$

where v_p is the pore volume and v_{ip} is the integration point volume; other components were set to zero. Equation 3.20 was then used to calculate the effective stiffness matrix, replacing d_{ii}^d with p_{ii} , which was then used in generating element stiffness matrices. Upon solution of the global finite element calculations, the effective stress was calculated by first evaluating the effective strain and then substituting into equation 2.28, i.e.

$$\tilde{\epsilon} = \mathbf{M}_p^{-T} : \epsilon \quad \text{and} \quad \tilde{\sigma} = \mathbf{E} : \tilde{\epsilon}, \quad (3.34)$$

where \mathbf{M}_p implies a damage effect operator for porosity.

3.2.6 Residual stress, damage initiation, and stress relaxation

In the paper by Lennon and Prendergast (2002)³, it was found that residual stress could initiate the damage accumulation process by forming pre-load cracks in porous bone cement. Although residual stress would raise the stress within the cement in the early part of the implant lifetime (results of the aforementioned study indicated stresses around pores may reach values of up to 24 MPa), they relax over the longer term. That such relaxation occurs was found in a Moiré interferometry study (see Lennon et al., 1999)⁴. Thus, a more complete interaction of residual stress with damage accumulation should include its amplification in the presence of pores and subsequent relaxation.

Shrinkage stresses were included in the computational scheme using the methodology described in Lennon and Prendergast (2002). Briefly, a temperature dependent heat generation model for polymerising bone cement was applied in a transient thermal finite element analysis. Upon determining the peak temperature achieved for each cement element, a reference temperature, to be used in a thermoelastic cooling to ambient temperature, was defined for that element.

Resulting shrinkage stresses, including the amplifying effect due to pores, were

³See Appendix A, pp. 168

⁴See Appendix A, pp. 179

calculated according to

$$\tilde{\boldsymbol{\sigma}}^s = \mathbf{M}_p : \boldsymbol{\sigma}^s , \quad (3.35)$$

where the superscript s denotes quantities due to shrinkage. However, this could not simply be added to load induced stresses as it would allow tensile stresses to be transferred across crack faces. This was avoided by calculating the effective strain due to shrinkage stresses, $\tilde{\boldsymbol{\epsilon}}^s$, as

$$\tilde{\boldsymbol{\epsilon}}^s = \mathbf{E}^{-1} : \tilde{\boldsymbol{\sigma}}^s . \quad (3.36)$$

Calculating the residual stress at a later timestep using the damaged stiffness tensor, $\tilde{\mathbf{E}}$, would then prevent tensile stresses across cracks since, by definition, $\tilde{\mathbf{E}}$ removes the stiffness for those planes under conditions of positive strain. Note that the effective strain could not be directly computed from the finite element strains of the thermoelastic analysis. As the stiffness reducing effect of porosity was not included in the constitutive model of the cement, application of $\tilde{\boldsymbol{\epsilon}}^s = \mathbf{M}_p^{-T} : \boldsymbol{\epsilon}^s$ would simply reduce the strain predicted from the already stiffer homogeneous cement, resulting in a decreased stress in the presence of a pore.

Damage initiation was achieved by comparing the principal values of the effective shrinkage stress to the ultimate tensile strength of the cement, σ_{uts} . Thus, initial damage for each integration point was calculated as

$$d_{ii}^{\tilde{\boldsymbol{\sigma}}^s} = \frac{\tilde{\sigma}_{ii}^s}{\sigma_{uts}} . \quad (3.37)$$

3.2.6.1 Stress relaxation

Stress relaxation was included using a linear viscoelastic model for a time dependent relaxation modulus (McCrum et al., 1988). Effective stiffness at time, t , was expressed as

$$\tilde{\mathbf{E}}(t) = \tilde{\mathbf{E}}_\infty + (\tilde{\mathbf{E}}_0 - \tilde{\mathbf{E}}_\infty) \exp\left(-\frac{t}{\tau_r}\right) , \quad (3.38)$$

where $\tilde{\mathbf{E}}_\infty$ is the fully relaxed stiffness, $\tilde{\mathbf{E}}_0$ is the unrelaxed stiffness, and τ_r is a stress relaxation time constant. Stresses were assumed to relax to very small values with respect to their original values in the long term; hence $\tilde{\mathbf{E}}_\infty = 0.01\tilde{\mathbf{E}}_0$ was used. A relaxation time of 450,000 s results in relaxation of $\sim 55\%$ at 100 h and $\sim 73\%$

at 1 week, which is comparable to values found in the literature (Huiskes, 1980, Yetkinler and Litsky, 1998). However, this is for bone cement at 37.5 °C. For other temperatures, θ , the relaxation constant, $\tau_{r\theta}$, was assumed to vary according to

$$\tau_{r\theta} = a\tau_{r_0} , \quad (3.39)$$

where τ_{r_0} is the relaxation constant at a reference temperature and a is calculated from the Arrhenius equation:

$$a = \exp \frac{\Delta H}{R} \left[\frac{1}{\theta} - \frac{1}{\theta_0} \right] , \quad (3.40)$$

where ΔH is the activation enthalpy of the relaxation, R is the Universal gas constant, and θ_0 is the reference temperature for τ_{r_0} (McCrum et al., 1988).

3.2.7 Summary of computational scheme

Calculation of the stress and strain fields was performed using the finite element method. A generic storage format for unstructured grids used by the Visualization Toolkit (VTK; Kitware, Inc., USA) was chosen as the file format for the finite element mesh. A separate file was then used to store element type data, material properties, nodal restraints, applied loads, and solution/convergence controls. Simulations of damage accumulation started by reading in these two files, see Fig. 3.9. An assembly strategy was adopted, with global arrays allocated based on data from the two input files; a symmetric profile-in skyline storage scheme was used to reference entries in the global stiffness matrix. A porosity distribution was then generated as described in section 3.2.5. If inclusion of residual stress was specified in the input file, shrinkage stresses were read in for cement nodes and interpolated to the integration points of elements defined by those nodes. Initial damages were then calculated for those integration points with porosity, as described in section 3.2.6. Applied loads were then retrieved from the input file and used to generate a loading vector, f . Some simulations required frictional contact modelling so that a capability for ramp loading had to be included by splitting the applied load into increments.

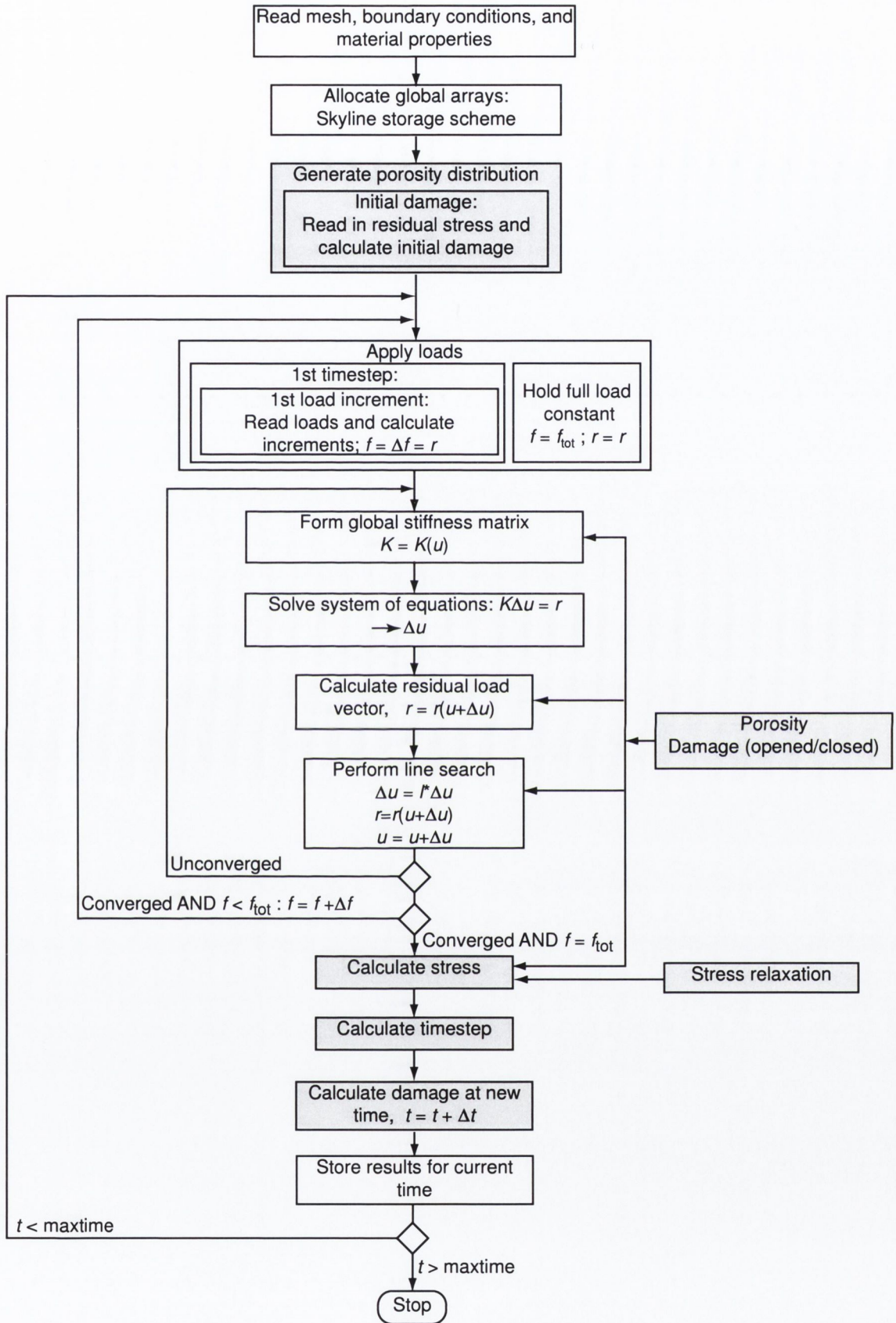


Figure 3.9. Summary of finite element algorithm for simulating damage accumulation in bone cement. Dark boxes indicate portions of the algorithm described in previous sections.

A Newton-Raphson iteration scheme was used to increment displacements until equilibrium was achieved. An out-of-balance (residual) load vector, r , was calculated as the difference between the applied load and the internal nodal restoring loads generated by the displacement increment (for the first iteration of the first timestep this amounted to the full load increment). The global stiffness matrix was then assembled as a function of the total displacement (this was due to the unilateral condition of opening and closing cracks as well as the need to include contact). A vendor provided symmetric skyline solver (Compaq Extended Math Library double precision symmetric skyline driver routine, DSSKYD) was used to calculate the displacement increment due to the residual load vector for a given iteration. A new residual was then calculated based on the prediction for the total displacement. The same subroutines used in calculating the integration point stiffness matrices when forming the global tangent stiffness were also used to calculate integration point stresses when forming the internal nodal load vectors. Discontinuities in the force-displacement response, due to crack opening/closing and contact, often resulted in oscillatory behaviour or slow convergence when only Newton-Raphson predictions were used—a line-search procedure was therefore used to augment convergence behaviour. Upon convergence, either the load was incremented (if the load was being ramped to the applied load), or the solution loop was exited (if the full load was already being applied).

Calculation of effective stresses (section 3.2.5), including relaxation of residual stresses if necessary, preceded timestep prediction according to the procedure described in section 3.2.4. Stress relaxation was applied only to the residual stress calculation as the relaxation constant was much greater than the duration of a loading cycle; this allowed any viscoelastic effect on the calculation of cyclic stress to be effectively neglected. Thus, the N-R procedure was only applied to the loading for the fatigue portion of the test as this was essentially independent from the slowly diminishing internal load due to residual stress relaxation. Damage tensors for all cement integration points were then updated for the new timestep (section 3.2.1.1). Points near failure were allowed to rupture in addition to the point for which the

timestep was predicted (a value of $D \geq 0.95$ was used as the rupture criterion); this reduced the number of timesteps required in a simulation and avoided very small timesteps. Results for the current timestep were stored in a VTK binary file format. Some additional data (total porosity fraction, volumes experiencing specified ranges of stress, total number of cracks, and the trace of integration point damage tensors summed over all cement integration points) were also stored in an ASCII format file. Time was incremented until either unstable displacement behaviour occurred, or a maximum time limit was exceeded.

3.3 Development of the experimental model

3.3.1 Features of model design

In designing the model, the main objective was to expose the cement layer so that quantitative measurement of damage accumulation could be made. At the same time, it was necessary to retain the most important geometric and loading features of a femoral hip replacement. The model presented here is a development of the one used by McCormack and Prendergast (1999) to investigate damage accumulation due to flexural loading. It consists of a proximally curved prosthesis encased between layers of cement and strips of cancellous bone that are, in turn, held between two aluminium covers (side-plates); these offer structural support in a manner similar to that provided by cortical bone (Fig. 3.10a). ‘Windows’ in the side-plates (Fig. 3.10b) expose the cement to allow direct observation of cracks. Features of the real system included were interdigitated cement-bone interfaces, achieved with the cancellous bone strips, and a trochanter-like process enabling the attachment of a muscle load through a lever. Prostheses of two surface roughnesses were manufactured: grit blasted (matt), with mean $R_a = 2.12 \mu\text{m}$, and polished, with mean $R_a = 0.04 \mu\text{m}$, see Fig. 3.11 for a photograph of each surface finish. Surface roughness was measured with a Zygo (Zygo Corp. USA) white light interferometer over 5 locations on each prosthesis within an area measuring $0.18 \text{ mm} \times 0.13 \text{ mm}$. Strength of the implant-cement interface is derived from both specific adhesion (molecular interactions) and mechanical interlock, with the latter being the main contributor

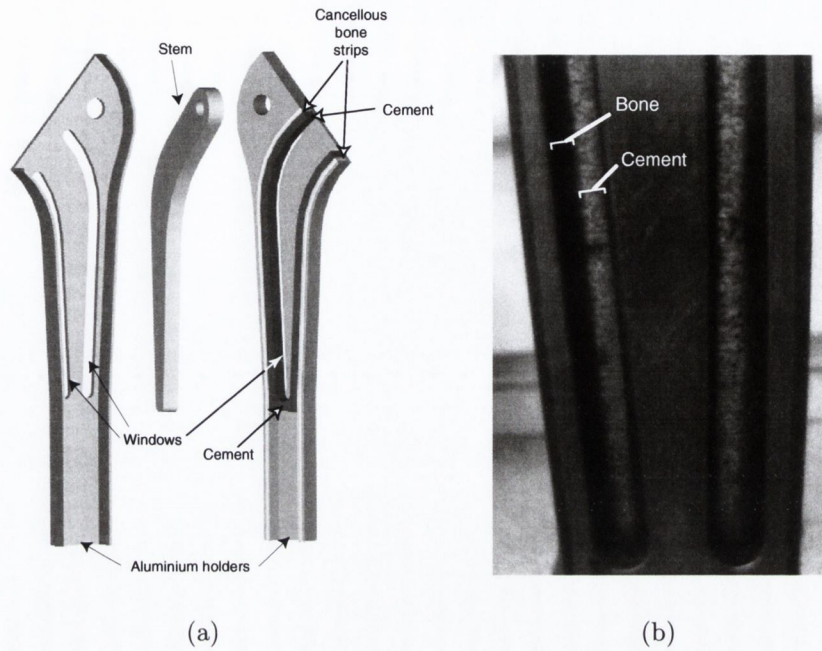


Figure 3.10. (a) Exploded view of the experimental model used for the study of damage accumulation around a femoral prosthesis under flexural loading. (b) Photograph of view through cutouts showing layered structure and silhouettes of pores.

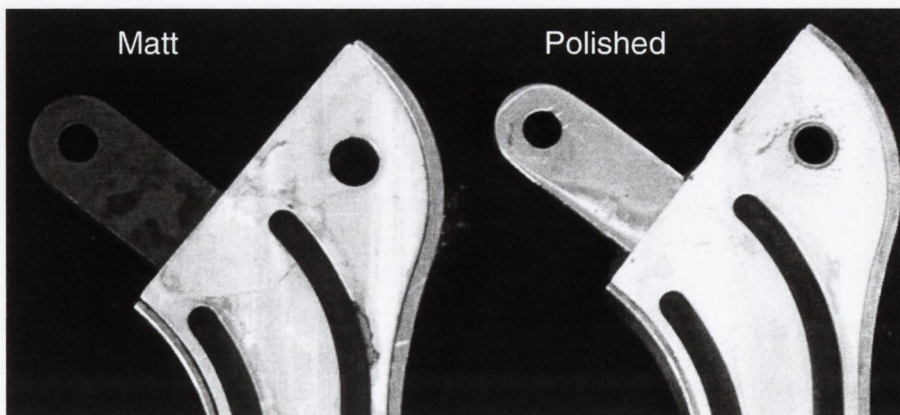


Figure 3.11. Photograph of surface finishes used in the study.

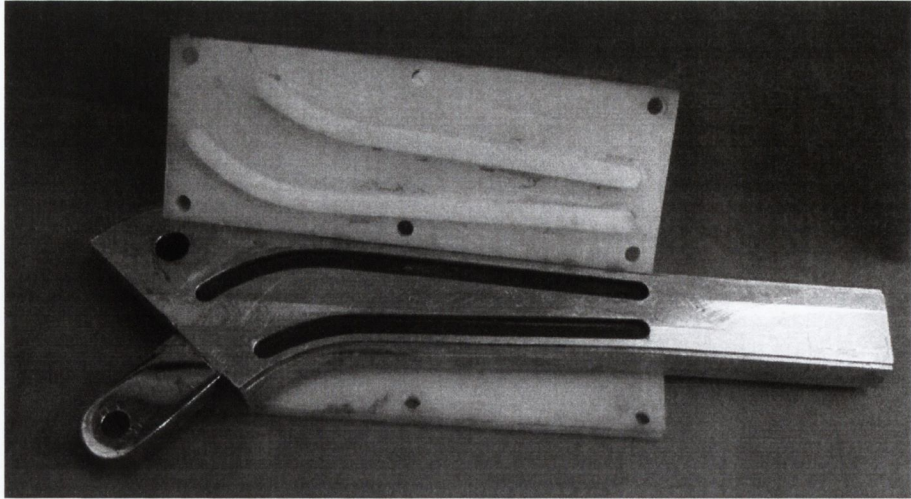


Figure 3.12. *Photograph showing polyethylene inserts used to prevent cement escape and to keep a continuous cement surface between prosthesis and cancellous bone*

to long term strength (Ahmed et al., 1984). Since mechanical interlock is a function of surface roughness, the polished prostheses were expected to debond to a much greater extent than the grit-blasted (matt) prostheses.

3.3.2 Model preparation

Cancellous bone strips were formed from bovine rib bone. These had been cut to remove their cortical exterior and ground to give a uniform cross-section of cancellous bone for the length of the inside cavity of the aluminium covers. These strips were clamped to the inside walls while still wet, to allow them to achieve conformity when dry, at which time they were glued using epoxy resin. Polyethylene inserts were placed in the windows of the inner holder to prevent cement escaping during preparation, and to keep the cement surface contiguous with the exposed stem and bone surfaces (Fig. 3.12). Simplex Rapid cement was hand mixed for approximately 60 s at 1 beat/s and then introduced into the cavity. This cement has the same composition as Simplex P bone cement without radio-opaque filler or antibiotic and is translucent so that stained cracks can be viewed by light transmission (McCormack and Prendergast, 1999). A second set of polyethylene inserts were placed in the windows of the outer holder and the specimen was allowed to cure in this state for 24 h. Specimens were stored in a fume cupboard and were not tested until at least 1 week after preparation.

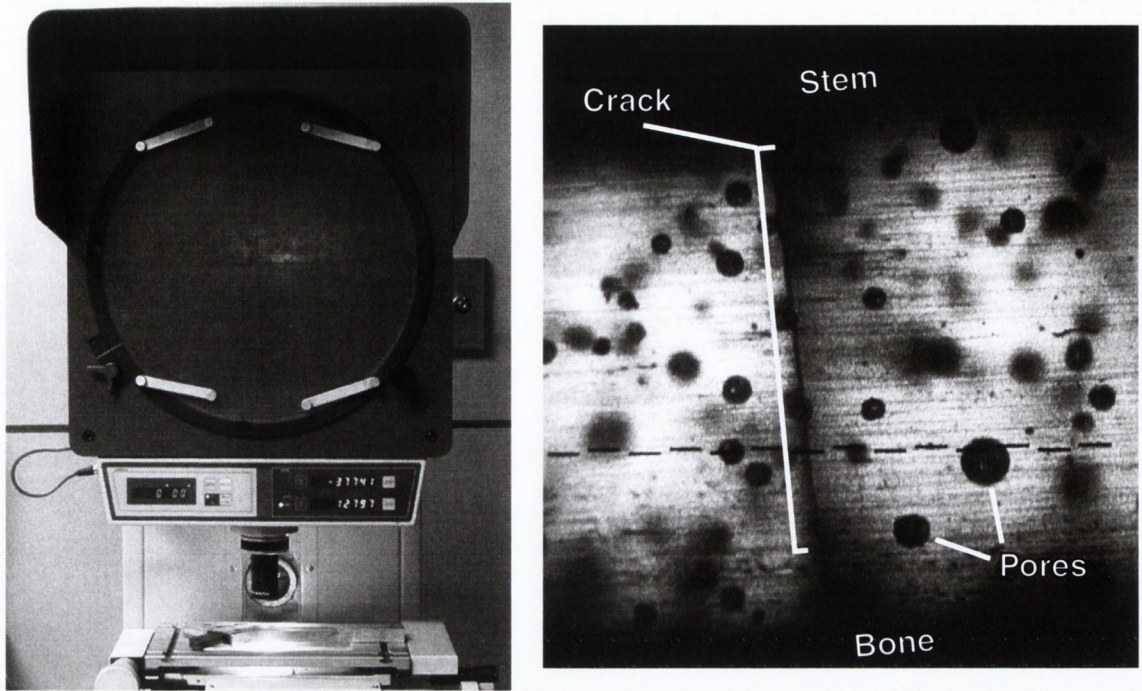
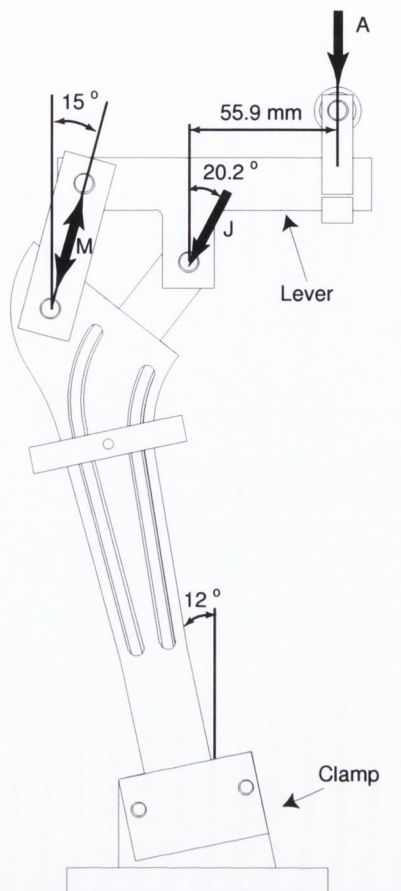


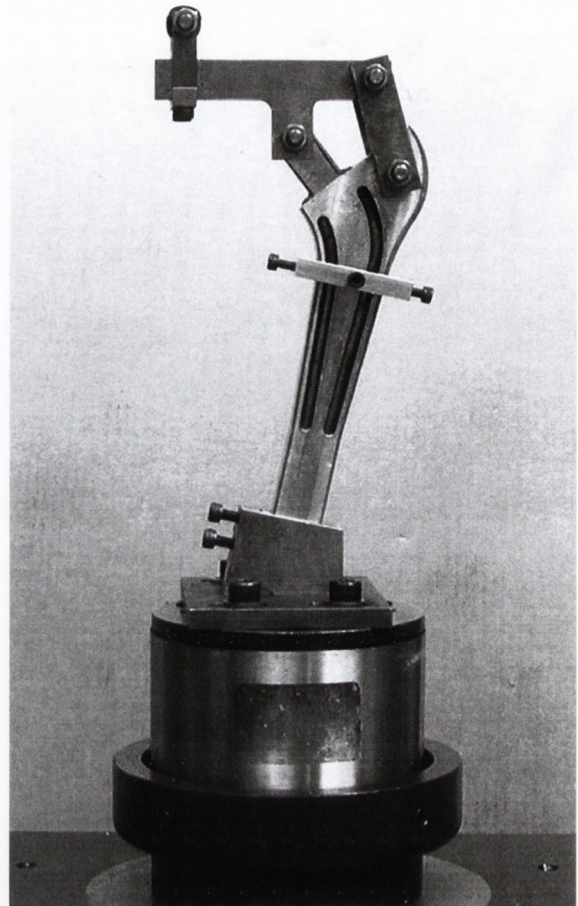
Figure 3.13. Photograph of optical comparator used for measuring cracks and a magnified view of a layer spanning crack as seen on the comparator screen.

3.3.3 Crack counting procedure

Dye penetrant was applied to the cement layers of each specimen before testing. This led to a conservative estimate of the number of cracks since only cracks which intersect the surface could be stained. An optical comparator with a $\times 20$ lens was used to project a magnified image of the cement surface onto a screen and each specimen was checked for cracks prior to testing (Fig. 3.13). Any observed cracks were traced onto acetate transparencies, which in turn had markings placed on them to allow referral to the comparator measurement system. If a crack was seen to extend below the surface, the focus was changed to assess the full projected length of the crack. All transparencies for each specimen were digitally scanned and thresholded to remove any background greyscale from the scanning operation. Image analysis was then used to calculate the position, length, and slope of each crack on an individual transparency; the results of all transparencies were referred to the comparator measurement system in order to assemble a complete spatial damage distribution for each specimen. The entire procedure was repeated for each specimen after testing.



(a)



(b)

Figure 3.14. (a) Schematic of the loading configuration used for fatigue testing of the experimental model; load A is applied via the actuator of the materials testing machine and induces a joint reaction, J , at the prosthesis head-centre, and a muscle load, M , in the plates connecting the lever to the specimen. (b) Photograph of specimen as mounted on the Instron materials testing machine.

3.3.4 Loading and test set-up

Prosthesis and muscle loading was applied simultaneously using a lever attached to the prosthesis head centre and the centre of the trochanter-like process of the aluminium holders. loosening could be observed (Fig. 3.14). The actuator of the materials testing machine was used to apply a load to an oil impregnated phosphor-bronze roller that could be adjusted to change the lever arm. Specimens were clamped in 12° adduction and a load was applied to the lever to produce a load of 2.9 kN through the prosthesis head-centre at 20.2° to the long axis of the stem and an abductor load of 1.6 kN at 15° to vertical. The joint load corresponds to 4.2 times

body weight, assuming a 70 kg body mass, and is comparable with measurements made by Bergmann et al. (1993). The abductor load (55% of joint load) is similar to that used by Burke et al. (1991) in a rig designed to simulate single leg stance. Specimens were tested at 5 Hz for 2×10^6 cycles (all specimens were tested in air at room temperature). Cyclic actuator displacements were monitored for 3 matt specimens and 5 polished specimens to assess if any

3.4 Damage accumulation simulations

Two experiments were analysed; the uniaxial fatigue tests of Murphy and Prendergast (2000a) and the tests on the experimental model described above in section 3.3. The uniaxial fatigue results were used to investigate the relationship between porosity and fatigue life. Simulations of the fatigue tests performed using the experimental model described in section 3.3 were performed to investigate damage accumulation under more realistic multiaxial stress states and interface conditions.

3.4.1 Uniaxial tension specimen

The aim was to generate S-N predictions for both hand-mixed and vacuum-mixed bone cement and to compare predicted failure lives with results of Murphy and Prendergast (2000a). As the damage accumulation model was dependent on a S-N curve (Sec. 3.2.1), this required a S-N curve for a pore-free cement. It was decided to estimate the S-N curve for the pore-free cement from the maximum values observed in the data for the less porous vacuum-mixed cement of Murphy and Prendergast (2000a). Three points that appeared to lie on a line of maximum life were chosen and a regression line was fitted to give the required S-N curve (Fig. 3.15 and Eq. 3.41).

$$\sigma = -6.04 \log_{10} N_F + 53.46 . \quad (3.41)$$

This line was not considered as a true prediction of maximum life of a pore-free cement since Murphy and Prendergast (2000a) found pores in all vacuum-mixed specimens in their study. Instead, it served as a device to investigate the hypothesis of a pore-free material and formed a common base for damage analysis of both hand-

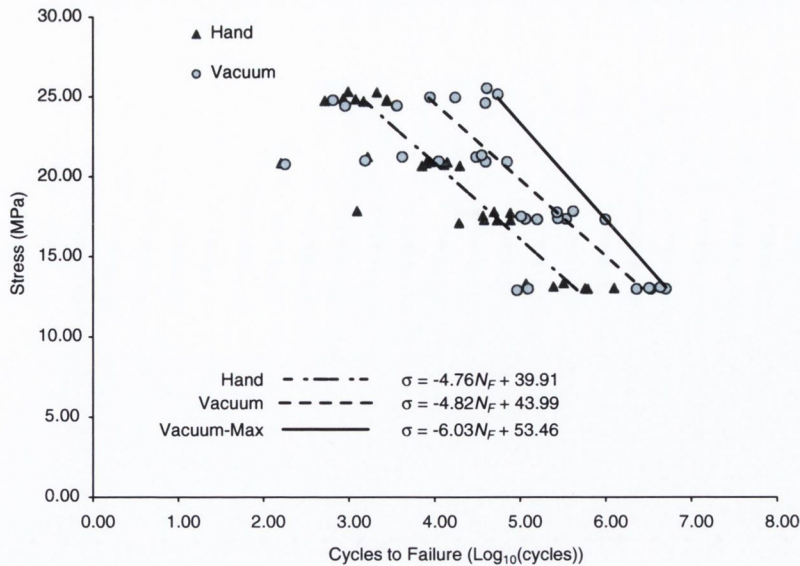


Figure 3.15. Raw data of uniaxial fatigue results of Murphy (2001) showing average regression curves fitted to hand-mixed and vacuum-mixed samples along with hypothetical pore-free curve. Only three of the stress levels (25, 17.5, and 13 MPa) that appeared to lie on a distinct line of maximum life were used to estimate the pore-free curve.

mixed and vacuum-mixed preparations. Only three points (the 25, 17.5, and 13 MPa maxima) were used as these appeared to lie on the most extreme line possible in the data, as would be expected for a material with less, if any, critical defects - i.e. the 21 MPa specimen with the highest life may have had a more critical defect compared with the highest life specimens of the other three stress levels.

Elastic properties for the pore-free cement were taken from the lower end of reported values for Perspex/Plexiglas, i.e. a Young's modulus of 2.8 GPa and a Poisson's ratio of 0.33.

A mesh of the uniaxial test specimen was generated using the Ansys preprocessor (Ansys, Inc., Canonsburg, USA) and converted to the file format required by the damage accumulation simulation (Fig. 3.16). Eight node hexahedral elements were used (total of 2,240 elements and 3,621 nodes) to represent the specimen. A uniform surface pressure was applied to the top face of the specimen to give the required nominal stress in the central section of a pore-free specimen. Nodes on the bottom face were restrained from motion in the vertical direction. Nodes defining the central axes of the restrained face were also restrained in the other two orthogonal directions to prevent rigid body motions.

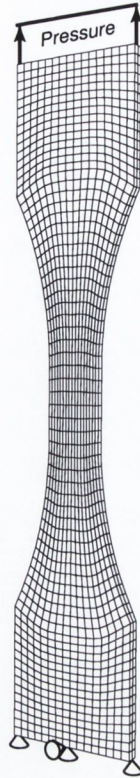


Figure 3.16. *Mesh used for study of uniaxial fatigue damage accumulation. Selected nodal restraints are included to illustrate the boundary conditions for the model.*

Eight simulations for each of four stress levels tested (25, 21, 17.5, and 13 MPa) were performed for each mixing method. Each of the 64 simulations had a different pore distribution generated by the methods described in section 3.2.5. As little data for the pore distributions obtained by Murphy and Prendergast was available, the values used to generate the pore distributions were obtained by a fitting procedure. Mean and standard deviation of total volume fraction of porosity as well as mean and standard deviation of pore radius were varied at one stress level (21 MPa). When similar maximum and minimum failure lives were predicted from a set of eight simulations the values were applied to the rest of the stress levels.

3.4.2 Experimental model of femoral replacement

The mesh used for the experimental model (Fig. 3.17) was also created from eight node hexahedra. To incorporate the potential for debonding of the prosthesis from the cement, contact elements had to be included. A surface-to-surface contact element, proposed by Beer (1985), constructed from underlying solid element faces at



Figure 3.17. *Mesh of experimental model used to study damage accumulation under conditions closer to a real femoral replacement. Selected nodal restraints are included to illustrate the boundary conditions for the model.*

the interface was implemented in the finite element code. As the mesh either side of the interface had matching node and element patterns, no contact search algorithm was included—instead contact pairs were identified during pre-processing of the mesh. A symmetrised frictional model, used in an implementation of Beer's element for modelling debonding of femoral replacements (Hefzy and Singh, 1997), was used to account for friction. Contact detection was performed at integration points and a penalty method was used to account for discontinuous force-displacement behaviour. Based on the normal gap and tangential slip, the element stiffness was calculated and assembled into the global stiffness matrix. The same elements were used to model bonded contact by applying constant normal and tangential stiffnesses. Due to the penalty implementation, convergence was affected by values of the stiffness coefficients. Acceptable results compared with contact analysis of the same mesh performed using Ansys was found for a normal stiffness of 1000 N/mm and a tangential stiffness of 10 N/mm (for fully bonded contact both coefficients were set to 1000 N/mm). A friction coefficient of 0.32 was used; this was determined from a pin-on-plate sliding test using a polished Co-Cr-Mo pin against a

Table 3.1. *Elastic properties used in damage accumulation simulations.*

Material	Young's Modulus (GPa)	Poisson's ratio
Prosthesis	200	0.30
Cement	2.8	0.33
Cancellous Bone	2	0.30
Aluminium	72	0.33

PMMA plate and was conducted in air⁵. In real joint replacements bodily fluids may provide lubrication to cause a reduction in the friction coefficient. However, the experimental model used in this study was only tested in air. Although other models exist, Coulomb friction has been widely used to simulate frictional sliding in contact between prostheses and cement (e.g Mann et al., 1995, Verdonschot and Huiskes, 1996) and has been found to give acceptable results when used to simulate push-out tests (Mann et al., 1991).

The mesh consisted of 3,404 nodes and 2,578 elements, including 468 cement elements and 312 contact elements between the prosthesis and cement, and prosthesis and aluminium. Nodes on the surfaces that were clamped in the real model were restrained from normal displacement. Applied forces were distributed among nodes at the points of contact with the shafts used to connect the lever to the specimen (Fig. 3.17). Elastic properties were assigned for a stainless steel, pore-free cement, average isotropic representation of cancellous bone, and a 7075 series aluminium (Tab. 3.1).

Previous work by Lennon and Prendergast (2002) showed that damage was initiated before the commencement of fatigue testing and was potentially caused by the interaction of porosity and thermal shrinkage of the cement. A preliminary thermoelastic analysis was therefore performed according to the previously developed method and the input file was altered to activate the relevant portions of the algorithm already described in sections 3.2.6 and 3.2.7.

Five tests were simulated for each of the interfacial conditions and simulations were halted once the elapsed time exceeded 2 million cycles. The effects of using

⁵Dr.-Ing. M. Pfeiderer, De Puy Johnson and Johnson, U.K., personal communication

Table 3.2. *Summary of damage accumulation simulations*

Model	No. Tests	Test time (cycles)
Uniaxial 25 MPa	2×8	To failure
Uniaxial 21 MPa	2×8	To failure
Uniaxial 17 MPa	2×8	To failure
Uniaxial 13 MPa	2×8	To failure
Experimental Model (Bonded—Monte Carlo)	5	2×10^6
Experimental Model (Debonded—Monte Carlo)	5	2×10^6
Experimental Model (Bonded—Deterministic)	1	2×10^6
Experimental Model (Debonded—Deterministic)	1	2×10^6

a deterministic approach were also investigated by performing one extra simulation for each of the interfacial conditions. For these simulations, the regression equation fitted to the average fatigue lives for the hand-mixed cement studied by Murphy and Prendergast (2000a) was used. To complete the deterministic approach, porosity, residual stress, and pre-load damage were removed. A summary of all simulations, including the uniaxial specimens, is presented in Table 3.2.

Chapter 4

RESULTS

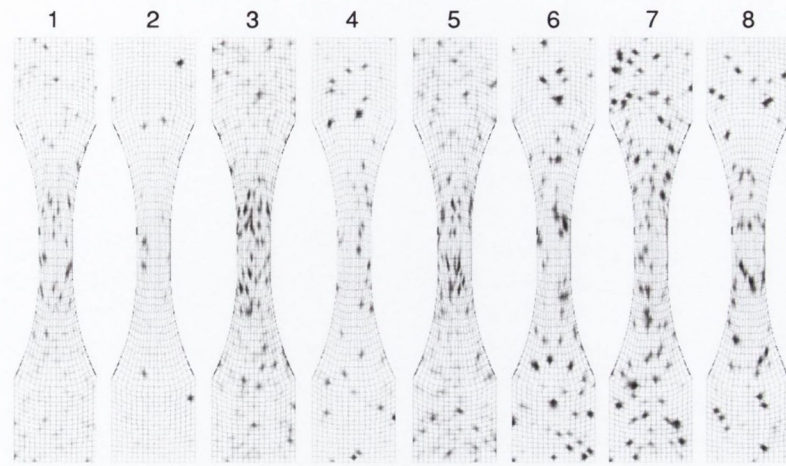
Contents

4.1 Numerical simulation of S-N curves	81
4.2 Experimental model of femoral replacement	90
4.2.1 Damage measurements	90
4.2.2 Migration measurements	99
4.3 Simulations of fatigue damage accumulation in the experimental model	102
4.3.1 Damage accumulation	102
4.3.2 Analysis of factors influencing damage accumulation	108
4.3.3 Migration	113
4.4 Recapitulation of main results	115

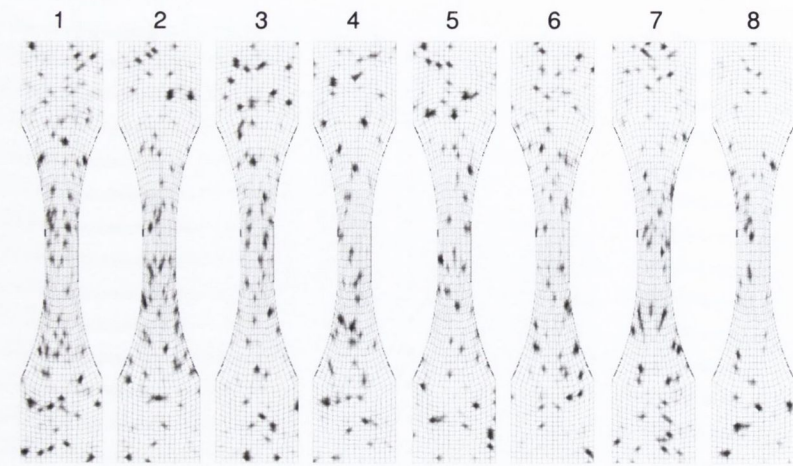
4.1 Numerical simulation of S-N curves

Different pore distributions were obtained by varying both average volume fraction of pores and average pore radius. The hand-mixed specimens were characterised by a large number of small pores whereas the vacuum-mixed specimens were characterised by a few pores of relatively large radius (Figs. 4.1 and 4.2). An example of typical porosity distributions for a hand-mixed and a vacuum-mixed specimen are shown in Fig. 4.3. The values used to achieve these distributions, and the resulting average total specimen volume fraction of pores, are shown in Tables 4.1 and 4.2.

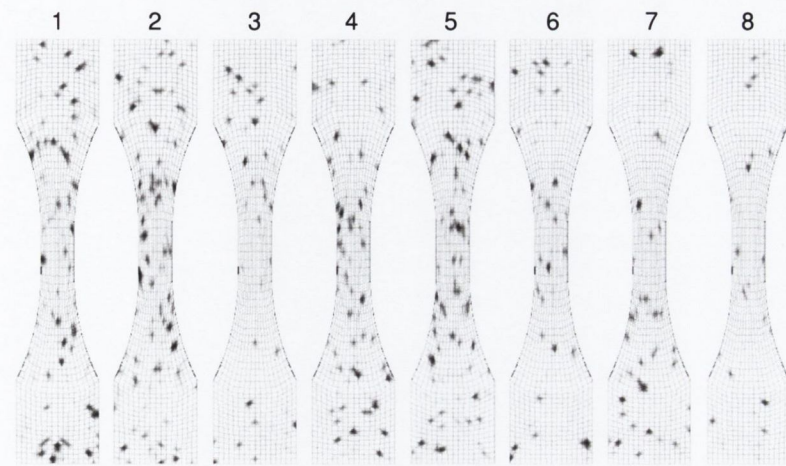
Porosity was seen to affect the initiation of damage accumulation differently for each mixing method. For hand-mixed specimens, damage always initiated from porosity, sometimes from several sites simultaneously (Fig. 4.4). However, vacuum-mixed specimens sometimes failed from the sites of nominal peak stress, predicted to be the point of tangency between the straight central section and the curved sections; these can be seen as the symmetric damage patterns in Fig. 4.5. In general, failure



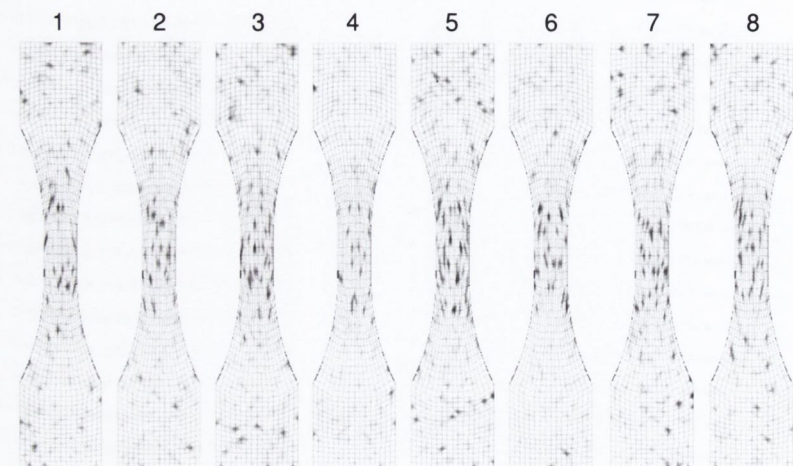
13 MPa



17.5 MPa



21 MPa



25 MPa

Figure 4.1. Simulated porosity distributions of hand-mixed specimens. Eight specimens were tested at each stress level. A greyscale from 0 to 1 is used to indicate pore volume fraction averaged at nodes.

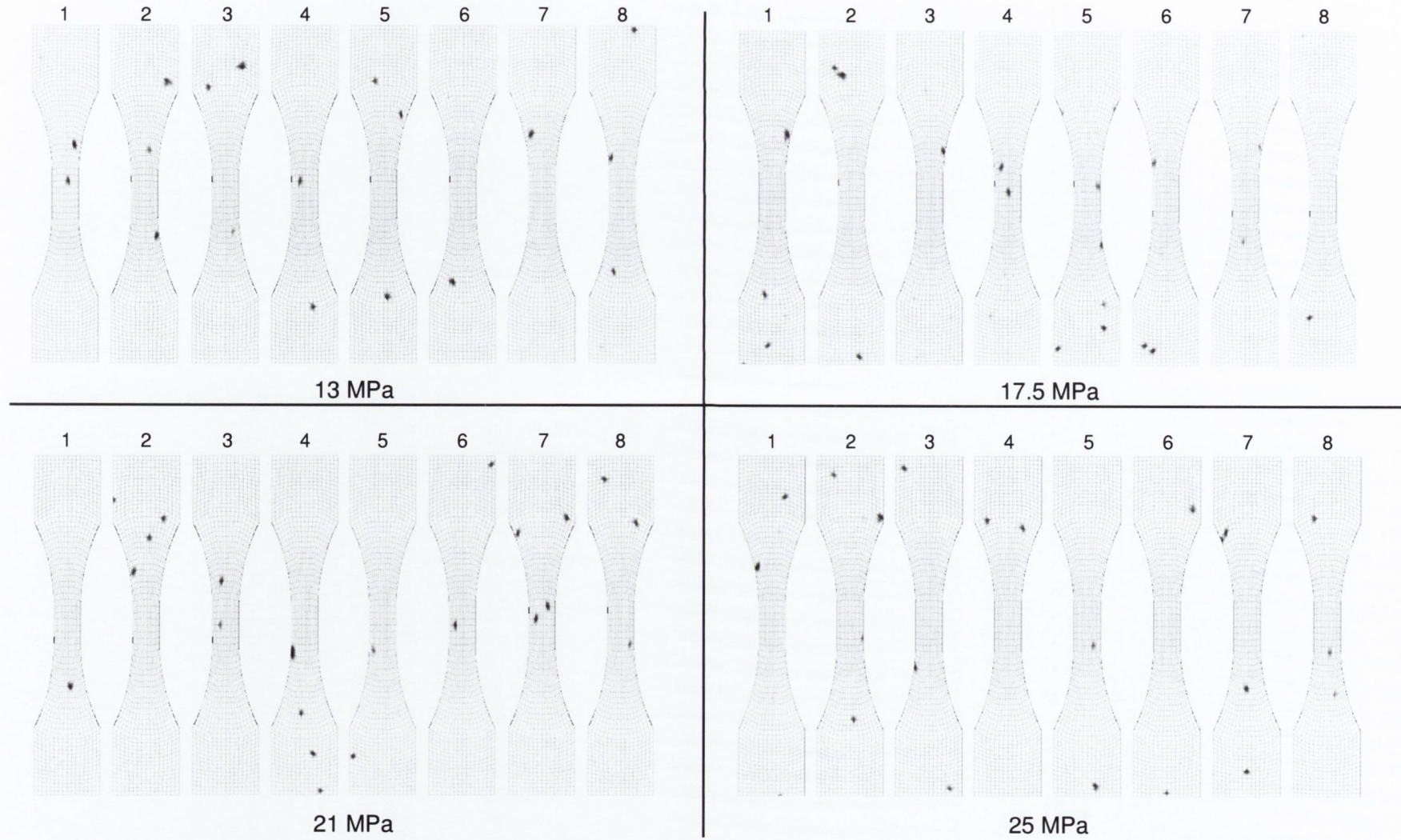


Figure 4.2. Simulated porosity distributions of vacuum-mixed specimens. Eight specimens were tested at each stress level. A greyscale from 0 to 1 is used to indicate pore volume fraction averaged at nodes.

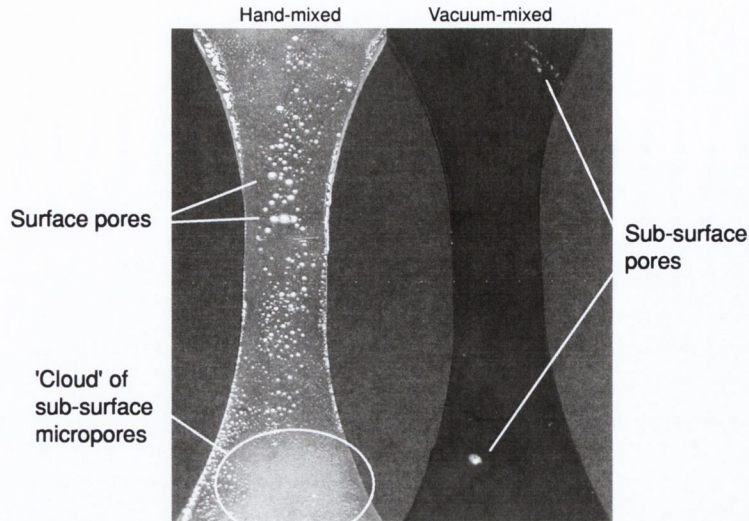


Figure 4.3. Photograph of typical porosity distribution for a hand-mixed and vacuum-mixed specimen. Translucency of the cement allows some visibility of sub-surface porosity. The hand-mixed specimen shows large amounts of both surface and sub-surface pores while the vacuum-mixed specimen contains a relatively large sub-surface pore in the lower half of the specimen and some smaller sub-surface pores in the upper half.

Table 4.1. Mean and standard deviation (S.D.) of porosity and pore radius used as input for each mixing method.

	Hand-mixed		Vacuum-mixed	
	Mean	S.D.	Mean	S.D.
% Porosity	5.00	(2.00)	0.25	(0.125)
Radius (mm)	0.20	(0.60)	1.75	(0.05)

Table 4.2. Mean and standard deviation (S.D.) of total specimen volume fraction of pores for each mixing method achieved in the simulations. No radius data is presented as only pore volume fractions were stored in the simulations.

	Hand-mixed		Vacuum-mixed	
	Mean	S.D.	Mean	S.D.
% Porosity	5.76	(2.02)	0.34	(0.10)

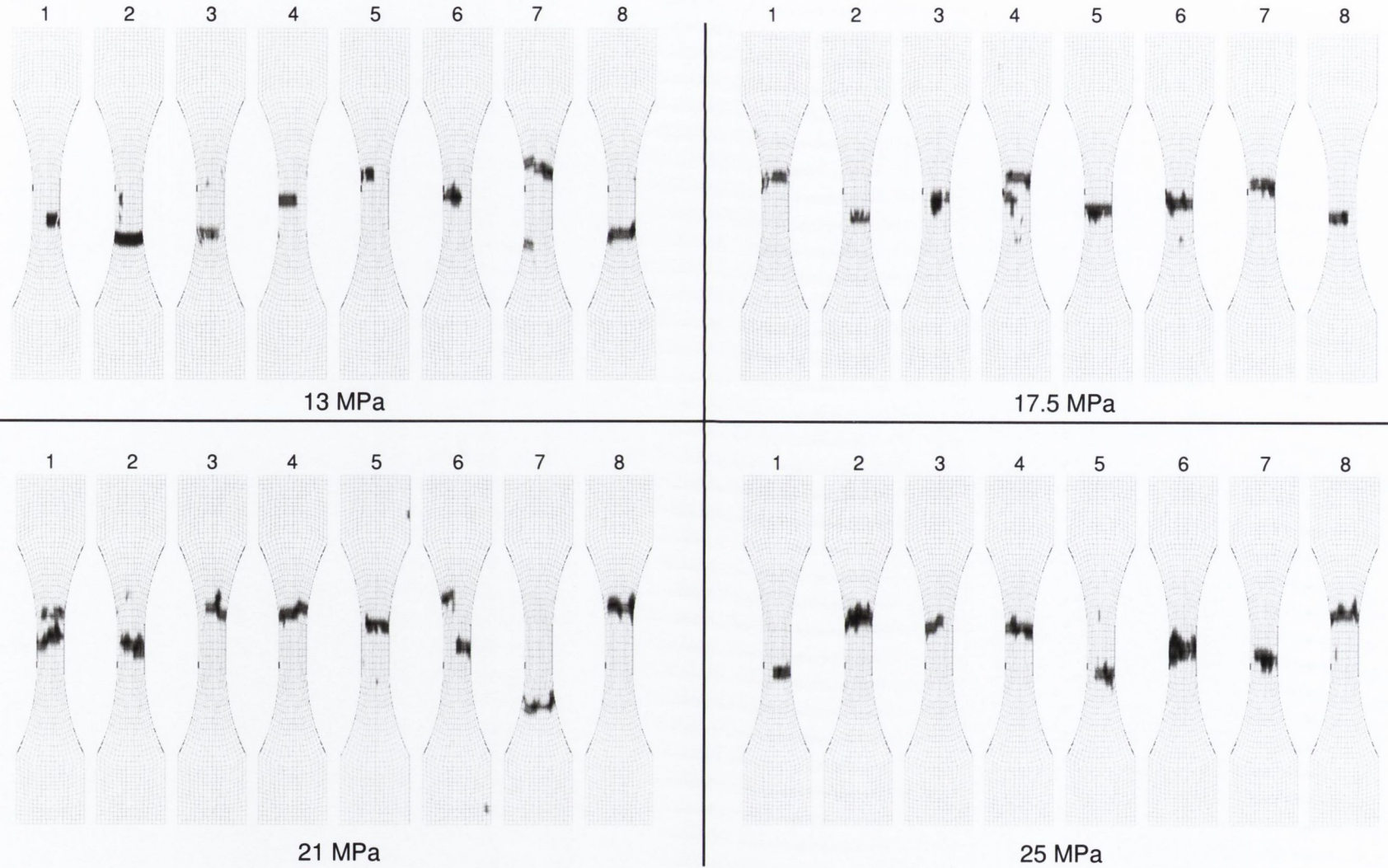
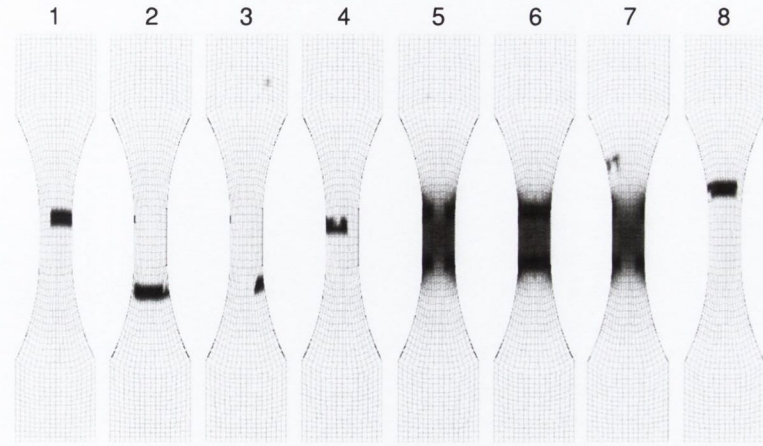
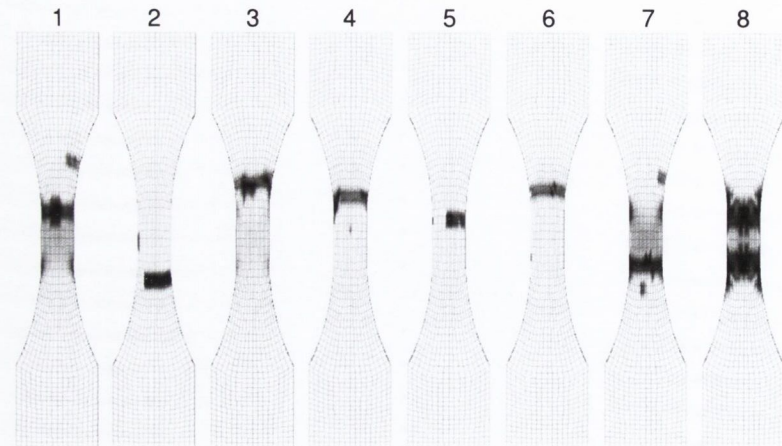


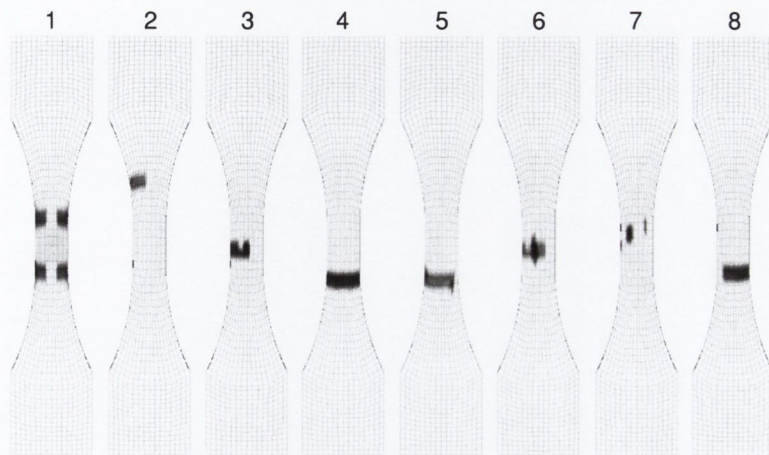
Figure 4.4. Spatial damage distributions for last stored timestep for each hand-mixed specimen. Not all plots show complete rupture because results were stored intermittently, due to storage limitations. Plots for such cases identify the crack leading to failure. A greyscale from 0 to 1 is used to indicate damage averaged at nodes.



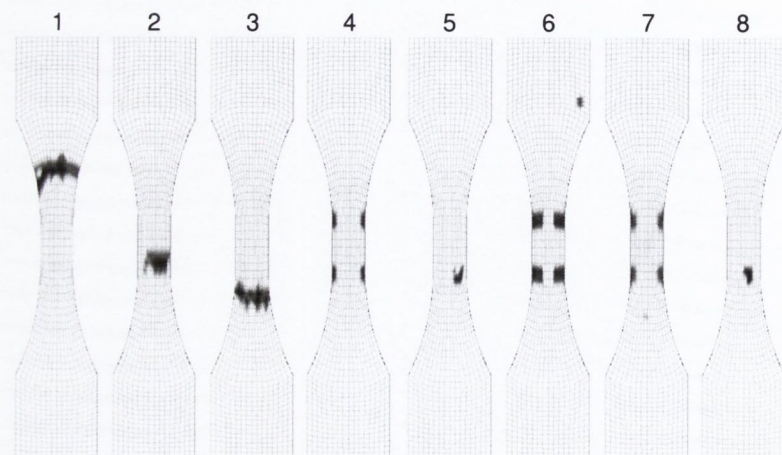
13 MPa



17.5 MPa



21 MPa



25 MPa

Figure 4.5. *Spatial damage distributions for last stored timestep for each vacuum-mixed specimen. Not all plots show complete rupture because results were stored intermittently, due to storage limitations. Plots for such cases identify the crack leading to failure. A greyscale from 0 to 1 is used to indicate damage averaged at nodes.*

occurred either within or near the gauge for both hand-mixed and vacuum-mixed specimens.

Considerable variability was observed for simulated failure life for both mixing methods (Fig. 4.6). This variability made comparison difficult as vacuum mixed specimens sometimes failed as early as some of the earliest hand-mixed failures. It should be emphasised that tuning of the model was limited to prediction of similar maximum and minimum failure life at one stress level, 21 MPa (see section 3.4.1). Given this, the similarity of the experimental and numerical results is noteworthy; to quantify this, least squares regression lines were used to compare the overall fit of the data. Comparison of the regression lines fitted through each data set show that similar S-N curves to those found experimentally by Murphy and Prendergast (2000a) were predicted for both hand-mixing and vacuum-mixing, see Fig. 4.7. The regression equations for the hand-mixed specimens are

$$\text{Experiment } \sigma = -3.76 \log_{10} N_F + 34.95 , \quad (4.1a)$$

$$\text{Simulated } \sigma = -3.58 \log_{10} N_F + 33.19 , \quad (4.1b)$$

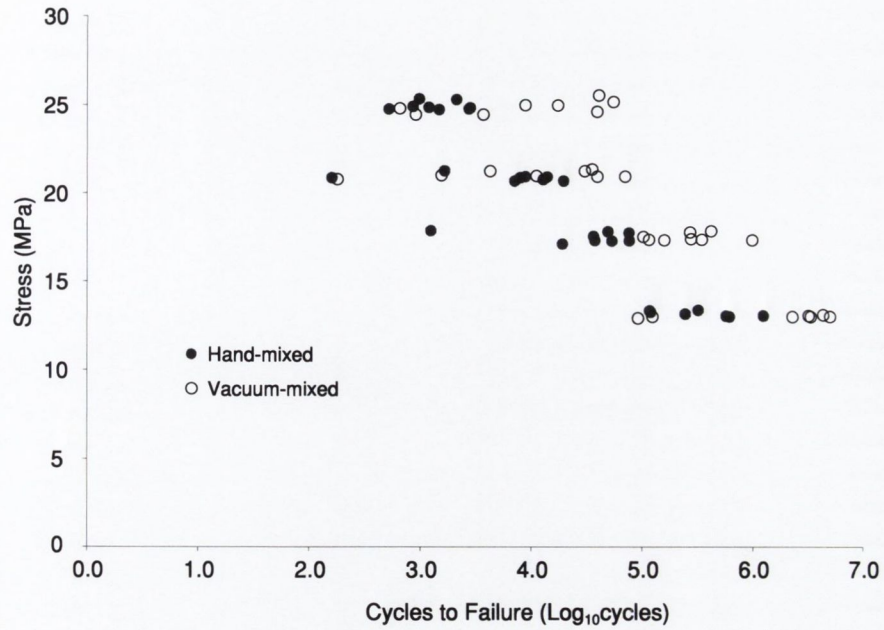
and for the vacuum-mixed specimens are

$$\text{Experiment } \sigma = -2.86 \log_{10} N_F + 33.06 , \quad (4.2a)$$

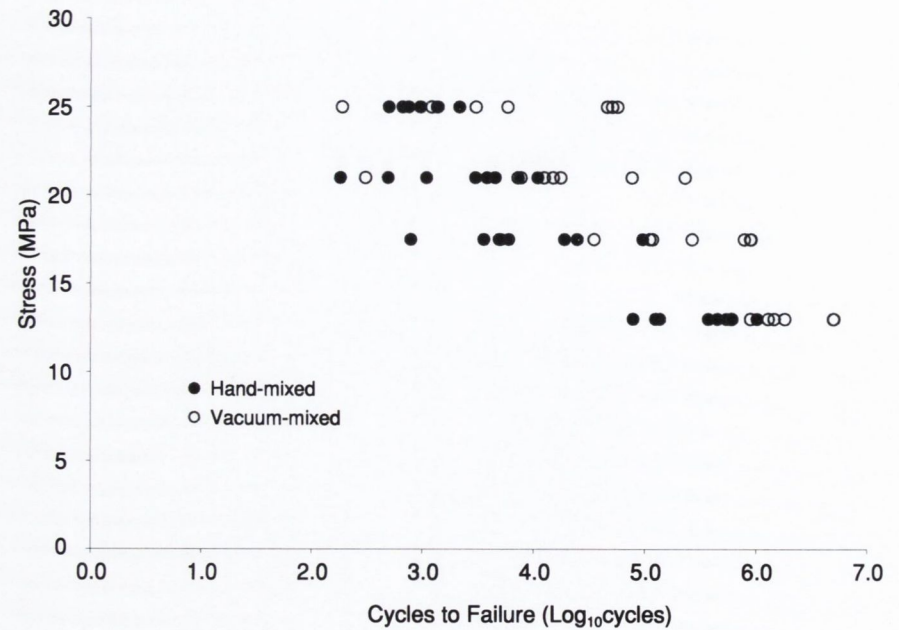
$$\text{Simulated } \sigma = -2.99 \log_{10} N_F + 33.97 , \quad (4.2b)$$

where σ is the applied stress and N_F is the number of cycles to failure.

Significance tests were used to determine if the simulated data could be statistically discriminated from the experimental data. For such a test significance values of $p > 0.05$ would indicate that the results from experiment and simulations cannot be distinguished and higher p values indicate a greater probability that the samples are similarly distributed. Direct comparison of average failure life at each stress level for the simulated data set with the corresponding experimental data set shows that the experimental and simulated results form part of the same distribution, although some stress levels do not show as good agreement as others (Table 4.3). Therefore, the distributions were not statistically different from each other and the simulations provide predictions with similar variability as the experiments. Furthermore, equiv-

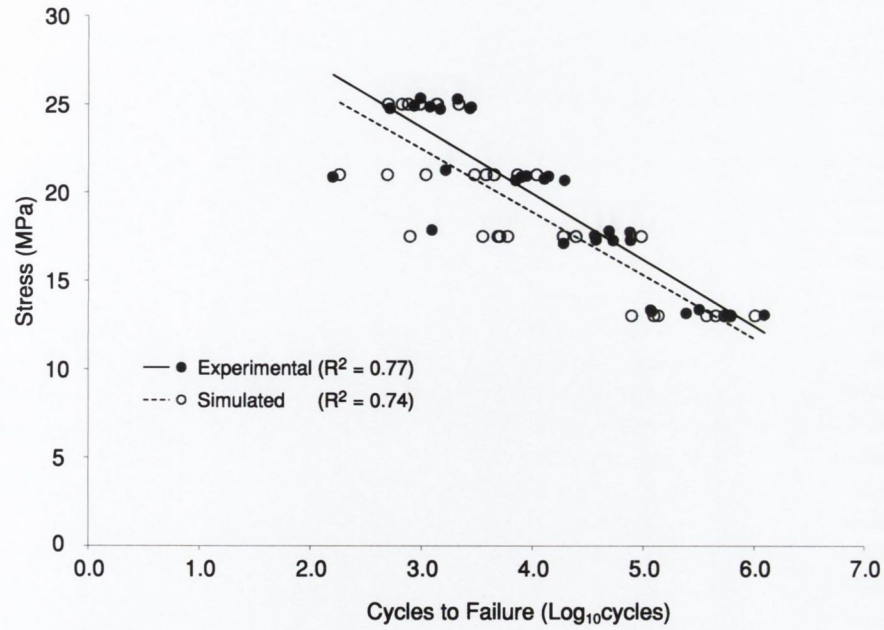


(a) Experiment of Murphy and Prendergast (2000a)

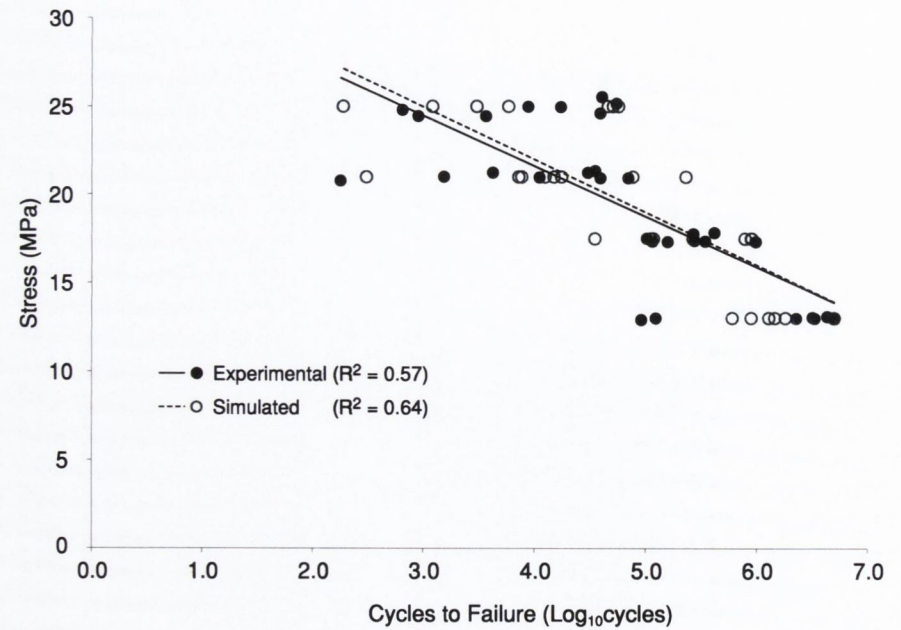


(b) Simulated results of present study

Figure 4.6. *S-N data from (a) experimental study of Murphy and Prendergast (2000a) and (b) simulated S-N data of present study.*



(a) Hand-mixed



(b) Vacuum-mixed

Figure 4.7. Comparison of (a) hand-mixed and (b) vacuum-mixed regression lines of Murphy and Prendergast (2000a) with regression lines fitted to vacuum-mixed data of simulated tests.

Table 4.3. Mean and standard deviation (S.D.) of failure life for each data set and significance (*p*) values for a Student's *t*-test

Stress (MPa)	HM-Exp		HM-Sim		<i>p</i>
	Mean	S.D.	Mean	S.D.	
13	462,054	(398,793)	419,686	(318,887)	0.82
17.5	43,683	(26,058)	20,006	(31,839)	0.12
21	8,985	(6,398)	3,918	(3,690)	0.07
25	1,580	(868)	1,097	(532)	0.20
	VM-Exp		VM-Sim		
13	2,628,680	(1,928,225)	2,652,430	(2,009,684)	0.98
17.5	333,132	(285,597)	503,379	(402,804)	0.35
21	23,841	(24,282)	46,329	(78,481)	0.45
25	20,631	(21,053)	26,961	(26,290)	0.60

alent or larger ranges in failure life were generally found for each stress level for experimental compared to simulated data, see Fig. 4.8.

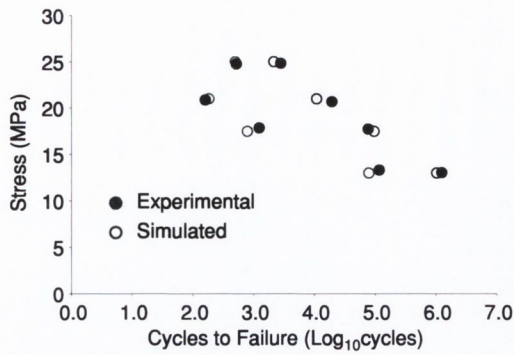
Trends are not immediately apparent from the raw data for either experimental or simulated data sets. To more clearly show differences in average failure life, Murphy and Prendergast (2000a) proposed that regression lines can be fitted through the average values for each distribution. Following this approach shows that the simulated data reproduces the increase in average fatigue life for vacuum-mixed cement specimens over hand-mixed specimens (Fig. 4.9).

4.2 Experimental model of femoral replacement

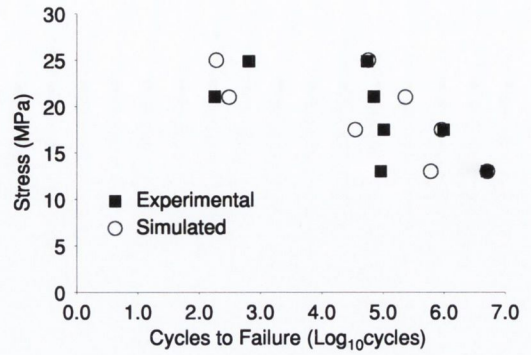
4.2.1 Damage measurements

Large numbers of cracks were found throughout the cement before the application of any load (i.e. due to residual stress). They were found in both the matt and polished prosthesis groups, see crack distributions of Fig. 4.10. Damage was also found to accumulate in a distributed fashion during testing without any obvious spatial pattern, see final crack distributions of Fig. 4.10.

The large variability in crack locations made it difficult to observe any pattern



(a) Hand-mixed



(b) Vacuum-mixed

Figure 4.8. Minimum and maximum failure life of (a) hand-mixed and (b) vacuum-mixed specimens at each stress level from experimental study of Murphy and Prendergast (2000a) and simulations of present study.

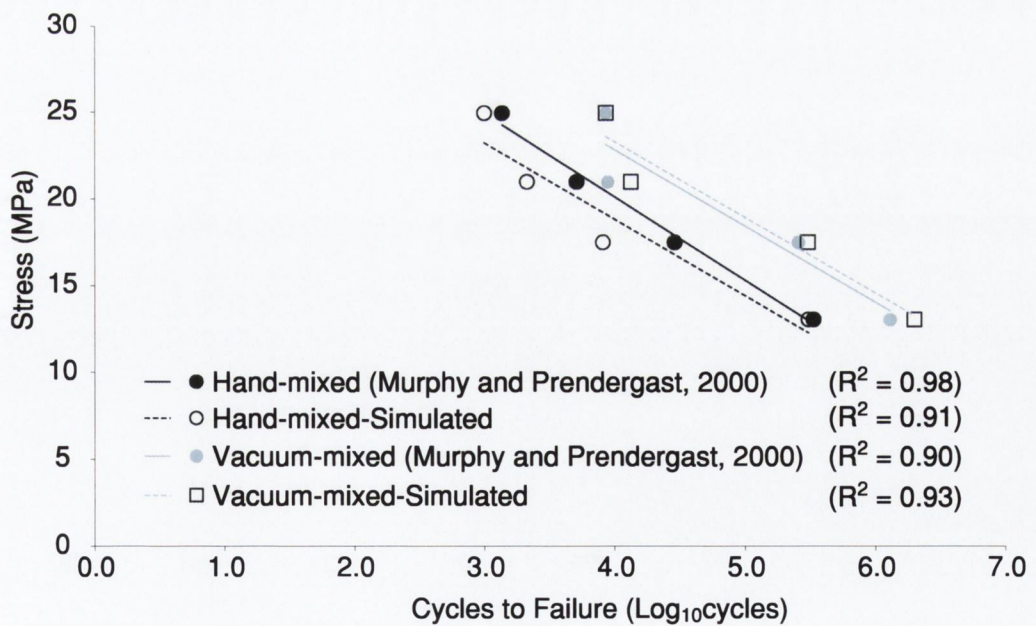


Figure 4.9. Average failure life at each stress level and resulting regression lines for hand-mixed and vacuum-mixed data of Murphy and Prendergast (2000a) and simulations of present study. The simulated data mimics the average increase in fatigue life for vacuum-mixing found experimentally.

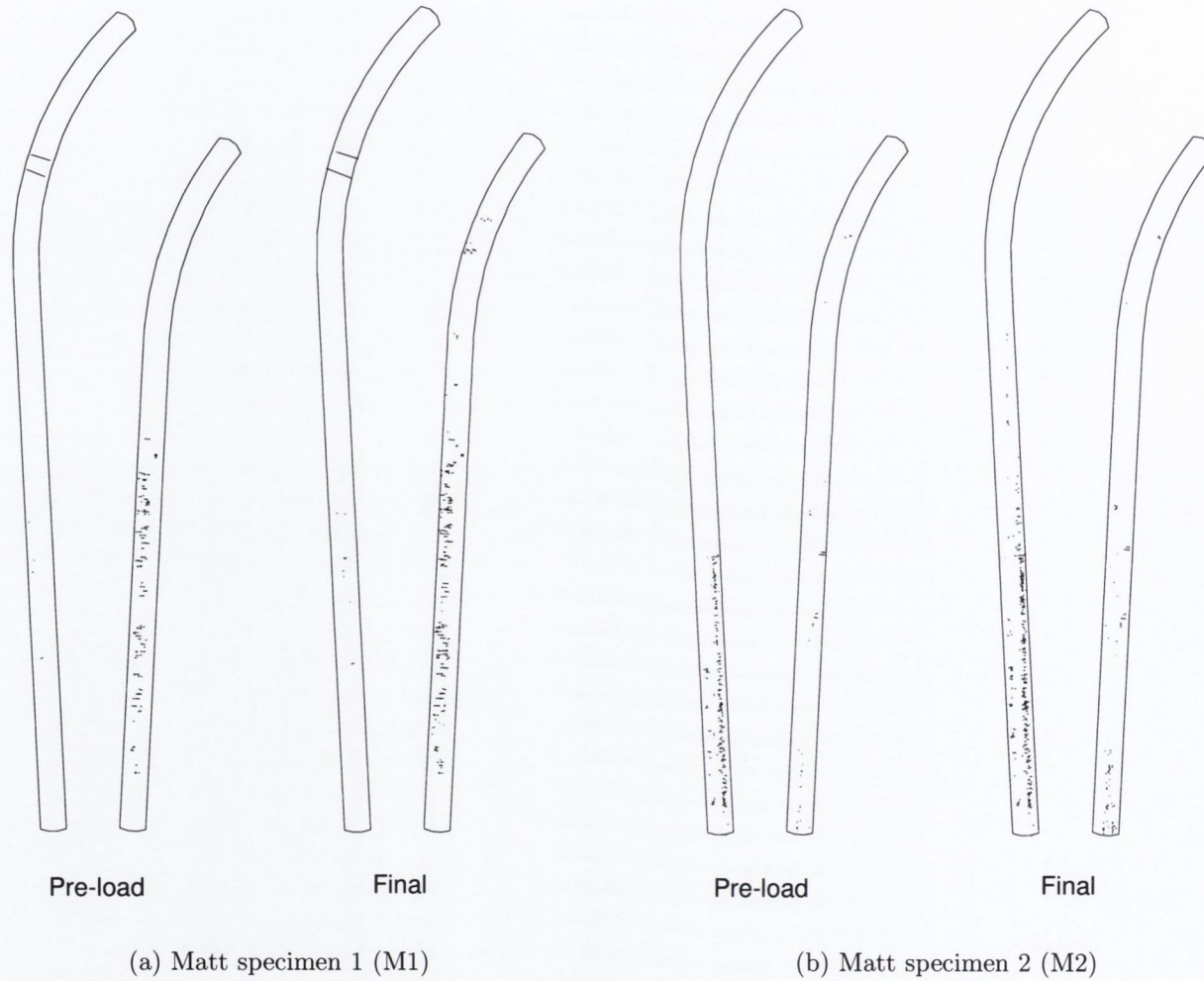


Figure 4.10. *Spatial crack distributions for each of the specimens tested before and after testing. First letter of specimen label indicates surface finish: M = Matt (grit blasted) and P = polished.*

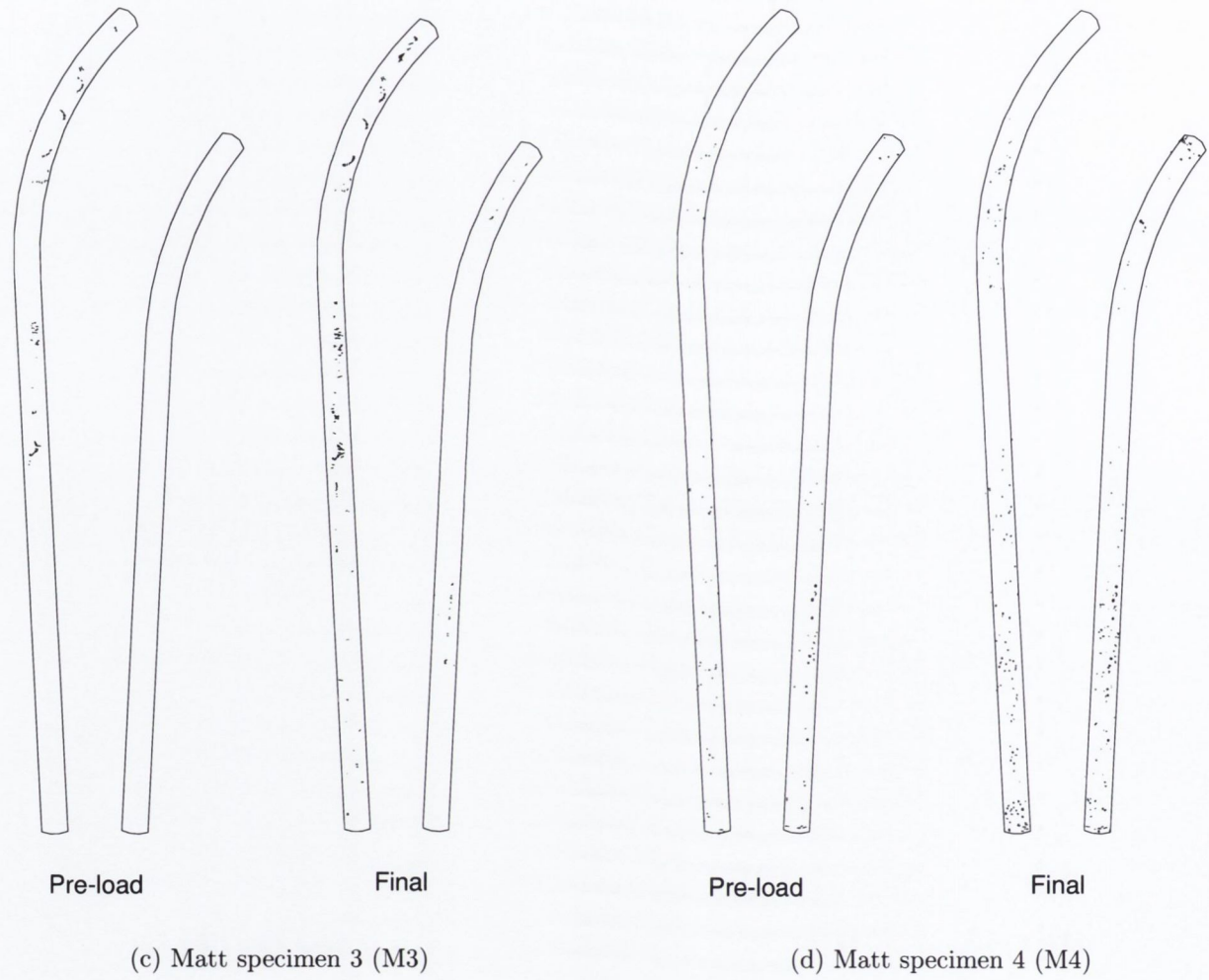
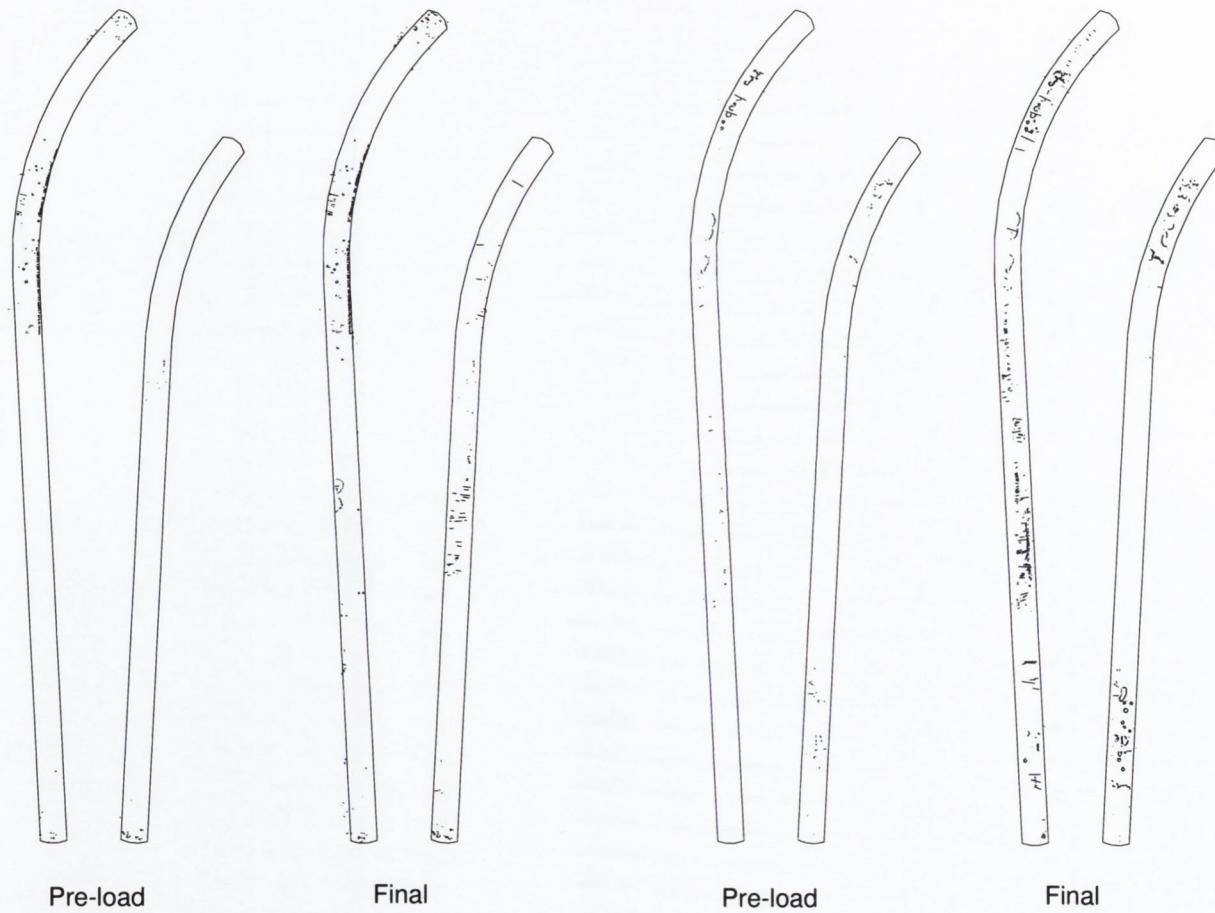


Figure 4.10. *cntd.* Spatial crack distributions for each of the specimens tested before and after testing. First letter of specimen label indicates surface finish: M = Matt (grit blasted) and P = polished.



(e) Matt specimen 5 (M5)

(f) Polished specimen 1 (P1)

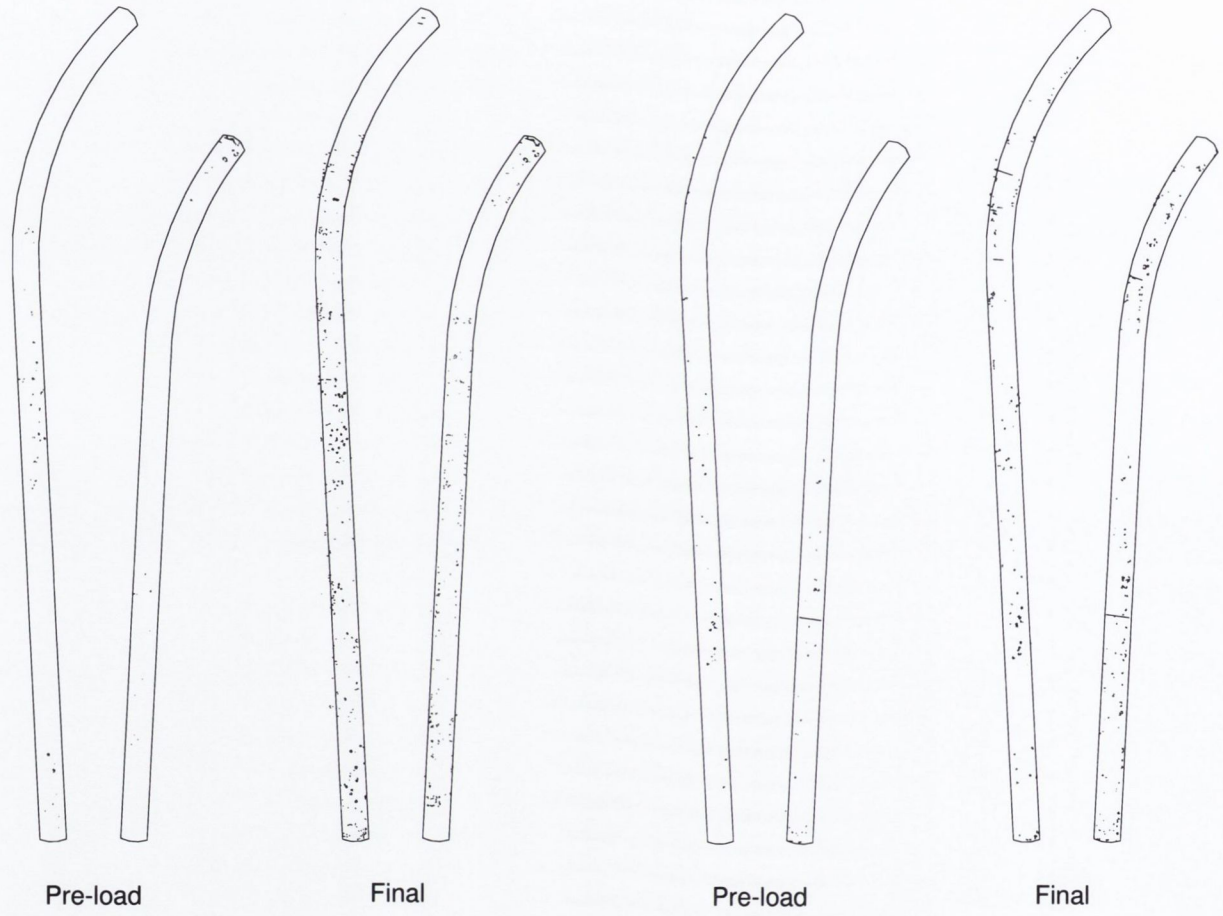
Figure 4.10. *cntd.* Spatial crack distributions for each of the specimens tested before and after testing. First letter of specimen label indicates surface finish: M = Matt (grit blasted) and P = polished. Some cracks occur outside the boundaries as the bone layer was not perfectly uniform and varied between specimens.



(g) Polished specimen 2 (P2)

(h) Polished specimen 3 (P3)

Figure 4.10. *cntd.* Spatial crack distributions for each of the specimens tested before and after testing. First letter of specimen label indicates surface finish: M = Matt (grit blasted) and P = polished.



(i) Polished specimen 4 (P4)

(j) Polished specimen 5 (P5)

Figure 4.10. *cntd.* Spatial crack distributions for each of the specimens tested before and after testing. First letter of specimen label indicates surface finish: M = Matt (grit blasted) and P = polished.

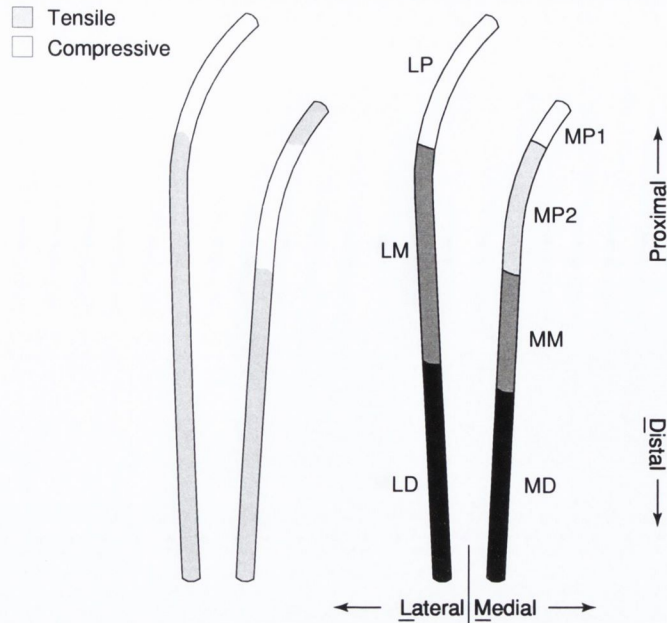


Figure 4.11. *Regions used to quantify damage. Anatomical terminology is used to denote position. Medial (M) implies toward the midline of the body, lateral (L) implies away from the midline of the body, proximal (P) implies toward the attachment point with the trunk, and distal (D) implies away from the attachment point with the trunk; when used as the second letter, M denotes middle. The medial proximal (MP) region was subdivided to account for a change from tensile to compressive maximum principal stress across this region, as predicted by a linear elastic finite element analysis with bonded interfaces (shown on the left).*

in damage accumulation within the cement. In an attempt to quantify a pattern of damage accumulation, the cement was partitioned into seven regions. The regions were chosen to correspond to where a finite element model had predicted tensile or compressive stress, see Fig. 4.11. Damage was quantified as the *sum-of-crack-lengths* for all cracks found to have their centroid inside a given region, i.e.

$$D = \sum_{i=1}^n a_i, \quad (4.3)$$

where a_i is the length of the i th crack occurring in a region with n cracks.

Regarding pre-load (residual stress) damage, similar damage amounts and variability for both matt and polished groups were found in all regions on the lateral side of the cement (Fig. 4.12). Although regions MP1 and MM appear to have clearly different averages between matt and polished groups on the medial side, these differences were not found to be significant at $p = 0.09$ and $p = 0.21$ respectively (Table 4.4).

Damage accumulated during testing also showed high variability. Higher average

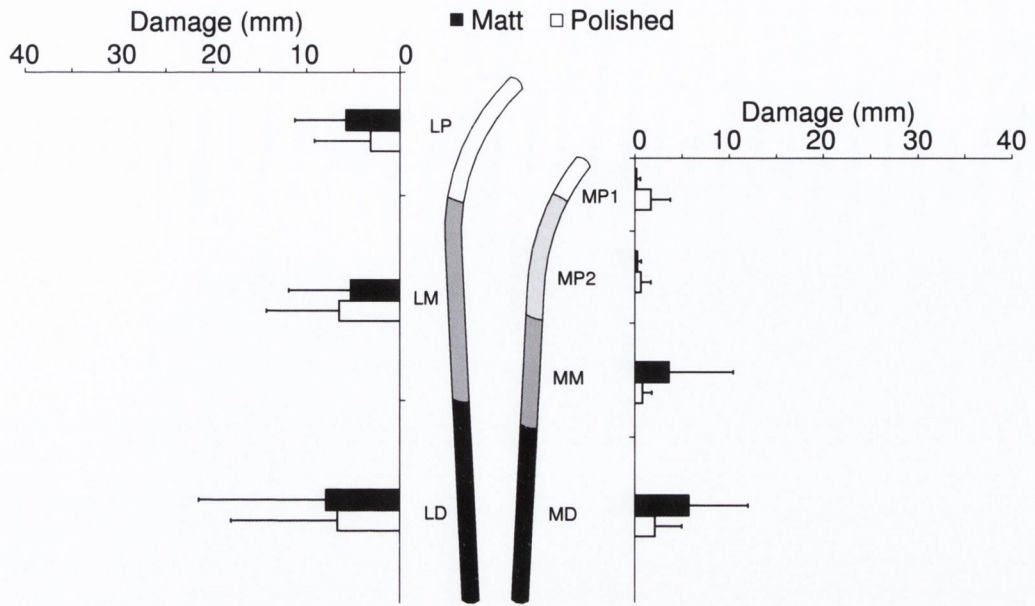


Figure 4.12. Mean pre-load damage according to region. Error bars indicate one standard deviation.

Table 4.4. Mean and standard deviation of pre-load damage and growth in damage during testing. Significance values are for a one-tail Student's T-test. For the letters succeeding a region name, M = matt and P = polished.

Region		Pre-load (mm)		<i>p</i>	Growth (mm)		<i>p</i>
MP1	Matt	0.19	(0.42)	0.09	0.97	(1.03)	0.14
	Polished	1.75	(2.08)		1.99	(1.60)	
MP2	Matt	0.30	(0.41)	0.23	2.01	(1.94)	0.09
	Polished	0.70	(1.03)		5.70	(5.08)	
MM	Matt	3.68	(6.80)	0.21	5.25	(7.84)	0.22
	Polished	0.86	(0.99)		2.16	(2.10)	
MD	Matt	5.77	(6.29)	0.15	5.29	(2.43)	0.17
	Polished	2.19	(2.84)		8.81	(6.88)	
LP	Matt	5.78	(5.46)	0.24	3.53	(3.82)	0.11
	Polished	3.14	(5.99)		7.49	(5.16)	
LM	Matt	5.25	(6.62)	0.40	6.40	(5.87)	0.02
	Polished	6.49	(7.74)		19.01	(8.92)	
LD	Matt	7.91	(13.52)	0.44	6.71	(5.88)	0.10
	Polished	6.69	(11.35)		18.72	(16.75)	

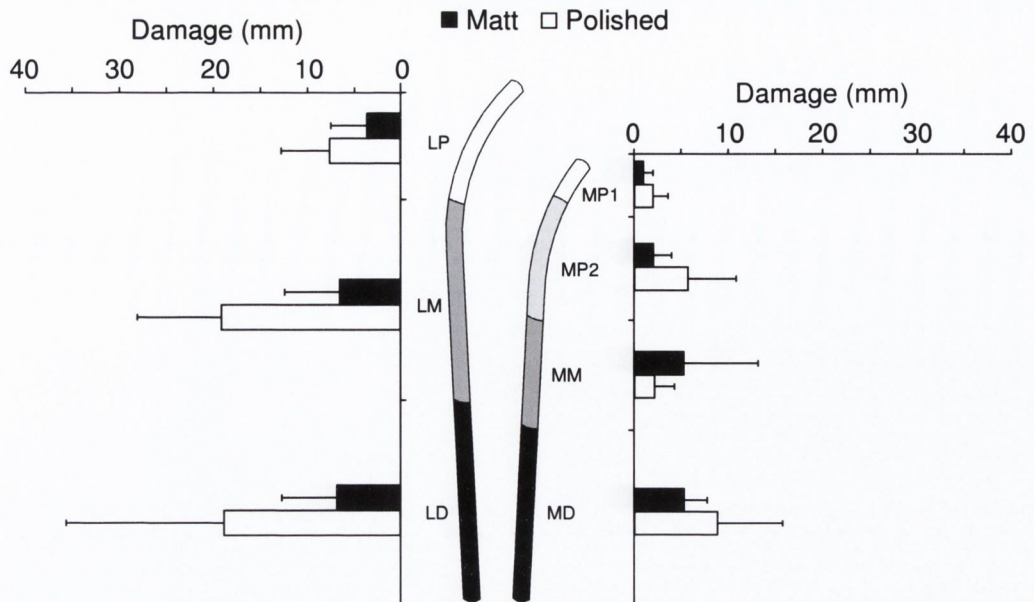


Figure 4.13. Mean damage accumulated during testing according to region. Error bars indicate one standard deviation.

damage growth was found for the polished group in all regions except the middle of the medial side (Fig. 4.13 and Table 4.4). However, the only region to show a statistically significant increase in damage accumulated during testing was the middle of the lateral side (Table 4.4).

Some further understanding of the source of the variability in the data can be achieved by examining the relationship between pre-load damage and damage accumulated during testing. When, pre-load damage is plotted against damage accumulated after two million cycles, a positive correlation is found, see Fig. 4.14. In general, the matt specimens show better correlation.

4.2.2 Migration measurements

Displacements of the actuator were also monitored for many of the tests but a complete history was only obtained for matt specimens 3–5 and polished specimens 3–5. Displacement of the actuator head at peak load was observed to change during testing relative to the displacement applied in the first cycle of loading for both matt and polished specimens. Greater variability was observed for the polished group (Fig. 4.15). As migration was expected to contain a considerable creep component, the peak-to-peak displacement of the actuator over a cycle was also monitored since the cyclic displacement should be more indicative of specimen stiffness. Also,

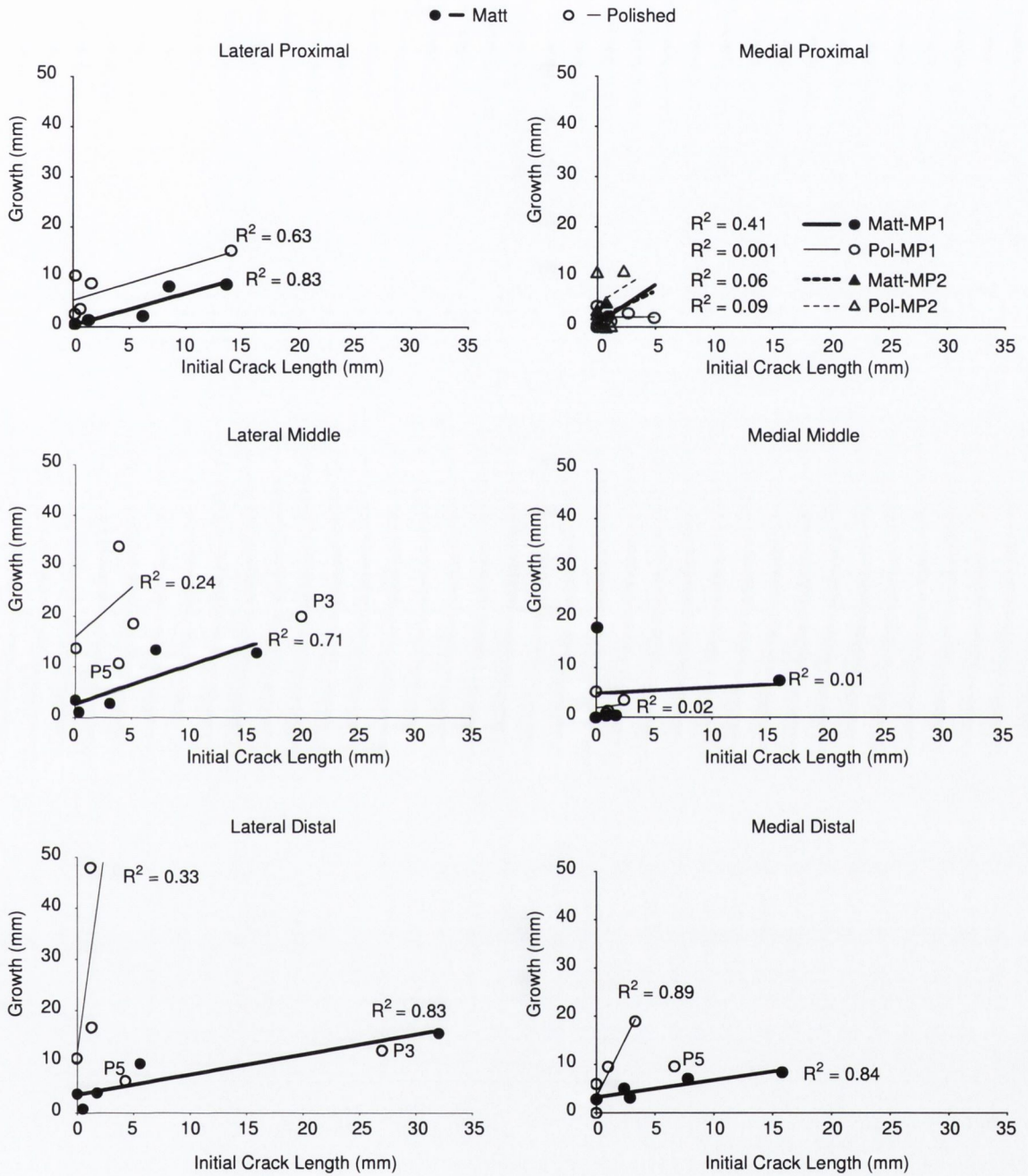


Figure 4.14. Relationship between pre-load damage and damage accumulated during testing for each region. The effect of excluding apparent outliers on correlation of the data is indicated in the middle of the lateral and both distal regions; the excluded data points are indicated by labelling. Regions MP1 and MP2 have been combined in a single plot and use a separate legend to the other regions.

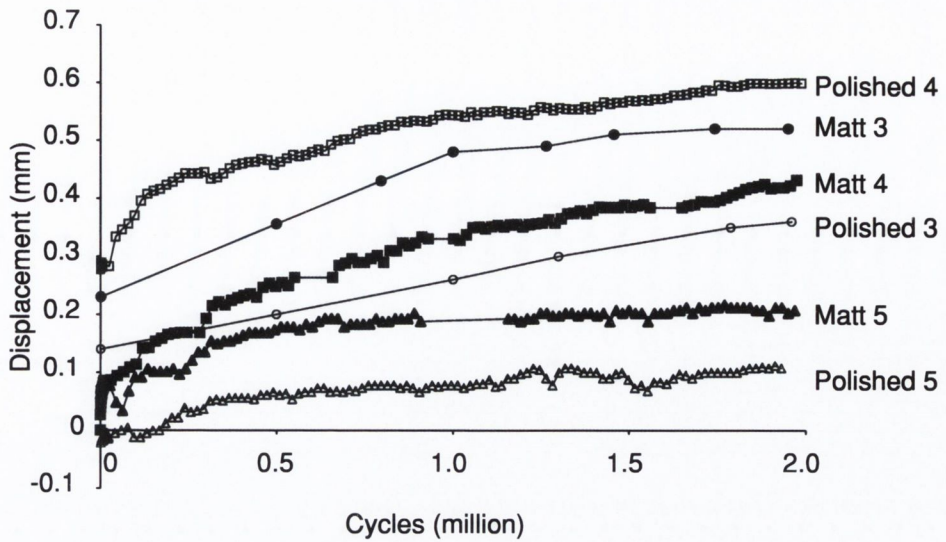


Figure 4.15. Migration of actuator displacement at peak load during testing of six specimens.

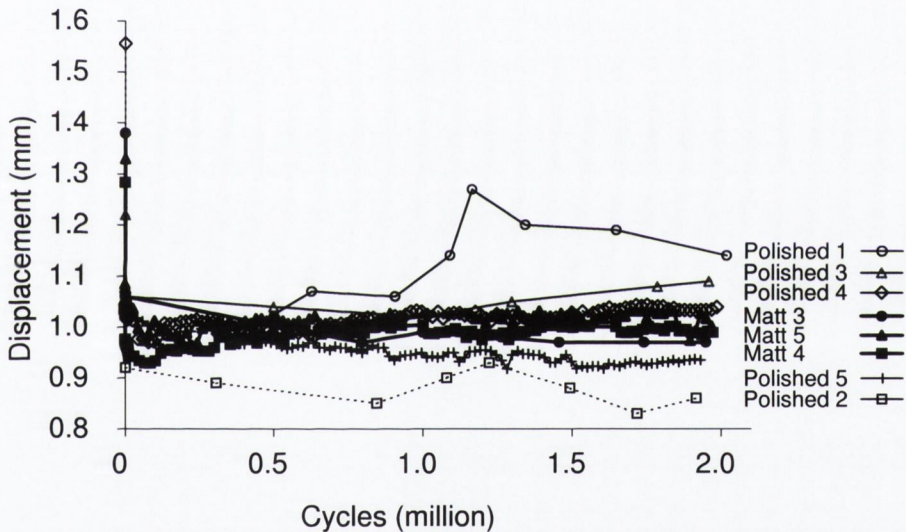


Figure 4.16. Inducible displacement of actuator during testing of eight specimens. Polished specimens are listed

because this did not require referencing with respect to displacements at the start of a test, data from the remaining two polished prostheses became available. This cyclic displacement, referred to as *inducible* displacement, was also observed to be more variable for the polished specimens (Fig. 4.16). The polished specimen that accumulated the most damage during testing (P1) also showed the largest inducible displacement. Furthermore, this specimen also exhibited more erratic evolution of inducible displacement. Specimen P2, which accumulated the least damage of the polished specimens, had the lowest inducible displacement.

A regression analysis was performed to further investigate a possible link be-

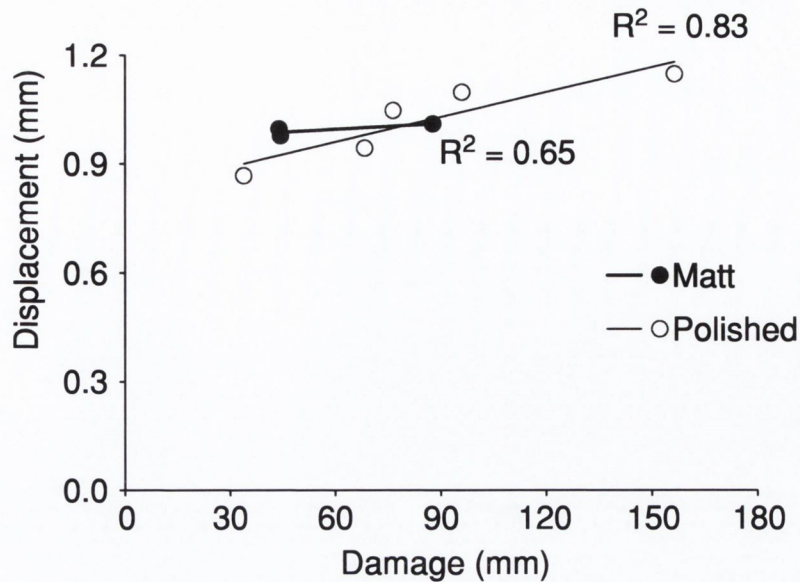


Figure 4.17. *Inducible displacement of actuator vs total damage at end of testing.*

tween inducible displacement and damage (Fig. 4.17). This shows that much of the variability for polished specimens may be attributed to total damage at the end of testing. As only three matt specimens were available for this analysis, they serve mainly as a comparison. However, the trend appears to be that inducible displacement for the matt specimens was less sensitive to damage.

4.3 Simulations of fatigue damage accumulation in the experimental model

4.3.1 Damage accumulation

Damage initiated by shrinkage stress was predicted in all regions of cement for both bonded and debonded specimens (e.g. Figs. 4.18 and 4.19¹). For comparison, the shrinkage stress distribution without the inclusion of pores is shown in Fig. 4.20. Because the shrinkage damage algorithm was based on the assumption that pores were required to initiate damage, the pre-load damage distributions correspond spatially with regions of high shrinkage stress caused by pores.

Damage accumulated during loading was also observed mainly in regions with pores and pre-load damage. Comparison of the stress distributions for the deter-

¹Remaining specimens are included in Appendix B, pp. 191.

Bonded (i.e. Matt): Predictions for Specimen #1

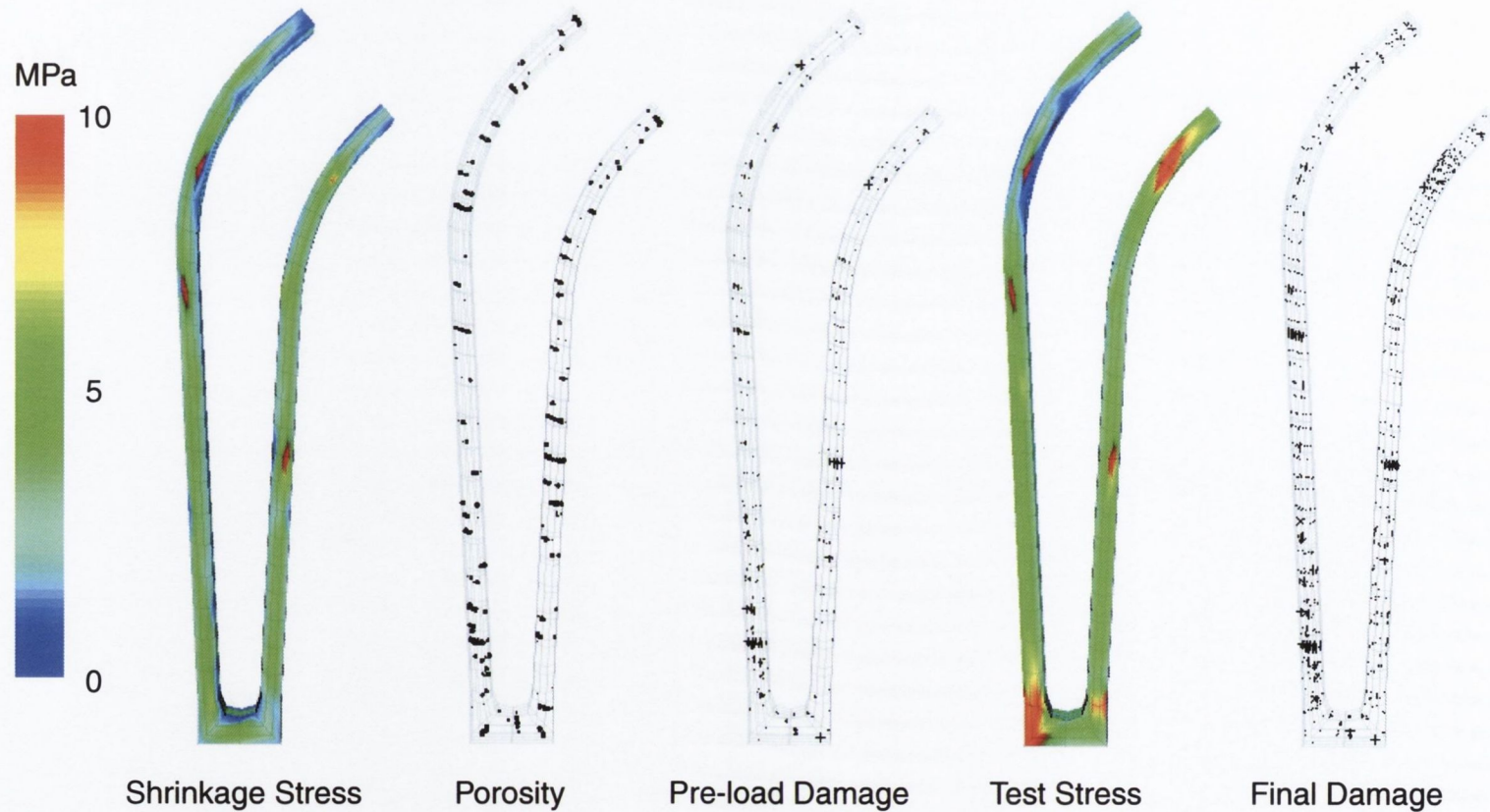


Figure 4.18. Stress, porosity, and damage distributions for bonded specimen no. 1. After generation of the pore distribution (pores are plotted as spheres scaled to the volume fraction of each pore), the residual stresses were read in from the original thermoelastic analysis. The pore distribution was then used to calculate local stress concentrations in the stress field. This new shrinkage stress distribution was used to initiate damage around pores (damage data are plotted as principal damage vectors) and the test load was then applied. Damage accumulation was calculated according to the timestep scheme of section 3.2.4 for the superposed stress distributions from loading and shrinkage (the stress distribution shown is for the first cycle of testing).

Debonded (i.e. Polished): Predictions for Specimen #1
 Note: specimens 2, 3, 4, 5 of each specimen type shown in Appendix B

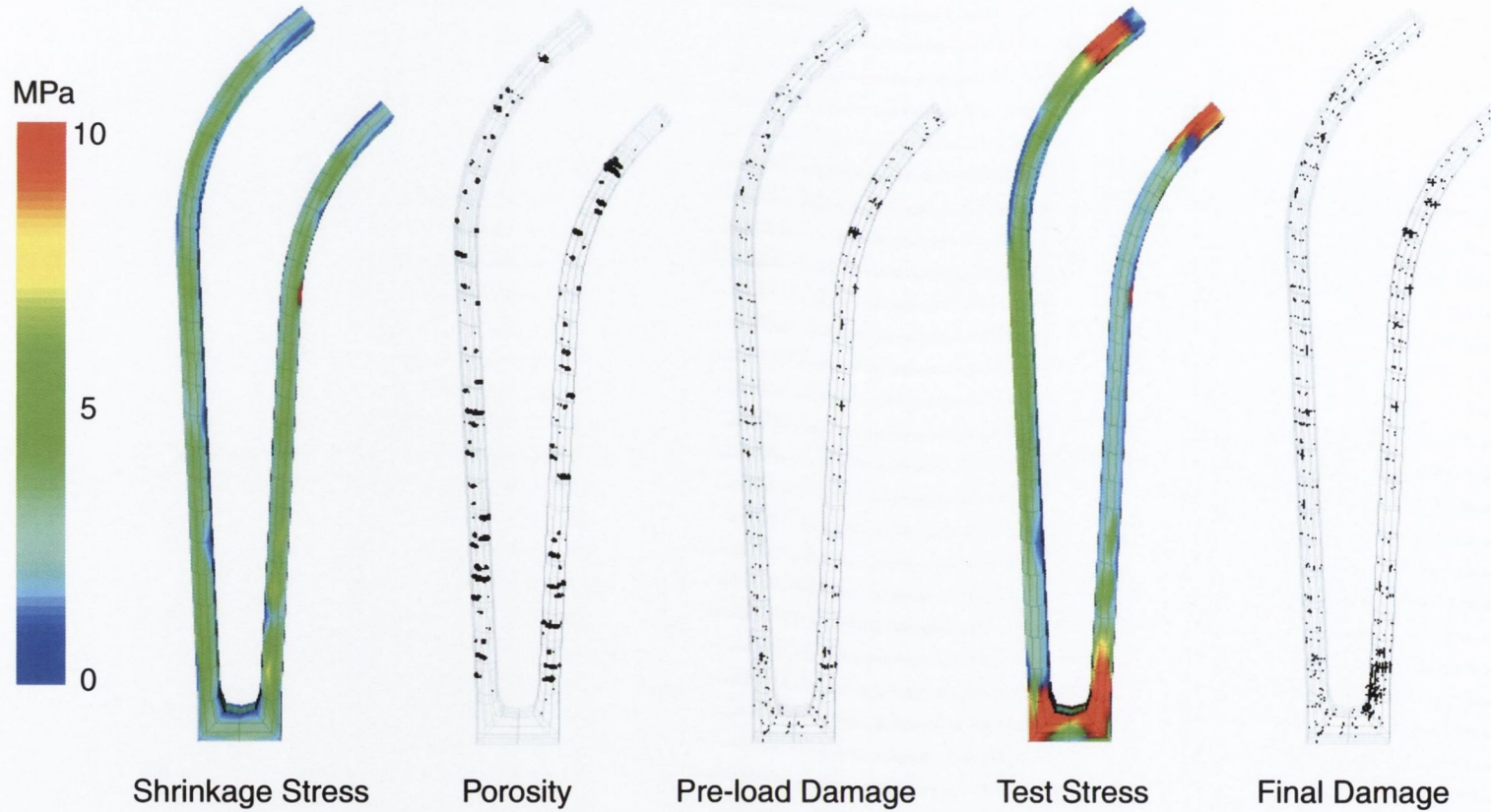


Figure 4.19. Stress, porosity, and damage distributions for debonded specimen no. 1. After generation of the pore distribution (pores are plotted as spheres scaled to the volume fraction of each pore), the residual stresses were read in from the original thermoelastic analysis. The pore distribution was then used to calculate local stress concentrations in the stress field. This new shrinkage stress distribution was used to initiate damage around pores (damage data are plotted as principal damage vectors) and the test load was then applied. Damage accumulation was calculated according to the timestep scheme of section 3.2.4 for the superposed stress distributions from loading and shrinkage (the stress distribution shown is for the first cycle of testing).

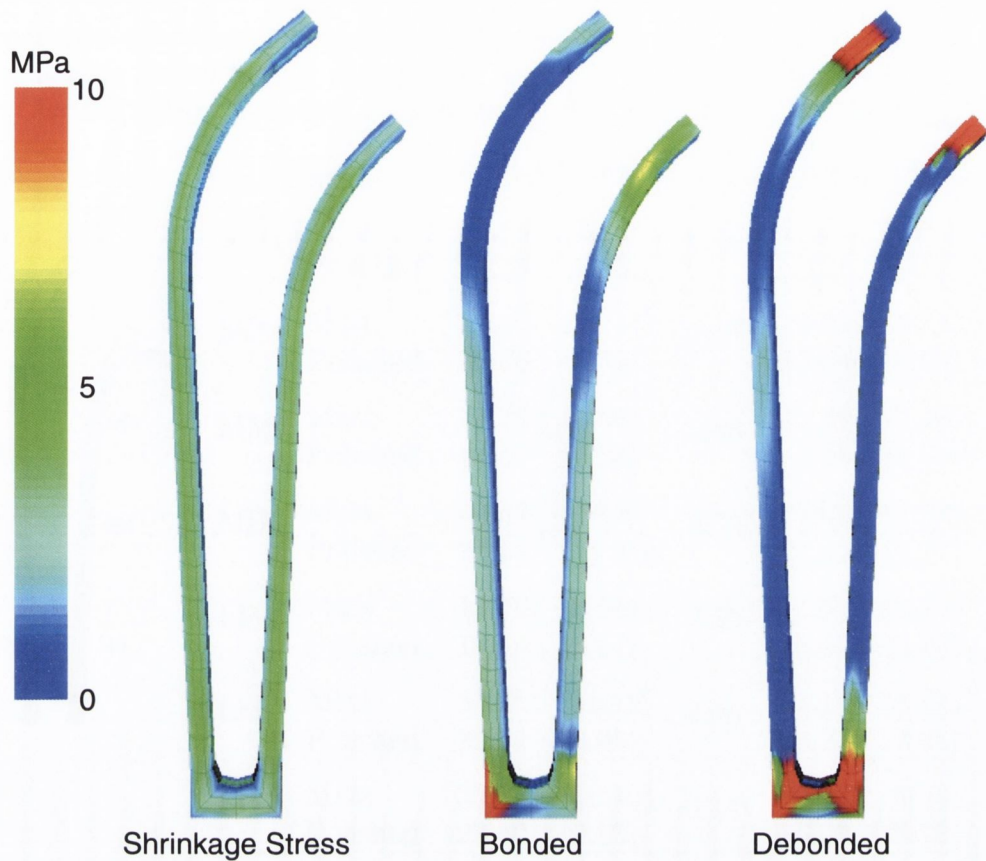


Figure 4.20. *Maximum principal stress for shrinkage of a pore-free cement and due to loading for the deterministic models.*

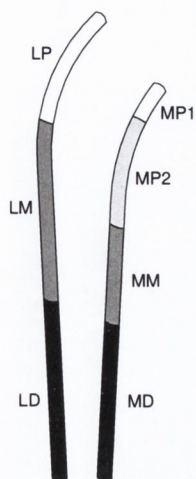
ministic specimens in Fig. 4.20 shows that stress was more uniformly distributed within the cement for a bonded prosthesis compared with a debanded prosthesis during testing. Debanding caused stresses to redistribute towards the cement surrounding the tip of the prosthesis and to the most proximal regions (Fig. 4.20). The region surrounding the tip, in particular on the medial side, was consistently observed to accumulate damage during loading in the debanded specimens (Figs. 4.19 and B.5–B.8).

Damage was segregated and quantified for the same regions used in the experimental tests (Fig. 4.11, page 97). This allowed more detailed comparison between stochastic specimens and deterministic specimens as well as with experimental results. Damage was quantified as the trace of the damage tensor at each integration point and was summed over all integration points for a given region, i.e.

$$D_{region} = \sum_{ip=1}^{n_{ip}} \sum_{i=1}^3 D_{ii}. \quad (4.4)$$

In addition to this measure, completely damaged (cracked) directions for each in-

Table 4.5. Mean and standard deviation of pre-load damage and growth in damage during testing. Significance values are for a one-tail Student's *t*-test. For the letters succeeding a region name, B = bonded and D = debonded.



The diagram shows two views of a prosthetic limb. The left view shows regions LP (top), LM (middle), and LD (bottom). The right view shows regions MP1 (top), MP2 (middle), MM (lower middle), and MD (bottom).

	Region		Pre-load (mm)	<i>p</i>	Growth (mm)	<i>p</i>	
MP1	Matt	4.86	(2.14)	0.48	15.81	(0.89)	0.00
	Polished	4.78	(1.91)		3.28	(0.57)	
MP2	Matt	22.30	(11.11)	0.47	14.55	(4.74)	0.23
	Polished	22.76	(8.70)		12.68	(2.24)	
MM	Matt	15.72	(5.88)	0.36	11.18	(1.46)	0.01
	Polished	14.10	(7.98)		8.20	(1.80)	
MD	Matt	20.34	(6.18)	0.38	18.08	(1.78)	0.00
	Polished	22.16	(11.41)		47.43	(6.27)	
LP	Matt	18.60	(6.68)	0.29	11.89	(3.75)	0.01
	Polished	16.18	(6.87)		19.67	(3.89)	
LM	Matt	22.24	(16.19)	0.46	21.15	(3.81)	0.19
	Polished	23.04	(8.05)		23.21	(3.05)	
LD	Matt	33.12	(16.19)	0.14	30.95	(2.30)	0.00
	Polished	23.40	(9.15)		18.00	(5.72)	

tegration point were also monitored, i.e. directions in which the principal damage had reached unity. Similar pre-load damage was predicted for both bonded and debonded specimens in all regions (Table 4.5). As would be expected, a Student's *t*-test could not discriminate between pre-load damage distributions of the bonded and debonded prostheses for any region. However, substantially different damage accumulation during loading between bonded and debonded specimens *was* predicted. Significance values indicate that only two regions, MP2 and LM, could not be distinguished from one another. Also, region MD clearly accumulated the highest average damage and was significantly higher for the case of a debonded prosthesis.

Comparison of these results with those of a deterministic model can also be made. These comparisons show that substantially different damage accumulation was predicted when the stochastic influences were removed (Fig. 4.21). For the deterministic bonded case, cracks were only found in one region, MP1 (Fig. 4.21a). For the deterministic debonded case, only regions LP and MD were predicted to accumulate cracks (Fig. 4.21b). When the sum-of-damage-trace measure was used, the deterministic specimens were predicted to experience damage accumulation through-

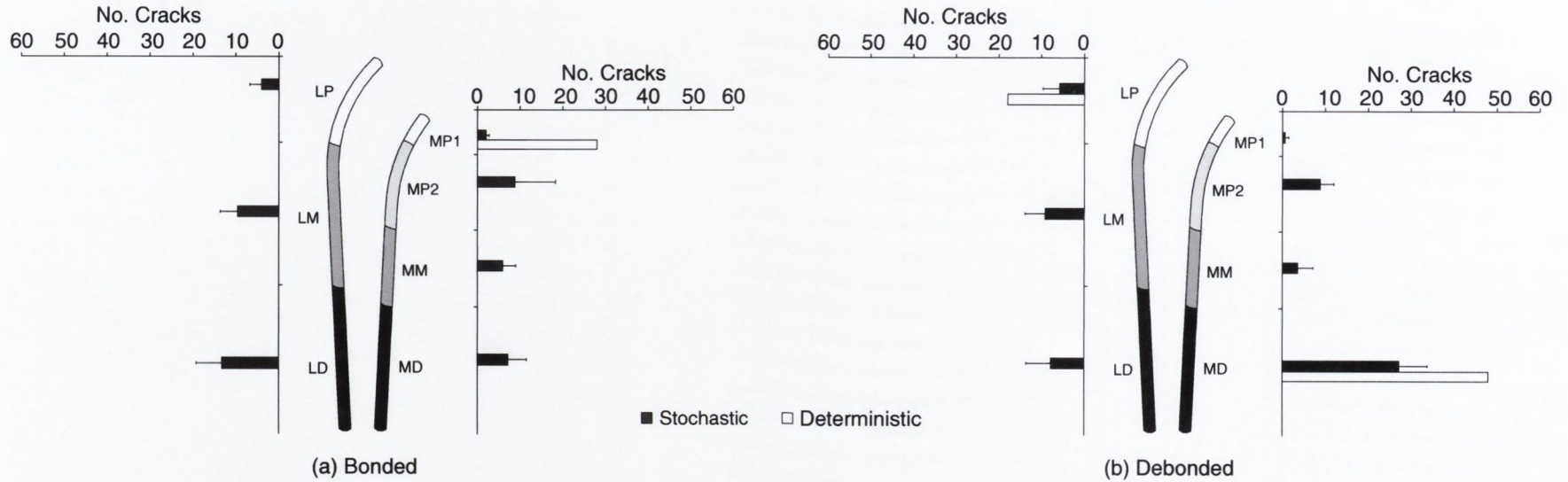


Figure 4.21. Mean number of integration point cracks (i.e. completely damaged directions) accumulated during loading according to region for (a) bonded and (b) debonded specimens for the stochastic simulations compared with predictions of a deterministic model. Error bars indicate one standard deviation. Deterministic models predicted considerably more localised damage accumulation than the stochastic models.

out the specimen (Fig. 4.22). However, the predicted trend of much greater damage accumulation in region MP1, for the bonded case, and regions LP and MD, for the debonded case, was repeated. Thus, much more localised damage accumulation was predicted by the deterministic models for both the number of cracks and total damage.

Comparisons were also made for evolution of damage accumulation (Fig. 4.23). Stochastic specimens were predicted to experience a rapid burst of damage accumulation at the start of testing for both total number of cracks (Fig. 4.23a) and total damage (Fig. 4.23b). Standard deviations of both bonded and debonded specimens increased over the period of testing. Debonded specimens were predicted to maintain a higher damage accumulation rate for a longer period, resulting in a significantly higher mean damage at the end of testing ($p = 0.012$ for total number of cracks and $p = 0.007$ for total damage). The deterministic models were not predicted to experience a rapid burst of damage accumulation at the start of testing (Fig. 4.23). The bonded specimen was predicted to accumulate damage almost linearly, but with a higher steady state damage accumulation rate than for either of the stochastic cases. Debonding resulted in a more rapid early damage growth but this still did not match the initial rate of damage accumulation predicted for the stochastic models. Therefore, considerably different evolution of damage accumulation was predicted for the stochastic and deterministic specimens.

4.3.2 Analysis of factors influencing damage accumulation

4.3.2.1 Pre-load damage

Regression analysis for simulated damage accumulation against pre-load damage showed a stronger relationship than was found for the experimental data (compare Fig. 4.24 with 4.14). However, there was still some variability that could not be attributed to pre-load damage. This meant that region MD, which had exhibited the clearest difference in mean damage growth between bonded and debonded conditions, had to have an outlier removed to establish a clear trend.

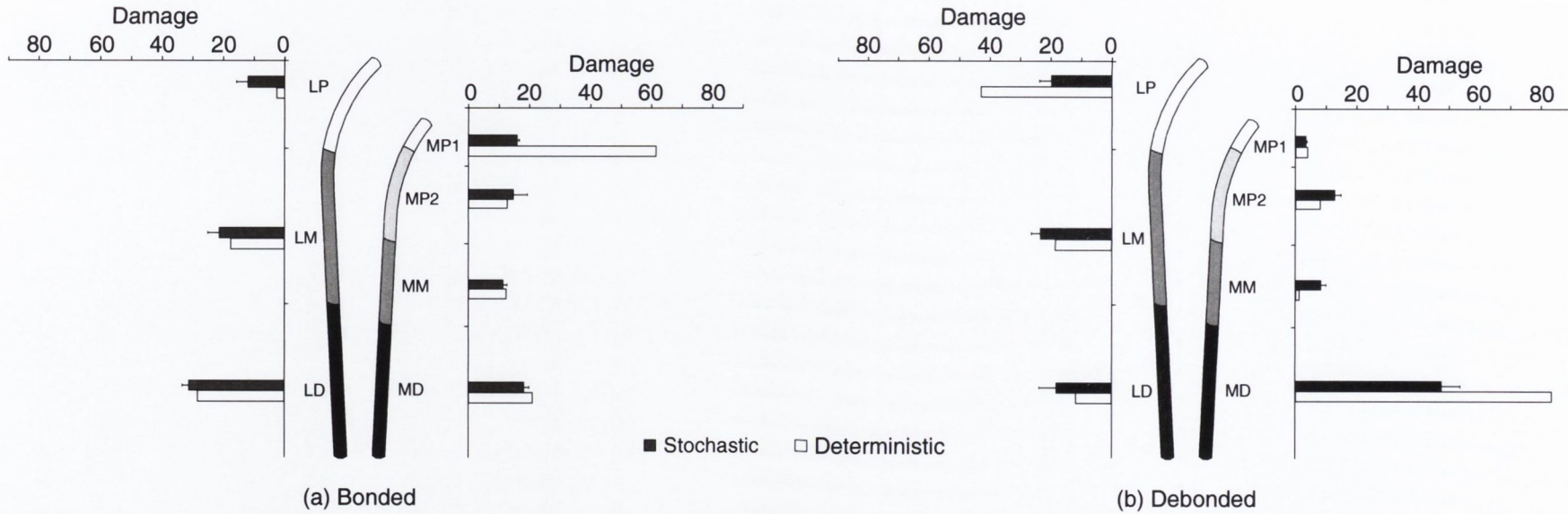
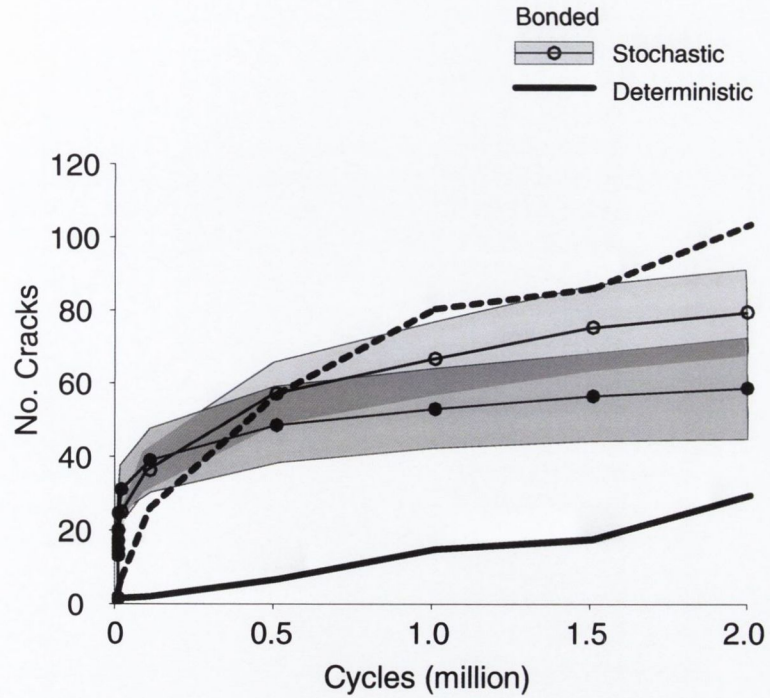
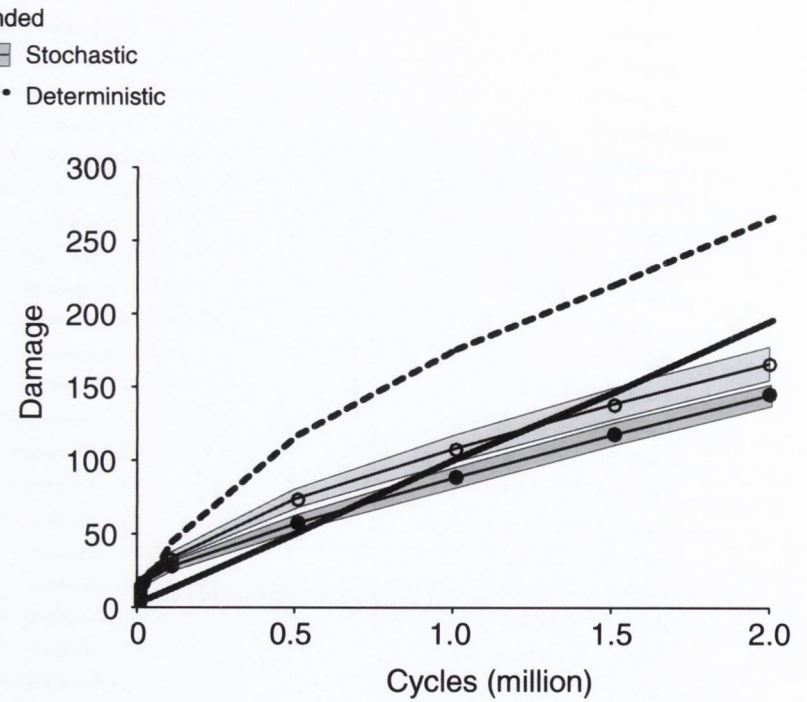


Figure 4.22. Mean damage accumulated during loading according to region for (a) bonded and (b) debonded specimens for the stochastic simulations compared with predictions of a deterministic model. Error bars indicate one standard deviation. Deterministic models predicted considerably more localised damage accumulation than the stochastic models, although the differences are not as evident as for the number of cracks accumulated in Fig. 4.21.



(a) Number of cracks



(b) Total damage (sum-of-damage-trace)

Figure 4.23. Evolution of (a) number of cracks (*i.e.* principal damage directions that had reached a value of one) and (b) total damage (*i.e.* as calculated according to (4.4)). Shaded regions represent one standard deviation about the mean values. The heavier lines indicate predictions of the deterministic simulations. Stochastic simulations predicted much higher growth at the start of testing than the deterministic specimens. They were also predicted to stabilise earlier to a lower steady state damage growth rate than the deterministic specimens.

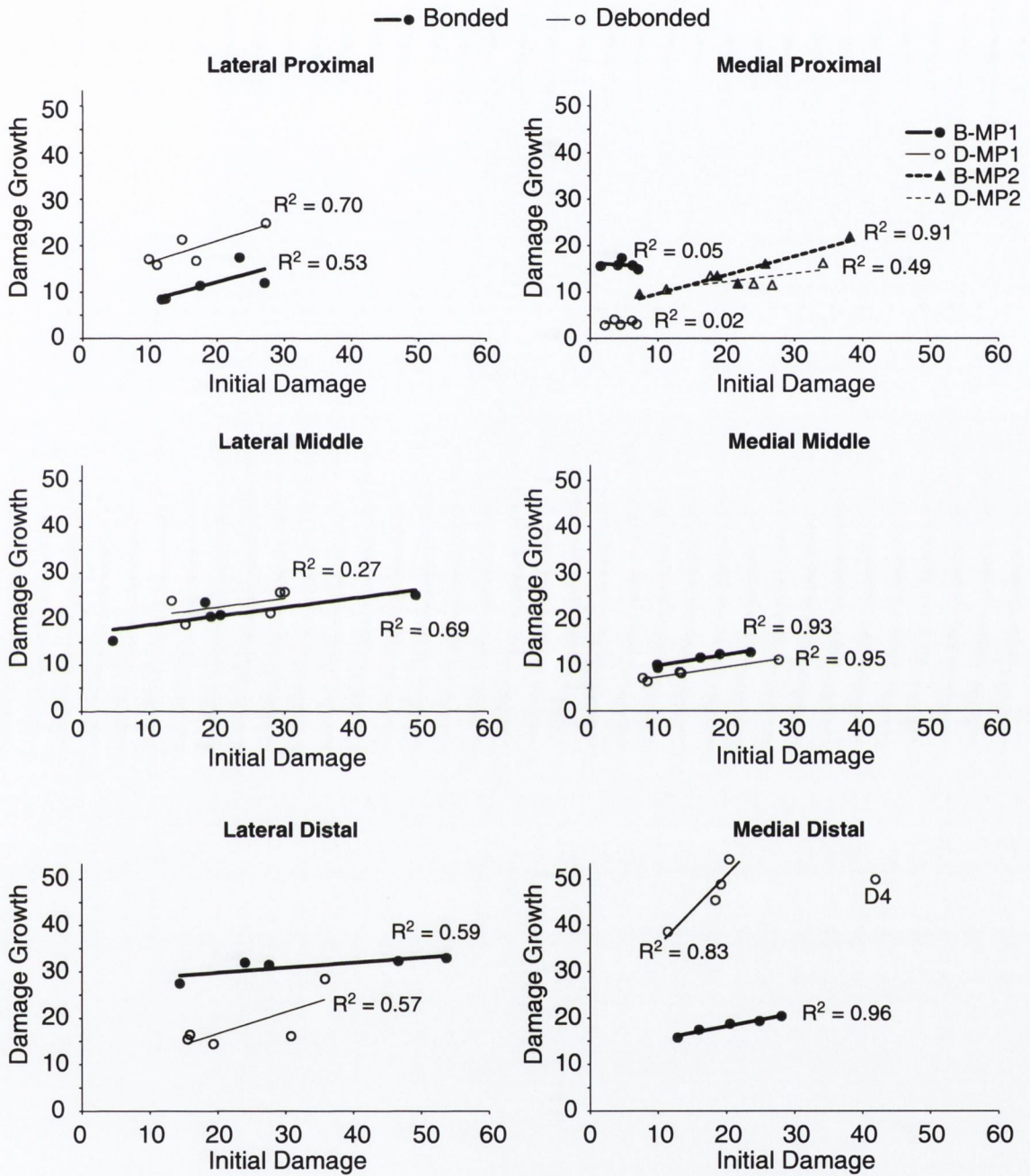


Figure 4.24. Relationship between damage accumulated during testing and pre-load damage. The effect of excluding a possible outlier on correlation of the data is indicated in the medial distal region; the excluded data point is indicated by labelling. Regions MP1 and MP2 have been combined in a single plot and use a separate legend to the other regions. Label D4 implies debonded specimen no. 4.

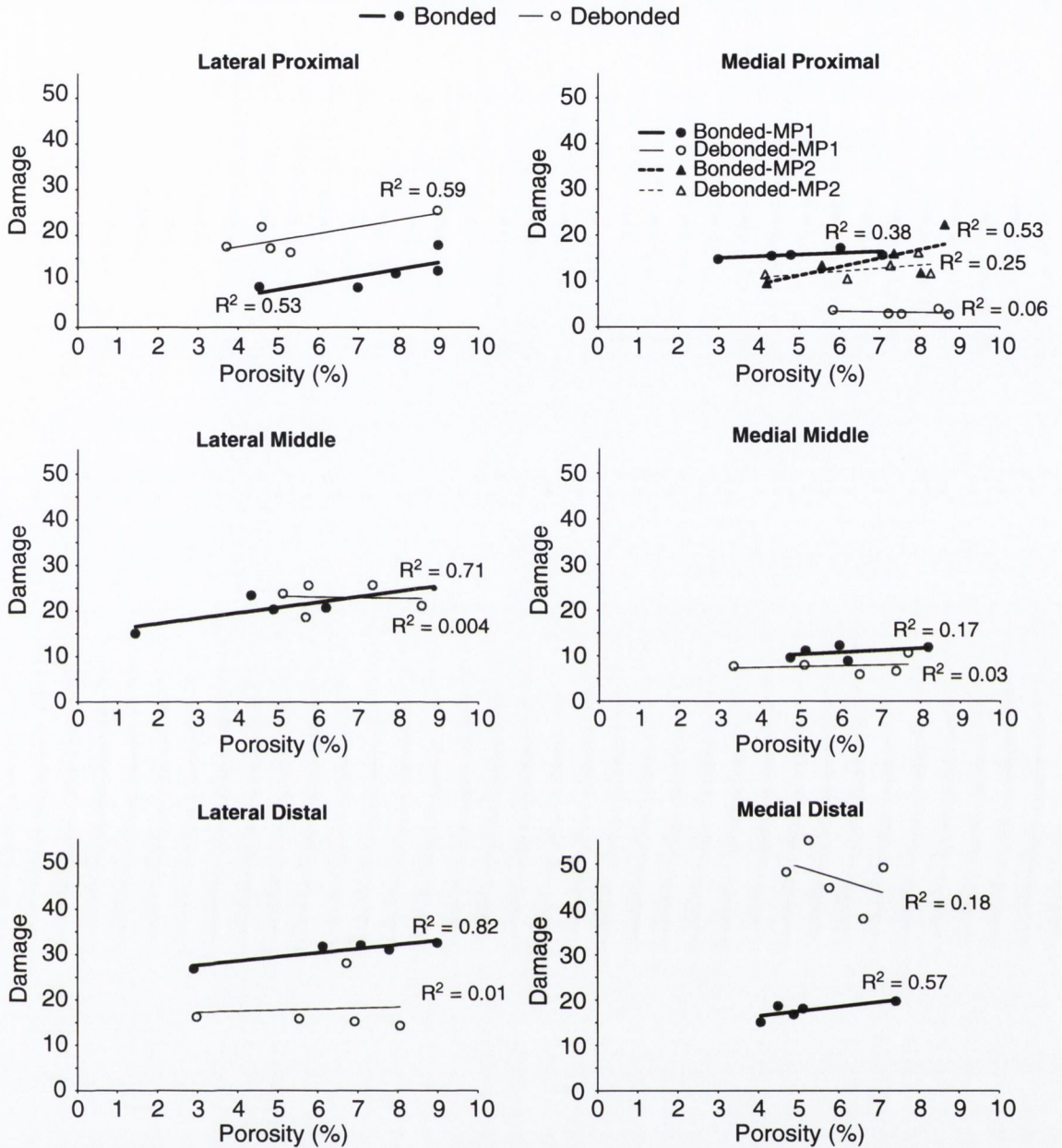


Figure 4.25. Relationship between damage accumulated during testing and porosity. Regions MP1 and MP2 have been combined in a single plot and use a separate legend to the other regions. Bonded specimens show better correlation but the overall trend is weak. Some regions show systematic differences.

4.3.2.2 Porosity

A regression analysis of damage accumulation against porosity shows that bonded specimens exhibited better correlation (Fig. 4.25). However, the overall trend appeared weak with no region showing a high sensitivity to the presence of pores. Several regions had regression lines that were offset from one another, with little or no overlap between maximum data points from the lower data set and minimum points from the higher data set, e.g. regions LP, LD, and MD. This indicates a

systematic difference for these regions.

4.3.2.3 *Stressed volume*

Increases in porosity and pre-load damage should increase the volume of cement experiencing higher stress. This would then drive the stress-dependent damage rule to cause greater damage accumulation for highly porous pre-damaged regions. Integration points had their volume summed to a given stress range. Ranges were divided into 1 MPa increments; e.g. a point experiencing a 0.5 MPa stress would have its volume added to the 0–1 MPa range. By comparing these stress ranges for the first cycle of loading to predicted damage accumulation, it was found that volumes experiencing stresses greater than 9 MPa were indicative of predicted damage accumulation (Fig. 4.26). The relationship between volumes stressed above this threshold and damage accumulation was further demonstrated by plotting a regression line through all the data points (Fig. 4.27). Thus, the effect of porosity and pre-load damage is better understood by considering their interaction with stress.

4.3.3 **Migration**

Debonded prostheses migrated noticeably during loading while bonded prostheses did not (Fig. 4.28). This was true for both medial and distal displacements. Both components of displacement show variability. For the case of debonding, the deterministic model predicted migration within the variability of the stochastic specimens. For a bonded prosthesis, a deterministic simulation predicted a small increase in migration, but this was still negligible compared with debonded predictions.

Regression analysis of migration against damage accumulation showed poor correlation, even with the removal of an outlier (Fig. 4.29). Consideration of the final damage, i.e. test damage plus pre-load damage, improved the correlation a little but still required the removal of an outlier. However, when only the cement in region MD and the regions not exposed by the viewing windows were considered, correlation for both damage measures improved (Fig. 4.30). Debonded specimen number 4 appeared to act as an outlier in all cases and correlation improved considerably when it was removed from the population sample.

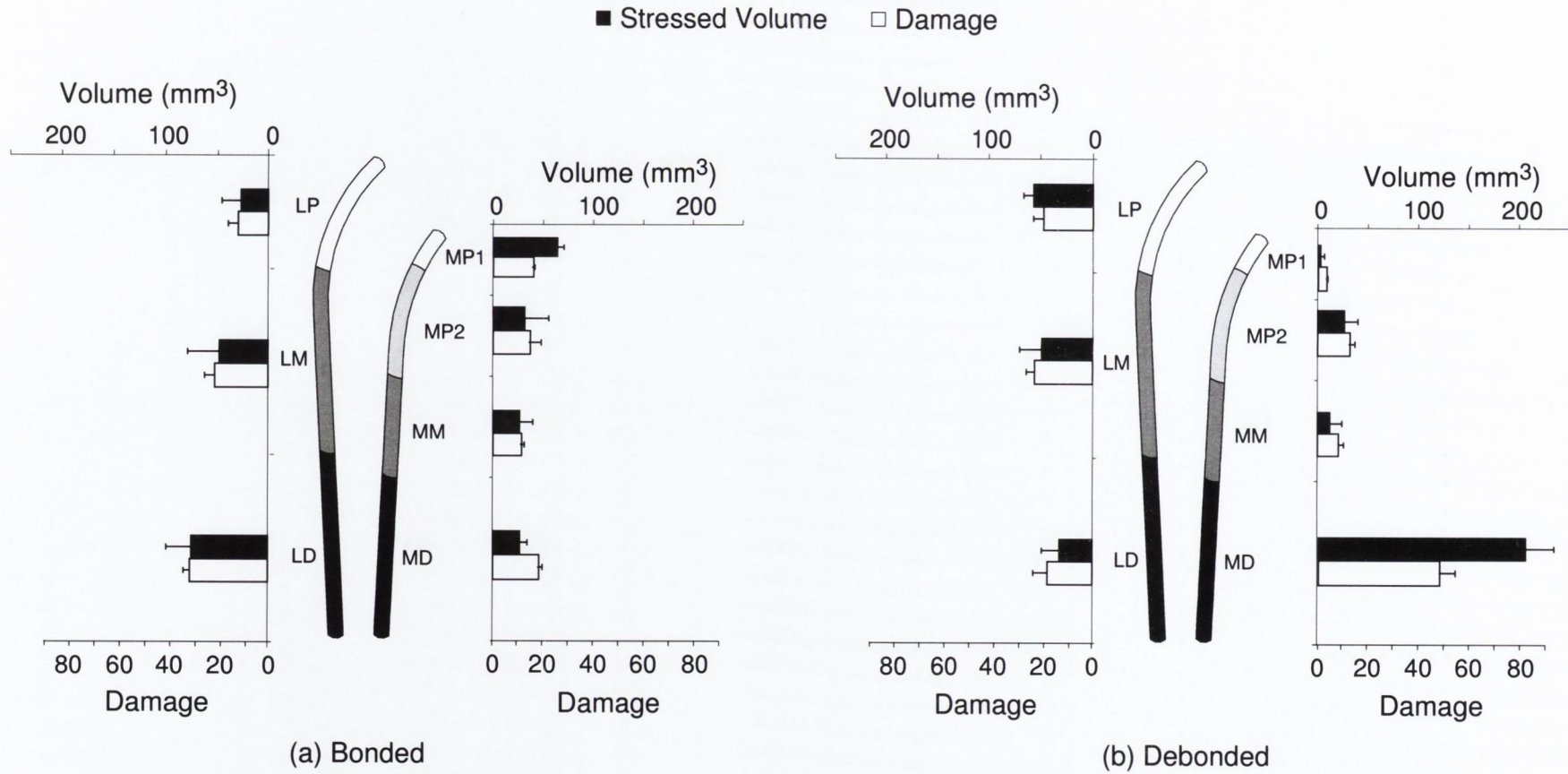


Figure 4.26. Stressed volume and damage growth patterns: average volumes of cement stressed above 9 MPa were found to indicate damage growth patterns for each region. Error bars indicate one standard deviation.

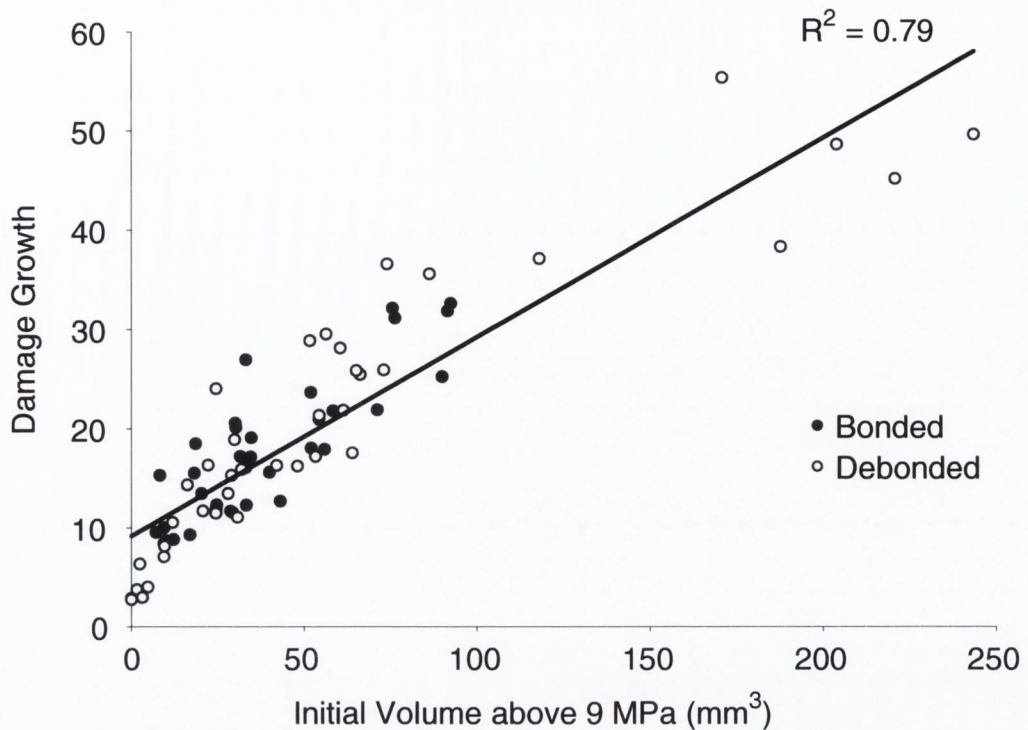


Figure 4.27. *Damage growth vs. volumes stressed above 9 MPa. The different regions for both specimen types were combined into one population sample and the regression line was fitted through all points.*

4.4 Recapitulation of main results

1. (a) The model of stochastic damage accumulation predicts similar average lifetimes and variability in lifetime for uniaxial tension experiments.
- (b) The model predicts similar differences between hand-mixed and vacuum mixed bone cement specimens.
- (c) Minimum lifetimes and range of lifetime were predicted close to the experimental results for almost all stress levels for both mixing methods.
2. (a) Experimental measurements of damage accumulation for the hip replacement model found pre-load (shrinkage) damage in addition to damage accumulation during testing.
- (b) For both polished surface and matt surface prostheses, damage was found to be distributed throughout the specimen and considerable variability was observed.
- (c) Displacement of the actuator of the materials testing machine at peak load showed that the prosthesis migrated during testing. This migration

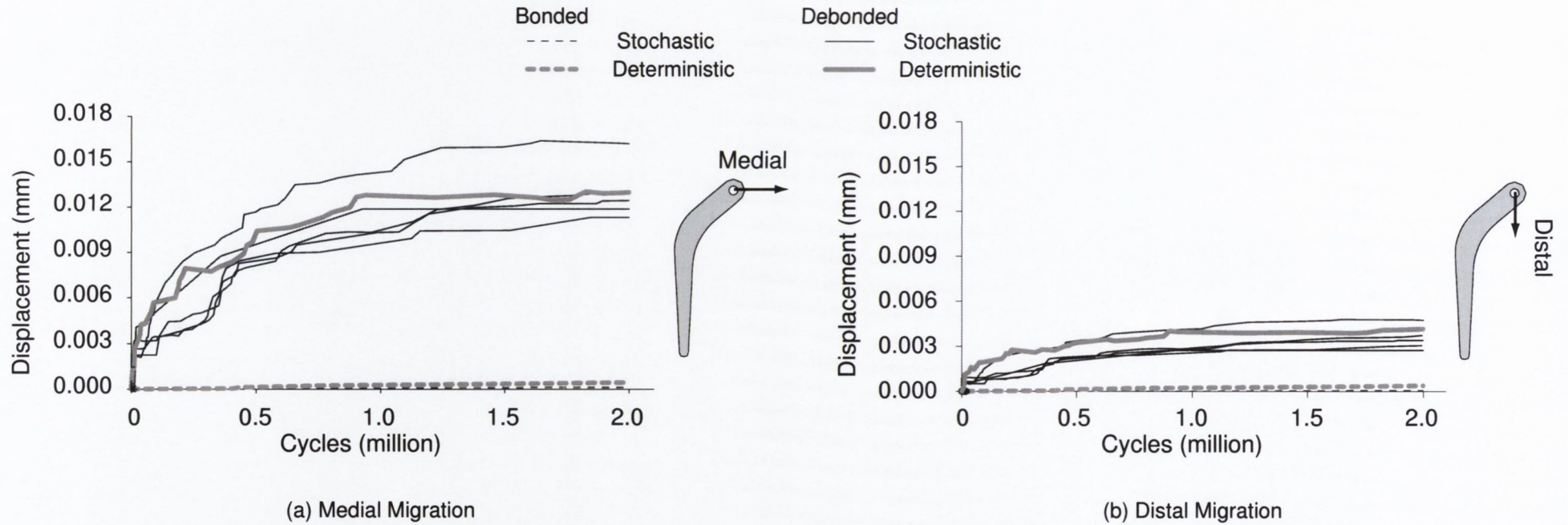


Figure 4.28. Migration over duration of test for each specimen for (a) medial and (b) distal directions. Labels: B = bonded, D = debonded, and digit = specimen number. Only debonded specimens were predicted to migrate noticeably during testing.

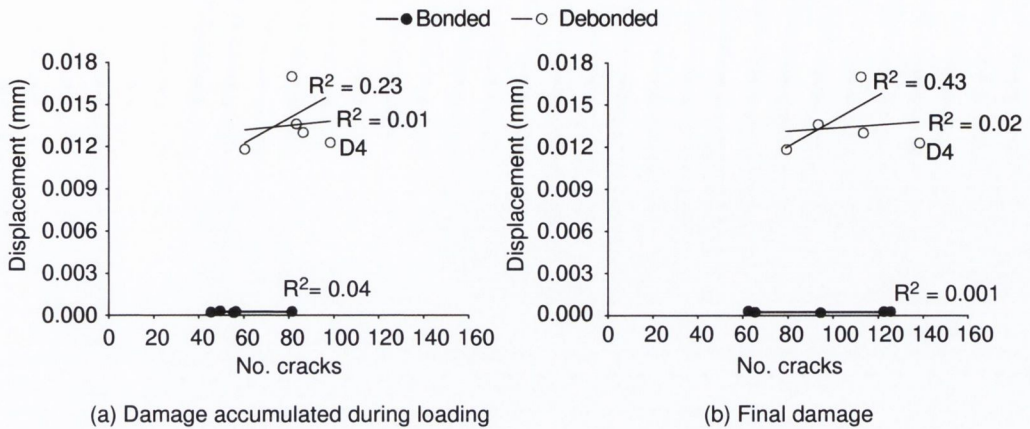


Figure 4.29. Dependence of resultant migration at end of test on damage accumulation: (a) for damage accumulated during loading and (b) for final damage at end of test, i.e. pre-load damage and test damage. Only cracked integration points were considered because of the mode of damage coupling used, i.e. stiffness loss only when damage completed. Debonded specimen number 4 (D_4) was seen to act as an outlier.

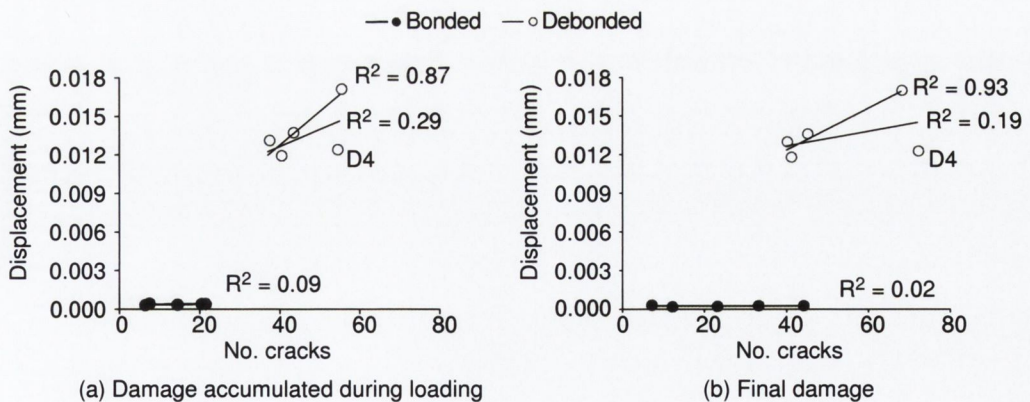


Figure 4.30. Dependence of resultant migration at end of test on damage accumulation in the cement surrounding the distal tip and in the most proximal regions. (a) for damage accumulated during loading and (b) for final damage at end of test. Debonded specimen number 4 (D_4) was seen to act as an outlier.

was more variable for polished specimens.

- (d) Inducible displacements (cyclic displacements) were found to vary more among polished specimens than matt specimens. Inducible displacement for polished specimens showed sensitivity to total accumulated damage.
3. (a) Monte Carlo simulations of porosity were found to predict damage distributed throughout the cement both prior to and during loading.
- (b) Volumes of cement stressed above a threshold stress (9 MPa in this case) were found to indicate relative levels of damage accumulation between bonded and debonded prostheses.
- (c) Displacements of the prosthesis head were found to migrate for debonded specimens and were most likely dependent on damage accumulation around the tip of the prosthesis and in the most proximal medial and lateral regions.
- (d) Bonded specimens showed virtually no migration. Any displacements that did occur showed negligible sensitivity to accumulated damage.
4. Deterministic simulations predicted localised damage accumulation whereas stochastic simulations that included pores predicted disperse damage accumulation. Differences were most evident when damage was measured as the number of cracks accumulated. The localised damage accumulation predicted by the deterministic simulations was not observed in the experimental data. Therefore, there may be good reason to reject deterministic modelling for this application.

Chapter 5

DISCUSSION

Contents

5.1	Introduction	120
5.2	Limitations	120
5.2.1	Damage rule, damage variable, and constitutive model	120
5.2.2	Porosity	123
5.2.3	Viscoelasticity	124
5.2.4	Interfaces	125
5.2.5	Finite element implementation	126
5.3	Assessment of the modelling approach	127
5.3.1	Confirmation of the model	127
5.3.2	Deterministic approach vs. stochastic approach	131
5.3.3	Implications for behaviour of the structure	135
5.4	Relevance to cemented fixation of orthopaedic implants	138
5.4.1	Use of migration as an assessment of fixation	138
5.4.2	Modelling of cemented fixation	139
5.5	Summary and perspectives	140

5.1 Introduction

In this thesis, the hypothesis that the random nature of bone cement porosity could explain the variable failure of cement-fixated orthopaedic implants is proposed. Since orthopaedic implants fail by damage accumulation, a computational scheme was developed to investigate damage initiation and growth in porous bone cement. Experimental studies were also carried out, but mainly to confirm that the computational scheme gave satisfactory results. The strength or weakness of the thesis stands on the predictive power of the computational tool. If that can be established, then a comparison of its prediction vis-à-vis the deterministic approach will give the evidence to accept or reject the hypothesised explanation of variability in orthopaedic implant longevity.

5.2 Limitations

The proposed computational scheme for modelling damage accumulation attempts to incorporate some important physical features of damage accumulation in acrylic bone cement. Nevertheless, the model remains a simplification. Before assessing results, a number of limitations must be considered.

5.2.1 Damage rule, damage variable, and constitutive model

First, the nonlinear fatigue damage rule was written in terms of maximum stress and assumes damage occurs only in tension. This is a significant simplification since it implies that fully reversed loading conditions will only affect damage accumulation during the tensile half of a cycle. As such, the model ignores the possibility that compressive stresses could act to resharpen a blunted crack in a way that would decrease fatigue life. An improvement in the model would be to incorporate stress amplitude and mean stress, e.g. as in the damage rule proposed by Chaboche and Lesne (1988).

A further assumption relating to stress was that local stress at integration points was used to drive damage accumulation while Murphy and Prendergast (2002) derived the damage rule in terms of the nominal stress for a relatively large region.

One way to investigate the effect of this would be to use a nonlocal definition of stress. Stolk et al. (1999) have shown that a benefit of such an implementation would be a decreased sensitivity to mesh discretisation.

This raises the point that crack interactions were not included in the damage rule. Thus, any predictions of the model are to be treated with caution when highly localised regions of damage appear. In reality, high densities of cracks are likely to cause interaction effects. It is also likely to make predictions mesh sensitive. It is unclear how the random distributions of porosity may affect such interactions. One reported effect of excluding localisation limiters is that damage tends to localise to the point of peak stress in a mesh. This was inhibited by the time dependency of damage growth and variable loading history in the current model. For example, any timestep will result in at least some damage for a given integration point, however small the stress it experiences from the current loading. If that integration point has sufficient damage from a previous loading, i.e. shrinkage in this case, then it may fail before points in the region of peak stress.

An important feature of the damage rule is the introduction of load-sequence effects in damage accumulation. It was shown that the form of load-sequence effects predicted by the model were shortened fatigue life for a low-high load sequence and increased fatigue life after a high-low (overload) sequence, see Fig. 3.3. However, Murphy and Prendergast (2002) did not perform multiple block testing, nor does any such data exist for bone cement in the literature, to the author's knowledge. Thus, the load-sequence effects predicted by the model cannot be validated at this time. Nevertheless, studies of fatigue crack propagation in PMMA have shown that overloads tend to retard crack growth due to an increase in size of the craze region (Imai et al., 1989). Also, notches are known to cause such fatigue retardation as a crack propagates from the high and localised stress environment of the notch root into the lower and more uniform stress state away from the notch (Hertzberg, 1996, Halford, 1997). Although not as severe as a notch, a pore could cause a similar effect—indeed, the bead-structure of the pore surface may approximate a notch quite well. Until data exists for multiple block fatigue loading of bone cement, these

questions remain unanswered.

In accounting for the multiaxial nature of damage, the uniaxial damage rule was simply applied independently to each tensile principal stress direction for a critical plane defined by the maximum principal stress for the cycle. Further investigation of these assumptions is needed by developing tests with nonproportional loads and rotating principal stress directions. A more advanced damage rule could be based on a combination of the invariants of the stress tensor to account for complex stress states. However, the brittle behaviour of bone cement at typical service loads makes the direct extrapolation of the uniaxial rule a reasonable first approximation.

Coupling of damage with elastic properties was assumed to occur only on achieving a critical rupture value. This was supported by experimental measurements showing little evidence of stiffness reduction during testing of uniaxial specimens (Fig. 3.4) and, therefore, the evolution of stiffness loss is likely to be highly nonlinear, with noticeable changes only occurring near failure. A change of variable in the damage growth rule has been suggested by Chaboche and Lesne (1988) to allow such a nonlinear continuous coupling for fatigue. However, it has been shown by Paas et al. (1990) that uncoupling of damage growth from stiffness in the manner used in the present study produces acceptable results when the relationship is highly nonlinear.

A further issue with regard to coupling is the form of the constitutive relationship. It was assumed that a second order damage tensor combined with a strain energy equivalence principle can represent the damaged state for bone cement. Orthotropic stiffness reduction is the greatest degree of anisotropy implied by this form of coupling. However, Kachanov (1992) has suggested that orthotropic damage is often an acceptable simplification for non-interacting cracks in an isotropic matrix. Alternatives to the use of strain energy equivalence in forming the constitutive relationship also exist. However, the uncoupled relationship between damage and stiffness until failure makes differences between models less apparent as the problem has simplified to the limiting case of full stiffness loss for a given plane. A criticism of the crack closure model used is that no shear stiffness is regained upon closing.

This neglects any possible frictional effects that may occur from closed and sliding crack faces. An improved framework for incorporating such effects and investigating greater degrees of anisotropy would be to use a micromechanical approach. As a considerable amount of physical crack measurements exist from the development of the damage rule (Murphy and Prendergast, 2002), an improved relationship could be developed using a crack density parameter and a statistical model of crack orientation generated from the crack data. Several approaches exist in the literature to achieve such a model, e.g. Horii and Nemat-Nasser (1983), Kachanov (1992), and Krajcinovic (1996).

These issues highlight the need for some form of validation of the constitutive model. A full field optical technique, e.g. digital speckle pattern interferometry (DSPI) or grating (Moiré) interferometry, is likely to provide the best means of assessing the intricate strain distributions arising in damaged regions. The author has performed some preliminary investigations on this (Lennon et al., 1999)¹.

5.2.2 Porosity

Considerations for the constitutive model of damage apply equally for the constitutive model of porosity. Unlike damage, porosity was coupled with elastic properties for all values of the volume fraction of pores at a given point. Thus, the need for some validation of stress-strain behaviour is potentially more important. The implementation of porosity also represents a paradox—i.e. the measurement scale for porosity could be viewed as violating the continuum assumption for the element sizes used in the study. This was highlighted by the need to include a procedure to allow a pore to extend outside of the region attributed to a single integration point. In addition to this, the constitutive model only accounts for an average stress over a representative volume element. Analytical solutions for the stress raising effects of voids predict complex stress states that cannot be accurately captured with the current approach (Timoshenko and Goodier, 1970, Tsukrov and Kachanov, 1997). In spite of these limitations, the model can account for stiffness loss and stress raising due to pores and provided qualitatively similar pore distributions for both

¹See Appendix A, pp. 179

hand-mixed and vacuum-mixed bone cements.

One criticism of the simulations of failure in hand-mixed cement uniaxial specimens is that the average lifetimes at two stresses, 17.5 and 21 MPa, are lower than the experimental results (Fig. 4.9 and Table 4.3). This may be due to the absence of occasional very large pores in the hand-mixed pore generation input. Occasional large pores were predicted to result in large variability and increase average fatigue life for the vacuum-mixed specimens, suggesting that hand-mixed cement may be better represented by a lower average porosity with large numbers of small pores and occasional large pores. Thus, a suggested improvement to the pore generation algorithm would be to use two separate distributions of small and large pores. Both distributions could then be varied independently to allow both small and large pore distributions to be specified for a particular mixing method. For example, the hand-mixed specimens could have mostly small pores with the occasional large pore. Vacuum-mixed specimens could have almost no small pores and a few large pores. Another possible and associated criticism is that the use of normal distributions may not be the most suitable. The use of other distributions should be investigated.

A related limitation to the simulations is the absence of other inclusions or regional changes in fatigue strength. It is likely that factors such as polymer bead morphology, distributions of molecular weight in the interstitial polymer matrix, and radiopacifiers will also cause variation in material strength. Their incorporation could also prevent the unrealistic symmetric fracture patterns observed in the vacuum-mixed specimens when pores were not critical enough to dominate failure (Fig. 4.5).

5.2.3 Viscoelasticity

Perhaps the most notable limitation of the computational scheme is the absence of creep. This has several implications. First, creep of the cement would allow migration of the prosthesis, causing a redistribution of stress that would undoubtedly affect the simulated damage accumulation. Second, pores could also be expected to interact with creep since creep is also a stress driven phenomenon. This could

cause variability to propagate, causing greater deviations from average behaviour. Third, damage will also affect creep due to the change in stress distributions as cracks initiate. Creep and damage should therefore be implemented as fully coupled phenomena, rather than a sequential coupling of creep prediction followed by damage prediction. A fourth issue brought about by creep would be the need to account for creep strain in determining crack closure.

5.2.4 Interfaces

In the simulations, the prosthesis-cement interface was modelled as either fully bonded or fully debonded. Thus, the model does not account for the time-course of prosthesis debonding from the cement. It could be argued that a polished prosthesis should debond completely over the duration of the test, making the assumption of debonding used in this study suitable only for long term lifetime prediction. The matt prosthesis would also be expected to debond in the longer term. Using an experimental model, Verdonschot and Huiskes (1998) have shown that matt prostheses can debond, and that damage accumulates near the interface due to abrasion. In cases of partial debonding, it is possible that more damaging stress states may arise during the debonding process, e.g. the interfacial crack front could increase local loading of the cement and lead to damage accumulation over wider areas as the crack front advances. Further investigation of this question would ultimately require the implementation of a fatigue failure model for the interface.

Porosity also affects the prosthesis-cement interface. Large numbers of pores have been observed at this interface and are most likely due to rheological properties of the doughy cement during prosthesis insertion (James et al., 1993, Bishop et al., 1996). A more sophisticated pore generation algorithm could be used to account for the greater pore density close to this interface. This would also act as an extra source of variability since its effect on interface failure would be likely to propagate through the coupled processes of creep and damage.

A limitation with considerable scope for further study is the representation of the cement-bone interface. Generally, it is modelled as a smooth and well defined

interface. However, in reality it is composed of interdigitated cement and trabecular bone (Maher and McCormack, 1999)—thus, it is not an interface in the usual sense of the word. This interdigitation acts as a site for crack initiation so that its absence from the computational model removes an important source of damage accumulation.

5.2.5 Finite element implementation

A finite element code was written by the author to implement the computational scheme. No validation of the code beyond the comparison with experimental results described in this thesis has been undertaken at this time. Simulations performed on benchmark problems would offer a better validation of the elastic and contact portions of the code.

A linear isoparametric hexahedral element was used for all analyses. This element is generally not suitable for capturing high strain gradients but can give good results when appropriate discretisation is applied to regions of higher gradients. Also, it is considerably less computationally expensive when used with material nonlinearities than higher order hexahedra. However, one known shortfall of this element is that it is too stiff in bending. Many commercial codes allow the augmentation of the shape functions for this element to allow so called ‘extra displacement shapes’. These can considerably improve the response of the element in bending. The current implementation of this element used in the modelling scheme does not employ such shape functions and therefore can be expected to underpredict bending deformation and stress.

The contact element used employed a penalty approach to enforce displacement continuity at the interface between two contacting elements. This approach is recognised as susceptible to ill-conditioning due to the sole reliance on interface stiffness. This usually results in convergence difficulties and is most prominent at high stiffness values. However, the accuracy of the prediction is dependent on the maintenance of as high a stiffness as possible since greater stiffness allows less penetration. The use of a line search algorithm considerably improved the ability to use higher stiffness co-

efficient. Nevertheless, the augmentation of the penalty method with a Lagrangian approach could be used to improve the accuracy of the element further.

A simplification used in the contact algorithm was the symmetrisation of friction. In general, different friction forces occur in the two tangential directions of the interface. When a Coulomb friction model is employed, i.e. stick-slip friction, the tangential stiffness components are varied to account for frictional sliding. This results in a nonsymmetric stiffness matrix. To avoid the considerable increase in computational cost associated with having to use nonsymmetric storage and solution strategies, the friction model was symmetrised based on the resultant frictional force.

5.3 Assessment of the modelling approach

The hypothesis put forward in this thesis is that modelling failure of bone cement requires a stochastic approach because a deterministic approach does not capture realistic damage accumulation. This hypothesis is investigated using a computational scheme that is first partially tuned to, and tested against, the fatigue data of Murphy and Prendergast (2000a). Predictions are compared to data from an experimental model with more complex structural features and loading, more similar to a cemented hip replacement. Predictions of both stochastic and deterministic models are then contrasted to illustrate the differing conclusions they each lead to. The implications of variability on structural behaviour of the experimental model are also discussed.

5.3.1 Confirmation of the model

5.3.1.1 Uniaxial fatigue

Several comparisons can be made between the predictions of the stochastic simulations and the experimental data of Murphy and Prendergast (2000a). The computational scheme proved capable of predicting:

1. similar averages and variability as shown by the correspondence between the predicted regression lines and the experimental regression lines (see Fig. 4.7),

2. equivalent ranges of failure life (see Fig. 4.8), and
3. the average increase in fatigue life for vacuum-mixed specimens over hand-mixed specimens (see Fig. 4.9).

The ability of the model to simulate the aforementioned phenomena does not necessarily increase our understanding of bone cement failure if the input bears no relationship to physical data. However, a comparison of the input porosity values with values found in the literature and some representative large pore radii taken from the study of Murphy and Prendergast (2000a) show that both hand-mixed and vacuum-mixed values fall within the range of values observed in the literature, see Table 5.1.

Due to the tuning procedure employed in generating the input parameters for the model, the comparison with experimental results cannot strictly be considered as a validation. However, the tuning was limited to simulating an approximately equivalent range in failure life for one stress level only, i.e. 21 MPa. In conclusion, the ability of pores to affect both range of fatigue life and average fatigue life of bone cement was demonstrated rather well.

5.3.1.2 *Pre-load damage and damage accumulation in a complex structure*

Under real service conditions, bone cement is subjected to a stress pattern that is continuously changing (e.g. bonded vs debonded prosthesis-cement interfaces) and is complex and intricate (due to the complex geometry and composite nature of the structure). Unlike the uniaxial fatigue specimens, in which tensile stress acted throughout the specimen, the more complex stress states of a cemented joint replacement can be expected to create complex damage accumulation patterns. The purpose of the experimental model was to provide a test of the computational scheme under more realistic loading and interfacial conditions.

Pre-load damage occurred in every region of the cement layer of the experimental models. Inclusion of porosity in the presence of shrinkage stress showed that similar distributed cracking could be predicted in the simulated models—compare pre-load cracking in Figs. 4.18 and 4.19 with the plots of Fig. 4.10. The experi-

Table 5.1. Comparison of mean porosity, mean pore radius (\bar{r}), and maximum pore radius (r_M) for hand-mixed and vacuum-mixed cements from several studies. Approximate values represent measurements taken from radiographs or fractographs presented in the studies. Hand-mixed values thus fall within the range of values observed in the literature. Vacuum-mixed porosity lies within the lower range of published values while mean pore radius lies in the higher range of peak radii. The average and maximum pore radius of Wixson et al. (1987) stands out as very low compared to other studies. This is most likely due to the fact that they measured pore radii at only two cross-sections of their specimens, thus decreasing the probability of finding a large pore.

	Hand-mixed			Vacuum-mixed		
	%	\bar{r} (mm)	r_M (mm)	%	\bar{r} (mm)	r_M (mm)
Input	5.00	0.25	—	0.20	1.75	—
Achieved	5.76	—	—	0.34	—	—
Murphy and Prendergast (2000a)	—	—	~1	—	—	0.75–1.5
Demarest et al. (1983)	4.8	—	—	1.2	—	—
Wixson et al. (1987)	7.2–9.4	—	1.5	0.1–0.8	0.015	0.075
Linden and Gillquist (1989)	10.0	—	—	0.5	—	—
Jasty et al. (1990)	9.39	0.24	1.66	—	—	—
Davies and Harris (1990)	—	—	—	—	—	~3.5
Kindt-Larsen et al. (1995)	5–15	—	5+	0.5–1.0	—	0.5–1.0
Wang et al. (1996)	4.8	—	—	0.1–1.3	—	~3
Lewis et al. (1997)	7.02	—	—	0.4	—	—

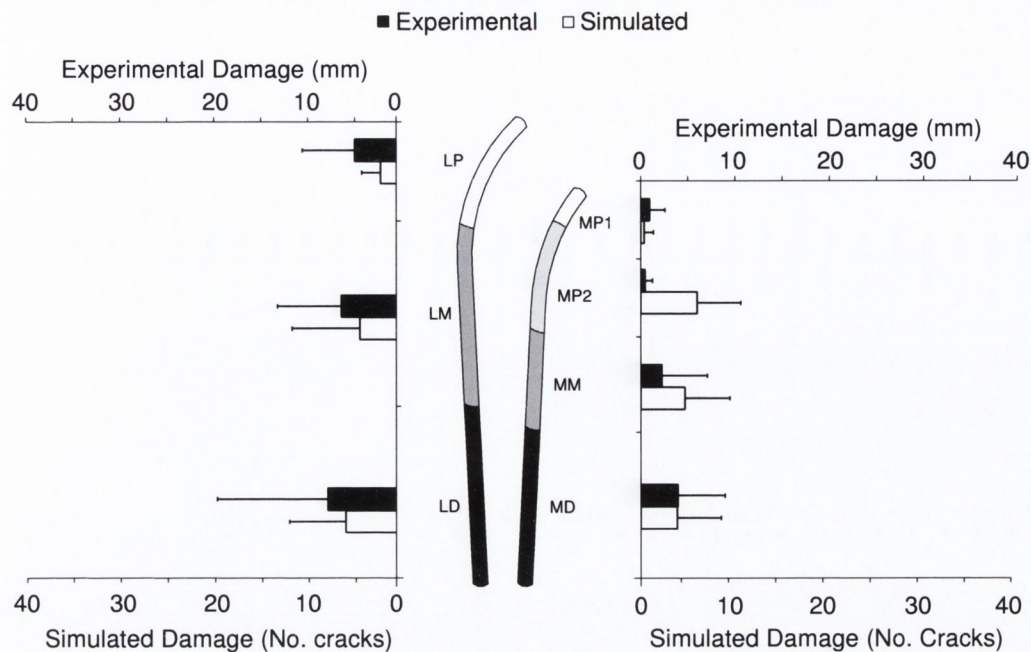


Figure 5.1. Comparison of experimental pre-load cracking with simulated pre-load cracks. Matt and polished specimens were combined to form an experimental population sample and bonded and debonded specimens were combined to form a simulated population sample. This was done because no significant difference was found between specimen types from experimental and simulated data sets, see Tables 4.4 and 4.5.

mental results also showed that no significant difference existed between matt and polished specimens (Table 4.4). This result was replicated in the simulated models when it was assumed that the shrinkage stress distribution for a bonded prosthesis applied to pre-load crack initiation. This suggests that both matt and polished specimens maintained intact interfaces prior to fatigue testing. A quantitative comparison of regional variation in pre-load damage for all stochastic specimens vs. all experimental specimens show similar trends for predicted pre-load cracking compared with measured pre-load cracking (Fig. 5.1). In making these comparisons it should be noted that the damage measure for the simulations does not correspond directly to the sum-of-crack-length measure of the experimental tests. Only one region, MP2, clearly deviated from the experimental trend. Limitations discussed in sections 5.2.1 and 5.2.2 could have contributed to this. Alternatively, the shrinkage stress prediction may overestimate the stress in this region. This could occur if some local debonding occurred at or near this region; an earlier study found the highest combination of tensile and shear interface stress in the most proximal portion of the

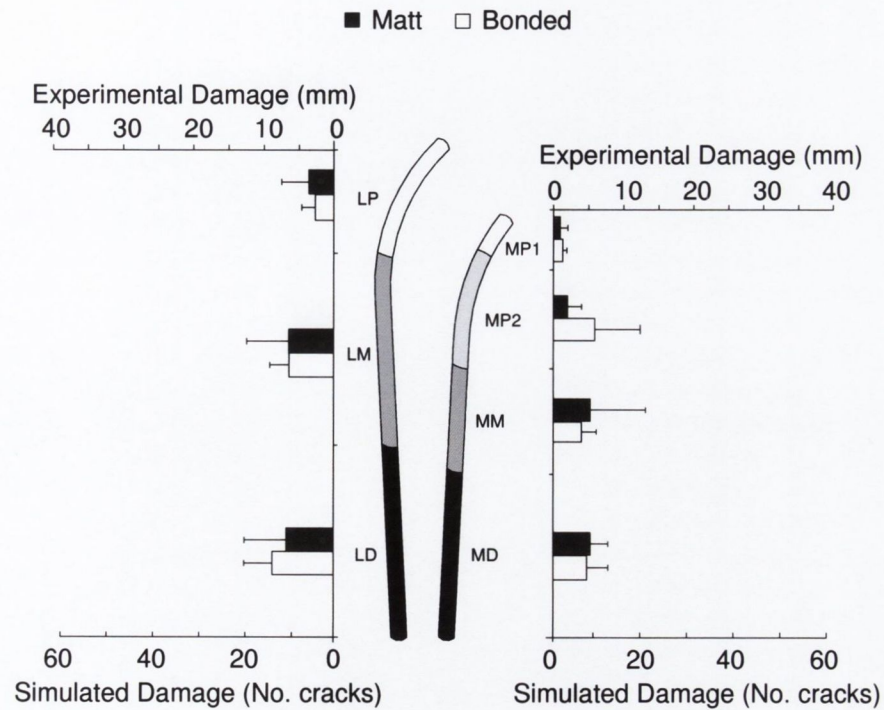
medial prosthesis-cement interface of this model (Lennon and Prendergast, 2002).

Damage in the cement was also found to accumulate in a distributed fashion during loading for the experimental specimens. This behaviour was replicated in the simulated specimens (Fig. 5.2). In general, predicted damage accumulation compares quite well with the experimental results. Only two regions showed clear deviation from the experimental trends: regions LM and LD of the polished specimens (Fig. 5.2b). These results suggest that the assumption of complete debonding for the duration of the test is incorrect for the polished specimens. A better assumption would be to include the evolution of debonding in simulations representing polished prostheses. In contrast, complete bonding appears to have been a reasonable assumption for the matt specimens.

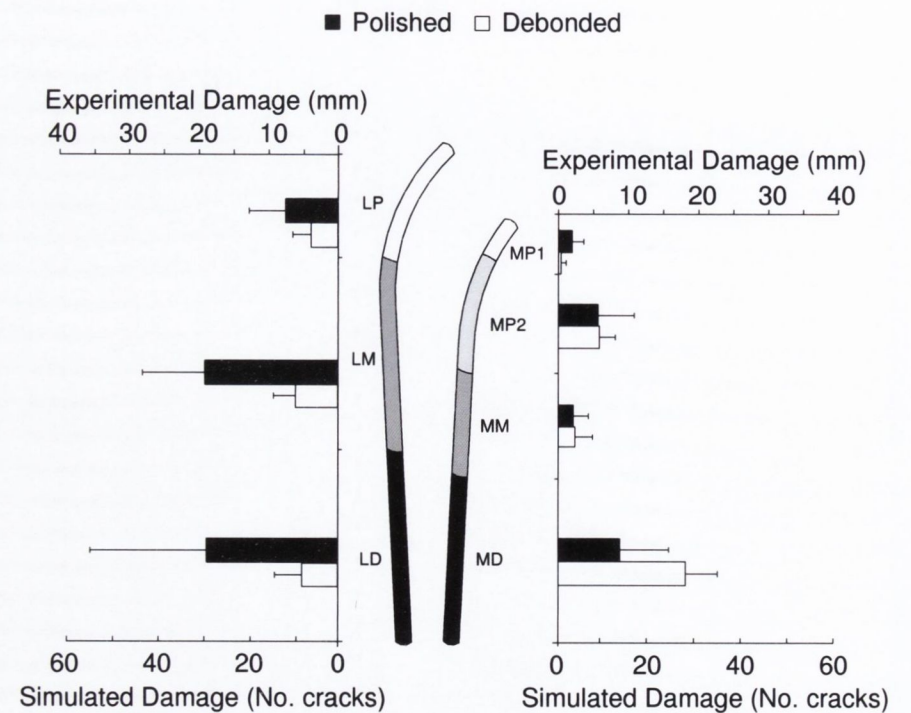
5.3.2 Deterministic approach vs. stochastic approach

The standard approach in deterministic fatigue models of bone cement has been to use the equation for a least squares regression line fit to fatigue data as input to a remaining-life damage rule, see e.g. Verdonschot and Huiskes (1997b), Colombi (2002a), Stolk et al. (2003). For a uniaxial test, such models predict the S-N curve that was used to generate them. Comparison of this prediction for the vacuum-mixed regression line at 21 MPa would result in an overprediction of approximately two orders of magnitude for the failure life of the earliest failing specimen (Fig. 5.3). Thus, if early failures are to be predicted, the use of a regression line fit to all the data is clearly unsuitable.

One could argue, therefore, that a regression line of minimum life should be used. This would avoid the complications and extra simulations of a stochastic approach. However, this would always result in damage accumulation from the point of nominal peak stress. That this does not usually occur has been shown by Davies et al. (1988); they found that for a set of fifteen notched uniaxial fatigue specimens, eleven specimens failed at a void rather than the notch. The stochastic approach used in this study predicted failure initiation and propagation outside the gauge length of the specimen in several cases (see Figs. 4.4 and 4.5). Even within



(a) Matt vs stochastic bonded predictions



(b) Polished vs stochastic debonded predictions

Figure 5.2. Comparison of damage accumulated during testing between (a) matt experimental specimens and bonded stochastic simulations and (b) polished experimental specimens and debonded stochastic specimens.

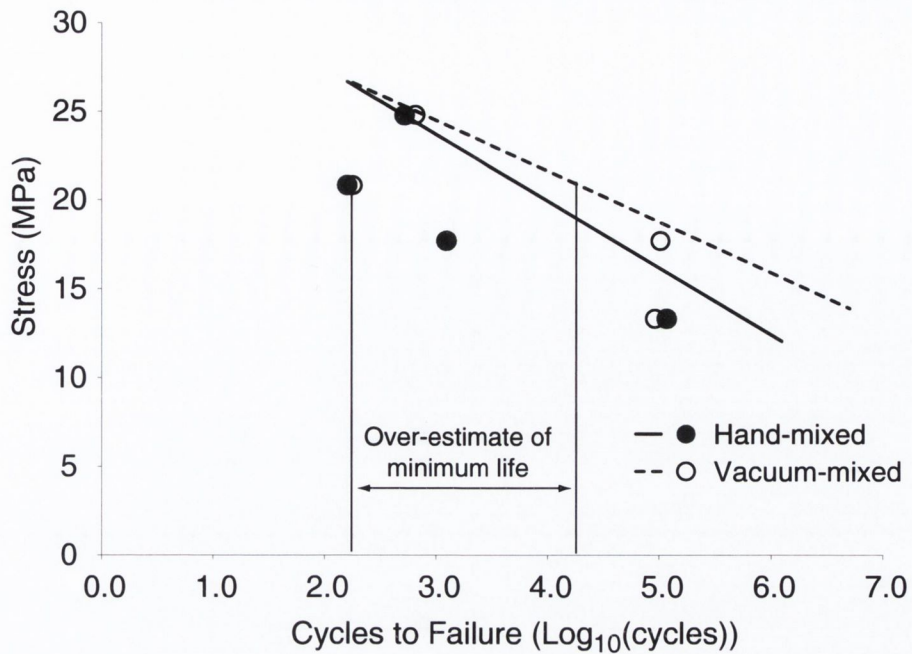
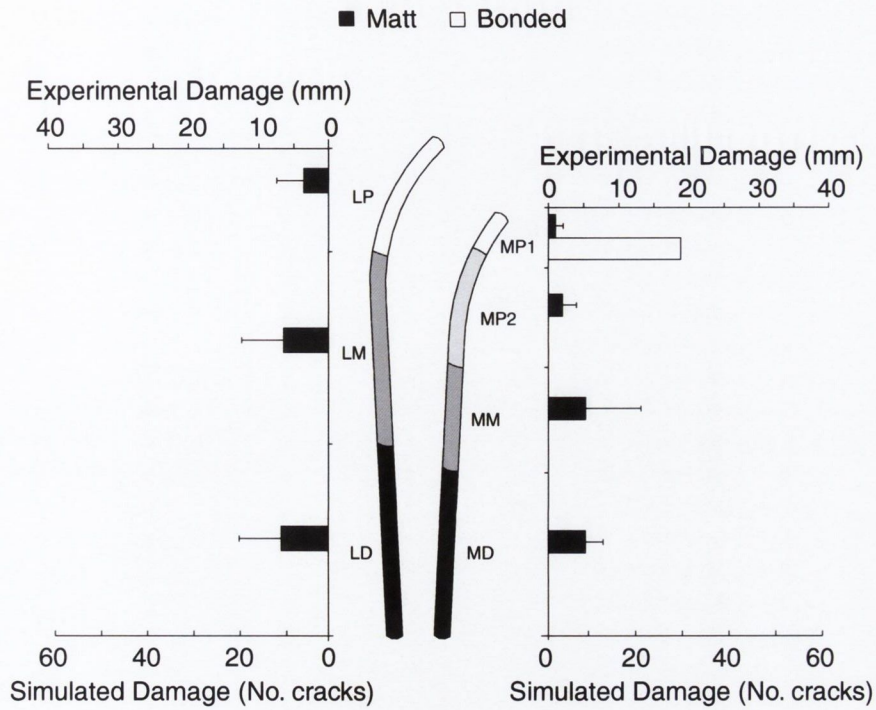


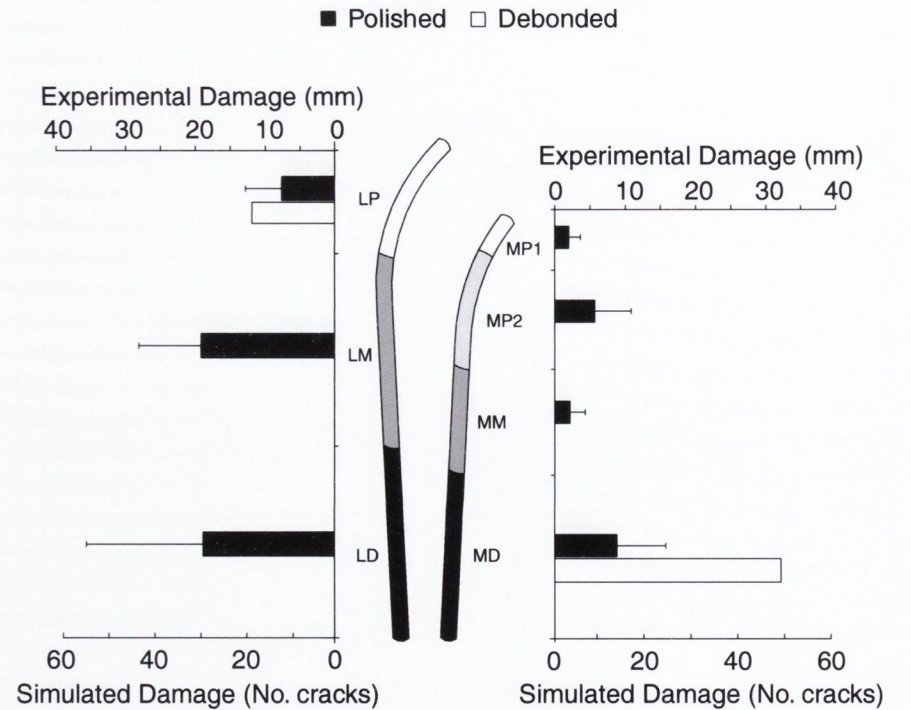
Figure 5.3. Comparison of minimum failure life at each stress level for hand-mixed specimens of Murphy and Prendergast (2000a) with vacuum-mixed specimens of the same study. The regression lines are the experimental regression lines reported by Murphy and Prendergast and represent the input for a deterministic damage model.

the gauge length, failure was only rarely predicted from the location of nominal peak stress, and this was mainly in the vacuum-mixed specimens. Multiaxial loading and intricate stress distributions in real joint replacements are likely to lead to more complex failure paths. For such cases, the inability of a deterministic model to account for variability in locations of damage accumulation may lead to errors in lifetime predictions.

Comparison of predictions of deterministic simulations with the experimental data clearly show the differences between the current stochastic approach and the deterministic approach that has previously been used to predict damage accumulation of bone cement (Fig. 5.4). The deterministic models could not account for the distributed nature of damage accumulation that was observed experimentally. Furthermore, it predicted greatest damage accumulation for the bonded specimens in region MP1, which was found in the experiments to have the lowest damage accumulation in both matt and polished specimens. In conclusion, the deterministic models were found to predict damage accumulation from regions of highest stress and resulted in much more localised damage accumulation. Such a damage



(a) Matt experimental specimens vs. deterministic bonded prediction



(b) Polished experimental specimens vs. deterministic debonded predictions

Figure 5.4. Comparison of experimental damage accumulated during testing with predictions of deterministic simulations: (a) matt experimental specimens vs. bonded deterministic simulations and (b) polished experimental specimens vs. debonded deterministic specimens.

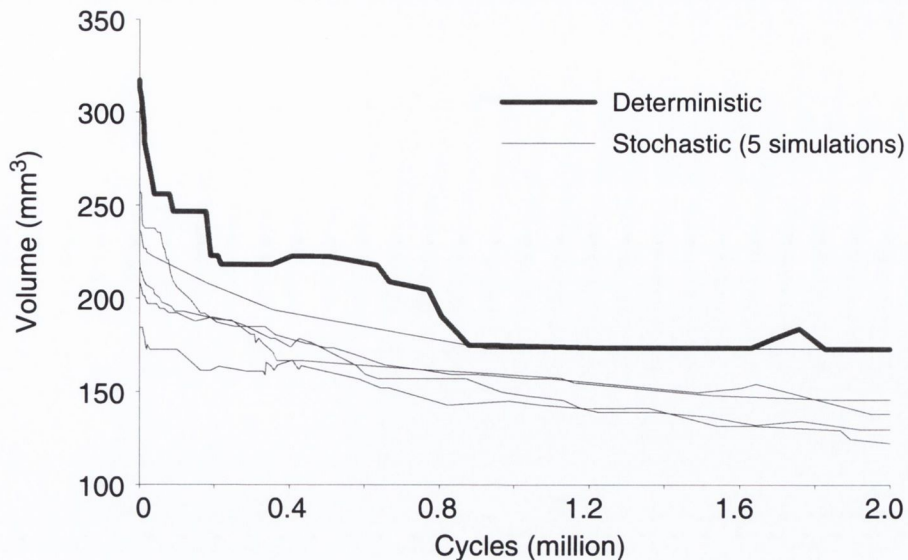


Figure 5.5. Change in stressed volumes during testing for region MD of the debonded specimens. For the stochastic debonded specimens, D1–D5, the volumes above 9 MPa are shown while for the deterministic model, DD, the volume above 4 MPa is plotted. The difference in the threshold stress is due to the lower *S-N* curve used for the deterministic model. Both values were chosen based on the results reported in section 4.3.2.3

accumulation pattern is not supported by the experimental data.

5.3.3 Implications for behaviour of the structure

5.3.3.1 Damage tolerance

It is evident from the comparison of stochastic and deterministic predictions that localised cracking occurs in regions of high stress for the deterministic specimens whereas it does not occur in the stochastic specimens for the same regions. Thus, the distribution of damage over wider areas appears to result in a reduction of damage accumulation in the most highly stressed regions (e.g. compare stochastic and deterministic predictions of debonded prostheses for region MD in Fig. 4.21). This form of damage tolerance may be due to the redistribution of stress from cracking in regions outside these ‘hot spots’ of damage accumulation. To investigate this further, stressed volume for one such region, MD in the debonded simulations, was examined (Fig. 5.5). All specimens show an initial rapid drop in the volume of cement above the relevant threshold stress as stress peaks were relieved by cracking. However, the deterministic specimen shows a subtle difference in that plateaus appear in the evolution, i.e. the volume remains relatively constant for extended periods. The

stochastic specimens, with the exception of debonded specimen number 4, show a more continuous decrease in stressed volume. This supports the hypothesis that the region was being shielded by damage occurring elsewhere.

The evolution of damage accumulation also suggests the existence of more damage tolerant failure mechanisms in the stochastic specimens. The three factors of porosity, residual stress, and pre-load damage result in a distribution of material with varying fatigue strength. Upon the application of loading, many regions are close to rupture and fail within a much shorter period than would be expected of a homogeneous cement subjected to the same loading (see initial rapid burst of damage growth in stochastic specimens of Fig. 4.23a). Relief and redistribution of peak stresses due to this sudden burst of damage accumulation over wide regions are likely to cause the subsequently observed decrease in damage accumulation rate. In the deterministic specimens, no cracking occurs outside the localised region to allow a shielding redistribution of stress. Thus, crack growth may continue at a greater rate than for the stochastic specimens in the longer term (indicated by the longer time to achieve steady state damage growth in Fig. 4.23a). These two types of behaviour are reminiscent of the parallel bar model of Krajcinovic (1996), reviewed in section 2.4.2.3. The introduction of wide bandwidths for these models resulted in much earlier damage accumulation that evolved in a more stable fashion than the bundles with almost identical failure strength.

While these predictions suggest that porosity may lead to damage tolerance, it should be noted that the porous specimens still contain higher amounts of damage when account is taken of pre-load damage. As the test is relatively short compared with the typical service life of a joint replacement, it is uncertain whether or not the higher overall damage of the stochastic specimens would result in an earlier failure. An interesting experimental observation that may relate to damage tolerance introduced by porosity was noted by Murphy and Prendergast (2000a). They found, using a Weibull analysis, that hand-mixed specimens were statistically more reliable than vacuum-mixed specimens, even though hand-mixed cement had more pores.

5.3.3.2 *Load-deflection behaviour of the structure*

Continuum damage models are often used to investigate the reduction in stiffness and load carrying capacity of materials. This can often be observed as increased deflections in response to constant loading amplitudes. However, when the damaging material is only one component of a more complex structure, the effect of damage on load sharing is more subtle. This was evident from the predictions that bonded prostheses migrated negligibly under peak load while debonded prostheses migrated by a comparatively large amount. This highlights the importance of interfaces for load transfer between individual materials for this kind of composite structure.

In contrast to the computer predictions, both matt and polished prostheses were observed to migrate to similar levels in the experimental tests. Creep of the cement is the most likely phenomenon to have caused similar migration for both matt and polished prostheses. However, creep does not explain why polished prostheses exhibited greater variability. As creep can be effectively neglected when considering displacement over a single cycle, the inducible displacement measurements offered insight into the potential relationship between damage and migration. A more erratic evolution of inducible displacements also suggested that they may be more strongly related to damage accumulation (Fig. 4.16). For the polished prostheses, it was found that inducible displacement at the end of testing could be correlated with the total amount of measured damage (Fig. 4.17). Matt prostheses showed a trend of comparatively little sensitivity to damage, although fewer matt specimens were available for this comparison. Thus, the inducible displacement measurements correspond better with the computational predictions than the total migration measurements since predictions showed that debonding was necessary to cause observable migration from damage accumulation (Figs. 4.29 and 4.30). This reinforces the earlier conclusion (section 5.3.1.2) that matt specimens remain bonded during testing and partial debonding occurs for polished prostheses.

5.4 Relevance to cemented fixation of orthopaedic implants

Ultimately, engineers wish to increase the longevity of cemented fixation of joint replacements by improving their design. As such, a number of conclusions can be drawn from the predictions of the computational scheme with respect to cemented fixation of prostheses, in particular the hip joint prosthesis.

5.4.1 Use of migration as an assessment of fixation

Lack of access to the components of a joint replacement that is in use has required the use of non invasive means of assessing the condition of the fixation. Much attention has been focussed on the assumption that failure of the materials and interfaces result in loosening of the prosthesis and that this is manifested by the observation of increased migration of the prosthesis on X-rays. The present study suggests a relatively minor role for damage accumulation in migration in comparison to interfacial failure and creep. The likely influence of creep is most evident from the fact that predicted migrations were low (distal components of 3–5 μm) in comparison to clinical and experimental measurements of migration. For example, Kärrholm et al. (2000) found distal subsidence of several different prosthesis designs ranging from approximately 50 μm to over 1 mm at two years for successful hip replacements. In an experimental test conducted for the same duration of two million cycles and in environmental conditions similar to the present study, i.e. conducted in air at room temperature, Maher and Prendergast (2002) found average distal subsidence of 43 μm and 113 μm , respectively, for Lubinus SPII and Müller prostheses. The inability of damage accumulation alone to explain clinical migration rates has also been noted by Verdonschot and Huiskes (1997b) and Colombi (2002a). In spite of the fact that damage accumulation plays little part in migration of the prosthesis, the inducible displacement results suggest that damage does affect loosening. Furthermore, they show that other factors, such as prosthesis-cement debonding influence inducible displacement. Some clinical evidence now exists to support a link between damage accumulation and inducible displacement—Cristofolini et al. (2002) have recently measured increased damage accumulation and inducible displacement for Müller

prostheses compared with Lubinus SPII prostheses in a similar test to that of Maher and Prendergast (2002). In conclusion, inducible displacement measurements are to be recommended over absolute migration measurements in assessing the condition of cemented fixations.

5.4.2 Modelling of cemented fixation

The present study supports the hypothesis that damage accumulation occurs from the moment of implantation rather than subsequent to loading of the joint. As such, the estimation of shrinkage stresses for the assessment of pre-load cracking behaviour is recommended in the analysis of cemented fixations. However, the need to include shrinkage stress in subsequent calculations may not be as necessary. This is due to considerably accelerated stress relaxation at body temperature in comparison to the temperature of the tests conducted in the present study. However, if any loading of the joint is likely to occur within the first 1–2 weeks it may be necessary to include shrinkage stress.

If realistic distributed damage accumulation in cemented fixations is to be simulated, then the inclusion of porosity, or some other mechanism of generating variation in the distribution of fatigue strength, must be recommended based on the results of the present study. Moreover, if the likely range in survival rate is the goal of the analyst, Monte Carlo simulations should also be carried out to assess the sensitivity to porosity and pre-load damage.

Time and computational resources may not always be sufficient to implement stochastic modelling for some analysts. Given the relationship that was predicted between stressed volumes and damage accumulation, useful information may still be obtained from linear elastic analyses. In particular, the reporting of stressed volumes for similar regions as were used for the experimental model of the present study would offer a useful quantitative comparison between cement stresses around different prosthesis designs. This would also avoid the often erroneous conclusions drawn from reporting only stress peaks, which may often be due to singularities arising in the mesh discretisation of the joint replacement and host bone (Lennon

and Prendergast, 2001)². However, some method of accounting for the greater size of cement layers between different implantations would be needed to make such reporting truly comparative.

The relationship between inducible displacement, damage accumulation, and interfacial bonding suggest the interactive nature of damage accumulation. By definition of its irreversibility, damage accumulation is a path dependent phenomenon. This suggests that its interaction with other processes could increase possible path bifurcations, resulting in increased variability; e.g. this may explain the outlying behaviour of one of the debonded specimens in the stochastic simulations. As such, the incorporation of all processes that could affect damage accumulation can be seen as a goal of a modelling scheme for predicting failure of cemented fixation. The stress driven nature of damage accumulation suggests that any process that is likely to cause stress redistribution should be seen as a candidate for inclusion in the computational scheme. In the present study, creep and the gradual debonding of the prosthesis-cement interface have been highlighted as two processes that can result in alteration of cement stress distributions. Another such process exists in real joint replacements. For most types of joint replacement, implantation of a cemented prosthesis alters the loading of the bone considerably. Bone is capable of adapting to this altered loading over time. Thus, the support provided by the bone to the cemented prosthesis will change over time, resulting in another source of cement stress redistribution. Although not a feature of bone cement itself, this can be considered another goal of a comprehensive model for predicting bone cement failure as used in cemented fixation of joint replacements.

5.5 Summary and perspectives

Throughout history, the development of new models for predicting material failure has been motivated by observation of new phenomena and failure modes in structures. Similarly, the development of the computational scheme of this study has been motivated by a desire to understand the very variable survival rates of

²Included in Appendix A, pp. 185

cemented joint replacements, as observed clinically.

It has been found that damage accumulation in bone cement is very variable in both spatial distribution and total damage accumulated. In this thesis, the author has used computational modelling to show that much of the variability in these features can be attributed to porosity introduced from mixing. Furthermore, it was shown that the difference in variability between hand-mixed and vacuum-mixed fatigue specimens of bone cement can be captured using porosity distributions representative of each mixing method.

Testing of the material under conditions that represent *in vivo* service conditions showed that damage can occur prior to any loading, due to residual stress caused by shrinkage. Furthermore, because porosity is random, pre-load damage shows considerable variability.

Testing of prostheses with different surface finishes offered a method of assessing the sensitivity of damage accumulation to prosthesis-cement debonding and the ability of the computational scheme to predict any changes caused by such debonding. Differences in damage accumulation could be observed experimentally between surface finishes, but the variability generally overwhelmed differences to render differences statistically insignificant. The computational model predicted qualitatively similar spatial damage distributions but variability was less than for the experimental specimens.

Results of the modelling scheme indicated that variability in porosity and pre-load damage can introduce damage tolerance in cemented joint replacements. Use of deterministic models, as proposed by Verdonschot and Huiskes (1997b), Colombi (2002a) and Stolk et al. (2003), did not predict realistic patterns of damage growth and did not provide a suitable framework for incorporating the type of pre-load damage observed experimentally. Therefore, stochastic models are a useful improvement on deterministic models for modelling failure of cemented joint replacements. Although deterministic models may predict a lower bound for failure in some cases, the path dependency of damage accumulation raises questions about the relationship of such a prediction to real failure modes.

Results of this study offer new insight into the process of damage accumulation in cemented joint replacements and signal new challenges for its modelling. The complexity of damage accumulation suggests that its coupling with other physical phenomena will increase variability even further. As greater variability is seen experimentally it is imperative that such features are incorporated into computer simulations if a better understanding of failure is to be achieved. Furthermore, application of the model in a simulated biological environment will open new avenues of interaction through the adaptation of bone to the redistribution of loading from creep and damage. These are considerable challenges to modelling, equally matched by the prospect of trying to complement investigation of these phenomena with experimental programmes. Nonetheless, an improvement in understanding attained from taking on these challenges can only benefit future development of joint replacement technology and represent a new insight for the orthopaedic field.

Chapter 6

CONCLUSIONS

Contents

6.1 Main conclusions	143
6.2 Conclusions relating to cemented hip replacement	143
6.3 Future work	144

6.1 Main conclusions

1. Use of purely deterministic modelling assumptions lead to an unrealistic conclusion of very localised failure for the cement layer of hip replacement prostheses.
2. Realistic distributions of damage accumulation are better explained by including pores as a source of variability in bone cement damage accumulation.

6.2 Conclusions relating to cemented hip replacement

- Damage accumulation begins before any external loading of a joint replacement and it can be attributed to crack initiation from pores in thermally shrinking cement.
- Damage accumulation is very variable throughout the cement layer and this variability can be simulated by including porosity, shrinkage stress, and pre-load damage.
- Small changes in implant design, surface finish in this case, can affect damage accumulation. Modelling can only predict resulting changes in behaviour if the complete path of damage accumulation is simulated, e.g. gradual interfacial failure simulated concurrently with bone cement damage accumulation.

- Debonding of the prosthesis is unlikely to occur prior to loading (i.e. it is unlikely to occur due to cement shrinkage).
- Migration of a prosthesis as damage accumulates is very sensitive to the interaction of damage accumulation with other processes, e.g. prosthesis debonding and cement creep, but not sensitive to damage accumulation alone.
- Inducible displacement provides a better indication of the influence of damage accumulation on prosthesis loosening than migration. This is because a migration measurement is likely to contain a considerable creep component.

6.3 Future work

- Other processes that can interact with cement damage accumulation could be included in the modelling scheme. The main candidates at this time are:
 1. fatigue failure of the prosthesis-cement interface,
 2. viscoelasticity of the bone cement, in particular creep, and
 3. bone remodelling due to the altered loading in the composite structure of the joint replacement.
- Experimental verification of the occurrence of load-sequence effects on fatigue life are required for the nonlinear damage rule. A programme of two-level tests could be used to achieve this. Sensitivity analysis of the influence of porosity on such a test could be used to design the experiment (i.e. determine suitable load sequences and the likely number of samples required to achieve a significant difference).
- Development of nonlocal damage variables would allow more robust analysis for cases of widely varying mesh densities and should be undertaken to minimise mesh sensitivity.
- The limited inclusion of stress relaxation and the absence of creep suggest a possible need to develop a general model of viscoelasticity coupled with damage

that can account for both creep and stress relaxation. A model with more basis in cement chemistry may also provide a consistent method of incorporating polymerisation induced stresses and ageing phenomena into the model.

- Simulations for prostheses for which clinical survival data is available should be performed to assess whether the modelling scheme can predict relative survival rates for different prosthesis designs. Important issues to be addressed in such a study are: (a) the inclusion of processes other than cement damage accumulation, (b) the use of realistic activities, including muscle load data, to account for possible load-sequence effects, and (c) determining a consistent criterion that best indicates the need for revision.
- Simulate cement damage accumulation in joint replacements other than hip replacement. Different modes of loading for other joints may produce new failure modes that would provide a more general test of the modelling scheme.

Bibliography

- Ahmed, A. M., Pak, W., Burke, D. L., and Miller, J., 1982a. Transient and residual stresses and displacements in self-curing bone cement—Part I: Characterization of relevant volumetric behaviour of bone cement. *Journal of Biomechanical Engineering, Transactions of the ASME*, **104**:21–27.
- Ahmed, A. M., Pak, W., Burke, D. L., and Miller, J., 1982b. Transient and residual stresses and displacements in self-curing bone cement—Part II: Thermoelastic analysis of the stem fixation system. *Journal of Biomechanical Engineering, Transactions of the ASME*, **104**:28–37.
- Ahmed, A. M., Raab, S., and Miller, J. E., 1984. Metal/cement interface strength in cemented stem fixation. *Journal of Orthopaedic Research*, **2**:105–118.
- Bathe, K. J., 1995. *Finite Element Procedures in Engineering*. Prentice Hall, 2nd edition.
- Bažant, Z. and Pijaudier-Cabot, G., 1988. Nonlocal continuum damage, localization instability and convergence. *Journal of Applied Mechanics*, **55**:287–293.
- Beer, G., 1985. An isoparametric joint/interface element for finite element analysis. *International Journal for Numerical Methods in Engineering*, **21**:585–600.
- Belytschko, T. and Lasry, D., 1988. Localization limiters and numerical strategies for strain-softening materials. In J. Mazars and Z. Bažant, eds., *Cracking and Damage—Strain Localization and Size Effect*, pages 349–362. France-US Workshop, Elsevier, London 1989, Cachan, France.
- Benallal, A., Billardon, R., and Geymonat, G., 1988. Some mathematical aspects of the damage softening rate problem. In J. Mazars and Z. Bažant,

- eds., *Cracking and Damage—Strain Localization and Size Effect*, pages 247–258. France-US Workshop, Elsevier, London 1989, Cachan, France.
- Bergmann, G., Graichen, F., and Rohlmann, A.**, 1993. Hip joint loading during walking and running, measured in two patients. *Journal of Biomechanics*, **26**(8):969–990.
- Bhattacharya, B. and Ellingwood, B.**, 1998. Continuum damage mechanics analysis of fatigue crack initiation. *International Journal of Fatigue*, **20**(9):631–639.
- Bhattacharya, B. and Ellingwood, B.**, 1999. A new CDM-based approach to structural deterioration. *International Journal of Solids and Structures*, **36**:1757–1779.
- Bishop, N. E., Ferguson, S., and Tepic, S.**, 1996. Porosity reduction in bone cement at the cement-stem interface. *Journal of Bone and Joint Surgery*, **78B**(3):349–356.
- Breysse, D.**, 1990. Probabilistic formulation of damage-evolution law of cementitious composites. *Journal of Engineering Mechanics*, **116**(7):1489–1510.
- Budiansky, B. and O’Connell, R. J.**, 1976. Elastic moduli of a cracked solid. *International Journal of Solids and Structures*, **12**:81–97.
- Burke, D. W., Gates, E. I., and Harris, W. H.**, 1984. Centrifugation as a method of improving tensile and fatigue properties of acrylic bone cement. *Journal of Bone and Joint Surgery*, **66-A**(8):1265–1273.
- Burke, D. W., O’Connor, D. O., Zalenski, E. B., Jasty, M., and Harris, W. H.**, 1991. Micromotion of cemented and uncemented femoral components. *Journal of Bone and Joint Surgery [Br]*, **73-B**:33–37.
- Chaboche, J. L.**, 1977. A differential law for non-linear cumulative damage. *Hors Serie 39*, (351):117–124.

- Chaboche, J. L.**, 1983. Le concept de contrainte effective appliqué à l'élasticité et à la viscoplasticité en présence d'un endommagement anisotrope. In J. P. Boehler, ed., *Colloques Internationaux du CNRS N° 295 — Comportement mécanique des solides anisotropes*, pages 761–774. Martinus Nijhoff, Boston.
- Chaboche, J. L.**, 1987. Continuum damage mechanics: present state and future trends. *Nuclear Engineering and Design*, **105**:19–33.
- Chaboche, J. L.**, 1988. Continuum damage mechanics: Part I—General concepts. *Journal of Applied Mechanics*, **55**:59–64.
- Chaboche, J. L.**, 1992. Damage induced anisotropy: On the difficulties associated with the active/passive unilateral condition. *International Journal of Damage Mechanics*, **1**:148–171.
- Chaboche, J. L.**, 1993. Development of Continuum Damage Mechanics for elastic solids sustaining anisotropic and unilateral damage. *International Journal of Damage Mechanics*, **2**:311–329.
- Chaboche, J. L.**, 1999. Thermodynamically founded CDM models for creep and other conditions. In H. J. Altenbach and J. J. Skrzypek, eds., *Creep and damage in materials and structures*, number 399 in International Centre for Mechanical Sciences (CISM) Courses and Lectures, pages 209–283. Springer-Verlag, Wien-New York.
- Chaboche, J. L. and Lesne, P. M.**, 1988. A non-linear continuous fatigue damage model. *Fatigue and Fracture of Engineering Materials and Structures*, **11**(1):1–17.
- Charnley, J.**, 1972. The long-term results of low-friction arthroplasty of the hip performed as a primary intervention. *The Journal of Bone and Joint Surgery*, **54** B(1):61–76.
- Charnley, J.**, 1979. *Low friction arthroplasty of the hip: Theory and practice*. Springer-Verlag, Berlin.

- Cheng, G. and Plumtree, A.**, 1998. A fatigue damage accumulation model based on continuum damage mechanics and ductility exhaustion. *International Journal of Fatigue*, **20**(7):495–501.
- Chudnovsky, A., Dolgopolsky, A., and Kachanov, M.**, 1987. Elastic interaction of a crack with a microcrack array—II. Elastic solution for two crack configurations (piecewise constant and linear approximations). *International Journal of Solids and Structures*, **23**(1):11–21.
- Colombi, P.**, 2002a. Fatigue analysis of cemented hip prosthesis: damage accumulation scenario and sensitivity analysis. *International Journal of Fatigue*, **24**:739–746.
- Colombi, P.**, 2002b. Fatigue analysis of cemented hip prosthesis: model definition and damage evolution algorithms. *International Journal of Fatigue*, **24**:895–901.
- Cook, R. D., Malkus, D. S., and Plesha, M. E.**, 1989. *Concepts and Applications of Finite Element Analysis*. John Wiley and Sons, 3rd edition.
- Cordebois, J. P. and Sidoroff, F.**, 1983. Damage induced elastic anisotropy. In J. P. Boehler, ed., *Colloques Internationaux du CNRS N° 295 — Comportement mécanique des solides anisotropes*, pages 761–774. Martinus Nijhoff, Boston.
- Costa Mattos, H. S. and Sampaio, R.**, 1995. Analysis of the fracture of brittle elastic materials using a continuum damage model. *Structural Engineering and Mechanics*, **3**(5):411–427.
- Cristofolini, L., Savigni, P., Teutonico, A. S., and Toni, A.**, 2002. Ex vivo and in vitro cement mantle fatigue damage around femoral stems: validation of a protocol to simulate real-life loading in hip replacement patients. *Acta of Bioengineering and Biomechanics.*, **4**(Suppl. 1):48.
- Culleton, T. P., Prendergast, P. J., and Taylor, D.**, 1993. Fatigue failure in the cement mantle of an artificial hip joint. *Clinical Materials*, **12**:95–102.

- Davies, J. P., Burke, W., O'Connor, D. O., and Harris, W. H., 1987. Comparison of the fatigue characteristics of centrifuged and uncentrifuged Simplex P bone cement. *Journal of Orthopaedic Research*, **5**(3):366–371.
- Davies, J. P. and Harris, W. H., 1990. Optimization and comparison of three vacuum mixing systems for porosity reduction of Simplex P cement. *Clinical Orthopaedics and Related Research*, pages 261–269.
- Davies, J. P., O'Connor, D. O., Burke, D. W., Jasty, M., and Harris, W. H., 1988. The effect of centrifugation on the fatigue life of bone cement in the presence of surface irregularities. *Clinical Orthopaedics and Related Research*, (229):156–161.
- Davison, L. and Stevens, A. L., 1973. Thermomechanical constitution of spalling elastic bodies. *Journal of Applied Physics*, **44**(2):668–674.
- Davy, K. W. and Braden, M., 1991. Residual monomer in acrylic polymers. *Biomaterials*, **12**(6):540–544.
- De-guang, S. and De-jun, W., 1998. A new multiaxial fatigue damage model based on the critical plane approach. *International Journal of Fatigue*, **20**:241–245.
- Demarest, V. A., Lautenschlager, E. P., and Wixson, R. L., 1983. Vacuum mixing of acrylic bone cement. In *9th Annual Meeting of the Society for Biomaterials*. Birmingham, Alabama.
- Diao, X., Ye, L., and Mai, T. W., 1997. Fatigue behaviour of CF/PEEK composite laminates made from commingled prepreg. Part II: statistical simulations. *Composites Part A*, **28A**:749–755.
- Dunne, N. J. and Orr, J. F., 2001. Influence of mixing techniques on the physical properties of acrylic bone cement. *Biomaterials*, **22**:1819–1826.
- Ebramzadeh, E., Mina-Araghi, M., Clarke, I. C., and Ashford, R., 1983. Loosening of well-cemented total-hip femoral prosthesis due to creep of the ce-

- ment. In A. C. Franker and G. D. Griffin, eds., *Corrosion and Degradation of Implant Materials*, pages 373–399. American Society for Testing and Materials, Philadelphia.
- Ellyin, F. and Golos, K.**, 1988. Multiaxial fatigue damage criterion. *Journal of Engineering Materials and Technology*, **110**:63–68.
- Eyerer, P. and Jin, R.**, 1986. Influence of mixing technique on some properties of PMMA bone cement. *Journal of Biomedical Materials Research*, **20**:1057–1094.
- Fatemi, A. and Yang, L.**, 1998. Cumulative fatigue damage and life prediction theories: a survey of the state of the art for homogeneous materials. *International Journal of Fatigue*, **20**(1):9–34.
- Fornasier, V., Wright, J., and Seligman, J.**, 1991. The histomorphological and morphometric study of asymptomatic hip arthroplasty. A postmortem study. *Clinical Orthopaedics and Related Research*, (271):272–282.
- Fornasier, V. L. and Cameron, H. U.**, 1976. The femoral stem/cement interface in total hip replacement. *Clinical Orthopaedics and Related Research*, **116**:248–252.
- Fowler, J. L., Gie, G. A., Lee, A. J., and Ling, R. S.**, 1988. Experience with the exeter total hip replacement since 1970. *Orthopaedic Clinics North America*, **19**:477–489.
- Freeman, M. A. R., Bradley, G. W., and Revell, P. A.**, 1982. Observations upon the interface between bone and polymethylmethacrylate cement. *The Journal of Bone and Joint Surgery*, **64-B**(4):489–493.
- Freitag, T. A. and Cannon, S. L.**, 1977. Fracture characteristics of acrylic bone cements. II. Fatigue. *Journal of Biomedical Materials Research*, **11**:609–624.
- Fritsch, E., Rupp, S., and Kaltenkirchen, N.**, 1996. Does vacuum-mixing improve the fatigue properties of high viscosity poly(methylmethacrylate) (PMMA)

- bone cement?. Comparison between two different evacuation methods. *Archives of Orthopaedic Trauma Surgery*, **115**:131–135.
- Germain, P., Nguyen, Q. S., and Suquet, P.**, 1983. Continuum thermodynamics. *Journal of Applied Mechanics*, **50**:1010–1020.
- Gilbert, J. L., Menis, D. W., Smith, S. M., Lautenschlager, E. P., and Wixson, R. W.**, 1990. Effect of pore size and morphology on fatigue crack initiation in acrylic bone cements. In *16th Annual Meeting of the Society for Biomaterials*, page 103. Charleston, South Carolina, USA.
- Ginebra, M. P., Albuixech, L., Fernández-Barragán, E., Aparicio, C., Gil, F. J., Román, J. S., Vázquez, B., and Planell, J. A.**, 2002. Mechanical performance of acrylic bone cements containing different radiopacifying agents. *Biomaterials*, **23**:1873–1882.
- Goldring, S. R., Jasty, M., Roelke, M. S., Rourke, C. M., Bringhurst, F. R., and Harris, W. H.**, 1986. Formation of a synovial-like membrane at the bone cement interface. Its role in bone resorption and implant loosening after total hip replacement. *Arthritis and Rheumatism*, **29**(7):836–841.
- Gordon, J. E.**, 1976. *The New Science of Strong Materials or Why You Don't Fall Through the Floor*. Penguin Books, London, England, 2nd edition.
- Gummert, P. R.**, 1999. General constitutive equations for simple and non-simple materials. In H. J. Altenbach and J. J. Skrzypek, eds., *Creep and Damage in Materials and Structures*, number 399 in International Centre for Mechanical Sciences (CISM) Courses and Lectures, pages 1–44. Springer-Verlag, Wien-New York.
- Halford, G. R.**, 1997. Cumulative fatigue damage modeling—crack nucleation and early growth. *International Journal of Fatigue*, **19**(Supp. No. 1):S253–S260.
- Hardinge, K.**, 1983. *Hip replacement. The facts*. Oxford University Press.

- Harris, W. H.**, 1991. The first 32 years of total hip arthroplasty. One surgeons perspective. *Clinical Orthopaedics and Related Research*, **274**:6–11.
- Hefzy, M. S. and Singh, S. P.**, 1997. Comparison between two techniques for modeling interface conditions in a porous coated hip endoprosthesis. *Medical Engineering and Physics*, **19**(1):50–62.
- Hertzberg, R. W.**, 1996. *Deformation and Fracture Mechanics of Engineering Materials*. John Wiley and Sons, 4th edition.
- Horii, H. and Nemat-Nasser, S.**, 1983. Overall moduli of solids with microcracks: load-induced anisotropy. *Journal of Mechanics and Physics of Solids*, **31**(2):155–171.
- Horii, H. and Nemat-Nasser, S.**, 1998. *Micromechanics: elastic properties of heterogeneous materials*. North-Holland.
- Huiskes, R.**, 1980. Some fundamental aspects of human joint replacement. *Acta Orthopaedica Scandinavica*. Suppl 185.
- Huiskes, R.**, 1993. Mechanical failure in total hip arthroplasty with cement. *Current Orthopaedics*, **7**:239–247.
- Huiskes, R. and Verdonshot, N.**, 1997. Biomechanics of artificial joints: The hip. In V. C. Mow and W. C. Hayes, eds., *Basic Orthopaedic Biomechanics*, pages 395–460. Lipincott-Raven, Philadelphia, 2nd edition.
- Imai, Y., Takase, T., and Nakano, K.**, 1989. Study of fatigue crack growth retardation due to overloads in polymethylmethacrylate. *Journal of Materials Science*, **24**:3289–3294.
- James, S. P., Schmalzried, T. P., McGarry, F. J., and Harris, W. H.**, 1993. Extensive porosity at the cement-femoral prosthesis interface: A preliminary study. *Journal of Biomedical Materials Research*, **27**:71–78.

- Jasty, M., Davies, J. P., O'Connor, D. O., Burke, D. W., Harrigan, T. P., and Harris, W. H.**, 1990. Porosity of various preparations of acrylic bone cements. *Clinical Orthopaedics and Related Research*, (259):122–129.
- Jasty, M., Maloney, W. J., Bragdon, C. R., O'Connor, D., Haire, T., and Harris, W. H.**, 1991. The initiation of failure in cemented femoral components of hip arthroplasties. *Journal of Bone and Joint Surgery*, **73-B**:551–558.
- Ju, J. W.**, 1990. Isotropic and anisotropic damage variables in continuum damage mechanics. *Journal of Engineering Mechanics*, **116**(12):2764–2770.
- Kachanov, M.**, 1987. Elastic solids with many cracks: a simple method of analysis. *International Journal of Solids and Structures*, **23**:23–43.
- Kachanov, M.**, 1992. Effective elastic properties of cracked solids: critical review of some basic concepts. *Applied Mechanics Review*, **45**(8):304–335.
- Kärrholm, J., Nivbrant, B., Thanner, J., Anderberg, C., Börlin, N., Herberts, P., and Malchau, H.**, 2000. Evaluation of hip implant design and surface finish. micromotion of cemented femoral stems. In *Scientific Exhibition presented at the 67th Annual Meeting of the American Academy of Orthopaedic Surgeons*. Orlando, USA.
- Kindt-Larsen, T., Smith, D. B., and Jensen, J. S.**, 1995. Innovations in acrylic bone cement and application equipment. *Journal of Applied Biomaterials*, **6**:75–83.
- Kine, B. B. and Novak, R. W.**, 1987. Acrylic and methacrylic ester polymers. In *Encyclopaedia of Polymer Science and Engineering*, volume 1, pages 234–299. John Wiley and Sons, 2nd edition.
- Krajcinovic, D.**, 1982. Statistical aspects of the continuous damage theory. *International Journal of Solids and Structures*, **18**(7):551–562.
- Krajcinovic, D.**, 1983. Constitutive equations for damaging materials. *Journal of Applied Mechanics*, **50**:355–360.

- Krajcinovic, D.**, 1996. *Damage Mechanics*, volume 41 of *North-Holland Series in Applied Mathematics and Mechanics*. Elsevier.
- Krajcinovic, D.**, 2000. Damage mechanics: accomplishments, trends and needs. *International Journal of Solids and Structures*, **37**:267–277.
- Krajcinovic, D. and Fonseka, G. U.**, 1981. The continuous damage theory of brittle materials. Part 1: General theory. *Journal of Applied Mechanics*, **48**:809–815.
- Krause, W. R. and Mathis, R. S.**, 1984. Fatigue properties of current acrylic bone cements. In *Second World Congress on Biomaterials. 10th Annual Meeting of the Society for Biomaterials*, page 349.
- Kühn, K. D.**, 2000. *Bone Cements*. Springer-Verlag.
- Kusy, R. P.**, 1978. Characterization of self-curing acrylic bone cements. *Journal of Biomedical Materials Research*, **12**:271–305.
- Lassen, T. and Sørensen, J. D.**, 2002. A probabilistic damage tolerance concept for welded joints. Part 1: data base and stochastic modeling. *Marine Structures*, page In Press.
- Lautenschlager, E. P., Wixson, R. L., Novak, M. A., and Bakir, N.**, 1986. Fatigue and fracture toughness of Simplex-P. In *32nd Annual Meeting of the Orthopaedics Research Society*.
- Lawn, B. R. and Marshall, D. B.**, 1998. Nonlinear stress-strain curves for solids containing closed cracks with friction. *Journal of Mechanics and Physics of Solids*, **46**(1):85–113.
- Laz, P. J. and Hillberry, B. M.**, 1998. Fatigue life prediction from inclusion initiated cracks. *International Journal of Fatigue*, **20**(4):263–270.
- Lee, A. J. C.**, 1994. Implants for fixation with and without a collar. In *Technical Principles, Design and Safety of Joint Implants*, pages 128–132. Hogrefe and Hubber, Bern.

- Lee, A. J. C., Ling, R. S. M., and Vangala, S. S., 1977. The mechanical properties of bone cements. *Journal of Medical Engineering and Technology*, pages 137–140.
- Lemaitre, J., 1985. A Continuous Damage Mechanics model for ductile fracture. *Journal of Engineering Materials and Technology*, **107**:83–89.
- Lemaitre, J., 2001. Petite histoire de l' experimentation en mecanique des solides. *Meccanica*, **36**:13–35.
- Lemaitre, J. and Chaboche, J. L., 1994. *Mechanics of Solid Materials*. Cambridge University Press.
- Lennon, A. B. and Prendergast, P. J., 2001. Evaluation of cement stresses in finite element analyses of cemented orthopaedic implants. *Journal of Biomechanical Engineering, Transactions of the ASME*, **123**:623–628.
- Lennon, A. B. and Prendergast, P. J., 2002. Residual stress due to curing can initiate damage in porous bone cement: experimental and theoretical evidence. *Journal of Biomechanics*, **35**:311–321.
- Lennon, A. B., Prendergast, P. J., Whelan, M. P., and Forno, C., 1999. Use of grating interferometry for validation of finite element models and to investigate residual strain in PMMA. In J. Middleton, M. L. Jones, N. G. Shrive, and G. N. Pande, eds., *Computer Methods in Biomechanics and Biomedical Engineering – 3*, pages 75–80. Gordon and Breach Science Publishers.
- Lennon, A. B., Prendergast, P. J., Whelan, M. P., Kenny, R. P., and Cavalli, C., 2000. Modelling of temperature history and residual stress generation due to curing in polymethylmethacrylate. In P. J. Prendergast, T. C. Lee, and A. J. Carr, eds., *12th Conference of the European Society of Biomechanics*, page 253. European Society of Biomechanics, Royal Academy of Medicine in Ireland, Dublin, Ireland.

- Lewis, G.**, 1999. Effect of two variables on the fatigue performance of acrylic bone cement: mixing method and viscosity. *Bio-Medical Materials and Engineering*, **9**:197–207.
- Lewis, G. and Austin, G. E.**, 1994. Mechanical properties of vacuum-mixed acrylic bone cement. *Journal of Applied Biomaterials*, **5**:307–314.
- Lewis, G., Nyman, J. S., and Trieu, H. H.**, 1997. Effect of mixing method on selected properties of acrylic bone cement. *Journal of Biomedical Materials Research*, **38**:221–228.
- Linden, U.**, 1989. Fatigue properties of bone cement. comparison of mixing techniques. *Acta Orthopaedica Scandinavica*, **60**(4):431–433.
- Linden, U. and Gillquist, J.**, 1989. Air inclusion in bone cement. *Clinical Orthopaedics and Related Research*, **247**:148–151.
- Ling, R. S. M.**, 1992. The use of a collar and precoating on cemented femoral stems is unnecessary and detrimental. *Clinical Orthopaedics and Related Research*, (285):73–83.
- Ling, R. S. M. and Lee, A. J. C.**, 1998. Porosity reduction in acrylic cement is clinically irrelevant. *Clinical Orthopaedics and Related Research*, (355):249–253.
- Lu, Z. and McKellop, H.**, 1997. Effects of cement creep on stem subsidence and stresses in the cement mantle of a total hip replacement. *Journal of Biomedical Materials Research*, **34**:221–226.
- Maher, S. A. and McCormack, B. A. O.**, 1999. Quantification of interdigitation at bone cement/cancellous bone interfaces in cemented femoral reconstructions. *Journal of Engineering in Medicine, Proceedings of the Institution of Mechanical Engineers, Part H*, **213**:347–354.
- Maher, S. A. and Prendergast, P. J.**, 2002. Discriminating the loosening behaviour of cemented hip prostheses using measurements of migration and inducible displacement. *Journal of Biomechanics*, **35**:257–265.

- Malchau, H., Herberts, P., Söderman, P., and Odén, A.,** 2000. Prognosis of total hip replacement: Update and validation from the Swedish National Hip Arthroplasty Registry 1979–1998. In *Scientific Exhibition presented at the 67th Annual Meeting of the American Academy of Orthopaedic Surgeons*, pages 1–16. Orlando, USA.
- Malvern, L. E.,** 1969. *Mechanics of Continuous Media*. Prentice-Hall.
- Mann, K. A., Bartel, D. L., Wright, T. M., and Burstein, A. H.,** 1995. Coulomb frictional interfaces in modeling cemented total hip replacements: a more realistic model. *Journal of Biomechanics*, **28**(9):1067–1078.
- Mann, K. A., Bartel, D. L., Wright, T. M., and Ingraffea, A. R.,** 1991. Mechanical characteristics of the stem-cement interface. *Journal of Orthopaedic Research*, **9**(6):798–808.
- Manson, S. S.,** 1979. Some useful concepts for the designer in treating cumulative fatigue damage at elevated temperatures. In *ICM 3*, pages 13–45. Cambridge, England.
- McCormack, B. A. O. and Prendergast, P. J.,** 1999. Microdamage accumulation in the cement layer of hip replacements under flexural loading. *Journal of Biomechanics*, **32**:467–475.
- McCormack, B. A. O., Prendergast, P. J., and O'Dwyer, B.,** 1999. Fatigue of cemented hip replacements under torsional loads. *Fatigue and Fracture of Engineering Materials and Structures*, **22**:33–40.
- McCormack, B. A. O., Walsh, C. D., Wilson, S. P., and Prendergast, P. J.,** 1998. A statistical analysis of microcrack accumulation in PMMA under fatigue loading: applications to orthopaedic implant fixation. *International Journal of Fatigue*, **20**:581–593.
- McCrum, N. G., Buckley, C. P., and Bucknall, C. B.,** 1988. *Principles of Polymer Engineering*. Oxford University Press, Oxford, UK.

- Miner, M. A.**, 1945. Cumulative damage in fatigue. *Journal of Applied Mechanics*, **A**:159–164.
- Muc, A. and Kędziora, P.**, 2001. A fuzzy set analysis for a fracture and fatigue damage response of composite materials. *Composite Structures*, **54**:283–287.
- Murakami, S.**, 1983. Notion of Continuum Damage Mechanics and its application to anisotropic creep damage theory. *Journal of Engineering Materials and Technology*, **105**:99–105.
- Murakami, S.**, 1988. Mechanical modelling of material damage. *Journal of Applied Mechanics*, **55**:280–286.
- Murphy, B. P.**, 2001. *On damage accumulation of acrylic bone cement*. Ph.D. thesis, University of Dublin, Trinity College, Dublin 2, Ireland.
- Murphy, B. P. and Prendergast, P. J.**, 1999. Measurement of non-linear microcrack accumulation rates in polymethylmethacrylate bone cement under cyclic loading. *Journal of Materials Science: Materials in Medicine*, **10**:779–781.
- Murphy, B. P. and Prendergast, P. J.**, 2000a. On the magnitude and variability of fatigue strength in acrylic bone cement. *International Journal of Fatigue*, **22**:855–864.
- Murphy, B. P. and Prendergast, P. J.**, 2000b. The relationship between stress, porosity, damage accumulation, and fatigue life in acrylic bone cement. In P. J. Prendergast, T. C. Lee, and A. J. Carr, eds., *Proceedings of the 12th Conference of the European Society of Biomechanics*, page 235. Royal Academy of Medicine in Ireland, Dublin, Ireland.
- Murphy, B. P. and Prendergast, P. J.**, 2002. The relationship between stress, porosity, and nonlinear damage accumulation in acrylic bone cement. *Journal of Biomedical Materials Research*, **59**(4):646–654.

- Onat, E. T. and Leckie, F. A., 1988. Representation of mechanical behaviour in the presence of changing internal structure. *Journal of Applied Mechanics*, **55**:1–10.
- Paas, M. H. J. W., Schreuers, P. J. G., and Janssen, J. D., 1990. The application of continuum damage mechanics to fatigue failure mechanisms. In J. F. Dijkstra and F. T. M. Nieuwstadt, eds., *Integration of Theory and Applications in Applied Mechanics*, pages 49–63. Kluwer Academic Publishers.
- Pal, S. and Saha, S., 1982. Stress relaxation and creep behaviour of normal and carbon fibre reinforced acrylic bone cement. *Biomaterials*, **3**:93–96.
- Palmgren, A., 1924. Die lebensdauer von kugellagern. *Zeitschrift des Vereines Deutscher Ingenieure*, **68**:339–341.
- Pascual, B., Gurruchaga, M., Ginebra, M. P., Gil, F. J., Planell, J. A., and Goñi, I., 1999. Influence of the modification of p/l ratio on a new formulation of acrylic bone cement. *Biomaterials*, **20**:465–474.
- Pascual, B. P., Vázquez, B., Gurruchaga, M., Goñi, I., Ginebra, M. P., Gil, F. J., Planell, J. A., Levenfeld, B., and Román, J. S., 1996. New aspects of the effect of size and size distribution on the setting parameters and mechanical properties of acrylic bone cements. *Biomaterials*, **17**:509–516.
- Raab, S., Ahmed, A. M., and Provan, J. W., 1981. The quasi-static and fatigue performance of the implant/bone-cement interface. *Journal of Biomedical Materials Research*, **15**:159–182.
- Radayev, Y. N., 1996. Thermodynamical models of anisotropic damage growth. Part I. Canonical dynamical state variables of continuum damage mechanics and thermodynamic functions of three-dimensional anisotropic damage state. *Journal of Non-Equilibrium Thermodynamics*, **21**(2):129–152.
- Radin, E. L., Rubin, C. T., Thrasher, E. L., Lanyon, L. E., Crugnola, A. M., Schiller, A. S., Paul, I. L., and Rose, R. M., 1982. Changes in the

- bone-cement interface after total hip replacement. An in vivo animal study. *The Journal of Bone and Joint Surgery*, **64-A**(8):1188–1200.
- Reckling, F. W., Asher, M. A., and Dillon, W. L.**, 1977. A longitudinal study of the radiolucent line at the bone-cement interface following total joint-replacement procedures. *The Journal of Bone and Joint Surgery*, **59-A**(3):355–358.
- Rice, J. R.**, 1993. Mechanics of solids. *Encyclopaedia Britannica*, **23**:734–747.
- Rinnac, C. M., Wright, T. M., and McGill, D. L.**, 1986. The effect of centrifugation on the fracture properties of acrylic bone cements. *The Journal of Bone and Joint Surgery*, **69-A**(2):281–287.
- Saanouni, K., Chaboche, J. L., and Lesne, P. M.**, 1989. On the creep crack-growth prediction by a non local damage formulation. *European Journal of Mechanics, A/Solids*, **8**(6):437–459.
- Saha, S. and Pal, S.**, 1984. Mechanical properties of bone cement: A review. *Journal of Biomedical Materials Research*, **18**:435–462.
- Sandler, S. R. and Karo, W.**, 1992. *Polymer Syntheses*, volume 1. Academic Press, 2nd edition.
- Seedhom, B. B. and Wallbridge, N. C.**, 1985. Walking activities and wear of prostheses. *Annals of Rheumatic Diseases*, **44**:838–843.
- Simo, J. C. and Ju, J. W.**, 1987. Strain- and stress-based continuum damage models—I. Formulation. *International Journal of Solids and Structures*, **23**:821–840.
- Singh, U. K. and Digby, P. J.**, 1989. A continuum damage model for simulation of the progressive failure of brittle rocks. *International Journal of Solids and Structures*, **25**:647–663.
- Skrzypek, J. J.**, 1999. Material damage models for creep failure analysis and design of structures. In H. Altenbach and J. J. Skrzypek, eds., *Creep and Damage*

in Materials and Structures, number 399 in International Centre for Mechanical Sciences (CISM) Courses and Lectures, pages 97–166. Springer-Verlag, Wien-New York.

Smeds, S., Goertzen, D., and Ivarsson, I., 1997. Influence of temperature and vacuum mixing on bone cement properties. *Clinical Orthopaedics and Related Research*, (334):326–334.

Spector, M., Shortkroff, S., Hsu, H. P., Lane, N., Sledge, C. B., and Thornhill, T. S., 1990. Tissue changes around loose prostheses. A canine model to investigate the effects of an anti-inflammatory agent. *Clinical Orthopaedics and Related Research*, (261):140–152.

Spiegelberg, S. and McKinley, G., 1999. A source of interfacial porosity in cemented femoral stems. In *46th Annual Meeting, Orthopaedic Research Society*. Anaheim, California, USA.

Starke, G. R., Birnie, C., and van den Blink, P. A., 1997. Numerical modelling of cement polymerisation and thermal bone necrosis. In J. Middleton, M. N. Jones, and G. N. Pande, eds., *Computer Methods in Biomechanics and Biomedical Engineering — 2*, pages 163–172. Gordon and Breach.

Stolk, J., Verdonschot, N., and Huiskes, R., 1999. Management of strain fields around singular points: comparison of FE simulations and experiments. In J. Middleton, M. N. Jones, and G. Pande, eds., *Computer Methods in Biomechanics and Biomedical Engineering — 3*, pages 57–62. Gordon and Breach.

Stolk, J., Verdonschot, N., and Huiskes, R., 2001. Final report workpackage 2. Preclinical testing of cemented hip replacement implants: pre-normative research for a European standard. Contract no. SMT4-CT96-2076. Technical report, Standards Measurements and Testing Program of the European Commission.

Stolk, J., Verdonschot, N., Murphy, B. P., Prendergast, P. J., and Huiskes, R., 2003. Finite element simulation of anisotropic damage accumu-

lation and creep in acrylic bone cement. *Engineering Fracture Mechanics*, page In press.

Timoshenko, S. P., 1983. *History of Strength of Materials*. Dover Publications, Inc., New York.

Timoshenko, S. P. and Goodier, J. N., 1970. *Theory of Elasticity*, chapter Axisymmetric Stress and Deformation in a Solid of Revolution: Local Stresses around a Spherical Cavity, pages 396–398. McGraw Hill, 3rd edition.

Topoleski, L. D., Ducheyne, P., and Cuckler, J. M., 1990. A fractographic analysis of in vivo poly(methyl methacrylate) bone cement failure mechanisms. *Journal of Biomedical Materials Research*, **24**(2):135–154.

Topoleski, L. D. T., Ducheyne, P., and Cuckler, J. M., 1993. Microstructural pathway of fracture in poly(methyl methacrylate) bone cement. *Biomaterials*, **14**(15):1165–1172.

Tsukrov, I. and Kachanov, M., 1997. Stress concentrations and microfracturing patterns in a brittle-elastic solid with interacting pores of diverse shapes. *International Journal of Solids and Structures*, **34**(22):2887–2904.

Verdonschot, N. and Huiskes, R., 1995. A combination of continuum damage mechanics and the finite element method to analyze acrylic cement cracking around implants. In *Second International Symposium on Computer Methods in Biomechanics and Biomedical Engineering*, pages 25–33. Gordon and Breach Publisher, The Netherlands.

Verdonschot, N. and Huiskes, R., 1996. Mechanical effects of stem cement interface characteristics in total hip replacement. *Clinical Orthopaedics and Related Research*, **329**:326–336.

Verdonschot, N. and Huiskes, R., 1997a. Acrylic cement creeps but does not allow much subsidence of femoral stems. *The Journal of Bone and Joint Surgery*, **79-B**(4):665–669.

- Verdonschot, N. and Huiskes, R.**, 1997b. The effects of cement-stem debonding in THA on the long-term failure probability of cement. *Journal of Biomechanics*, **30**(8):795–802.
- Verdonschot, N. and Huiskes, R.**, 1998. Surface roughness of debonded straight-tapered stems in cemented THA reduces subsidence but not cement damage. *Biomaterials*, **19**:1773–1779.
- Voyiadjis, G. Z. and Kattan, P. I.**, 1992. A plasticity-damage theory for large deformation of solids—I. Theoretical formulation. *International Journal of Engineering Science*, **30**(9):1089–1108.
- Voyiadjis, G. Z. and Park, T.**, 1997. Anisotropic damage effect tensors for the symmetrization of the effective stress tensor. *Journal of Applied Mechanics*, **64**:106–110.
- Wang, J. S., Toksvig-Larsen, S., Müller-Willie, P., and Franzeén, H.**, 1996. Is there any difference between vacuum mixing systems in reducing bone cement porosity? *Journal of Biomedical Materials Research*, **33**:115–119.
- Whelan, M. P., Kenny, R. P., Cavalli, C., Lennon, A. B., and Prendergast, P. J.**, 2000. Application of optical fibre Bragg grating sensors to the study of PMMA curing. In P. J. Prendergast, A. J. Carr, and T. C. Lee, eds., *Proceedings of the 12th Conference of the European Society of Biomechanics*, page 252. Royal Academy of Medicine in Ireland, Dublin, Ireland.
- Willert, H. G., Ludwig, J., and Semlitsch, M.**, 1974. Reaction of bone to methacrylate after hip arthroplasty. A long-term gross, light microscopic, and scanning electron microscopic study. *The Journal of Bone and Joint Surgery*, **56-A**(7):1368–1382.
- Wixson, R. L., Lautenschlager, E. P., and Novak, M. A.**, 1987. Vacuum mixing of acrylic bone cement. *The Journal of Arthroplasty*, **2**(2):141–149.
- Wright, T. M. and Robinson, R. P.**, 1982. Fatigue crack propagation in polymethylmethacrylate bone cements. *Journal of Material Science*, **17**:2463–2468.

- Xia, Z. H. and Curtin, W. A.**, 2001. Multiscale modeling of damage and failure in aluminum-matrix composites. *Composites Science and Technology*, **61**:2247–2257.
- Xiao, Y. C., Li, S., and Gao, Z.**, 1998. A continuum damage mechanics model for high cycle fatigue. *International Journal of Fatigue*, **20**(7):503–508.
- Yetkinler, D. N. and Litsky, A. S.**, 1998. Viscoelastic behaviour of acrylic bone cements. *Biomaterials*, **19**:1551–1559.
- Zavattieri, P. D. and Espinosa, H. D.**, 2001. Grain level analysis of crack initiation and propagation in brittle materials. *Acta Materiala*, **49**:4291–4311.

Appendix A

Relevant published articles

- Modelling of temperature history and residual stress generation due to curing in polymethylmethacrylate 167
- Residual stress due to curing can initiate damage in porous bone cement: experimental and theoretical evidence 168
- Use of grating interferometry for validation of finite element models and to investigate residual strain in polymethylmethacrylate 179
- Evaluation of cement stresses in finite element analyses of cemented orthopaedic implants 185

Modelling of temperature history and residual stress generation due to curing in polymethylmethacrylate

A. B. Lennon¹, P. J. Prendergast¹, M. P. Whelan², R. P. Kenny², C. Cavalli²

¹Department of Mechanical Engineering, Trinity College, Dublin 2, Ireland. ²Photonic Technologies and Diagnostics Sector, Institute for Systems Informatics and Safety, European Commission Joint Research Centre, 21020 Ispra (VA), Italy.

Introduction

Residual stress in polymethylmethacrylate (PMMA) due to curing is one possible mechanism of damage initiation within the cement mantles of orthopaedic joint replacements. Although relaxation of these stresses may decrease their effect in the long term [1], they may have a significant effect in the immediate post-operative period. Modelling of the generation of these stresses has focussed on thermal expansion and contraction of the material during and after polymerisation [2]. This study assumes that residual stress is due primarily to shrinkage of the cement from the excited thermal state existing at the end of polymerisation.

Materials and methods

A PTFE mould was produced to cast a small rectangular block of PMMA cement as part of a companion study [3]. The mould was open at one of its narrow ends to facilitate pouring. Extra fixturing at the opposite end allowed either a fibre Bragg sensor, a thermocouple or both to be held near the mould-centre during casting. Internal temperature and strain variations were then monitored for a period of 2000s during the cure.

A quarter model of the experimental set-up (Fig. 1, left) was generated in ANSYS (SAS IP Inc., USA). Beam elements, with nodal degrees of freedom coupled to the corresponding cement nodes, were used to model the fibre. Since only axial strain of the fibre could be predicted with the beam elements, a sub model (Fig.1, right) of a portion of the fibre and the surrounding cement was also generated, in order that radial and hoop strains could also be calculated.

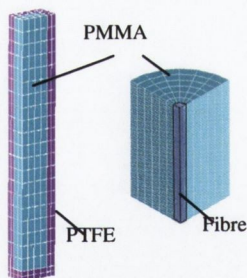


Fig. 1: Global and sub-model meshes

All heat generated was considered to be due to polymerisation of the monomer [4], which was assumed to progress with time in the form of an S-curve (Fig. 2). The polymerisation curve was discretised into linear segments and the required heat generation rate due to polymerisation calculated for each segment. Convection cooling was assumed at the exterior surfaces of mould and cement. Properties for both thermal and structural analyses are shown in Tab. 1 (the silica fibre was not modelled in the thermal analysis).

Elements were divided into 30 groups based on their peak temperature at the end of polymerisation. These values were then input as the reference temperature for thermal stress calculations and the specimen temperature was set to ambient conditions (23°C). Frictional contact was assumed for the PTFE-PMMA interface and modelled using surface to surface contact elements.

Tab. 1: Thermal and structural material properties

Material:	PMMA	PTFE	Silica
ρ (kg/mm ³)	1.19×10^{-6}	2.2×10^{-6}	—
c (J/kgK)	1450	1000	—
K (W/mmK)	0.18×10^{-3}	0.25×10^{-3}	—
E (GPa)	2.4	0.41	70
ν	0.33	0.33	0.17
α (10 ⁻⁶)	70	130	0.5

Results

A rapid rise in temperature is predicted for the cement (Fig. 2) with a peak of 89°C (Tab. 2) compared with a measured value of 93°C [3]. Large compressive strain of the fibre due to shrinkage of the cement is also predicted (Tab. 2)—significant compression of the fibre was also found in the experimental study [3], but could not be compared directly with the FE prediction, as the three dimensional strain sensitivity of the fibre is not currently known. The interior cement itself is predicted to be in tension (Tab. 2).

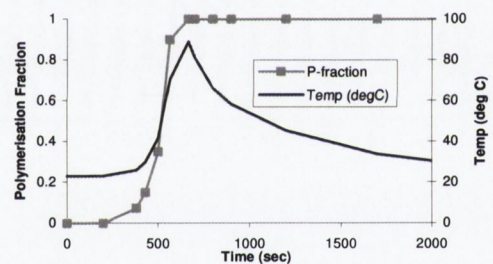


Fig.2: Temperature history of node at specimen centre and corresponding polymerisation curve.

Tab.2: Peak temperature, fibre microstrain and peak residual cement stress

Temp (°C)	Axial $\mu\epsilon$	Radial $\mu\epsilon$	Hoop $\mu\epsilon$	Stress(MPa)
89	-4027	-4555	-5264	3.1

Discussion and conclusions

Significant tensile stress, of 3.1 MPa, is predicted for the interior cement. This is because the outer cement elements do not reach as high temperatures as the interior and hence shrink less in approaching ambient temperature. The interior cement elements are therefore restrained from shrinking and tension results. Although stress relaxation will reduce this value over time [1], the value is almost equal to that generated by loading.

References

- [1] A.B. Lennon, P.J. Prendergast, M.P. Whelan and C. Forno (1999) Proc. 4th Int. Sym. on Comp. Meth. in Biomech. and Biomed. Eng., Lisbon. [2] A.M. Ahmed, W. Pak, D.L. Burke and J. Miller (1982) J. Biomech. Eng. 104: 21-37. [3] M. P. Whelan, R. P. Kenny, C. Cavalli, A. B. Lennon and P.J. Prendergast (2000) Proc. ESB2000. [4] R. Huiskes (1980) Acta Orthop. Scand., suppl. 183.

Residual stress due to curing can initiate damage in porous bone cement: experimental and theoretical evidence

A.B. Lennon, P.J. Prendergast*

Department of Mechanical Engineering, Trinity College, Dublin 2, Ireland

Accepted 17 October 2001

Abstract

Residual stress due to shrinkage of polymethylmethacrylate bone cement after polymerisation is possibly one factor capable of initiating cracks in the mantle of cemented hip replacements. No relationship between residual stress and observed cracking of cement has yet been demonstrated. To investigate if any relationship exists, a physical model has been developed which allows direct observation of damage in the cement layer on the femoral side of total hip replacement. The model contains medial and lateral cement layers between a bony surface and a metal stem; the tubular nature of the cement mantle is ignored. Five specimens were prepared and examined for cracking using manual tracing of stained cracks, observed by transmission microscopy; cracks were located and measured using image analysis. A mathematical approach for the prediction of residual stress due to shrinkage was developed which uses the thermal history of the material to predict when stress-locking occurs, and estimates subsequent thermal stress. The residual stress distribution of the cement layer in the physical model was then calculated using finite element analysis. Results show maximum tensile stresses normal to the observed crack directions, suggesting a link between residual stress and pre-load cracking. The residual stress predicted depends strongly on the definition of the reference temperature for stress-locking. The highest residual stresses (4–7 MPa) are predicted for shrinkage from maximum temperature; in this case, magnitudes are sufficiently high to initiate cracks when the influence of stress raisers such as pores or interdigitation at the bone/cement interface are taken into account (up to 24 MPa when calculating stress around a pore according to the method of Harrigan and Harris (*J. Biomech.* 24(11) (1991) 1047–1058)). We conclude that the damage accumulation failure scenario begins before weight-bearing due to cracking induced by residual stress around pores or stress raisers. © 2002 Elsevier Science Ltd. All rights reserved.

Keywords: Arthroplasty; Numerical model; Damage; Intramedullary prostheses; Residual stress; Cemented prostheses

1. Introduction

Prostheses for joint arthroplasty are often fixated into the bone using an acrylic polymer called polymethylmethacrylate (PMMA). Although it is widely used, certain aspects of the mechanical behaviour of the polymer in situ have not yet been elucidated. One such aspect is the residual stress due to shrinkage of the PMMA as it polymerises. Shrinkage stresses of sufficient magnitude could cause cracking before the joint is loaded and therefore could contribute to starting a damage accumulation failure scenario within the reconstructed joint (Huiskes, 1993). However, this has not yet been definitively demonstrated.

Pre-load cracks have been observed in physical models (McCormack and Prendergast, 1999) and it has also been proven statistically that the rate of damage accumulation is proportional to the number of pre-load cracks (McCormack et al., 1998). The exact mechanism of stress generation during polymerisation of a two phase mixture of PMMA/MMA is difficult to ascertain. The polymerisation reaction itself is affected by several factors; first, as polymerisation progresses the mixture becomes viscous as the polymer chains grow longer and, second, the monomer remains relatively mobile leading to an auto-acceleration of the polymerisation rate (Kine and Novak, 1987). Thirdly, the reaction is highly exothermic and the release of heat further accelerates the polymerisation rate. The highly nonlinear reaction rate during the process makes it difficult to know exactly from when the material becomes capable of supporting stress. The problem is simplified greatly if the mechan-

*Corresponding author. Tel.: +353-1-608-1383; fax: +353-1-679-5554.

E-mail address: pprender@tcd.ie (P.J. Prendergast).

ism of stress-locking can be considered as a result of thermal deformations only. Some reference time from which stress-locking occurs is then needed. One such time is the peak temperature reached during polymerisation. A rationale for this is that the heat generation pulse is severe enough that thermal excitation will be sufficient to delay locking until the maximum temperature is reached.

A number of approaches have been used to estimate the level of shrinkage stress in bone cement around femoral replacements. Huiskes (1980) predicted radial temperature rise in a cement layer and calculated the residual stress due to shrinkage from the peak temperature distribution. Mann et al. (1991) assumed shrinkage from a uniformly distributed maximum temperature. Ahmed et al. (1982a) developed a model which predicted transient stresses during polymerisation, as well as locked-in stress due to shrinkage, and applied it to an axisymmetric model. However, if stress-locking occurs at the time when peak temperature is reached then stresses during the expansion phase of polymerisation, i.e. before the point of stress-locking, are likely to be relaxed and therefore not significant. If this is true, then the approach used by Huiskes (1980) is suitable for the estimation of cement residual stress. However, his calculation used a time-dependent polymerisation function which did not account for any temperature dependence in the polymerisation rate. Therefore, in his model, all regions of cement polymerised at the same rate, and this may lead to inaccuracies in the timing and magnitude of the maximum temperature. This was noted by Huiskes (1980) when comparing predictions of his model with results from an experimental study by Meyer et al. (1973).

Baliga et al. (1992) developed an empirical model for the prediction of heat generation in polymerising cement as a function of temperature and fraction of monomer polymerised. They also showed large deviations from measured behaviour if account was not taken of the dependence of polymerisation rate on local instantaneous temperature. Starke et al. (1997) implemented the model of Baliga et al. (1992) as an iterative numerical scheme suitable for finite element modelling and predicted that interior regions of cement experienced greater and more rapid temperature rise than regions nearer the interfaces — the temperature distribution was then used to predict thermal bone necrosis. Indeed, much analysis of polymerisation has been focussed on the prediction of thermal bone necrosis, (e.g. Huiskes, 1980; Starke et al., 1997), while the subject of residual stress has often been neglected because it has been assumed that residual stress will relax due to the viscoelastic properties of the cement. However, the presence of pre-load cracks in cement layers implies that the initial residual stress, although it relaxes over time, may have an immediate effect. In this paper, we use both

experimental and computational models to test the hypothesis that shrinkage-induced residual stresses can cause pre-load cracking in femoral components of hip replacements. It is hypothesised that residual stresses create measurable amounts of damage. Even if residual stress later disappears due to stress relaxation, the pre-load damage created may initiate the damage accumulation failure scenario, as described by Huiskes and Verdonschot (1997).

2. Methods

2.1. Physical model

2.1.1. Description and preparation

A physical model has been developed to investigate damage accumulation around cemented femoral components of total hip replacements (Fig. 1). The model consists of a medial and lateral layer of cement encased between a layer of bone and an implant, with the whole construction held together between two aluminium covers (sideplates). The model was developed from the earlier work of McCormack and Prendergast (1999). Windows in the sideplates expose the cement layer around the stem, allowing direct observation of damage accumulation, while the sideplates themselves support the structure in a manner similar to that of cortical bone. Bovine rib bone was used to form cancellous bone margins on the inside walls of the aluminium covers. The model has proximal curvature and a trochanter-like process for the attachment of an abductor load for later studies (Lennon and Prendergast, 2001). A description of the design of the model is given in Lennon et al. (1998).

Hand mixed Simplex Rapid cement was used for all specimens as it is sufficiently translucent to allow microscopic observation of cracks by light transmission. A standard mixing ratio of 2 g : 1 ml powder to liquid was used. Polyethylene covers were inserted into the cut-outs of the inner cover to prevent cement escaping, as well as to keep the cement surface contiguous with the stem and bone surfaces. Mixing was carried out for approximately 60 s at 1 beat/s and the cement was introduced into the specimen cavity once a doughy state had been achieved. The specimen was then allowed to cure and was kept encased between the polyethylene covers for a further 24 h.

2.1.2. Crack counting

Crack measurement was achieved by staining the sample with dye penetrant (Johnson and Allen Ltd., UK) — cracks can then be seen under magnification and light transmission through the translucent cement. Since only those cracks which intersect the exposed surfaces can be stained, this method leads to a

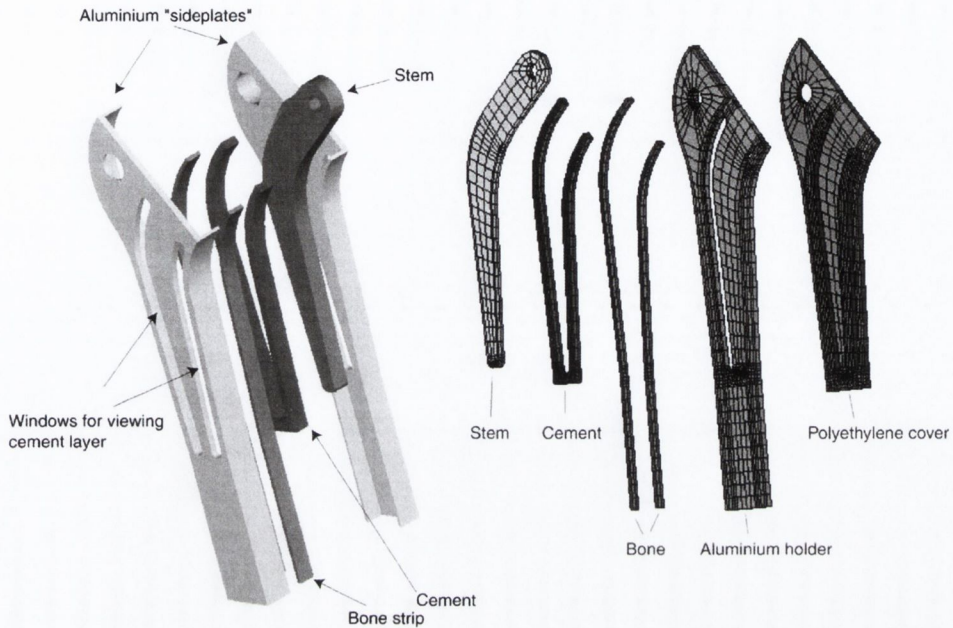


Fig. 1. Schematic view of physical model for damage accumulation study showing cut-outs for viewing of cement layers, trochanter feature with muscle attachment capability, cancellous bone strips, and aluminium side-plates. The cement thickness is 8 mm in the sagittal plane and approximately 3–4 mm in the frontal plane (i.e. between implant and bone surfaces). In addition, the mesh used for the finite element study is shown on the right.

conservative count of the number of cracks as those that are entirely within the mantle cannot be stained. A Mitutoyo optical comparator with a $\times 20$ lens was used to examine the specimen for pre-cracks by imaging cracks at the specimen surface, unless it could be seen that a crack extended further below the surface — in such a case focus was changed to the plane in which the crack was clearly seen to be longest. Each crack observed was traced onto an acetate transparency from which a digital image of the crack distribution was obtained. The images were then thresholded to separate cracks from any background level of greyscale, due to the scanning operation, and image analysis software (Image Tool, UTHSCSA, USA) was used to fit an ellipse to each crack; the major axis length, slope, and centroid were recorded for all cracks. The coordinates of the centroids were then transferred from the image coordinate system to a reference system, and the endpoints for each crack were calculated.

2.2. Numerical model

2.2.1. Algorithm for heat generation and residual stress prediction

The model presented here is an adaptation of that used by Baliga et al. (1992) and Starke et al. (1997). The polymerisation fraction p of the curing mass of cement is

defined as the ratio of heat generated, Q , at time t , to the total amount of heat generated on completion of polymerisation, Q_{total} ; i.e.

$$p = \frac{1}{Q_{total}} \int_0^t \dot{Q} dt \approx \frac{1}{Q_{total}} \sum_{i=1}^n \dot{Q}_i \Delta t_i, \quad (1)$$

where the heat generation rate \dot{Q} expressed per unit volume of cement, is assumed to be a function of temperature θ and polymerisation fraction p . The form of the heat generation rate can be approximated as

$$\dot{Q} = R(\theta)(p - p^2), \quad (2)$$

where $R(\theta)$ is a rate function which depends on temperature. Baliga et al. (1992) determined, empirically, the following expression for $R(\theta)$:

$$R(\theta) = 4.4 \times 10^6 \left[a_0 + a_1 \left(\frac{\theta}{100} \right) + a_2 \left(\frac{\theta}{100} \right)^2 + a_3 \left(\frac{\theta}{100} \right)^3 + a_4 \left(\frac{\theta}{100} \right)^4 + a_5 \left(\frac{\theta}{100} \right)^5 \right] \quad (3)$$

with units of $J/(m^3s)$. The coefficients a_i are: $a_0 = -23.89 J/(m^3s)$, $a_1 = 296.74 J/(^{\circ}Cm^3s)$, $a_2 =$

$-1352.97 \text{ J}/(^{\circ}\text{C}^2\text{m}^3\text{s})$, $a_3 = 2894.76 \text{ J}/(^{\circ}\text{C}^3\text{m}^3\text{s})$, $a_4 = -2806.62 \text{ J}/(^{\circ}\text{C}^4\text{m}^3\text{s})$, and $a_5 = 1009.84 \text{ J}/(^{\circ}\text{C}^5\text{m}^3\text{s})$.

2.2.2. Finite element model

The mesh (Fig. 1) was generated from eight-noded hexahedral elements as a half-model, due to symmetry about the frontal plane of the specimen. ANSYS (Canonsburg, PA, USA) was used to solve both thermal and structural analyses. The polyethylene covers were included in the model (Fig. 1) for the thermal portion of the analysis but were removed for the structural analysis. This was felt to be acceptable as it was found that very little effort was required in removing the covers, implying that little or no bonding existed between the cement and polyethylene. All interfaces were assumed to be bonded for the thermal analysis and a convection load (surface convection coefficient of $2 \times 10^{-5} \text{ J}/\text{mm}^2$) was applied to the external surfaces of the specimen. Ambient temperature was assumed to be 23°C and all materials were assumed to be at this temperature at the start of the analysis. All remaining interfaces in the structural analysis were also treated as bonded. Material properties for the model are given in Table 1.

Since pores are a feature of bone cement (Murphy and Prendergast, 2000; Tepic and Soltesz, 1998), the stress distribution was also calculated using the method of Harrigan and Harris (1991). This approach assumes stress on the surface of the pore is independent of pore size and that the pore is much smaller than the region in which it is found — this leads to less than a 10% error if the distance between the pore and an interface is less than three times the pore diameter (Harrigan and Harris, 1991). The maximum stress around a pore was estimated for every element in the observable cement and averaged at each node — although it is unrealistic to assume that a pore would be present in every element, this approach allows areas of cement susceptible to crack initiation from pores to be identified.

2.2.3. Description of the iterative procedure

A polymerisation rate, $\Delta p_1/\Delta t_1$, was assumed for the first time increment and the required heat generation rate obtained by solving for \dot{Q} in Eq. (1). For

subsequent time increments, the temperature and polymerisation fraction for each cement element at the end of the increment were used to calculate the heat generation rate (from Eqs. (2) and (3)) for the next increment. Fig. 2 shows an outline of the algorithm used. The time required to achieve a 5% increase in polymerisation fraction for each element was calculated; the minimum was used as the time step so that time stepping was governed by the fastest polymerising element. Once polymerisation had completed, the time step size was increased and the analysis was stopped at 2000 s. During the solution procedure, the centroid temperature and polymerisation fraction of each element was recorded and the maximum value was stored. The time an element finished polymerisation, as well as its temperature, was also recorded.

The next stage of the numerical modelling procedure was to determine the shrinkage stresses. These were determined for (i) shrinkage from the maximum temperature attained by each element and (ii) shrinkage from the temperature at the end of polymerisation. As the level of stress predicted will depend strongly on the amount of shrinkage that occurs, particular attention was paid to the thermal expansion coefficient. The temperature dependence of the thermal expansion coefficient of PMMA immediately after polymerisation has been approximated as a bilinear curve by Ahmed et al. (1982b) with a knee point at 43°C (Table 1). Therefore, using the temperature distribution obtained from the thermal analysis, the coefficient of thermal expansion was set for each element based on its temperature during each time step of the cooling phase.

3. Results

3.1. Pre-load crack distributions

Pre-load cracks were found in almost every region of the mantle over all five specimens, although there is great variability between individual specimens, see Fig. 3. Total crack length for specimens 1, 2, 4, and 5 were comparable, while specimen 3 exhibited approxi-

Table 1
Structural and thermal properties

Material	ρ (kg/mm ³)	k (W/mmK)	c (J/kgK)	E (GPa)	ν	α ($\times 10^{-6}\text{K}^{-1}$)
Cement	1.19×10^{-6}	0.18×10^{-3}	1450	2.4	0.33	72.2 ($\theta < 43^{\circ}\text{C}$) 87.6 ($\theta > 43^{\circ}\text{C}$)
Stem	7.8×10^{-6}	14×10^{-3}	460	210	0.33	12.5
Cancellous bone	1.3×10^{-6}	0.29×10^{-3}	2292	2	0.3	0.1
Aluminium	2.8×10^{-6}	0.1255	925	73	0.33	20
Polyethylene	0.92×10^{-6}	0.5×10^{-3}	1900	1	0.3	150

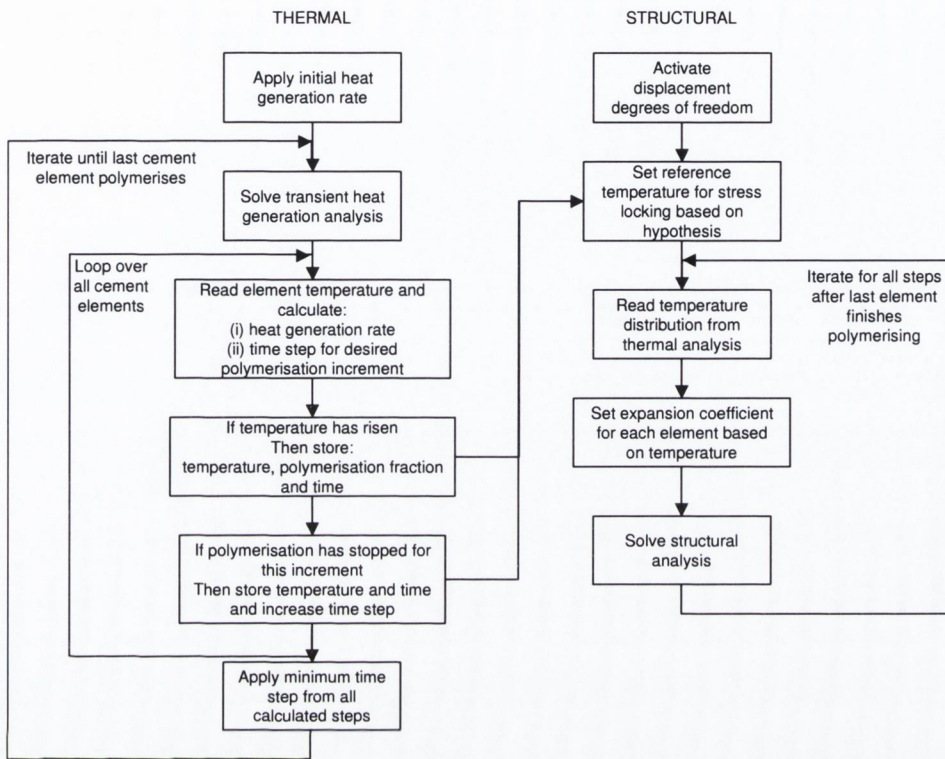


Fig. 2. Algorithm for thermoelastic analysis. The thermal analysis is performed first and results are subsequently used as inputs for the structural analysis stage. Reference temperatures for shrinkage are set according to either peak temperature or temperature upon completion of polymerisation. Iteration in the structural portion is required because of the use of a bilinear expansion coefficient — element temperature must be compared during each step to temperature at which change of coefficient occurs (43°C for this study).

mately double the amount (Table 2). Complete cracking of the cement layer was observed in the proximo-lateral region of specimen 1 and in the mid- to disto-lateral and disto-medial regions of specimen 3. Cracks were predominantly oriented normal to the interfaces, although many short cracks, which often originated from pores, were oriented in other directions. Cracks were also observed to originate from both stem–cement and bone–cement interfaces. Initiation sites near the interfaces were difficult to discern due to a high degree of staining at the bone–cement interface, and regions of numerous small pores near portions of the stem–cement interface.

3.2. Temperature and polymerisation history

Cement temperature rose rapidly between 6.7 and 10 min, after which it cooled quickly (Fig. 4a) with the rate of cooling dependent on deviation from ambient conditions. The middle of the cement layer exhibited greater and more rapid temperature rise than cement

near the interfaces (Fig. 4b). Temperature rise was lowest at the stem–cement interface because heat loss was greatest here, while the bone–cement interface experienced somewhat higher temperature rise due to the bone's lower heat transfer coefficient. The time taken to reach the peak temperature increased slightly for cooler regions; the stem–cement interface reached it last (Fig. 4b), but, in general, the time at which the peak occurred was spread over a relatively small interval (Fig. 5 — the entire cement layer reached its peak temperature within a time interval of 38 s. The through-layer temperature distribution was approximately uniform over the length of the observable cement (Fig. 4c) at the time that the peak occurred, with temperatures along the length of the central region of the layer ranging between approximately 46–53°C (Fig. 4c). Temperature of the PMMA at the end of polymerisation ranged between 27–29°C. Complete polymerisation of the entire acrylic cement occurred almost simultaneously, with 97.5% of cement elements polymerising before 697 s.

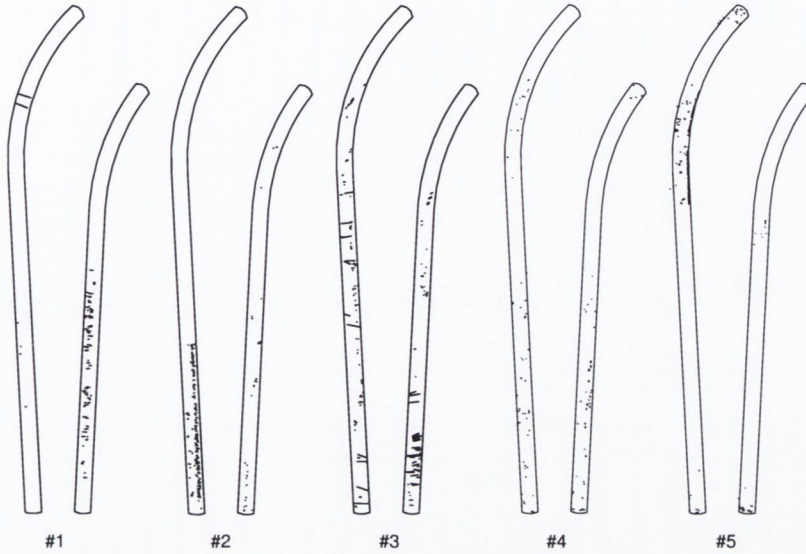


Fig. 3. Pre-load cracks for each of the five specimens. Dots and lines indicate cracks.

Table 2
Number of cracks and sum of crack lengths for each specimen

Specimen no.	1	2	3	4	5
No. cracks	136	218	173	350	315
Sum of crack lengths	38.332	31.111	84.136	19.319	34.756

3.3. Shrinkage stresses

Whether stress-locking was implemented at the peak temperature or at the end of polymerisation had a pronounced effect on the predicted residual stress in the cement. The maximum principal stresses are in the range of 1–2 MPa for the case of shrinkage from the temperature at the end of polymerisation, whereas they are in a range of 4–7 MPa for the case of shrinkage from the maximum temperature reached (compare Fig. 6 (a) and (b)). The effect of the inclusion of porosity in the stress calculation had a pronounced effect on maximum stress (Fig. 6(c)), resulting in increases of approximately three times the predicted maximum principal stress in many regions.

The maximum residual stresses were predominantly directed parallel to the interfaces, which fits the experimental observation of crack direction, as shown in Fig. 7. Some deviation occurs for short cracks radiating from pores.

Shrinkage also creates normal tensile stresses on the interfaces. These were predicted to range from 1.6 to 3.6 MPa on the lateral stem–cement interface (Fig. 8). Shear stresses in the longitudinal direction at the same

interface were low: approximately -0.2 – 0.4 MPa (Fig. 8); those in the direction perpendicular to the exposed cement surface were somewhat higher, at approximately -0.8 MPa (Fig. 8). The same pattern of stress was found for the medial stem–cement interface.

4. Discussion

That the stress and crack orientations are mutually orthogonal (Fig. 7) give a strong indication that residual stresses initiate cracks prior to mechanical loading. The level of pre-cracking supports the conclusion that stresses are relatively high, and suggests that the peak temperature is probably the correct reference point for residual stress calculations. Some experimental data exists to support the case of shrinkage from peak temperature — Whelan et al. (2000), using embedded fibre optic sensors, attempted to measure strain within the interior of a curing block of PMMA and did not find strain in the sensor until the peak temperature was reached.

Limitations apply to both physical and computational models. As the cement mantle is modelled by layers, the

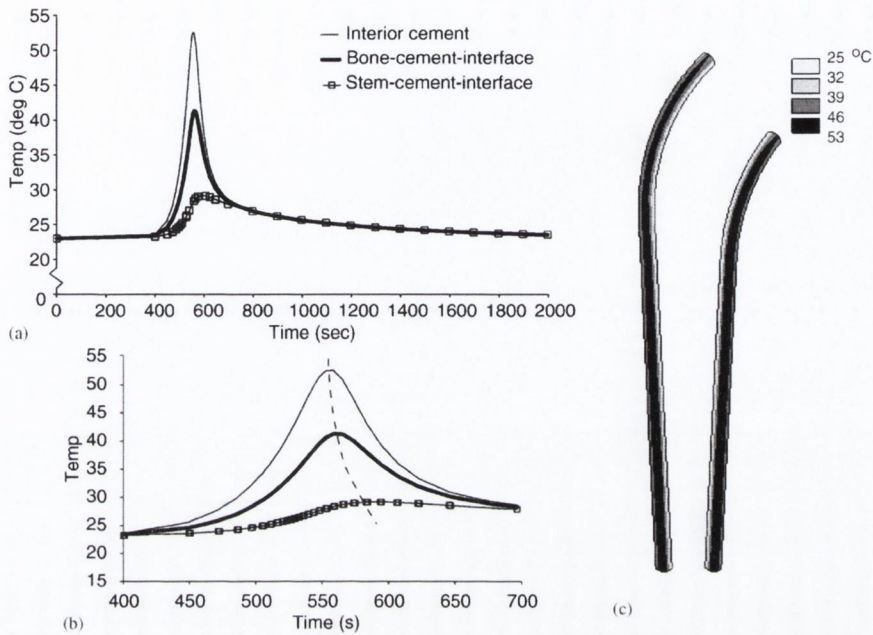


Fig. 4. (a) Temperature history at three points through the cement thickness, between stem-cement and cement-bone interfaces; (b) time-magnified view of temperature pulse to illustrate peak temperature of each region occurring at different times; (c) contour plot of temperature in viewable cement for iteration when highest temperature was reached ($t = 536$ s).

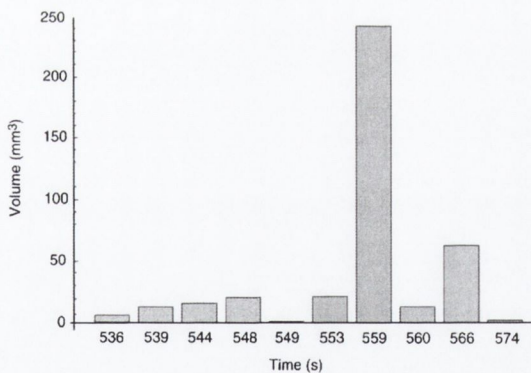


Fig. 5. Histogram showing volumes of cement that reached their peak temperature at different times. Time intervals are not uniform because time-step procedure was based on fastest polymerising element. Notice that most cement peaks at 559 s while the entire interval is 38 s.

effect of hoop stress is not incorporated in either model. Because the dye penetrant needs access to surfaces in order to stain cracks, only cracks that intersect with the exposed surface can be measured. Another understimation of cracks relates to the use of an orthographic projection of the crack to measure its length; i.e. there is no guarantee that the longest projection of the crack is

being imaged. The measurement method is therefore conservative in its estimate of summed length of cracks. Also, to retain transparency, cement without radio-opaque filler or other additives was used. In the computational model there is significant uncertainty with respect to the thermal properties of cancellous bone and its interface with the cement, as well as the stem-cement interface — this has been mentioned previously by others (Huiskes, 1980; Starke et al., 1997). Furthermore, the bone in this model was at room temperature (approximately 23°C compared with 37.5°C for the physiological case). Therefore, Less shrinkage might be expected for the in vivo case. However, the higher ambient in vivo temperatures would likely result in an increased temperature pulse — if large enough, this could compensate for the higher final temperature and result in a similar temperature drop as for our in vitro case. A further limitation could be the arbitrarily-assigned initial polymerisation rate; however, since the heat generation rate depends on the total polymerisation fraction, which is set as a 5% increment, rather than its rate (Eq. (2)), subsequent polymerisation is insensitive to this assumption. Finally, the model calculates stress due to thermal shrinkage from element reference temperatures but these temperatures occur at different times for different elements. An element which has just begun to lock-in stress, but is surrounded by others

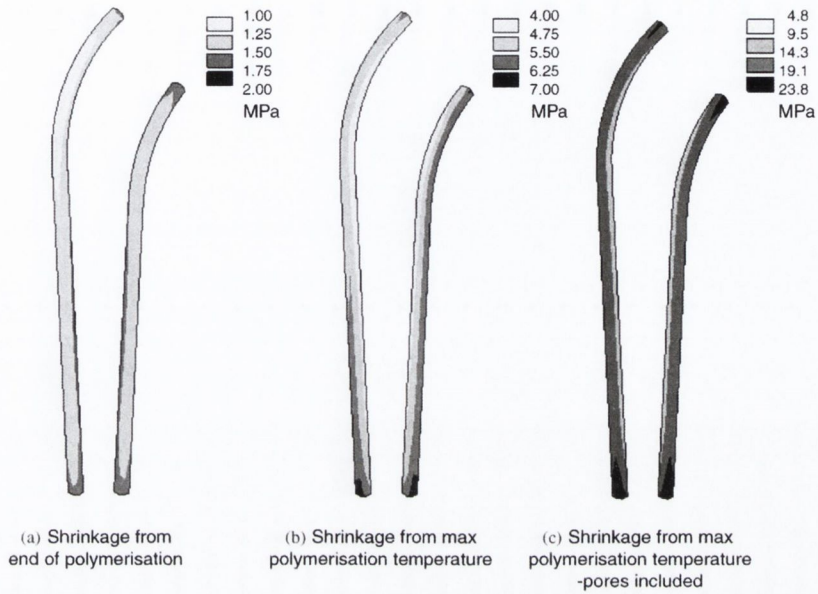


Fig. 6. Maximum principal stress in cement cut-outs for shrinkage of each element relative to (a) its temperature at the time it completed polymerisation, (b) its maximum temperature achieved during polymerisation, and (c) the maximum stress for a uniform pore distribution for stress state of case (b).

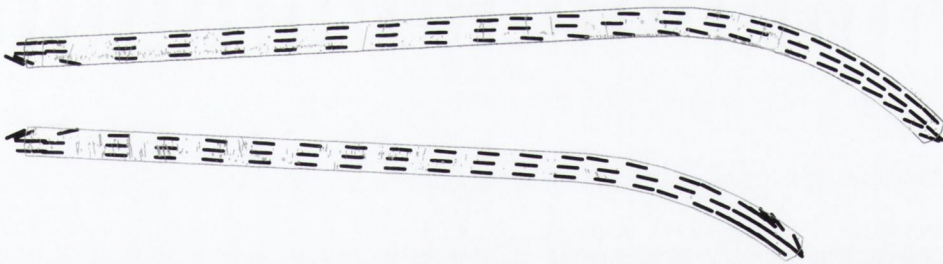


Fig. 7. Plot of maximum principal stress vectors (heavy lines) superimposed on combined crack distribution (light lines) for all five specimens.

which have not yet stress-locked, may not experience significant resistance to shrinkage and little stress would be induced in it. The model is unable to account for this and assumes that elements which have not yet stress-locked will provide the same resistance to neighbouring elements as those that have. However, as the time span for stress-locking is quite small (Fig. 5), the effect will not be too severe. To overcome this deficiency it would be necessary to account for the changing material properties during polymerisation. Such a constitutive relation for chemically hardening materials has been proposed by Shaffer and Levitsky (1974) and implemented for bone cement by Ahmed et al. (1982a).

The thermal analysis predicted differing rates of temperature rise, as well as different peak temperature times, within the mantle, which agrees with the results of other studies, e.g. Meyer et al. (1973), Baliga et al. (1992), and Starke et al. (1997). Other studies have predicted a maximum rise of 45–50°C (Huiskes, 1980), 40°C (Baliga et al., 1992), and 36–45°C (Starke et al., 1997). Peak temperature rise for the observable cement of our physical model was approximately 30°C, which is therefore between 6°C and 20°C lower than the predicted *in vivo* temperature rise from the aforementioned studies. Thus, the predicted temperature drop of our model, in spite of its lower final temperature, is

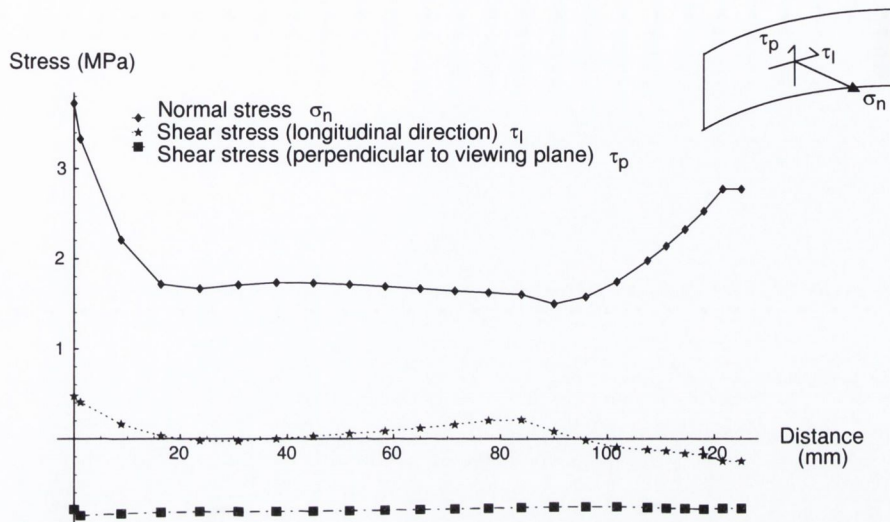


Fig. 8. Plot of normal and shear interface stresses (from distal to proximal) along lateral stem-cement interface for case of shrinkage from maximum temperature. Inset illustrates local interface coordinate system and stress legend.

likely to be of similar proportions to that occurring in vivo. If the drop from peak temperature is a controlling factor in residual stress generation, then the observable cement of the physical model could be expected to develop similar residual stress, and hence pre-load cracking, as occurs in vivo (a higher temperature rise (48°C) was predicted below the distal tip of the stem but this region was not observable for the purposes of crack measurement).

If shrinkage is the primary mechanism of stress generation, then a number of factors should further influence the level of stress generated:

- (i) The expansion coefficient of PMMA is quite high, and is likely to be even higher prior to complete polymerisation — this would have the effect of further increasing residual stresses, especially if stress-locking occurs at the maximum temperature.
- (ii) Geometry and boundary conditions of the structure will affect the shrinkage direction. Consider, first, an annulus of cement — if unrestrained, it would shrink radially inward. However, if an implant is placed inside, the cement will constrict around the implant and the interior surface of the cement will be radially compressed against it and, in the absence of the bone, the external surface would be free of stress in the radial direction. However, if a stiff cortical shell could bond to the external cement, it would restrain the cement from shrinking inwards resulting in radial tension. Thus, in a real hip prosthesis cement mantle, radial cement stress can be expected to vary from compression at the

stem-cement interface to tension at the cement-bone interface, provided a bond can be maintained with the bone. These geometric and interfacial constraints would also induce tensile hoop stresses as the cement shrinks.

A different mode of shrinkage occurs in the physical model presented above. Shrinkage in the transverse and longitudinal directions depend on the total temperature drop in each direction. Consider three points, oriented in a transverse direction, from the central region of the slice (Fig. 9) — because of the temperature gradient in this direction, a lower average temperature change occurs in the transverse direction compared with the perpendicular set of points in the longitudinal orientation. The points in the longitudinal direction are at a more uniform, and higher, average temperature. Greater shrinkage would then occur in the longitudinal direction. Therefore, provided this shrinkage was restrained (e.g. under conditions of plane strain and interfacial bonding), greatest tensile stresses would be expected in the longitudinal direction, as predicted (see Fig. 7).

We predict somewhat higher maximum tensile stresses, but still of the same order of magnitude, as those predicted by other authors; i.e. a range of 4–7 MPa compared with a range of 1–5.5 MPa found in: Huiskes (1980), Mann et al. (1991), and Ahmed et al. (1982a). The level of shrinkage stress for both cases is still an order of magnitude below the static strength of the cement (approx. 25 MPa for Simplex-P according to a review by Saha and Pal, 1984), suggesting that pre-

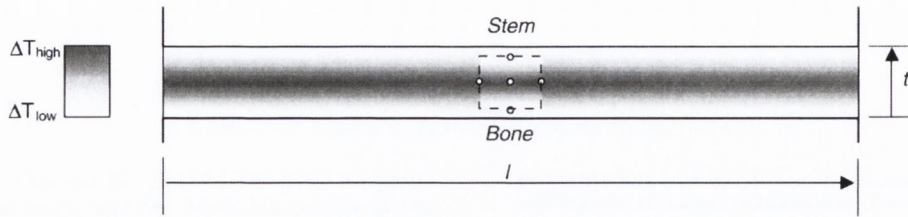


Fig. 9. Schematic illustrating difference between transverse and longitudinal temperature change in a longitudinal slice of cement — the average temperature change of three points in a representative square at the centre of the bar depends on which orientation is examined. Effective temperature drop is thus greater for the longitudinal direction (legend on left of figure illustrates peak temperature change — i.e. peak temperature occurs in central region).

cracking is dependent on a combination of factors, rather than residual stress alone. One such factor is likely to be porosity — this was demonstrated by the estimation of stress around a pore for each element (Fig. 6 (c)) which raised the stress very close to the ultimate tensile strength of the material. This method of stress calculation did not include interaction effects so, in practice, stresses could be even higher around clusters of pores. As porosity takes on an apparently random distribution, at least away from the cement–metal interface, it would be expected that pre-load crack distributions would also appear to be random and would have higher densities in regions with pores. Such a random distribution of cracks is evident from the data for the individual specimens (see Fig. 3) but any study of direct correlation with pores could not be undertaken as the pore distributions were not quantified.

Another possible initiation site for cracks is the bone–cement interface — the interdigitated cement is likely to contain stress concentrations which may, possibly in combination with nearby pores, be sufficient to initiate cracks in the shrinking cement. Unfortunately, it was not possible to observe any such behaviour directly, as this interface stained so heavily that it became opaque (due to easier access to the dye penetrant through the cancellous bone).

In contrast to the nonporous stress predictions for the bulk cement, magnitudes of normal interface stresses (1.6–3.6 MPa) were of the same order of magnitude as, although lower than, static interface strength; e.g. 6.9 MPa for ultimate tensile strength (Raab et al., 1981), and could be expected to accelerate interfacial debonding upon cyclic loading of the specimen. This interface was difficult to observe in the specimens due to increased opacity, created mainly by shadows from the high density of pores in this region, and so no definitive assessment of interface integrity could be made. As the stem had a matt surface finish ($R_a \leq 3 \mu\text{m}$), and hence relatively high strength, the interface was expected to remain bonded under the predicted interface stresses. If debonding were to occur it would be expected to relieve residual stresses since the cement would not be

restrained from shrinking. For the *in vivo* case this is not likely to be an issue since, as mentioned above, an annular cement mantle shrinking around an implant is likely to result in radial compressive stresses at the implant interface, as predicted by Huiskes (1980).

The presence of shrinkage stress immediately post-operatively can create damage only, it seems, if pores are present. The importance of vacuum mixing would seem to be emphasised by the present results, since it decreases the number of pores and increases the average fatigue life (Murphy and Prendergast, 2000). Similarly, methods to reduce interfacial porosity, such as pre-heating the stem (Bishop et al., 1996) could be explored using the present analysis since the amount of conduction towards the stem would be reduced by increasing stem temperature; however, the beneficial effect of reduced interface porosity may be offset by the likely increase in magnitude of the temperature pulse. The use of lower exotherm cements, which would result in a reduced temperature pulse, could also be a method of reducing shrinkage stresses but experience with cements such as Boneloc (Biomet, UK) indicate that it may be difficult to develop such a cement with sufficient fatigue strength (Nilsson and Dalen, 1998; Walczak et al., 2000).

In conclusion, residual stress is a factor in pre-load cracking of cement mantles of orthopaedic joint replacements but requires additional factors, such as porosity, stress concentrations, or excessive heat generation, to initiate large cracks. The peak temperature reached for a given region of cement appears to have a significant effect on the level of residual stress that occurs, indicating that control of polymer/monomer ratios as well as ambient conditions during polymerisation are critical in controlling the phenomenon of pre-load cracking due to shrinkage.

Acknowledgements

The authors would like to acknowledge the Standards Measurements and Testing Program of the European

Comission (Contract No. SMT4-CT96-2076) for partly funding the work, and the MediLink project of the PRTLII programme for financial support of one of us (A.B. Lennon). The Department of Mechanical Engineering, University College Dublin, who kindly allowed us to use their optical comparator and prep room.

References

- Ahmed, A.M., Pak, W., Burke, D.L., Miller, J., 1982a. Transient and residual stresses and displacements in self-curing bone cement — Part II: thermoelastic analysis of the stem fixation system. *Journal of Biomechanical Engineering* 104, 28–37.
- Ahmed, A.M., Pak, W., Burke, D.L., Miller, J., 1982b. Transient and residual stresses and displacements in self-curing bone cement — Part I: characterization of relevant volumetric behaviour of bone cement. *Journal of Biomechanical Engineering* 104, 21–27.
- Baliga, B.R., Rose, P.L., Ahmed, A.M., 1992. Thermal modelling of polymerizing polymethylmethacrylate, considering temperature-dependent heat generation. *Journal of Biomechanical Engineering* 114, 251–259.
- Bishop, N.E., Ferguson, S., Tepic, S., 1996. Porosity reduction in bone cement at the cement-stem interface. *Journal of Bone and Joint Surgery* 78B (3), 349–356.
- Harrigan, T.P., Harris, W.H., 1991. A three-dimensional non-linear finite element study of the effect of cement-prosthesis debonding in cemented femoral total hip components. *Journal of Biomechanics* 24 (11), 1047–1058.
- Huiskes, R., 1980. Some fundamental aspects of human joint replacement. *Acta Orthopaedica Scandinavica* 185, 10–208.
- Huiskes, R., 1993. Mechanical failure in total hip arthroplasty with cement. *Current Orthopaedics* 7, 239–247.
- Huiskes, R., Verdonschot, N., 1997. *Biomechanics of Artificial Joints: The Hip*. In: Mow, V.C., Hages, W.C. (Eds.), *Basic Orthopaedic Biomechanics*, 2nd ed. Lippincott-Raven Publishers, Philadelphia, pp. 395–460.
- Kine, B.B., Novak, R.W., 1987. Acrylic and Methacrylic Ester Polymers. In: Mark, H.F., Bikales, N.M., Overberger, C.G., Menges, G., Kroschwitz, J.I. (Eds.), 2nd ed., *Encyclopedia of Polymer Science and Engineering*, Vol. 1. John Wiley & Sons, New York, pp. 234–299.
- Lennon, A.B., McCormack, B.A.O., Prendergast, P.J., 1998. Development of a physical model of a cemented hip replacement for investigation of cement damage accumulation. In: 11th Conference of the European Society of Biomechanics, *Journal of Biomechanics* 31 (Suppl. 1), 129.
- Lennon, A.B., Prendergast, P.J., 2001. Sub-task 6.3: Cement/metal interface fatigue damage accumulation. Final Report for contract SMT4-CT96-2076 Preclinical Testing of Cemented Hip Replacement Implants: Pre-normative Research for a European Standard. Nijmegen, The Netherlands.
- Mann, K.A., Bartel, D.L., Wright, T.M., Inghraffa, A.R., 1991. Mechanical characteristics of the stem-cement interface. *Journal of Orthopaedic Research* 9 (6), 798–808.
- McCormack, B.A.O., Prendergast, P.J., 1999. Microdamage accumulation in the cement layer of hip replacements under flexural loading. *Journal of Biomechanics* 32, 467–475.
- McCormack, B.A.O., Walsh, C.D., Wilson, S.P., Prendergast, P.J., 1998. A statistical analysis of microcrack accumulation in PMMA under fatigue loading: applications to orthopaedic implant fixation. *International Journal of Fatigue* 20, 581–593.
- Meyer, J.R., Lautenschlager, E.P., Moore, E.K., 1973. On the setting properties of acrylic bone cement. *Journal of Bone and Joint Surgery* 55-A, 149–156.
- Murphy, B.P., Prendergast, P.J., 2000. On the magnitude and variability of fatigue strength in acrylic bone cement. *International Journal of Fatigue* 22, 855–864.
- Nilsson, K.G., Dalen, T., 1998. Inferior performance of Boneloc (R) bone cement in total knee arthroplasty — a prospective randomized study comparing Boneloc (R) with Palacos (R) using radiostereometry (RSA) in 19 patients. *Acta Orthopaedica Scandinavica* 69 (5), 479–483.
- Raab, S., Ahmed, A.M., Provan, J.W., 1981. The quasi-static and fatigue performance of the implant/bone-cement interface. *Journal of Biomedical Materials Research* 15, 159–182.
- Saha, S., Pal, S., 1984. Mechanical properties of bone cement: a review. *Journal of Biomedical Materials Research* 18, 435–462.
- Shaffer, B.W., Levitsky, M., 1974. Thermoelastic constitutive equations for chemically hardening materials. *Journal of Applied Mechanics* 96, 652–657.
- Starke, G.R., Birnie, C., van den Blink, P.A., 1997. Numerical modelling of cement polymerisation and thermal bone necrosis. In: Middleton, J., Jones, M.N., Pande, G.N. (Eds.), *Third International Symposium on Computer Methods in Biomechanics and Biomedical Engineering*. Gordon and Breach, London, pp. 163–172.
- Tepic, S., Soltész, U., 1998. Fatigue of bone cement with simulated stem interface porosity. *Journal of Materials Science — Materials in Medicine* 9 (12), 707–709.
- Walczak, J.P., D'Arcy, J.C., Ross, K.R., James, S.E., Bonnici, A.V., Koka, S.R., Morris, R.W., 2000. Low-friction arthroplasty with Boneloc bone-cement — outcome at 2 to 4 yr. *Journal of Arthroplasty* 15 (2), 205–209.
- Whelan, M.P., Kenny, R.P., Cavalli, C., Lennon, A.B., Prendergast, P.J., 2000. Application of optical fibre Bragg grating sensors to the study of pmma curing. In: Prendergast, P.J., Carr, A.J., Lee, T.C. (Eds.), *Proceedings of the 12th Conference of the European Society of Biomechanics*. Royal Academy of Medicine in Ireland, Dublin, p. 252.

USE OF GRATING INTERFEROMETRY FOR VALIDATION OF FINITE ELEMENT MODELS AND TO INVESTIGATE RESIDUAL STRAIN IN POLYMETHYLMETHACRYLATE

A.B. Lennon^{1,2}, P.J. Prendergast¹, M.P. Whelan² and C. Forno²

1. ABSTRACT

Important physical aspects of cemented joint reconstructions, such as residual stress, cement viscoelasticity, porosity, and interdigitation of bone-cement interfaces, are often neglected in finite element models, with little knowledge of the effect of their omission. Grating interferometry was used to measure in-plane displacement of two designs of polymethylmethacrylate (PMMA) beam specimens loaded in three-point bending—(i) a plain PMMA beam, and (ii) a steel-PMMA-foam-PVC layered beam. Measurements were also taken when the load was removed. Displacements predicted from finite element models incorporating the aforementioned simplifications, were compared to the measured displacements for both specimens. No residual stress was evident in specimen (i) and good agreement was obtained between finite element and measured displacements. On the other hand specimen (ii) exhibited increasing displacements over time, even in the unloaded state. This indicates relaxation of residual cement stresses, and prevented good agreement between the finite element and measured displacements for this specimen. Porosity that is substantially smaller than the dimension of the cement layer did not significantly affect structural behaviour.

2. INTRODUCTION

The use of finite element modelling has become widespread in the analysis of orthopaedic joint replacements, with frequent attention focused on the stress experienced by the cement used to fixate the implant [1]. However, many physical features of the reconstruction, such as residual stress, viscoelasticity, porosity in the cement layer, and interdigitation of the bone-cement interface, are often omitted from the models. Studies that have investigated these features [2, 3, 4, 5, 6] are the exception rather than the rule.

The enclosed nature of the cement mantle around implants requires either indirect

KEYWORDS: Polymethylmethacrylate, optical strain measurement, residual stress, porosity.

¹ Bioengineering Group, Department of Mechanical Engineering, Trinity College, Dublin 2, Ireland

² Photonics Technologies and Diagnostics Sector — Institute for Systems Informatics and Safety, European Commission Joint Research Centre, Ispra (VA), Italy

validation, by measurement of strains on accessible surfaces [7, 8], or direct validation, by means of embedded strain gauges [9, 10]. The use of strain gauges, however, has some disadvantages—namely, that measurements are made only at discrete points of the structure, and the gauge can locally reinforce the region under investigation. This limits the value of comparison between finite element and experimental models. Optical techniques, on the other hand, involve little or no contact with the measured surface and are full-field. A review of the application of several optical strain measurement techniques to biomechanics can be found in Orr and Shelton [11]. The use of *in vitro* models that expose the cement layer [12, 13, 14] offer an opportunity for the application of optical methods to measurement of PMMA deformation.

One such optical technique is grating interferometry [15,16], in which interferograms of surface displacement are produced by the interaction of an applied specimen grating and a reference grating. The reference grating is produced by the interference of two collimated wavefronts of coherent light at the surface of the specimen. Fringe formation is based on the combination of two diffracted wavefronts, but a simple explanation can be made in terms of *moiré*. Using the intensity distributions obtained from several interferograms at known phase intervals from each other, the phase ambiguity can be removed from the measurements [17] and a full-field distribution of surface deformations can be obtained by scaling phase values to displacements. The sensitivity and full-field nature of the technique is such that it is very suitable for investigating complex displacement and strain distributions.

By comparing displacement distributions from finite element models that do not incorporate residual stress, viscoelasticity, porosity or interdigitated interfaces, with measured displacement distributions from physical models, the effect of the omission of these physical features can be investigated.

3. METHODS

Two designs of specimen (Fig. 1) were investigated under 3 point bending configurations. The first was a beam made only of PMMA (specimen (i)), and the second was a composite beam of steel, PMMA, polyurethane foam and PVC (specimen (ii)). The latter specimen was representative of a layered structure such as a hip reconstruction. Hand-mixed Simplex Rapid acrylic cement was used for both specimens. The composite specimen contained a large subsurface pore (Fig. 1).

Gratings of 1200 lines/mm were applied to the specimens shortly after the cement had cured and cooled. A three-point-bend rig was used which had a mirror system attached to it in such a way as to allow measurement of both horizontal and vertical displacements. Phase-stepping of the laser beam was achieved using a piezoelectric actuator attached to an adjustable mirror frame. Images of the interferograms were captured at the exit pupil of the interferometer using a CCD camera. Fringe Application Ver. 3.2 (Warsaw, Poland) was used to calculate the wrapped and unwrapped phasemaps as well as converting phase to displacement. A five-step algorithm was used which required the phase-steps to be increments of $\pi/2$.

Both specimens were tested in 3-point bending with a 100 N load. To investigate shrinkage and stress relaxation phenomena, interferograms were recorded for the composite beam at three different times (1, 10 and 21 days) without any load being

applied. This was not necessary for specimen (i) as no change in the displacements were observed over time while load was removed.

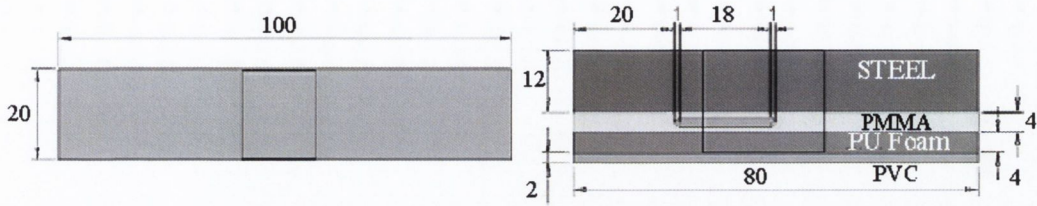


Figure 1: Schematic of Specimen (i) (left) and composite (right) beams. Grating areas shown as rectangular outlines in central region of each specimen. Dimensions in mm. Out-of-plane thickness is 8 mm.

Finite element models were constructed in ANSYS 5.4 (SAS IP, Inc., USA) for both of the specimens. Eight-node, isoparametric brick elements were used for all analyses. Material properties are shown in Table 1. The pore was assumed to be 20 mm long (the length of the visibly translucent portion of the cement), 4 mm wide, and 2 mm high and to sit centrally in the cement layer. As its dimensions were the same order of magnitude as those of the cement layer, the presence of the large pore in the mesh was acceptable.

Material	Steel	PMMA	PU Foam	PE	PVC
E (GPa)	215	2.4	0.15	0.85	2
ν	0.3	0.33	0.3	0.33	0.3

Table 1: Material properties used in finite element analysis

4. RESULTS

Displacements in the horizontal direction for specimen (i) at 100 N load are shown in Figure 2. Comparison of the displacement maps shows that the finite element analysis predicted displacements to within approximately $0.75 \mu\text{m}$ of the measured displacements.

For the loaded composite, the FEM predicts a horizontal displacement value which is $2.1 \mu\text{m}$ less than in the actual specimen (Figure 3). Some qualitative difference in the distribution can also be seen, particularly in the lower regions of the plots. Vertical displacements are shown in Figure 4. The displacement range predicted is $-2.8 \mu\text{m}$ compared to the measured range of $-7.3 \mu\text{m}$. However, the general trend of increasing displacement magnitude from left to right is captured in the finite element result. Again the lower portion of the measured displacement plot differs qualitatively (particularly in the lower left corner) due to the presence of the pore.

Figures 5 and 6 show comparisons between interferograms of the specimen in an unloaded state at three different times after the application of the grating: 1, 10, and 21 days. It can be seen that for both directions there is increased displacement over time, indicating a stress relaxation process. The distributions also become more complex over time. After 3 weeks no more change in the interferograms was apparent. Figure 7 shows unwrapped and scaled phasemaps of the displacements for the last time point in Figures 5 and 6.

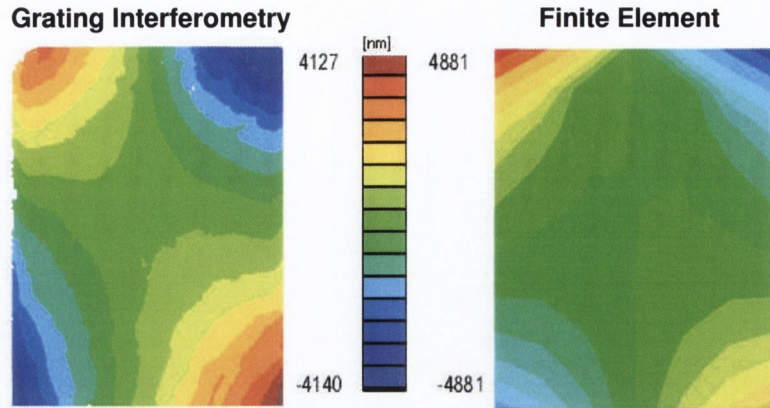


Figure 2: Measured and predicted *horizontal* displacements for specimen (i), the PMMA beam, at 100 N load. Displacements in nm.

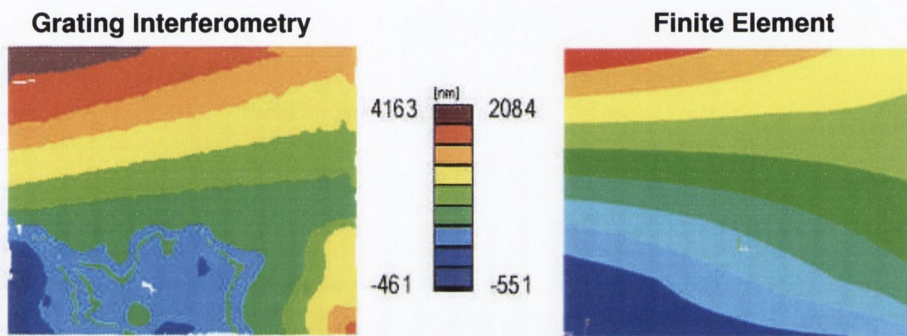


Figure 3: Measured and predicted *horizontal* displacements for specimen (ii), the composite beam, at 100 N load. Displacements in nm.

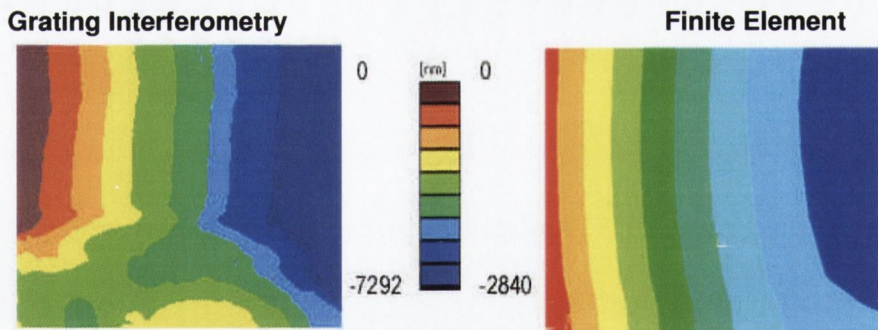


Figure 4: Measured and predicted *vertical* displacements for specimen (ii), the composite beam, at 100 N load. Displacements in nm.

5. DISCUSSION

Agreement between predicted and measured displacements is good for specimen (i), showing that the finite element method is suitable for modelling the deformation behaviour of bulk cement (Figure 2). Specimen (ii) does not show similar quantitative agreement but does achieve some similarity in a qualitative sense (Figures 3 and 4). However, there are still significant differences between predicted and measured behaviour. Comparison of Figure 7 with Figures 3 and 4, shows that the regions differing most are those near the pore. Increasing displacements over time, with no applied load, were exhibited for the entire cement and interdigitated cement-foam layers (Figures 5 and 6). These are almost certainly due to relaxation of residual stress. The magnitude of these displacements indicates that the omission of residual stress and its

relaxation are the main reasons for discrepancy between measured and predicted results. As no relaxation was apparent in specimen (i), in which the cement was not constrained from shrinking, it is likely that the main factor in generation of residual stress is thermal shrinkage. Approaches based on such an assumption were used by Huiskes [2], and Mann *et al.* [3]. A study by Ahmed *et al.* [4], however, suggests that stress is also generated during the curing phase, mainly due to thermal expansion. The pore had the further effect of increasing the complexity of displacements generated by stress-relaxation. Poor knowledge of the exact morphology of the subsurface pore, the possible existence of other subsurface pores, and the interdigitated nature of the 'bone'-cement interface, could also lead to discrepancy between predicted and measured displacements. Omission of small pores (as are commonly found in hand-mixed cement) did not affect the results from the finite element model as their presence is normally accounted for in the material properties for the relevant cement. The effect of the interdigitated interface led mainly to an increase in complexity of the displacement distributions near the interface and became more apparent as displacements were induced by stress-relaxation (Figures 5 and 6).

6. CONCLUSIONS

- (i) Residual stresses occur in the immediate post-operative period and relax in approximately 3 weeks. This has consequences for damage accumulation from pores and interfaces in the immediate post-operative period.
- (ii) Features, such as porosity, approaching the dimension of the cement thickness should not be neglected for stress analysis.
- (iii) Grating interferometry indicates that thermal shrinkage may have a significant effect on stress levels in bone cement. This fact needs to be borne in mind when reconciling theoretical and experimental analyses of damage accumulation in total joint arthroplasty.



Figure 5: Horizontal component interferograms for three different times after grating application for specimen (ii), the composite beam. Time is in days

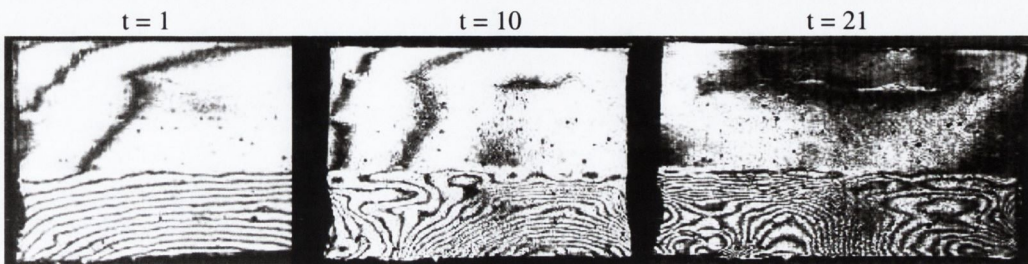


Figure 6: Vertical component interferograms for three different times after grating application for specimen (ii), the composite beam.

7. ACKNOWLEDGEMENTS

We acknowledge financial support from the Standards Measurements and Testing Program of the European Commission (SMT4-CT96-2076). The first author also acknowledges a STAGIARE contract from the Photonics Technologies and Diagnostics Sector - ISIS, European Commission Joint Research Centre, Ispra (VA), Italy, where the experimental work was undertaken.

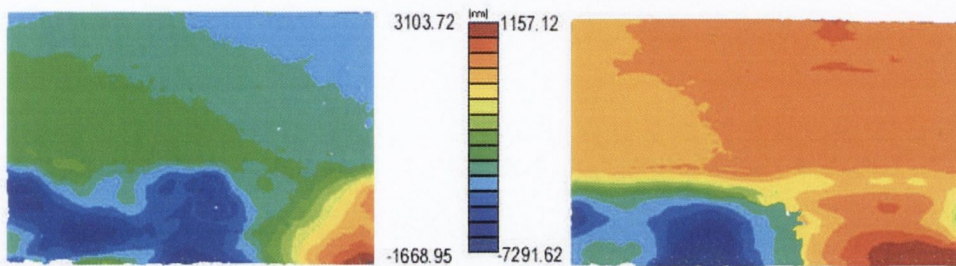


Figure 7: Horizontal (left) and vertical (right) displacements for composite beam due to relaxation of cement residual stresses

8. REFERENCES

1. Prendergast, P.J., Finite element models in tissue mechanics and orthopaedic implant design. *Clin. Biomech.*, 1997, Vol. 12, 343–366.
2. Huiskes, R. Some fundamental aspects of human joint replacement. *Acta Orthop. Scand.* 1980
3. Mann, K.A., Bartel, D.L., Wright, T.M., and Inghraffa, A.R. Mechanical characteristics of the stem-cement interface. *J. Orthop. Res.*, 1991, Vol. 9, 798–808.
4. Ahmed, A.M., Pak, W., Burke, D.L., and Miller, J. Transient and residual stresses and displacements in self-curing bone, Parts I and II. *J. Biomech. Eng.*, 1982, Vol. 104, 21–37.
5. Harrigan, T.P. and Harris, W.H. A three-dimensional non-linear finite element study of the effect of cement-prosthesis debonding in cemented femoral total hip components. *J. Biomech.*, 1991, Vol. 24, 1047–1058.
6. Mann, K.A., Werner, F.W., and Ayers, D.C. Modeling tensile failure at the cement-bone interface using non-linear fracture mechanics. In *Trans. 42nd ORS*, 1996, 519.
7. Rohlmann, A., Mössner, U., Bergmann, G., and Kölbl, R. Finite-element-analysis and experimental investigation in a femur with hip endoprosthesis. *J. Biomech.*, 1983, Vol. 16, 727–742.
8. Akay, M. and Aslan, N. Numerical and experimental stress analysis of a polymeric composite hip joint prosthesis. *J. Biomed. Mat. Res.*, 1996, Vol. 31, 167–182.
9. Cristofolini, L. and Viceconti, M. Development and validation of a technique for strain measurement inside PMMA. *J. Strain Anal. Eng. Des.*, in press.
10. Colgan, D., McTague, D., O'Donnell, P., and Little, E.G. Three-dimensional embedded strain gauge analysis of the effect of collared versus collarless prosthesis on cement mantle stresses in a femoral model of a total hip replacement. *J. Strain Anal.*, 1996, Vol. 31, 329–339.
11. Orr, J. F. and Shelton, J. C. *Optical Measurement Methods in Biomechanics*. Chapman and Hall, 1997.
12. McCormack, B.A.O., Prendergast, P.J., and Gallagher, D.G. An experimental study of damage accumulation in cemented hip prostheses. *Clin. Biomech.*, 1996, Vol. 11, 214–219.
13. McCormack, B.A.O., and Prendergast, P.J. Microdamage accumulation in the cement layer of hip replacements under flexural loading. *J. Biomech.*, 1999, Vol. 32, 467–475.
14. McCormack, B.A.O., Walsh, C.D., Wilson, S.P., and Prendergast, P.J. A statistical analysis of microcrack accumulation in PMMA under fatigue loading: applications to orthopaedic implant fixation. *Int. J. Fatigue*, 1998, Vol. 20, 581–593.
15. Post, D. Moiré Interferometry, In: *Handbook of Experimental Mechanics*, ed. A.S. Kobayashi, Soc. Exp. Mech., 1987.
16. Forno, C. and Whelan, M. JRC Report, ISIS/SAIA, 11/99
17. Creath, K. Phase-measurement interferometry techniques. In *Progress in Optics XXVI*, Elsevier Science Publishers B.V., 1988, 349–393.

Evaluation of Cement Stresses in Finite Element Analyses of Cemented Orthopaedic Implants

A. B. Lennon

P. J. Prendergast

Department of Mechanical Engineering,
Trinity College,
Dublin 2, Ireland

Stress analysis of the cement fixation of orthopaedic implants to bone is frequently carried out using finite element analysis. However, the stress distribution in the cement layer is usually intricate, and it is difficult to report it in a way that facilitates comparison of implants for pre-clinical testing. To study this problem, and make recommendations for stress reporting, a finite element analysis of a hip prosthesis implanted into a synthetic composite femur is developed. Three cases are analyzed: a fully bonded implant, a debonded implant, and a debonded implant where the cement is removed distal to the stem tip. In addition to peak stresses, and contour and vector plots, a stressed volume and probability-of-failure analysis is reported. It is predicted that the peak stress is highest for the debonded stem, and that removal of the distal cement more than halves this peak stress. This would suggest that omission of the distal cement is good for polished prostheses (as practiced for the Exeter design). However, if the percentage of cement stressed above a certain threshold (say 3 MPa) is considered, then the removal of distal cement is shown to be disadvantageous because a higher volume of cement is stressed to above the threshold. Vector plots clearly demonstrate the different load transfer for bonded and debonded prostheses: A bonded stem generates maximum tensile stresses in the longitudinal direction, whereas a debonded stem generates most tensile stresses in the hoop direction, except near the tip where tensile longitudinal stresses occur due to subsidence of the stem. Removal of the cement distal to the tip allows greater subsidence but alleviates these large stresses at the tip, albeit at the expense of increased hoop stresses throughout the mantle. It is concluded that a thorough analysis of cemented implants should not report peak stress, which can be misleading, but rather stressed volume, and that vector plots should be reported if a precise analysis of the load transfer mechanism is required. [DOI: 10.1115/1.1412452]

1 Introduction

Advances in prosthesis design have the ultimate objective of increasing implant longevity so that one replacement will suffice for the remaining life of the patient. However, this has not yet been achieved despite the considerable advances in prosthesis design and fixation technique. The rate of revision is significantly dependant on implant design; for example, one study of hip prostheses reports a survival rate, at six years, of 86.9 percent (± 21.1 percent) for the Müller design and 96.1 percent (± 1.4 percent) for the Stanmore design [1]. The predominant failure mode for cemented hip reconstruction is aseptic loosening of the femoral stem [2]. Usually, loosening is caused by fatigue failure of the cement mantle under cyclic loading. To reduce loosening rates, the mechanical integrity of the cement must be maintained for as long as possible.

Maintaining the mechanical integrity of the mantle is not simply a matter of reducing the *peak* stress in the cement mantle or on the cement/bone and cement/prosthesis interfaces, although this criterion can be used to optimize a stem profile [3]. Mechanical integrity can only be maintained if the overall stress within the mantle is kept below some threshold over time. Many design concepts have been proposed to achieve this, including polished tapered stems to achieve support via a wedging action [4], stems pre-coated with polymethylmethacrylate to maintain stem/cement bonding [5], and stems with various surface features such as dimples, undercuts, etc. [6]. Each design concept creates a signifi-

cantly different distribution of stress within the mantle [7]; however these data cannot, as yet, be correlated directly with the likelihood of implant loosening.

In theory, finite element modeling is the ideal tool for determining the stresses in the cement and hence the durability of the implant fixation. However, a significant problem is that the stress distribution in a cement mantle around an orthopaedic implant is very intricate, and furthermore, the influence of cement porosity may dominate the effect of the stress to a degree that failure may not occur at the site of peak stresses in the cement mantle, but rather may occur where the pores are largest [8]. In this respect, the peak stress may give an incorrect picture of the potential durability of a cemented fixation since it only occurs in a very small volume of the cement mantle. Finally, there are several problems in reporting of cement stresses, viz.

(i) the critical peak stresses may occur at singularities in the stress field,

(ii) high peak stresses may be dissipated by localized cement failure (damage formation or creep) *in vivo* so that they will not initiate failure. In this way, stresses may be distributed away from high stress regions so that through-mantle cracking may not finally occur in the region initially under the peak stress [9].

One approach used to solve the first problem is to use a non-local definition of stress [10], or to determine the volume of cement stressed above a certain level [11]. Stolk et al. [12] showed that volume percentages of cement above a specified stress tended to converge quickly with mesh density. To address the second problem, Verdonschot and Huiskes [13] used continuum damage mechanics to model the progressive failure of the cement due to fatigue, and predicted that high cement stresses exist predomi-

Contributed by the Bioengineering Division for publication in the JOURNAL OF BIOMECHANICAL ENGINEERING. Manuscript received by the Bioengineering Division August 24, 2000; revised manuscript received July 10, 2001. Associate Editor: R. T. Hart.

nantly in the early part of the life of the replacement, after which they are dissipated—the damaged cement, being less stiff, allows a “stress bypass” to the surrounding undamaged cement, thereby dissipating the peaks in cement stress. They also found that debonding can affect peak stresses [13]. In femoral hip replacement, the consequences of debonding can be further complicated by the presence or absence of cement distal to the cement tip, since subsidence of a stem can be increased when no distal cement exists to support it. A further approach to avoiding the problems associated with using peak values when reporting cement stress is to relate volumetric distribution of cement with the probability-of-failure within a specified time frame, by using cement volumes at specified stress levels the probability of that volume failing within a certain lifetime can be estimated [11]. Furthermore, the use of the average stresses for an element diminishes the influence of any singularities in the stress field.

In this paper, the stress field in the cement mantle surrounding a hip replacement is analyzed with the goal of evaluating stress-based criteria used to predict durability of orthopaedic implant fixation. The stress is quantified in three ways; peak stress, stressed volume, and probability-of-failure; and it is hypothesized that each measure may lead to differing conclusions regarding durability of the fixation. If this is true, then results reported in the literature for comparing orthopaedic implants by finite element analysis need careful interpretation.

2 Methods

The finite element models were generated using ANSYS finite element software (Rhode Island, USA). The standardized femur was used as a basis for a finite element model of a composite femur [14]. A combination of automatic and manual meshing was used to generate an 11,807 element model of the proximal half of the femur. An IGES file of a tapered stem prosthesis and its cement mantle was placed within the composite femur geometry. This prosthesis has a straight lateral border, and a medial border that was straight distally and curved proximally; it had a wedge-shaped cross section. Figure 1 shows a view of the mesh for the prosthesis and bone. Three cases of prosthesis fixation were examined in the study: (i) a fully bonded stem–cement interface, (ii) a debonded stem–cement interface with friction, retaining the cement distal to the stem tip, and (iii) a debonded stem–cement interface with friction, with the distal cement removed.

Node-to-surface contact was used to model debonding of the metal stem from the cement. Both sticking (elastic) and sliding (inelastic) friction were included. The coefficient of friction used was 0.32 and was determined from a pin-on-plate sliding test (Dr.-Ing. M. Pfeleiderer, De Puy Johnson and Johnson, U.K., personal communication).

The loading applied was taken from the work of Bergmann et al. [15], and Duda et al. [16]. As only the proximal femur was used and loads had to be applied in the same coordinate system for intact and implanted femurs, a different but comparable coordinate system from that of Bergmann et al. was used¹—this coordinate system was then assumed to be equivalent to that defined in [15] for the purpose of defining loads. Using Bergmann et al. [15], the maximum gait resultant force (F) on the head of the femur is given by 4.6 times body weight (BW) at 45 percent of the gait cycle. This can be resolved into

¹The z axis used had a femoral axis defined by connecting the “point where the femoral midline crosses the midsection of the femoral shaft” to the “saddle point between the greater trochanter and femoral head”; see Fig. 1. This resulted in a long axis for the standardized femur that was approximately parallel to one that would have been defined using the intersection of the femoral midline with the intercondylar notch and neck axis (as proposed by Bergmann et al. [15]). The femur was then rotated about this axis until the neck axis formed an angle in the transverse plane that was equivalent to the one it would form with an x axis passing through the medial and lateral condyles (as defined by Bergmann et al. [15]). This allowed the $x-z$ plane to be defined and the y direction was defined using the right-hand rule.

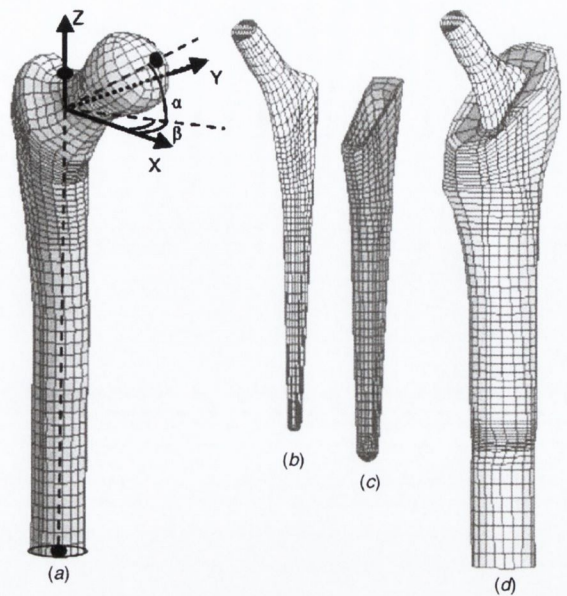


Fig. 1 Finite element meshes of: (a) intact femur, (b) hip prosthesis femoral component, (c) cement mantle, and (d) implanted femur. Note that x axis points medially, y anteriorly, and z superiorly. α is the angle the neck axis makes with its own projection onto the transverse ($x-y$) plane. β is the angle the neck axis makes with the x axis in the transverse ($x-y$) plane.

$$F_x = -1.80BW, \quad F_y = -0.74BW, \quad \text{and} \quad F_z = -4.17BW$$

where the xyz axes are indicated in Fig. 1.

At 45 percent of the gait cycle, a simplified set of active muscles are the abductor muscles, located on the greater trochanter (*Gluteus medius* and *Gluteus minimus*), and the ilio-tibial band (*Gluteus maximus* and *Tensor fascia latae*). In this study, body mass was taken as 70 kg giving the joint and muscle forces shown in Table 1. All nodes on the most distal section (Fig. 1) were fully restrained against displacements in all directions.

For the prosthesis, Young's modulus equals 210 GPa, Poisson's ratio equals 0.3. For the cement, Young's modulus equals 2.28 GPa, Poisson's ratio equals 0.3. Table 2 shows the material properties for the composite femur; those of real bone are shown alongside for comparison.

Table 1 Joint and muscle load magnitudes applied to the finite element model

Force Component	Joint (N)	Gluteus Medius (N)	Gluteus Minimus (N)	Ilio-tibial band (N)
F_x	1236	259	279	59
F_y	508	160	269	74
F_z	2864	319	134	58

Table 2 Elastic properties for composite and real bone

	Composite material properties		Real bone material properties	
	Young's modulus (GPa)	Poisson's ratio	Young's modulus (GPa)	Poisson's ratio
Cortical Bone	11.5	0.4	17	0.33
Cancellous bone	0.413	0.3	1.5	0.33

3 Results

3.1 Stress Generated in the Bone Cement

3.1.1 Contours of Cement Stress. For the case of a completely bonded cement/metal interface, the stresses are generally low, with highest stress regions occurring on the medial and lateral sides, with only the proximal and distal anterior regions experiencing comparable tensile stress (Fig. 2(a)). Debonding increases the size of regions experiencing higher stress, especially the proximal medial side, and the region surrounding the distal tip where the peak occurs (Fig. 2(b)). Removal of the cement distal to the stem tip obviously eliminates this peak; the same stress distribution as the previous case occurs except that there is a slight increase in the size of the highly stressed proximal medial region (Fig. 2(c)).

3.1.2 Direction of Stress in Cement Mantle. The loading behavior can be better understood when the direction of the maximum principal stress is considered. For the bonded case, the cement mantle is subjected to a combination of bending (shown by the longitudinally oriented lateral tensile stresses in the middle

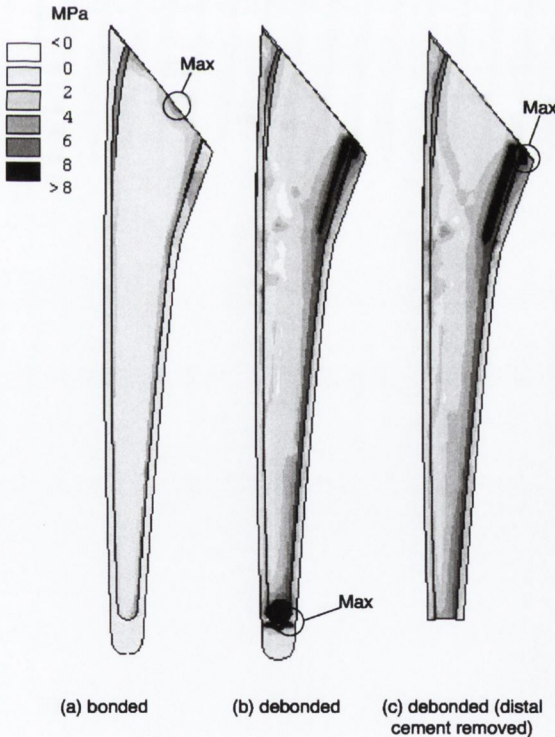


Fig. 2 Comparison of maximum tensile principal stresses in the anterior half of the cement using the scale for the bonded case. The stresses are extrapolated from the integration points and averaged at the nodes. Cases are: (a) bonded, (b) debonded, and (c) debonded with distal cement removed.

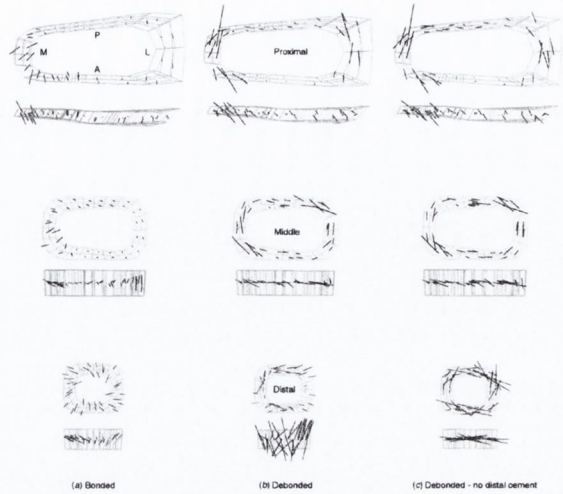


Fig. 3 Tensile maximum principal stress vectors in proximal, middle, and distal sections of cement mantle for: (a) bonded, (b) debonded, and (c) debonded with distal cement removed. The proximal and distal sections were taken a small distance away from the layers containing the peak stresses so as to give a more representative illustration of principal stress vectors (this is why the anterior vectors in (a) and distal vectors in (b) do not appear as maxima). P=posterior, M=medial, A=anterior, and L=lateral.

and distal sections of Fig. 3(a)) and a small degree of wedging action (shown by the hoop components in the lateral region of the proximal section and posterior side of the middle section; see Fig. 3(a)).

Debonding fundamentally alters the stress state in the cement mantle; the bending action is replaced by a wedging action of the prosthesis as illustrated by the increase in cement hoop stress in the proximal and middle sections (Fig. 3(b)). However, the cement surrounding the distal tip is subjected to significant longitudinal tensile stresses; these are not due to bending, however, but are a result of a tensile force exerted by the cement below the stem tip on the cement surrounding the tip (Fig. 3(b)). This is caused by the distal displacement of the prosthesis. The removal of the distal cement relieves the high longitudinal stresses in this region. This also allows greater distal displacement of the stem; this was verified by calculating the displacement of the tip, which increased from 122 μm for the debonded case to 209 μm for the case with distal cement removed. When the distal cement is removed, significant hoop stresses appear in the distal region; see Fig. 3(c).

3.1.3 Peak Cement Stresses. The peak tensile cement stress for the bonded prosthesis was 7.6 MPa (Table 3). It was located on the anterior side of the proximal region (Fig. 2(a)) and was oriented primarily in the radial direction (Fig. 3(a)). Debonding increases the magnitude of the peak tensile stress to 38 MPa (Table 3), shifting the site of the maximum to the region surrounding the distal tip of the stem (Fig. 2(b)) in a predominantly longi-

Table 3 Comparison of results of each criterion

Measure	Bonded	Debonded	No Distal Cement
Peak Stress (MPa)	7.6	38.0	15.8
% Volume > 3 MPa	3.0	21.1	25.8
% Volume Pf = 1.0	0.0	0.35	0.18

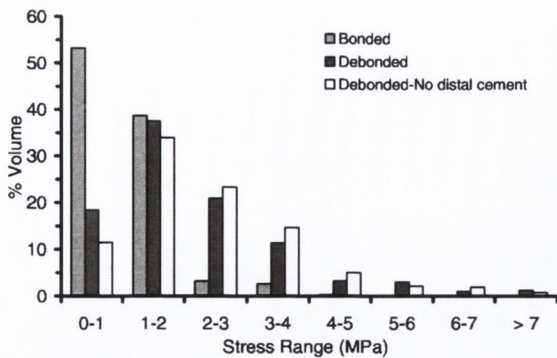


Fig. 4 Distribution of percent volume of cement over a stress range of 0–8 MPa, for the three implanted models

tudinal direction (Fig. 3(b)). Removal of the cement distal to the stem tip reduces the peak tensile stress to 15.8 MPa (Table 3) and moves it to the medial proximal region (Fig. 2(c)) and a hoop orientation (Fig. 3(c)). The scale of the changes in the stress is considerable, as illustrated in Table 3.

3.1.4 Volume of Cement at a Given Stress. For the bonded stem, much of the cement is stressed within the 0–2 MPa range, as shown by the contour plots in Fig. 2(a). To examine the distribution of stress quantitatively within the cement mantle, the average stress for each element in the cement mantle was recorded and the elements were divided into groups with stress ranges between 0 and 1 MPa, 1 and 2 MPa, etc. The total volume of elements for each stress range could then be plotted, as shown in Fig. 4. An increase in the proportion of cement experiencing higher stresses due to debonding is well illustrated using this method. Furthermore, if a stress level, e.g., of 3 MPa, is considered critical, it is clearly shown that the volume of endangered cement is greater when the distal cement is removed (~26 percent for debonded stem with distal cement removed compared with ~21 percent for debonded stem with cement distal to the tip, as illustrated in Table 3). This result would not be expected from the peak stresses only.

3.1.5 Probability-of-Failure. In order to compare the results from the different models, a probability-of-failure analysis was performed on the cement mantle. A relationship between the probability-of-survival at 10 million cycles and the applied stress was used to calculate the probability-of-survival for each element of the cement mantle, based on the experimental data from Murphy and Prendergast [17]. The regression polynomial has the following form:

$$P_S = -0.0005\sigma^3 + 0.0202\sigma^2 - 0.3304\sigma + 1.8365 \quad (1)$$

This regression curve applies within the stress range of 3–11 MPa. The probability-of-failure can then be expressed as

$$P_F = 1 - P_S \quad (2)$$

The percentage volume of cement was then plotted against probability-of-failure to compare the three cases of fixation.

This approach gives interesting results because it becomes apparent that, for the bonded stem, none of the cement has a failure probability greater than 0.4, whereas debonding shifts a small but significant amount of the cement towards higher failure probabilities (Fig. 5). The volume fractions of cement predicted to fail within 10 million cycles (i.e., $P_F=1$) for the cases of debonding, and debonding without distal cement, were 0.35 percent and 0.18 percent (Table 3), respectively. Therefore, while the debonded stem with distal cement removed has the highest percentage of cement above the failure threshold (i.e., percent volume >3 MPa)

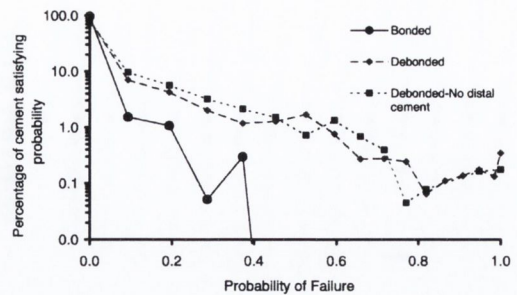


Fig. 5 Percentage volume of cement satisfying probability-of-failure survival at 10 million cycles ($P_F=1$ predicts failure within 10 million cycles and $P_F=0$ predicts survival for the same time period)

at 10 million cycles, it is the debonded stem retaining cement distal to the tip which has the highest percentage predicted to fail (i.e., $P_F=1$) within the same period of loading.

4 Discussion

This paper presents a detailed examination of the stress in the cement mantle around a hip prosthesis. The primary question of interest is how best to report the stress pattern so that the results can be interpreted in a way that allows intercomparison of implants. This is important if finite element models are to achieve their potential as pre-clinical testing tools [6]. To investigate this question, a finite element model based on a composite femur bone was used. The suitability of the composite femur has been demonstrated using experimental methods by Cristofolini et al. [18] who showed that no significant differences in mechanical behavior were found between composite femora and two groups of cadaveric specimens, while the inter-femur variability for the composite specimens was 20–200 times lower than the cadaveric groups; this allows reduction in required sample sizes for mechanical testing while increasing sensitivity in characterising differences in behavior. Although the composite femur can never match the behavior of an individual bone exactly, its use is seen as central to the pre-clinical analysis of new implant designs since it offers a consistent geometry for comparing results across studies. Furthermore, a strain gauge analysis was carried out (see appendix), in which it was found that the finite element model predicts strains reasonably similar to those found experimentally. Similar results have been obtained by others [19,20]. It should be noted, however, that all materials were assumed to be isotropic in the present study. In reality both the fiber-reinforced epoxy of the composite femur and the cancellous and cortical bone are known to be anisotropic. The strain gage validation showed that acceptable results can be achieved without including the directional material properties, as has been shown previously by Vichnin and Batterman for a cadaveric femur [21].

Apart from the assumption of material isotropy a number of other simplifications have been made. Firstly, only one load case (45 percent of the gait cycle during walking), has been included, even though this means that the higher stresses that may occur during stair climbing or stumbling are not included in the failure prediction. It is noteworthy that a recent study [22] has shown that walking accounts for approximately 80 percent of dynamic load cycles for the hip and so represents a significant contribution to the fatigue loading of the cement. Another simplification is that the interdigitation at the bone/cement interface is not included; this is commonly made in finite element models due to the difficulty in including sufficient geometric detail; the result is a smoothing of the stress distribution in this region. Since it is consistent for each model it should not affect the comparisons made between implants. The same argument applies to other limitations

such as the exclusion of residual stress, creep, and biological response of bone to stress shielding or thermal shock due to cement curing. Finally, a limitation to the use of stressed volumes and probability-of-failure as comparative criteria is that only cement mantles of very similar volumes can be correctly compared, i.e., mantles of much greater volume are likely to have greater volumes of cement with high failure probability and vice versa.

Debonding has a number of effects on the cement mantle that would not be captured by reporting of peak stresses alone. The increased wedging action of the stem is immediately apparent when examining the vector plots (Fig. 3). These clearly predict that greater hoop stresses occur in the bulk of the cement mantle for both debonded stems compared to the bonded case. The relief of the proximal anterior stresses is captured well by both stress contours and vectors while the increase in overall stress levels is apparent, qualitatively, from the contour plots and, quantitatively, from the shift in volumetric distribution of stress towards higher ranges (i.e., growth of volumes in the 2–7 MPa range, illustrated in Fig. 4). This shift toward higher stress explains the increase in probability-of-failure for the two debonded models. Because the maximum tensile stresses in these regions are in the hoop direction, damage is most likely to be in the form of radial cracking.

Peak stresses are likely to be dissipated by local damage and/or localized plastic deformation due to creep. Therefore, peak stresses may be considered to indicate damage initiation events in the early life of the joint replacement. The combination of magnitude and direction information for the peak tensile stresses therefore offers useful insight into possible sites of damage initiation or critical cracks. Cement failure probability, however, offers a more general indication of the performance of the prosthesis in the longer term. When combined with direction vectors and volumetric distribution of stress, a reasonably complete understanding of the possible long-term nature of damage accumulation in the cement mantle can be obtained. As the probability-of-survival of the cement mantle is based on the volume of cement above a specified stress level, it is less mesh sensitive than the peak stress [12]; this makes it better for comparing different designs of replacement. In other words, if two implants generate the same peak stress, then the implant with the greater volume of cement stressed to that level will have the greatest likelihood of failure. If peak stress is reported without volume this information is missed.

The novelty of the probability-of-failure measure is not so apparent in this particular case because of the near linear relationship between probability-of-failure and stress, in the stress range of interest here (see Eq. (1)), resulting in similar results to the stressed volume approach, but it contains a number of advantages: (i) it identifies, for a given lifetime, the appropriate stress levels that should be used in reporting stressed volumes; in this way the probability-of-failure adds information about lifetime and is therefore more meaningful than the stress data alone; (ii) comparison of contour plots is made more readily using probability-of-failure than stress because of a defined (zero to one) scale for generating displays and thus avoiding the need to rescale contours for models with different stress ranges (a common problem when attempting to compare contour plots from different studies); (iii) it may also be useful when attempting to convey results to clinicians who may better understand the significance of a probability-of-failure level than a stress magnitude.

The most important point of this paper for clinicians and implant designers is that the measures used to report the stresses in the cement mantle around orthopaedic implants can give quite different information, and could lead to different conclusions (Table 3). Reporting of peak stress alone would seem to be a particularly misleading measure of the durability of a cemented fixation. On the other hand, the approaches based on stressed volume (e.g., Fig. 4) and probability-of-survival (Fig. 5) show the general nature of the change in the distribution in stress; however, the fact that cement volume will be different for different designs needs to be accounted for. Finally, the stress vector plot (Fig. 3)

shows that polished implants (such as the Exeter prosthesis) have an entirely different load transfer mechanism than matt bonded implants.

Given the primary value of finite element modeling as a tool for intercomparison of implants [7,8,23–25], it is suggested by these results that stressed volume is the best way to compare stresses generated in cement mantles surrounding orthopaedic implants. This information could be supplemented with vector plots if a detailed comparison of the stress transfer mechanism is required.

Acknowledgments

This research was partly funded by the Higher Education Authority under the MediLINK program and partly by DePuy Johnson and Johnson, Leeds, U.K. Thanks are extended to Dr. Damien Lacroix for the strain gaging analysis.

Appendix

Experimental Collaboration of Numerical Results

Method. Rosette strain gages were used in this study. Three rosettes were applied at the proximal, middle, and distal levels on each of the anterior, posterior, medial and lateral surfaces, i.e., 12 rosettes in total, as shown in Fig. 6. The type of rosette used was EA-06-062RB-120 strain gage (Micro-measurements Group Inc., USA). The composite femur was held in a clamp, 44 mm distally, and was inclined at an angle of 20 deg in the coronal plane and rotated anticlockwise by 20 deg about the long axis of the femur. A compressive load was applied on the femoral head; see Fig. 7. This configuration provides strains not only on the medial and lateral surfaces but also on the anterior and posterior surfaces, and hence it can confirm the FE model more completely than a strictly physiological load. Loads of 0.3, 0.45, 0.6, and 0.9 kN were applied to the femoral head. Loading was repeated three times for each gage to assess repeatability of the measurement. For each

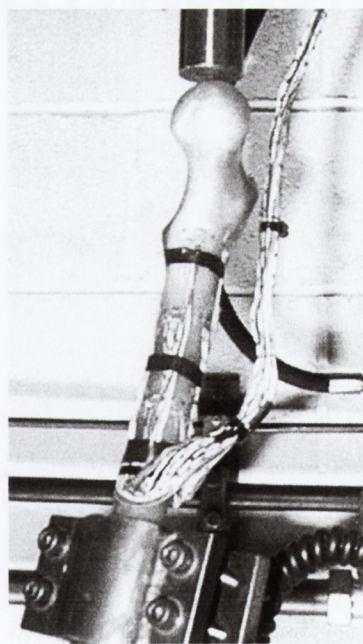


Fig. 6 A photograph of the strain gaged femur in the Instron testing machine. Note, there are four gages on each of the anterior, posterior, medial and lateral surfaces (i.e., 12 altogether)

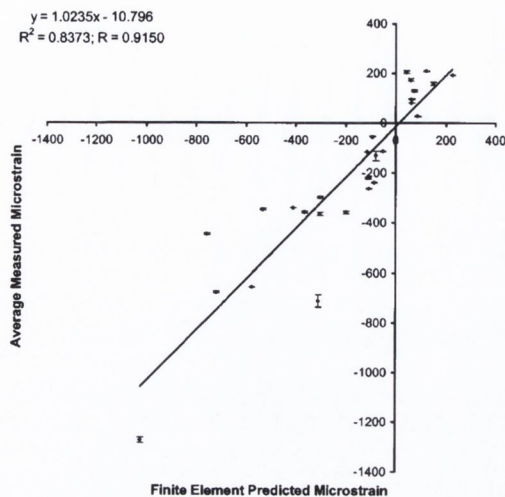


Fig. 7 Strain measured in the femur (strain gauges) versus strain predicted in the femur (finite element). Error bars show \pm two standard deviations of each mean gage measurement from three readings.

strain measurement the output voltage was measured before loading, during loading and after loading to check zeroing of the circuit.

Results. A plot of the measured microstrain against the predicted microstrain at each gage site shows that the predicted strain and measured strain are reasonably well correlated (Fig. 7). A linear regression analysis was performed to find the line of least-squares fit between the experimentally measured values and the calculated finite element prediction. If the measurements and predictions were equal at all points, all the data would lie on a line of slope equal to one. The slope of the line was found to be 1.02 and the y intercept found to be very low at -10.84 microstrain.

References

[1] Britton, A. R., Murray, D. W., Bulstrode, C. J., McPhearson, K., and Denham, R. A., 1997, "Pain Levels After Total Hip Replacement. Their Use as End-points for Survival Analysis," *J. Bone Jt. Surg.*, **79B**, pp. 93–98.
 [2] Malchau, H., and Herberts, P., 1998, "Prognosis of Total Hip Replacement. Revision and Re-revision Rate in THR. A Revision Risk Study of 148,359 Primary Operations," scientific exhibition presented at the 65th Annual Meeting of the American Academy of Orthopaedic Surgeons, New Orleans, LA.
 [3] Huiskes, R., and Boeklagen, R., 1989, "Mathematical Shape Optimization of Hip Prosthesis Design," *J. Biomech.*, **22**, pp. 793–804.
 [4] Lee, A. J. C., 1994, "Implants for Fixation With and Without a Collar," in:

Technical Principles, Design, and Safety of Joint Implants, G. H. Buchhorn and H.-G. Willert, eds., Hogrefe & Hubber, Bern, pp. 128–132.

[5] Harris, W. H., 1992, "Is It Advantageous to Strengthen the Cement Metal Interface and Use a Collar for Cemented Femoral Components of Total Hip Replacements," *Clin. Orthop. Relat. Res.*, **285**, pp. 67–72.
 [6] Prendergast, P. J., "Bone Prostheses and Implants," in: *Bone Mechanics Handbook*, (S. C. Cowin, ed.), Boca Raton, FL, Chap. 35.
 [7] Huiskes, R., 1990 "The Various Stress Patterns of Press-Fit, Ingrown, and Cemented Femoral Stems," *Clin. Orthop. Relat. Res.*, **261**, pp. 27–38.
 [8] Harrigan, T. P., Kareh, J. A., O'Connor, D. O., Burke, D. W., and Harris, W. H., 1992, "A Finite Element Study of the Initiation of Failure of Fixation in Cemented Femoral Total Hip Components," *J. Orthop. Res.*, **10**, pp. 134–144.
 [9] McCormack, B. A. O., and Prendergast, P. J., 1999, "Microdamage Accumulation in the Cement Layer of Hip Replacements Under Flexural Loading," *J. Biomech.*, **32**, pp. 467–475.
 [10] Stolk, J., Verdonshot, N., and Huiskes, R., 1999, "Management of Strain Fields Around Singular Points: Comparison of FE Simulations and Experiments," in: Middleton, J., Jones, M. N., and Pande, G. N., eds., *Proc. 4th International Symposium on Computer Methods in Biomechanics and Biomedical Engineering*, Gordon and Breach, pp. 57–62.
 [11] Verdonshot, N., and Huiskes, R., 1996, "Mechanical Effects of Stem Cement Interface Characteristics in Total Hip Replacement," *Clin. Orthop. Relat. Res.*, **329**, pp. 326–336.
 [12] Stolk, J., Verdonshot, N., and Huiskes, R., 1998, "Sensitivity of Failure Criteria of Cemented Total Hip Replacements to Finite Element Mesh Density," in: *11th Conference of the European Society of Biomechanics, Journal of Biomechanics*, **31**, pp. 165.
 [13] Verdonshot, N., and Huiskes, R., 1997, "The Effects of Cement-Stem Debonding in THA on the Long-Term Failure Probability of Cement," *J. Biomech.*, **30**, pp. 795–802.
 [14] Viceconti, M., 1997, curve2_3.igs, from: The ISB Finite Element Repository, Istituti Rizzoli; http://www.cineca.it/hosted/LTM/back2net/ISB_mesh.
 [15] Bergmann, G., Graichen, F., and Rohlmann, A., 1993, "Hip Joint Loading During Walking and Running, Measured in Two Patients," *J. Biomech.*, **26**, No. 8, pp. 969–990.
 [16] Duda, G., 1998, "Influence of Muscle Forces on Femoral Strain Distribution," *J. Biomech.*, **31**, pp. 841–846.
 [17] Murphy, B. P., and Prendergast, P. J., 2000, "On the Magnitude and Variability of the Fatigue Strength of Acrylic Bone Cement," *Int. J. Fatigue*, **22**, pp. 855–864.
 [18] Cristofolini, L., Viceconti, M., Capello, A., and Toni, A., 1996, "Mechanical Validation of Whole Bone Composite Femur Models," *J. Biomech.*, **29**, pp. 525–535.
 [19] McNamara, B. P., Cristofolini, L., Toni, A., and Taylor, D., 1997, "Relationship Between Bone-Prosthesis Bonding and Load Transfer in Total Hip Reconstruction," *J. Biomech.*, **30**, No. 6, pp. 621–630.
 [20] Stolk, J., Verdonshot, N., Cristofolini, L., Firmati, L., Toni, A., and Huiskes, R., 2000, "Strains in Composite Hip Joint Reconstructions Obtained Through FEA and Experiments Correspond Closely," *Trans. 46th Annu. Meet.—Orthop. Res. Soc.*, p. 515.
 [21] Vichnin, H. H., and Batterman, S. C., 1986, "Stress Analysis and Failure Prediction in the Proximal Femur Before and After Total Hip Replacement," *ASME J. Biomech. Eng.*, **108**, pp. 33–41.
 [22] Bergmann, G., and Duda, G., 1998, "Development of the Loading Configuration: Report for Contract SMT4-CT96-2076 Preclinical Testing of Cemented Hip Replacement Implants: Pre-normative Research for a European Standard," Nijmegen, the Netherlands.
 [23] Chang, P. B., Mann, K. A., and Bartel, D. L., 1998, "Cemented Femoral Stem Performance—Effects of Proximal Bonding, Geometry, and Neck Length," *Clin. Orthop. Relat. Res.*, **355**, pp. 57–69.
 [24] Prendergast, P. J., 1997, "Finite Element Models in Tissue Mechanics and Orthopaedic Implant Design," *Clin. Biomech.*, **12**, pp. 343–366.
 [25] Fagan, M. J., and Lee, A. J. C., 1986, "Materials Selection in the Design of the Femoral Component of Cemented Total Hip Replacement," *Clin. Mater.*, **1**, pp. 151–167.

Appendix B

Appended results

Bonded (i.e. Matt): Predictions for Specimen #2

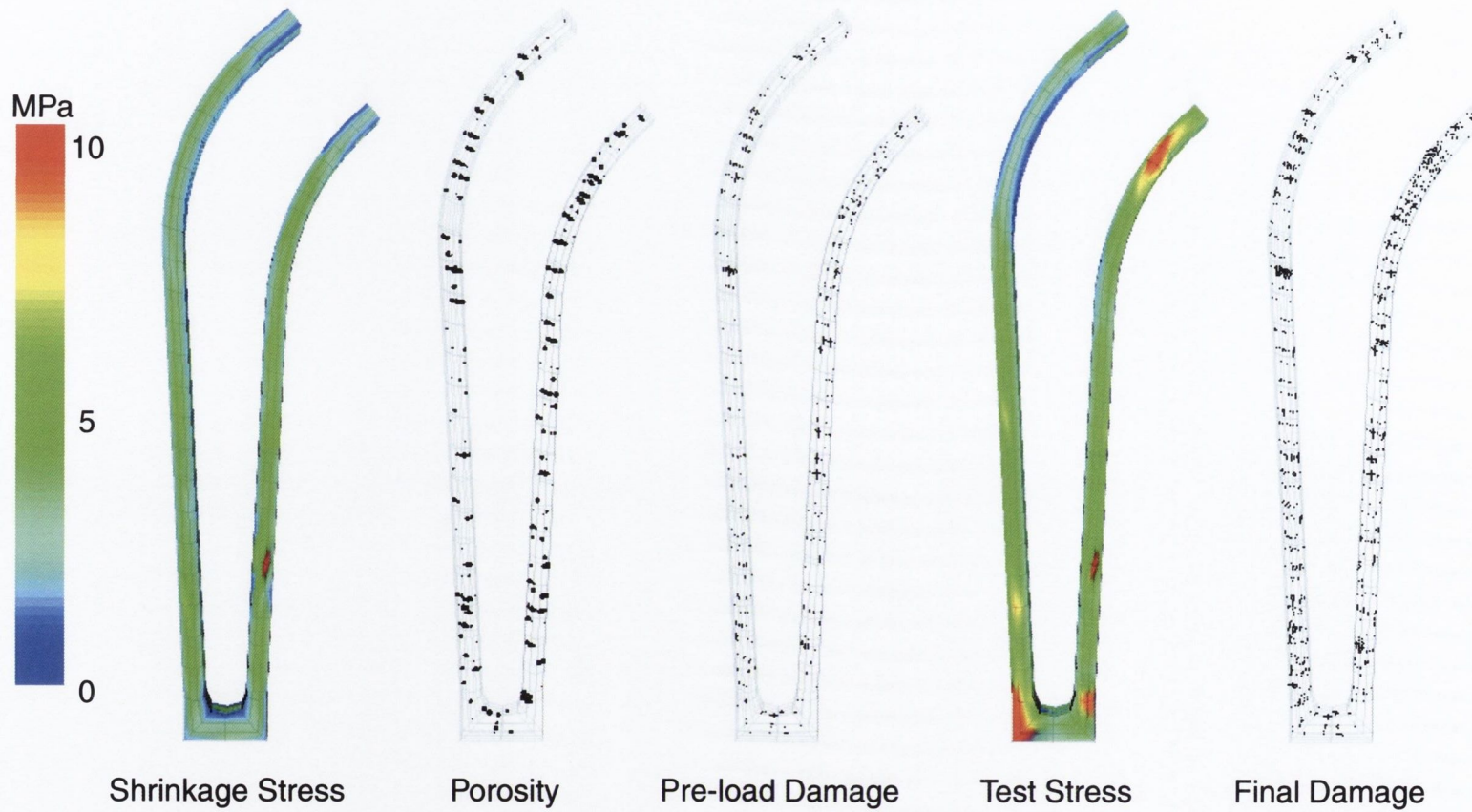


Figure B.1. Stress, porosity, and damage distributions for bonded specimen no. 2. After generation of the pore distribution (pores are plotted as spheres scaled to the volume fraction of each pore), the residual stresses were read in from the original thermoelastic analysis. The pore distribution was then used to calculate local stress concentrations in the stress field. This new shrinkage stress distribution was used to initiate damage around pores (damage data are plotted as principal damage vectors) and the test load was then applied. Damage accumulation was calculated according to the timestep scheme of section 3.2.4 for the superposed stress distributions from loading and shrinkage (the stress distribution shown is for the first cycle of testing).

Bonded (i.e. Matt): Predictions for Specimen #3

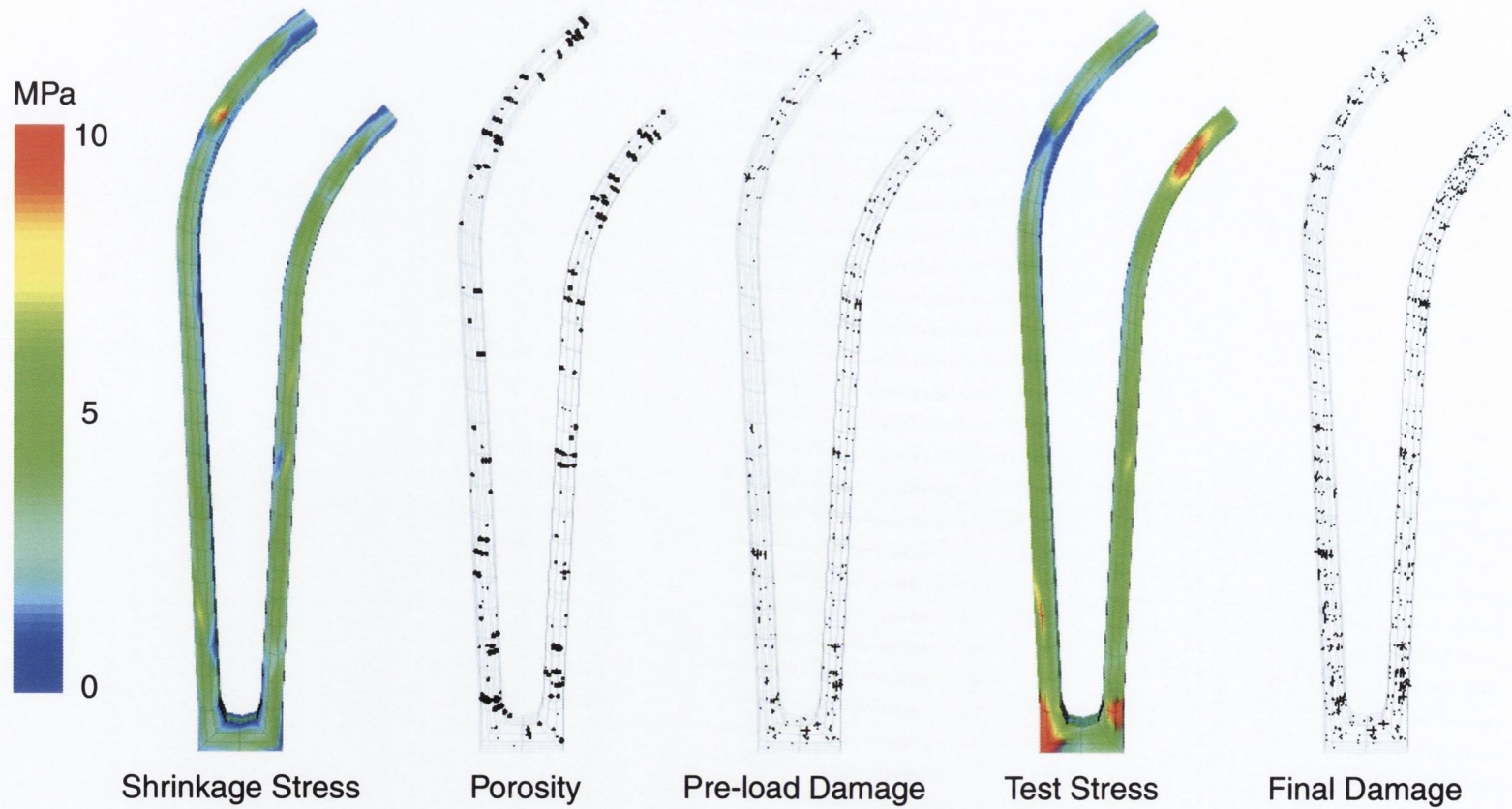


Figure B.2. Stress, porosity, and damage distributions for bonded specimen no. 3. After generation of the pore distribution (pores are plotted as spheres scaled to the volume fraction of each pore), the residual stresses were read in from the original thermoelastic analysis. The pore distribution was then used to calculate local stress concentrations in the stress field. This new shrinkage stress distribution was used to initiate damage around pores (damage data are plotted as principal damage vectors) and the test load was then applied. Damage accumulation was calculated according to the timestep scheme of section 3.2.4 for the superposed stress distributions from loading and shrinkage (the stress distribution shown is for the first cycle of testing).

Bonded (i.e. Matt): Predictions for Specimen #4

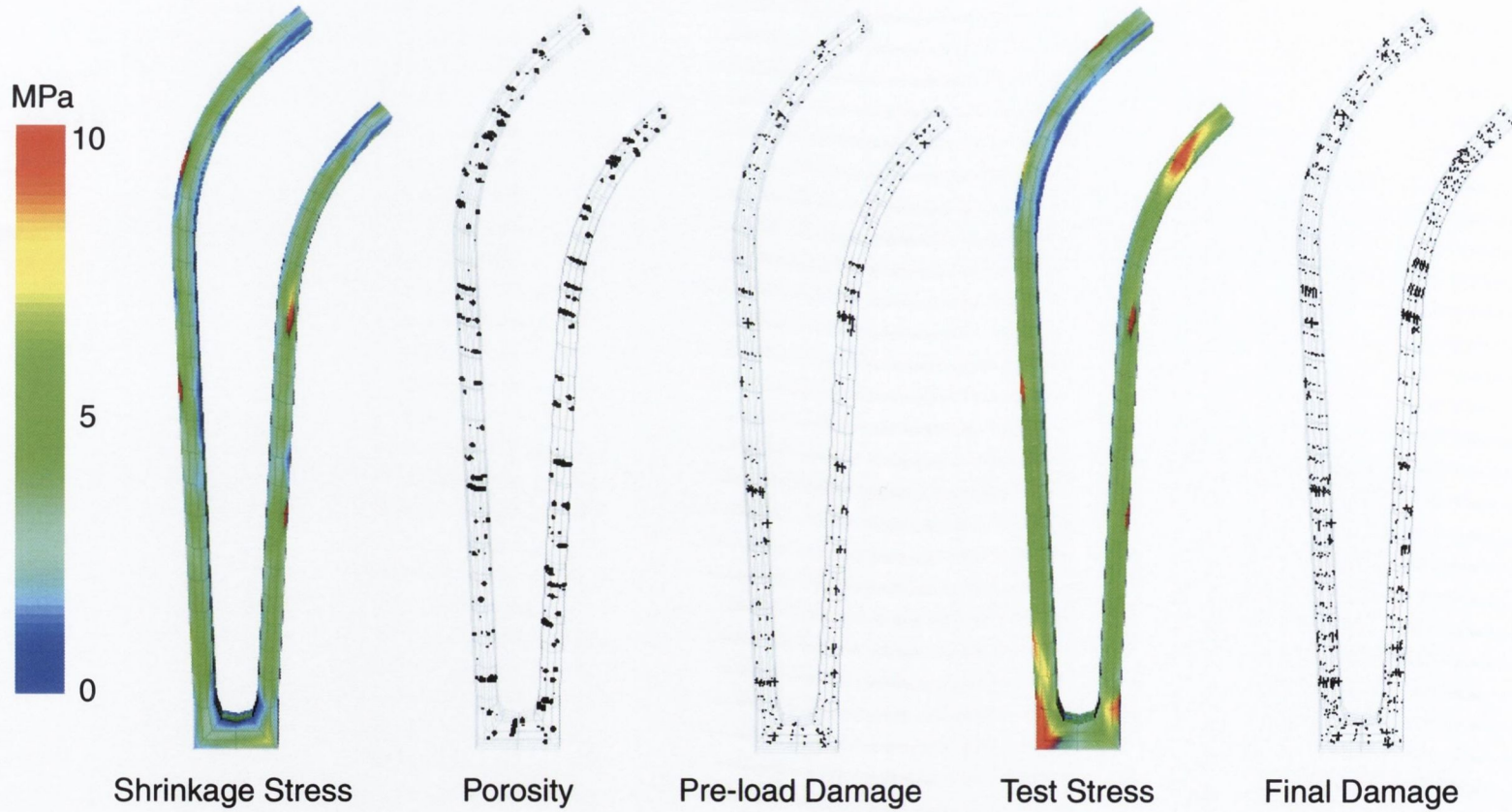


Figure B.3. Stress, porosity, and damage distributions for bonded specimen no. 4. After generation of the pore distribution (pores are plotted as spheres scaled to the volume fraction of each pore), the residual stresses were read in from the original thermoelastic analysis. The pore distribution was then used to calculate local stress concentrations in the stress field. This new shrinkage stress distribution was used to initiate damage around pores (damage data are plotted as principal damage vectors) and the test load was then applied. Damage accumulation was calculated according to the timestep scheme of section 3.2.4 for the superposed stress distributions from loading and shrinkage (the stress distribution shown is for the first cycle of testing).

Bonded (i.e. Matt): Predictions for Specimen #5

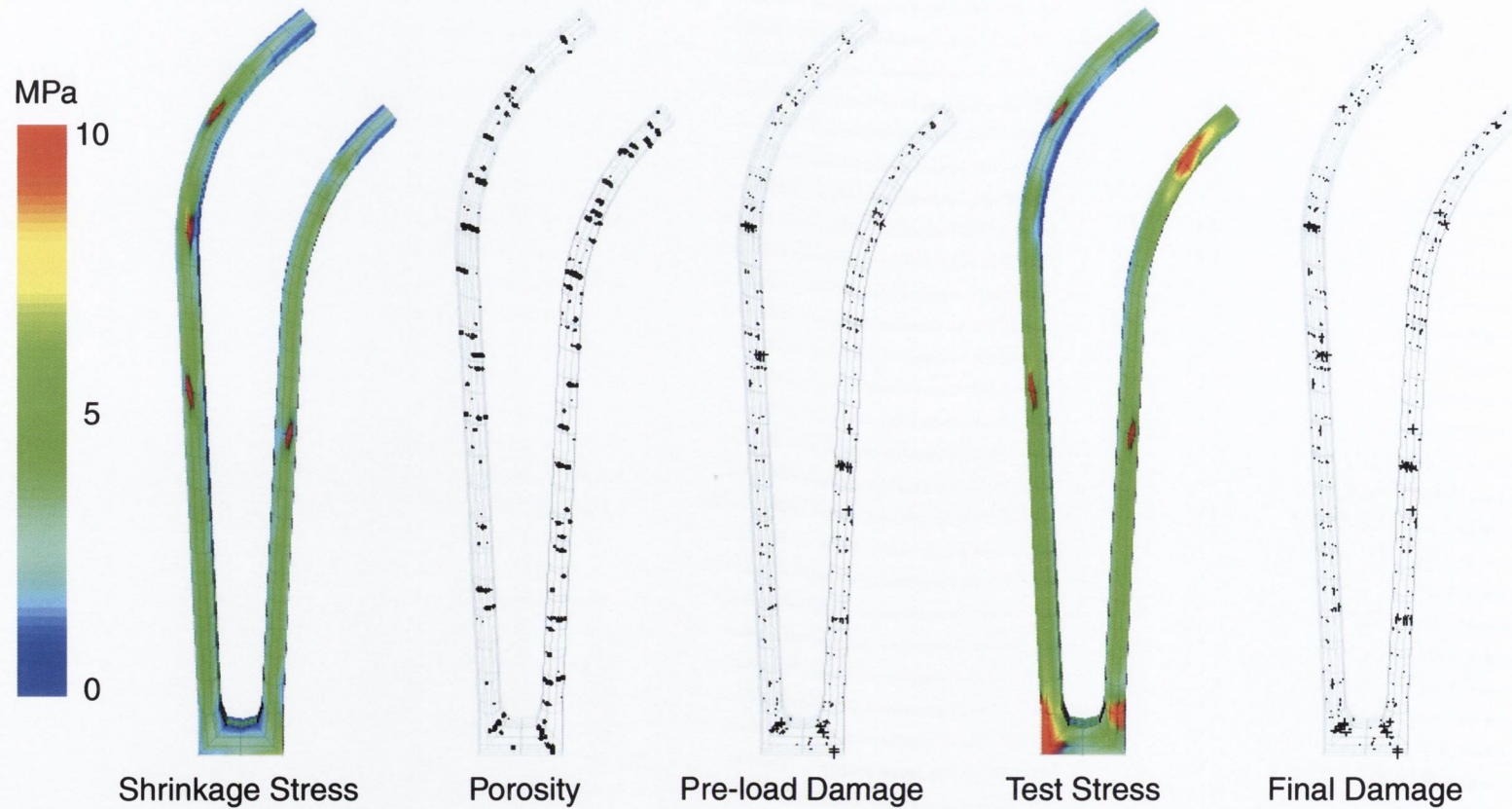


Figure B.4. Stress, porosity, and damage distributions for bonded specimen no. 5. After generation of the pore distribution (pores are plotted as spheres scaled to the volume fraction of each pore), the residual stresses were read in from the original thermoelastic analysis. The pore distribution was then used to calculate local stress concentrations in the stress field. This new shrinkage stress distribution was used to initiate damage around pores (damage data are plotted as principal damage vectors) and the test load was then applied. Damage accumulation was calculated according to the timestep scheme of section 3.2.4 for the superposed stress distributions from loading and shrinkage (the stress distribution shown is for the first cycle of testing).

Debonded (i.e. Polished): Predictions for Specimen #2

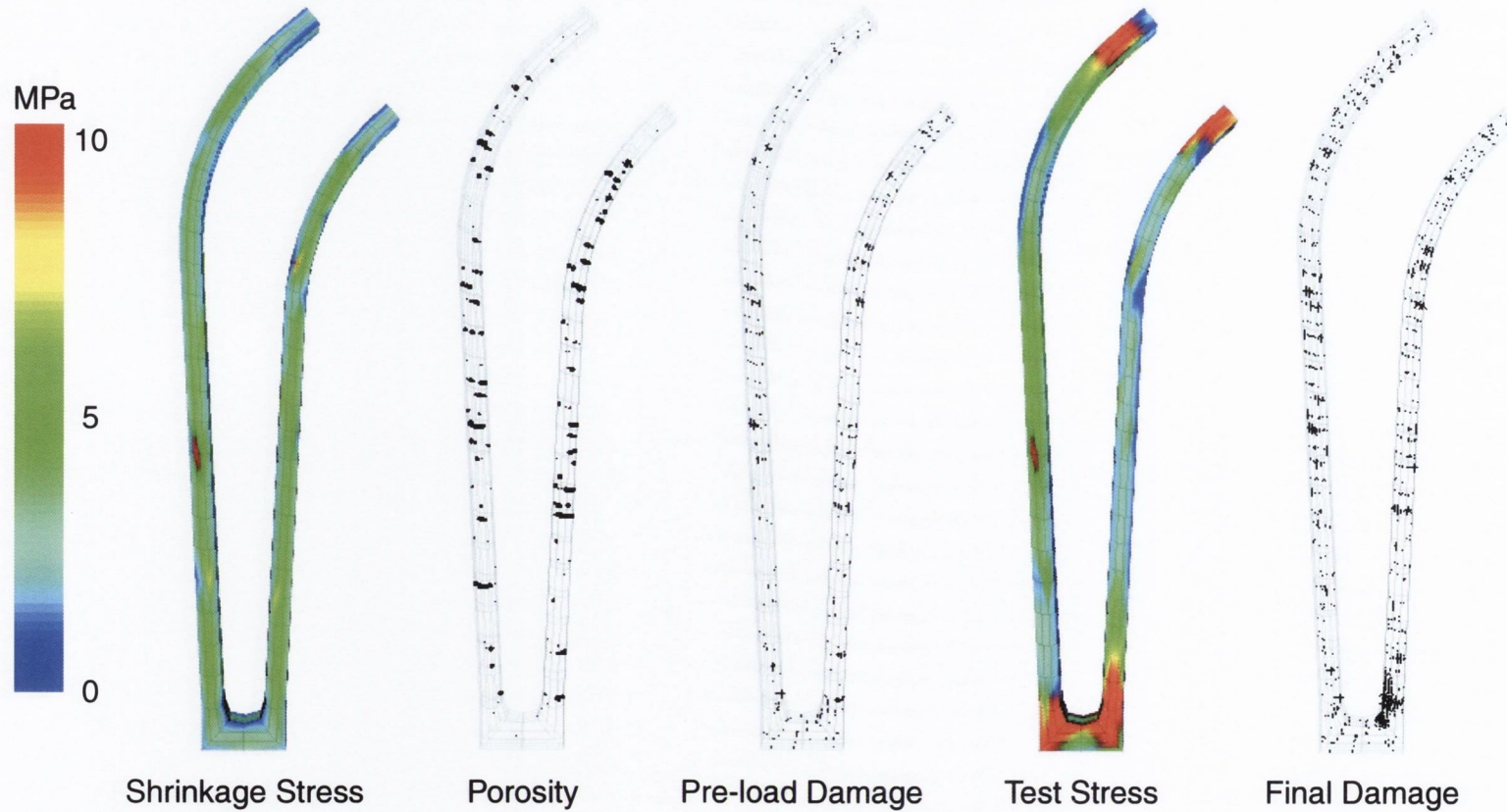


Figure B.5. Stress, porosity, and damage distributions for debonded specimen no. 2. After generation of the pore distribution (pores are plotted as spheres scaled to the volume fraction of each pore), the residual stresses were read in from the original thermoelastic analysis. The pore distribution was then used to calculate local stress concentrations in the stress field. This new shrinkage stress distribution was used to initiate damage around pores (damage data are plotted as principal damage vectors) and the test load was then applied. Damage accumulation was calculated according to the timestep scheme of section 3.2.4 for the superposed stress distributions from loading and shrinkage (the stress distribution shown is for the first cycle of testing).

Debonded (i.e. Polished): Predictions for Specimen #3

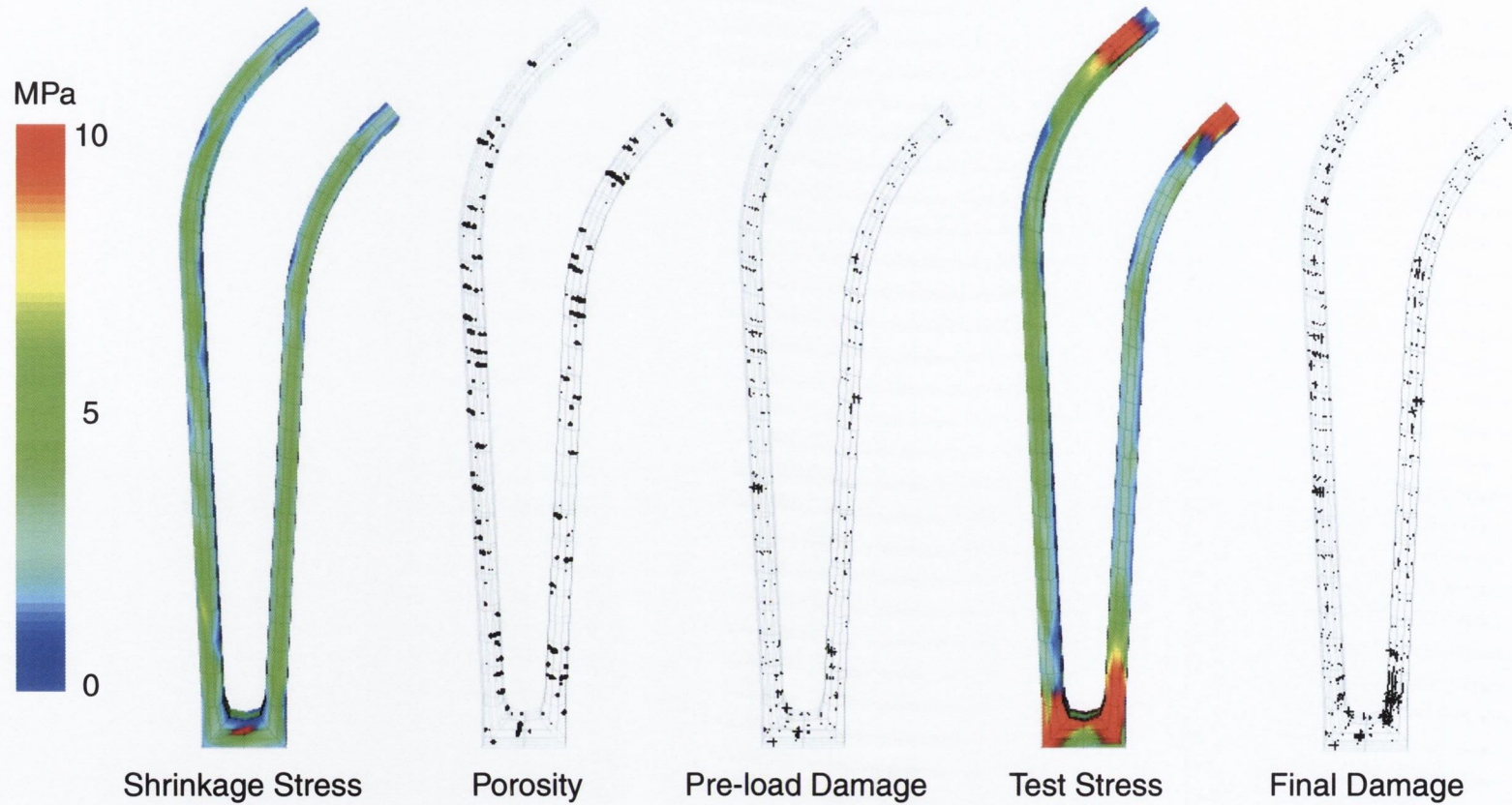


Figure B.6. Stress, porosity, and damage distributions for debonded specimen no. 3. After generation of the pore distribution (pores are plotted as spheres scaled to the volume fraction of each pore), the residual stresses were read in from the original thermoelastic analysis. The pore distribution was then used to calculate local stress concentrations in the stress field. This new shrinkage stress distribution was used to initiate damage around pores (damage data are plotted as principal damage vectors) and the test load was then applied. Damage accumulation was calculated according to the timestep scheme of section 3.2.4 for the superposed stress distributions from loading and shrinkage (the stress distribution shown is for the first cycle of testing).

Debonded (i.e. Polished): Predictions for Specimen #4

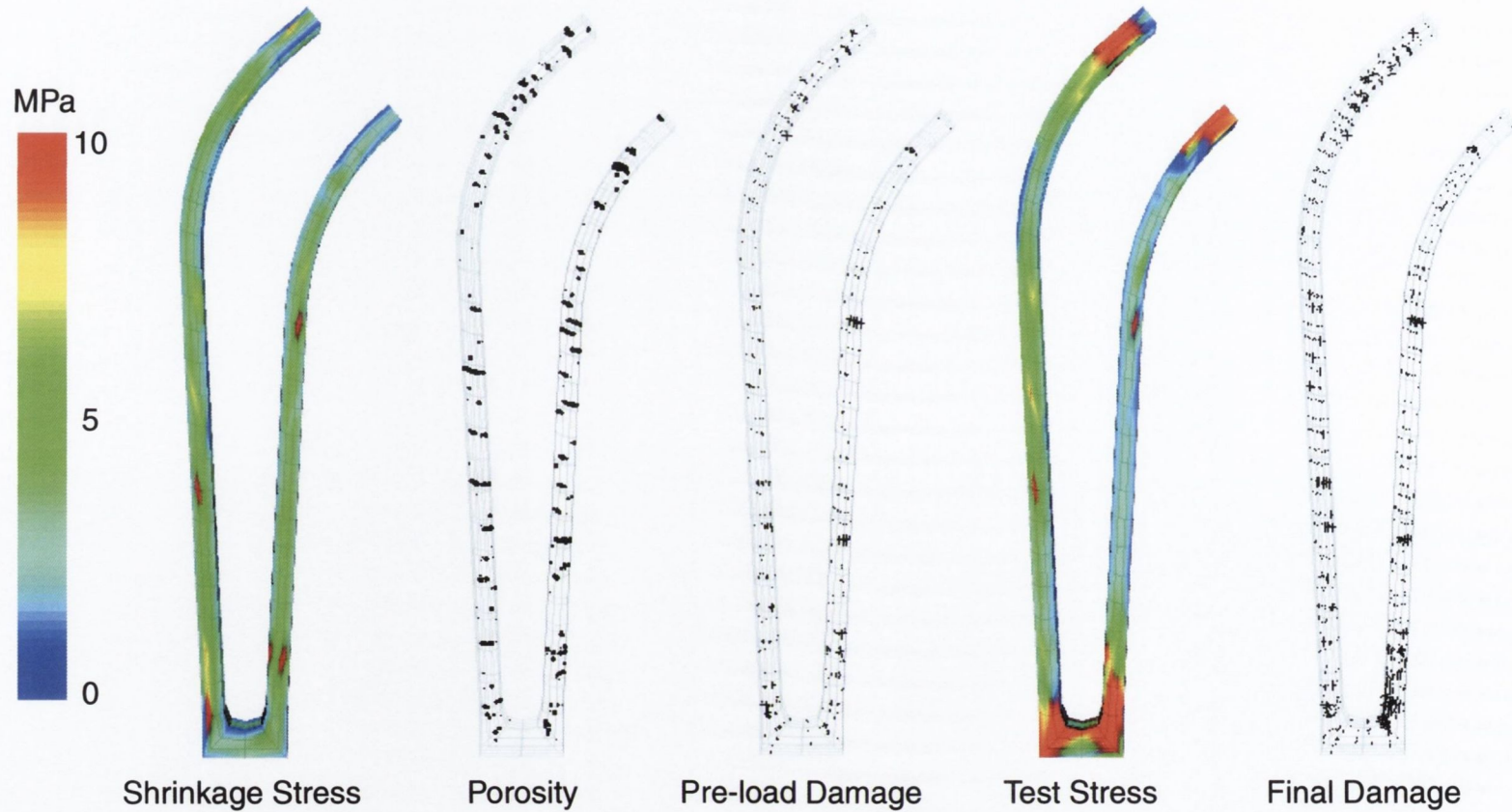


Figure B.7. Stress, porosity, and damage distributions for debonded specimen no. 4. After generation of the pore distribution (pores are plotted as spheres scaled to the volume fraction of each pore), the residual stresses were read in from the original thermoelastic analysis. The pore distribution was then used to calculate local stress concentrations in the stress field. This new shrinkage stress distribution was used to initiate damage around pores (damage data are plotted as principal damage vectors) and the test load was then applied. Damage accumulation was calculated according to the timestep scheme of section 3.2.4 for the superposed stress distributions from loading and shrinkage (the stress distribution shown is for the first cycle of testing).

Debonded (i.e. Polished): Predictions for Specimen #5

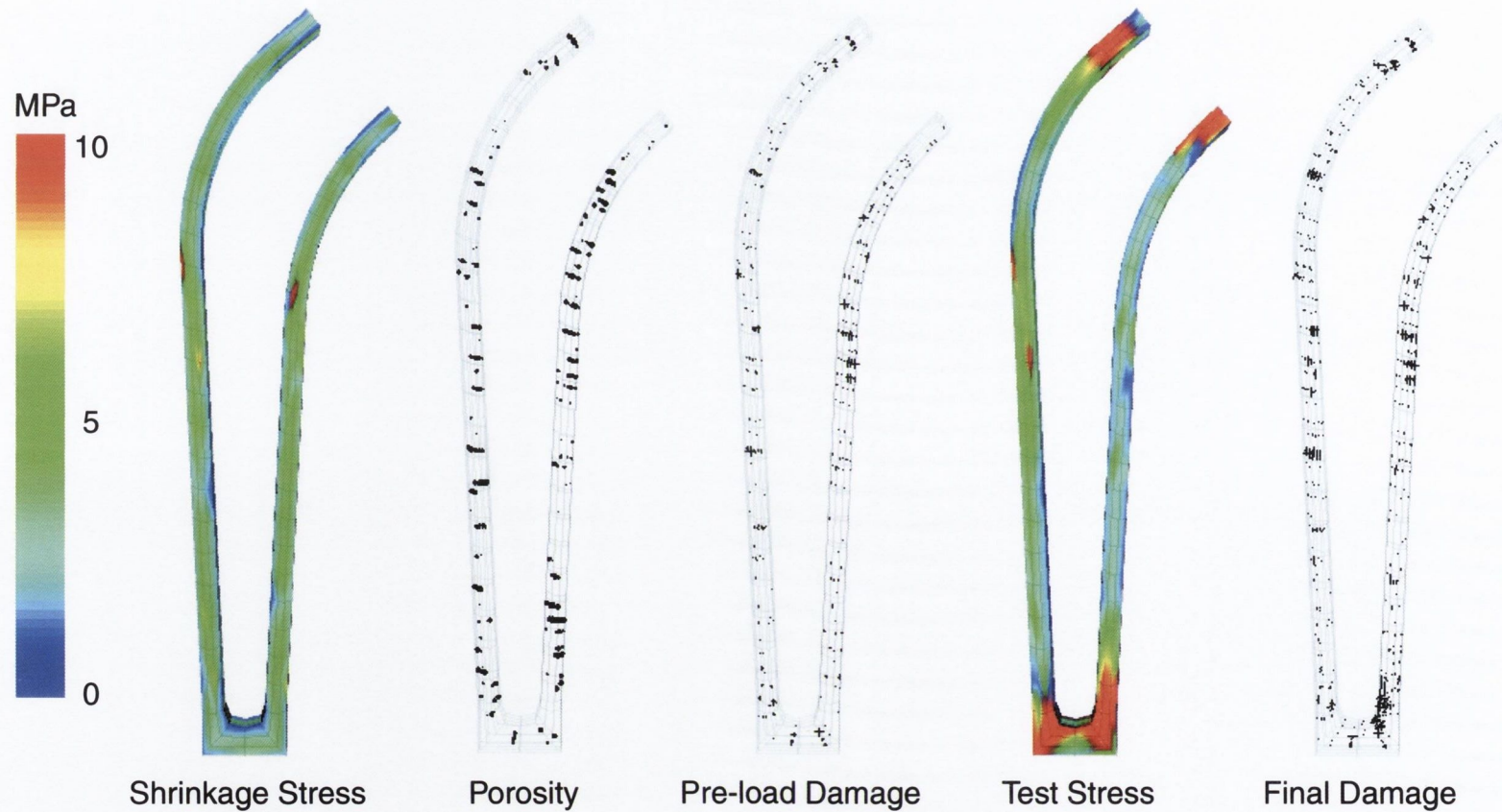


Figure B.8. Stress, porosity, and damage distributions for debonded specimen no. 5. After generation of the pore distribution (pores are plotted as spheres scaled to the volume fraction of each pore), the residual stresses were read in from the original thermoelastic analysis. The pore distribution was then used to calculate local stress concentrations in the stress field. This new shrinkage stress distribution was used to initiate damage around pores (damage data are plotted as principal damage vectors) and the test load was then applied. Damage accumulation was calculated according to the timestep scheme of section 3.2.4 for the superposed stress distributions from loading and shrinkage (the stress distribution shown is for the first cycle of testing).

Digitized by Google

Appendix C

Finite element code

The computational scheme was implemented using the finite element method, coded in Fortran 90. A static library (`Fem_Lib.lib`) was compiled from 12 modules, each of which consist of several subroutines to accomplish associated tasks within a given module, e.g. element shape functions and element strain-displacement matrix formation within a given element module. A main program (`main.f90`) contains all the necessary I/O and the general algorithm of the computational scheme (described briefly in Fig. 3.9). This was then compiled and linked with the library to give a single executable containing any procedures from the library required to implement the algorithm. A brief description of the various modules is given below. All code in `main.f90` and `Fem_Lib.f90` is then listed. Originally, some subroutines from a library accompanying the book "Programming the Finite Element Method" by I.M. Smith and D.V. Griffiths (1998, John Wiley and Sons) were used to develop the algorithm. Some features remain but most of the library has been reimplemented and many new subroutines have been added.

- General FEM modules

`pre-ops` — pre-processing of the mesh, e.g. assigning degree-of-freedom numbers to nodes, convert between node numbering to internal degree-of-freedom numbering, and set up some volumetric search trees

`elements_tot` — subroutines that pass through all elements, e.g. global stiffness matrix assembly, calculation of global internal nodal load vector

`solution` — subroutines related to solution stage, only line-search subroutine at this time

`solvers` — common operations required to interface to a linear system solver, e.g. store stiffness coefficients in skyline matrix, interface to vendor

Math library solver

- Element modules

`hex8_xtr` — subroutines required for 8 node hexahedron, e.g. Gauss pt. sampling, shape functions, etc.

`elContactSurf` — surface-to-surface contact element subroutines

- Material modules

`linelastic` — isotropic linear elasticity

`damage` — damage growth, timestep prediction, and constitutive model

`closedPorous` — elastic medium with non interconnected porosity

`viscoelastic` — only stress relaxation at this time

- Helper modules

`tensor_ops` — useful operations for handling tensors and their matrix representations, e.g. calculate and order principal values/vectors, form rotation and projection matrices

`post_ops` — subroutines used to read/write global arrays, e.g. results or mesh geometry, and filter some of the data, e.g. calculate stressed-volumes

program cdm_fem

Program for reading vtk format file of general unstructured mesh
and subsequent damage analysis for a single pore

variables

mesh stuff

nels = array of no. elements of each type
nn = no. nodes
nlp = array of no. gauss pts per element type
nod = no. nodes per element
nshare = node sharing array (indicates no. elements sharing a node)
nodof = no. dof per node
ndof = array of no. dof per element type
ndim = no. of dimensions
g_coord = global nodal coordinate array
g_ipCoord = global gauss pt coordinate array
g_ipSurf = ditto for contact elements
g_num = global element node no.s array
g_nconn = global array of node-element connectivities
max_elconn = max no. elements connected to another element
g_elconn = global array of element-element connectivities
r_bkt = radius of spatial buckets around gauss pts
max_ipbkt = max no. of gauss pts in a bucket
g_ipbkt1 = global array of level 1 gauss pt buckets within radius, r_bkt
g = element steering vector
g_g = global element steering array
eltypes = no. element types
etype = array of element types id's
mat_id = array of element material id's
g_ipMat = array of gauss pt material id's
vol_el = array of solid element volumes
g_ipvol = array of solid element gauss pt volumes
totvol = array of total volume of each individual material

materials properties

np_types = no. material property types
nprops = max no. of constants req'd from all occurring types
e.g. isotropic elastic only requires two but 3D contact surface
requires 3
e,mu = Young's modulus and Poisson's ratio
ndam = no. of damaging materials
mat_dam = array of damaging material id's
uts = y-intercept of S-N curve
sn_slope = slope of S-N curve
nels_dam = no. elements that can damage
ncpore = no. closed porous materials
mat_cpore = array of closed porous material id's
cpore_m = array of mean of 'cpore' distribution for each cpore material
cpore_std = ditto for standard deviation
prad_m = array of mean pore radii for each 'cpore' material
prad_std = standard deviations of pore radii for each 'cpore' material
nels_cpore = no. elements that can contain closed pores
ks,kn,mu = shear and normal stiffnesses, and friction coefficient for
contact surfaces
solDee = matrix of elastic constants for solid
surfDee = ditto for surface

loads, boundary conditions and global stiffness

nr = no. restrained nodes
nf = nodal freedom matrix
g_k = global stiffness matrix stored as vector
loads = vector of nodal loads (gets overwritten by displacements after
solution in order to save storage space)
elforce = array of gauss pt surface loads on an element face
npres = no. applied element surface pressure load
nodPres = no. nodes per face

! nvec = normal vector of surface
! lcoord = array of nodal coordinates rotated to local surface csys
(with normal, nvec)
! Pval = array of pressures applied to a given node on an element face
! ipPres = pressure at a gauss pt on element face
! force = force vector at a gauss pt

element stuff

element = character variable describing element type (e.g. hexahedron)
solpoints = array of sampling pt coords for a solid element
surfpoints = ditto for surface contact element
weights = vector containing weighting for each gauss pt
km = element stiffness matrix
coord = element nodal coordinates matrix
num = vector of element node no.s
numip = vector of element gauss pt no.s
g_num = global array of all elements' num vectors
g_numip = ditto for numip vectors
g = element steering vector
eld_inc = element displacement vector
solFun = array of shape functions for solid element
surfFun = ditto for surface
ntot = array of shape functions for all dof of element
(used for contact elements)
solDer = array of shape derivatives wrt natural csys for solid
surfDer = not needed at this time
solDeriv = array of shape derivatives wrt global csys for solid
surfDeriv = not needed at this time
solBee = strain displacement matrix for solid
contBee = ditto for surface
gap_tot = total relative displacement vector at interface gauss pt
gap_inc = incremental relative displacement vector at interface gauss pt
gaptol = tolerance above which gap is said to be open
bonding = status of interfacial bonding
1 => fully bonded ; 0 => debonded

solver stuff

strategy = character variable for solver strategy (i.e. mesh-free, 'mfree'
or assembly, 'assem')
solver = character variable describing type of solver:
'ban' (constant bandwidth), 'sky'(skyline), and
'pcg' (preconditioned conjugate gradient)
nband = half bandwidth of global stiffness matrix
neq = no. dof in mesh
g_k = banded or skyline global stiffness matrix stored as vector
kb = banded global stiffness matrix stored as vector
kdiag = vector of diagonal positions for skyline strategy
skyRhs = rhs loads vector for skyline solver
storkm = array used to store mesh-free element stiffness matrices
diag_precon = diagonal preconditioner used for pcg solver

iteration stuff

arrays

! aload = applied forces vector
! lastdisp = displacements from last iteration
! totdisp = accumulated displacements
! ipload = gauss pt internal body load
! eload = accumulated gauss pt internal body load for element
! bdylds = accumulated internal body loads for mesh
! resid = residual load vector
! lastresid = residual for last N-R iteration
! resdisp = residual displacement

scalars

! time = elapsed time (not real time)

```

! maxtime = maximum allowable time
! tstep = time step
! astep = analysis step
! iter = iteration no.
! maxiter = max allowable iterations per step
! totiter = total no. elapsed iterations
! c_val = integer controlling which convergence controls are present
!         0 => none; 1 => displacement; 2 => disp + load
! d_crit = displacement convergence criterion
! l_crit = load convergence criterion
! ltoler = load tolerance
! dtoler = displacement tolerance
! lnorm = euclidean load norm
! idnorm = euclidean displacement increment norm
! idnorm0 = " for initial load step
! tdnorm = euclidean total displacement norm
! rlnorm = euclidean residual load norm
! rdnorm = euclidean residual displacement norm

-----
! logical
! converged = true if rnorm < ltoler
! mlconv = master load convergence on/off
! lconv = analysis step load convergence on/off
! dconv = displacement convergence on/off

-----
! state variables and their associated variables-----
! v_scheme = voigt storage scheme (i.e. 12 or 23)

-----
! g_ipEptot = global array of gauss pt total strains
!            (stored as vectors vectors)
! eptot = individual gauss pt. voigt strain vector
! g_ipSigma = global array of gauss pt stresses for solid elements
! g_ipSigcnv = global stress array for last converged solution
! sigma = individual gauss pt. voigt stresses
! sigma_el = array of all gauss pt stress for an element
! g_stress = global array of nodal stresses
! stress = individual node voigt stresses

-----
! g_ipDam = global array of gauss pt damages
! solDam = individual gauss pt. voigt damage vector
! dam_el = array of all gauss pt damages for an element
! g_dam = global array of nodal damages
! tstep_fail = array indicating failure in less than the min allowable tstep
!             1 => gauss pt is flagged for failure for current tstep
!             0 => " " " " " " " " " " " "
! crackstat = array of gauss pt crack status
!             0 => no cracks present; 1 => 1 crack present
!             2 => 2 " " " " " " " " " " " "
!             3 => all directions ruptured

-----
! cpores = array of pores (1=> present, 0=> not). Length of array matches
!         (no. cpore materials) x (no. elements) x (no. gauss pts/elem)
! poros = porosity values (1st implementation is isotropic, i.e. equal
!         diagonal values only)
! g_ipPoros = global array of gauss pt porosities
! g_poros = global array of nodal porosities
! poros_el = element array of gauss pt porosities
! poros_vol = array of gauss pt volumes which belong to porous materials
!            is overwritten by random pore generator to give array of
!            gauss pt void ratios (i.e. porosities)
! tot_pvol = array of total volume of pores for each porous material
! tot_poros = array of total porosity (i.e. void/vol ratio) for each porous
!            material

-----
! output stuff-----
! ftype = file type for output (ascii or binary as specified in vtk
!         datafile format

```

```

! ipout = character array used to assign whether gauss pt results are
!        extrapolated or copied to nodes
! nout_inc = no. time increments between output of nodal results to
!           datafiles; default is 1
! ipout_inc = no. time increments between output of gauss pt results to
!            datafiles; default is 1
! ostep = next analysis step no. flagged for output of nodal results
! ipstep = next analysis step no. flagged for output of gauss pt
!         results
! sigvol_tab = array of stress-volume data for a given step
! sigMin,sigMax = min and max values defining stress range for output
! sigRange = range of stress intervals for sigvol output
!            i.e. sigMax - sigMin + 1, where extra '1' is for stresses
!            above sigMax
! fs = shear force at a contact gauss pt
!      (used when calculating shear stress at interface)
! iarray = dummy integer array used for output (ensure to deallocate
!         after use so that can be used again for different output
!         quantities)

-----
! df98 provided libraries
! use df1ib
! user supplied libraries for general mesh, tensor, or results manipulation
! use pre_ops; use tensor_ops; use post_ops;
! solver and solution tools
! use solvers; use solution
! general tasks for elements (tangent stiffness and nodal restoring load calc)
! use elements
! user supplied elements
! use hex8_xtr; use elcontactsurf
! user supplied materials
! use linelastic; use damage; use closedPorous

!
! implicit none
integer :: neq,nband,nn,nodof,nr,neltot,max_elconn,max_ipbkt,&
!         iel,inod,ip,ipore,imat,lastmat,g_ip,g_ipSurf,niptot,i,j,k,&
!         astep,iter(-1:0),totiter,maxiter,itime,time_last,&
!         ndim,disp_shapes,loaded_nodes,fixed_nodes,npres,nodPres,&
!         nprops,np_types,ndam,ncpore,nvisc,nels_dam,&
!         v_scheme,nout_inc,ipout_inc,ostep,ipstep,&
!         eltypes, cell_size, cell_pts,ncrack,optype,lastcrack,dummy,width, &
!         file_id, m_1, sigMin,sigMax,sigRange,listnodes,maxshare
integer(2) :: status,c_val,bonding,load_incs,l_inc,loading,bisects,init_stress
real(8) :: det,tstep,tstep_min,tstep_min2,t_solve_in,t_solve_out,t_solve,&
!         & t_iter_in,t_iter_out,t_iter,t_inc_in,t_inc_out,t_inc,t_astep_in,&
!         & t_astep_out,t_astep,time,t_visc,maxtime,max_inc,l_inc_r(-1:0),&
!         & delta_l_inc,l_inc_fact,d_crit,l_crit,ltoler,dtoler,lnorm,lnorm_ap,&
!         & tdnorm,idnorm,idnorm0,rlnorm,rdnorm,last_rlnorm,ls_norm(1:3),ls_slope,&
!         & ls_y0,ls_param(1:3),lsearch,e,mu,notch,maxsig,ks,kn,mu,gaptol,ipPres,&
!         & fs,r_bkt,cnv_ratio,cnv_err,lbound
logical :: mlconv,lconv,dconv,converged,load_loop,initiate
character*3 :: solver; character*4 :: suffix; character*5 :: ftype,strategy
character*6 :: ipout; character*24 :: fname; character*12 :: element,word1,word2,word3

!
! -----
! dynamic arrays-----
real(8), allocatable :: coord(:,,:),g_coord(:,,:),g_ipCoord(:,,:),solpoints(:,,:)&
! & surfpoints(:,,:),solvts(:,,:),surfwts(:,,:),jac(:,,:),jac2D(:,,:),solFun(:,)&
! & solDer(:,,:),solDeriv(:,,:),surfFun(:,,:),ntot(:,,:),solBee(:,,:),&
! & contBee(:,,:),eld_inc(:,,:),eld_tot(:,,:),g_ipEptot(:,,:),g_ipEplnit(:,,:),&
! & g_eptot(:,,:),eptot(:,,:),epinc(:,,:),gap_tot(:,,:),gap_inc(:,,:),gap_elas(:,)&
! & gap_plas(:,,:),solDee(:,,:),incDee(:,,:),surfDee(:,,:),srf_incD(:,,:),km(:,)&
! & & g_k(:,,:),kb(:,,:),prop(:,,:),loads(:,,:),skyRhs(:,,:),apload(:,,:),apl_inc(:,)&
! & & lastdisp(:,,:),cnvdisp(:,,:),totdisp(:,,:),incdisp(:,,:),tot_inc(:,,:),value(:,)&
! & & Pval(:,,:),nvec(:,,:),lcoord(:,,:),ipload(:,,:),eload(:,,:),elforce(:,,:),&
! & & frc_inc(:,,:),bdylids(:,,:),lastresid(:,,:),ls_resid(:,,:),resdisp(:,)&
! & & ls_incdisp(:,,:),sigma(:,,:),sigma_el(:,,:),pr_el(:,,:),pr_mod(:,,:),sig_inc(:,)&
! & & sigvol_tab(:,,:),cnvlds(:,,:),dummysig(:,,:),g_ipSigma(:,,:),g_ipSigcnv(:,,:),&
! & & g_ipPrSig(:,,:),g_PrSig(:,,:),stress(:,,:),g_stress(:,,:), stressTensor(:,)&
! & & solDam(:,,:),dam_el(:,,:),g_ipDam(:,,:),g_ipRefDam(:,,:),ipDam(:,,:),g_dam(:,)&

```

```

&,g_iprDam(:,:),g_PrDam(:,:),uts(:,sn_slope(:),totdam(:),refdam(:,:),&
& newdam(:,:),prindam(:),prdamtens(:,:),prbasis(:,:),prinrot(:,:),&
& actrot(:,:),maxpvec(:),cpore_m(:),cpore_std(:),prad_m(:),prad_std(:),&
& poros(:),g_ipPoros(:,:),g_poros(:,:),poros_el(:,:),poros_vol(:,:),&
& avPoros(:),vol_el(:),g_ipvol(:),totvol(:),diag_precon(:),storkm(:,:),:))
real(4), allocatable :: sgl_vec(:), sgl_tens(:,:),sgl_arr(:,:), nlls_arr(:)
integer, allocatable :: nels(:),nod(:),nst(:),ndof(:),nip(:),nf(:,:),kdiag(:)&
&,nshare(:),g_nconn(:,:),g_elconn(:,:),g_ipbkt1(:,:),g(:),num(:),&
& numip(:),g_num(:,:),g_numip(:,:),g_g( : , :),g_ipMat(:),g_ipType(:),&
& no(:),sense(:),node(:),mat_id(:),etype(:),tstep_fail(:),crackstat(:),&
& mat_dam(:),totcrack(:),mat_cpore(:),mat_visc(:),nels_cpore(:),&
& cpores(:),iarray(:,:),nlls_idarr(:)
-----input and initialisation-----
! set up file-prefix naming variable
call getarg(1,fname,status)
! Display message describing initial phase
print *, "Reading in grid from vtk file and building finite element mesh..."
! open up files required for generating mesh and outputting results and
! runtime messages:
! mesh file in vtk unstructured grid format
fname = fname(1:status)//'.vtk'
open (10,file=fname,status='old',action='read')
! types, loads etc.
fname = fname(1:status)//'.gen'
open (11,file=fname,status='old',action='read')
! nodal results file
fname = fname(1:status)//'.res'
open (20,file=fname,status='replace',action='write')
! integration pt. results
fname = fname(1:status)//'.ipr'
open (22,file=fname,status='replace',action='write')
! runtime messages
fname = fname(1:status)//'.mes'
open (21,file=fname,status='replace',action='write')
! results summary table
fname = fname(1:status)//'.tbl'
open (23,file=fname,status='replace',action='write')
! solution convergence summary table
fname = fname(1:status)//'.cnv'
open (24,file=fname,status='replace',action='write')
! linesearch data for debugging purposes
fname = fname(1:status)//'.lsr'
open (25,file=fname,status='replace',action='write')
! ascii results listing for individual nodes
fname = fname(1:status)//'.nls'
open (26,file=fname,status='replace',recl=1024,action='write')
-----
!--read/write headers for vtk formatted input/results files-----
! note: word1 string is often read in just to advance to the next record
! get output file format from 'gen' file first
read (11,*) word1, word2, word3, word1
read (11,*) ftype, ipout, nout_inc, ipout_inc
!header
read (10,*) word1
do i=20,22,2
  write (i,'(a)') "# vtk DataFile Version 3.2"
enddo
! title
read (10,*) word1
do i=20,22,2
  write (i,'(a)') "3D FE Analysis - Structural"
enddo
! data type
read (10,*) word1
do i=20,22,2
  select case (ftype)

```

```

case ('ascii');
write (i,'(a)') "ASCII" ; write (i,*)
case ('binar')
write (i,'(a)') "BINARY"; write (i,*)
end select
enddo
! mesh geometry/topology. this will generally be of type 'unstructured_grid'
read (10,*) word1
do i=20,22,2
  write (i,'(a)') "DATASET UNSTRUCTURED_GRID"
enddo
-----GEN file initiation-----
! read analysis dimensionality, no. dof/node, no. element types
read (11,*) word1,word2,word3
read (11,*) ndim, nodof, eltypes
allocate (nels(eltypes),nod(eltypes),nip(eltypes),nst(eltypes),ndof(eltypes))
! read element type data
! assigns hex8 as type 1 and contact surf as type 2 (if present)
read (11,*) word1,word2,word3,word1
do i=1,eltypes
  element type name, nodes/elem, no. gauss pts, no. stresses
  read (11,*) element,nod(i),nip(i),nst(i)
! calc no. dofs/element
  ndof(i)=nod(i)*nodof
end do
read (11,*) word1,disp_shapes
! read contact options if contact elements exist
if (eltypes .ne. 1) then
  read (11,*) word1, gaptol
endif
! read stress range for stress volume tabular output data
read (11,*) word1, word2
read (11,*) sigMin, sigMax
sigRange = sigMax - sigMin + 1
! read list of nodes to track for displacement output listings
read (11,*) word1, listnodes
if (listnodes .ne. 0) then
  allocate(nlls_idarr(listnodes),nlls_arr(ndim*listnodes))
  read (11,*) nlls_idarr
endif
! read no. of material types and no. constants for type with highest no.
! of constants
read (11,*) word1 ; read (11,*) word1,word2
read (11,*) np_types, nprops
allocate (prop(nprops,np_types),sigvol_tab(np_types*sigRange,9),&
& totvol(np_types))
! read material property array
do i=1,np_types
  read (11,*) prop(1:nprops,i)
end do
-----Read in mesh, set up array allocations and results file-----
! read/write point data headers for node results files
read (10,*) word1, nn, word2 ! point data header
write (20,'(a,i7,TR1,a)') 'POINTS', nn, 'float'
100 format (f15.6,tr1,f15.6,tr1,f15.6)
! assign voigt storage scheme variable as '12 23 31' form
v_scheme = 12
! allocate and read/write global nodal coordinate matrix
allocate (g_coord(ndim,nn),sgl_vec(ndim),sgl_tens(ndim,ndim))
read (10,*) g_coord
call write_globArray (g_coord,ftype,20,ndim,v_scheme,fname(1:status),'.res')
!
! read connectivity array header (i.e. no. elements and total size of array)
read (10,*) word1, neltot, cell_size

```

```

! allocate arrays
allocate (nf(nodof,nn),nshare(nn),num(nod(1)),numip(nip(1)),&
g_num(nod(1),neltot), coord(nod(1),ndim),&
solpoints(nip(1),ndim),solwts(nip(1)),jac(ndim,ndim),jac2D(2,2),&
vol_el(neltot),&
solDee(nst(1),nst(1)),incDee(nst(1),nst(1)),km(ndof(1),ndof(1)),&
ipload(ndof(1)),eload(ndof(1)),&
eld_inc(ndof(1)),eld_tot(ndof(1)),&
g_eptot(nst(1),nn),epinc(nst(1)),eptot(nst(1)),&
sigma(nst(1)),sigma_el(nst(1),nip(1)),sig_inc(nst(1)),&
pr_el(ndim,nip(1)),pr_nod(ndim),&
g_stress(nst(1),nn),g_PrSig(ndim,nn),&
stress(nst(1)),stressTensor(ndim,ndim),&
g(ndof(1)),g_g(ndof(1),neltot),mat_id(neltot),etype(neltot), &
solDam(nst(1)),dam_el(nst(1),nip(1)),ipDam(nst(1)),&
g_dam(nst(1),nn),g_PrDam(ndim,nn),&
refdam(ndim,ndim),newdam(ndim,ndim),prdamtens(ndim,ndim),&
actrot(ndim,ndim),prbasis(ndim,ndim),prindam(ndim),&
prinrot(ndim,ndim),maxpvec(ndim),&
poros(nst(1)),poros_el(nst(1),nip(1)),g_poros(nst(1),nn))
! allocate element arrays depending on whether extra displacement shapes are
! used: 1 => extra shapes; 0 => standard formulation
if (disp_shapes .eq. 1) then
  allocate (solFun(nod(1)+ndim),solDer(ndim,nod(1)+ndim),&
solDeriv(ndim,nod(1)+ndim),solBee(nst(1),ndof(1)+ndim*nodof))
else
  allocate (solFun(nod(1)),solDer(ndim,nod(1)),&
solDeriv(ndim,nod(1)),solBee(nst(1),ndof(1)))
endif
! allocate arrays for contact elements if present
if (eltypes .eq. 2) then
  allocate (surfpnts(nip(2),ndim),surfwts(nip(2)),surfFun(nod(2)/2),&
ntot(nodof,nod(2)*nodof),gap_tot(nodof),gap_inc(nodof),&
gap_elas(nodof),gap_plas(nodof),contBee(nst(2),ndof(2)),&
surfDee(nst(2),nst(2)),srf_incD(nst(2),nst(2)),&
force(nodof),frc_inc(nodof))
endif
! write connectivity array header to nodal results file
write (20,*) : write (20,'(a,tr1,i7,tr1,i7)') word1, neltot, cell_size
! read/write global element node no. array
call rv_vtkCells (g_num,10,20,ftype,fname(1:status),'.res')
! convert vtk cells numbering to Fem90 global element node no. array
g_num = g_num + 1
! read/write vtk 'cell types' section (i.e. what element type is each cell)
! 12 = 8 node hex solid (ok since contact is just a hex8 with zero volume)
read(10,*) word1, neltot ; write(20,*) ; write(20,*) word1, neltot
read(10,*) etype
call write_globArray(etype,ftype,20,fname(1:status),'.res')
!-----Material input section-----
! read/write element material no.s (vtk scalar cell dataset) from/to 'cell
! data' section
! CELL_DATA ncells (overall header for cell data section--i.e. not repeated
! for types)
read (10,*) word1, dummy ; write (20,*)
write (20,'(a,i5)') "CELL_DATA", neltot
! SCALARS Materials int ncells
read (10,*) word1,word2,word3,dummy
write (20,'(a,TR1,a,TR1,a,i5)') word1,word2,word3,dummy
! LOOKUP_TABLE default
read (10,*) word1, word2 ; write (20,'(a,TR1,a)') word1, word2
! material no.s
read (10,*) mat_id
call write_globArray(mat_id,ftype,20,fname(1:status),'.res')
! set mat id for elements which can damage

```

```

read (11,*) word1
read (11,*) ndam
if (ndam .ne. 0) then
  allocate (mat_dam(ndam),uts(ndam),sn_slope(ndam),totdam(ndam),&
& totcrack(ndam))
  do i=1,ndam
    read(11,*) mat_dam(i), uts(i), sn_slope(i)
  end do
endif
! set mat id for elements which can contain closed pores
read (11,*) word1
read (11,*) ncpore
if (ncpore .ne. 0) then
  allocate (mat_cpore(ncpore),nels_cpore(ncpore),&
cpore_m(ncpore),cpore_std(ncpore),&
prad_m(ncpore),prad_std(ncpore),&
avPoros(ncpore))
  do i=1,ncpore
    read(11,*) mat_cpore(i),cpore_m(i),cpore_std(i),prad_m(i),prad_std(i)
  end do
endif
! set mat_id for viscoelastic materials (stress relaxing for the moment)
read (11,*) word1
read (11,*) nvisc
if (nvisc .ne. 0) then
  allocate (mat_visc(nvisc))
  do i=1,nvisc
    read(11,*) mat_visc(i)
  enddo
endif
! check if initial stress results are to be used
! 1=> use initial stress (to be read in); 0=> no initial stress
read (11,*) word1, init_stress
if (init_stress .eq. 1) read (11,*) notch
!-----Element types section and Gauss pt sampling-----
! read/write element type id's (vtk scalar cell dataset) from/to 'cell data'
! section
! 1 = 8 node hex solid
! 2 = 4-node-surface to 4-node-surface contact element
! (note: still an 8 node element)
! SCALARS Materials int ncells
read (10,*) word1,word2,word3,dummy
write (20,'(a,TR1,a,TR1,a,TR1,i5)') word1,word2,word3,dummy
! LOOKUP_TABLE default
read (10,*) word1, word2 ; write (20,'(a,TR1,a)') word1, word2
! element type no.s
nels = 0
do iel=1,neltot
  read (10,*) etype(iel)
  nels(etype(iel)) = nels(etype(iel)) + 1
end do
call write_globArray(etype,ftype,20,fname(1:status),'.res')
! calc total no. of gauss pts
if (eltypes .eq. 2) then
  niptot = nels(1)*nip(1) + nels(2)*nip(2)
else
  niptot = nels(1)*nip(1)
endif
! allocate global 'ip' arrays
allocate(g_ipcoord(ndim,niptot),g_numip(maxval(nip),neltot),&
g_ipType(niptot),g_ipMat(niptot),g_ipvol(niptot),&
g_ipEptot(nst(1),niptot),g_ipEplnit(nst(1),niptot),&
g_ipSigma(nst(1),niptot),g_ipSigcgv(nst(1),niptot),&
g_ipPrSig(ndim,niptot),g_ipPrDam(ndim,niptot),&
g_ipDam(nst(1),niptot),g_ipRefDam(nst(1),niptot),&
crackstat(niptot),g_ipPoros(nst(1),niptot),&
tstep_fail(niptot))

```

```

! calculate all gauss pt coordinates for mesh and set up gauss pt number,
! property, type, and volume arrays as well as no. porosity and damage elems
g_ip = 1
nshare = 0 ; nels_dam = 0 ; nels_cpore = 0
g_ipvol = .0 ; vol_el = .0 ; g_numip = 0
call hex8GaussSample(nip(1),solpoints,solwts)
if (eltypes .eq. 2) then
  call surfSampleGaussPtsLocal (nip(2),surfpoints,surfwts)
endif
!loop elements
do iel=1,neltot
  num = g_num(:,iel); coord = transpose(g_coord(: , num))
  ! fill node-share array
  nshare(num) = nshare(num) + 1
  ! Hex8
  if (etype(iel) .eq. 1) then
    ! loop gauss pts
    do ip=1,nip(1)
      call hex8ShapeFun (solpoints(ip,:),solFun)
      g_ipCoord(:,g_ip) = matmul(solFun(1:nod(1)),coord)
      ! fill global element ip-number vectors
      g_numip(ip,iel) = g_ip
      ! get shape derivatives wrt local elem csys
      call hex8ShapeDer(solpoints(ip,:),solDer)
      ! calculate jacobian and determinant
      jac=matmul(solDer(:,1:nod(1)),coord)
      call gen_det(jac,det)
      ! fill in volume arrays
      g_ipvol(g_ip) = det
      vol_el(iel) = vol_el(iel) + det
      totvol(mat_id(iel)) = totvol(mat_id(iel)) + g_ipvol(g_ip)
      ! fill global ip material id vector
      g_ipMat(g_ip) = mat_id(iel)
      g_ipType(g_ip) = etype(iel)
      g_ip = g_ip + 1
    end do
    ! check if dam material and update relevant variables
    if (ndam .ne. 0) then
      do i=1,ndam
        if (mat_dam(i) .eq. mat_id(iel)) then
          nels_dam = nels_dam + 1
        endif
      end do
    endif
    ! check if cpore material and update relevant variables
    if (ncpore .ne. 0) then
      do i=1,ncpore
        if (mat_cpore(i) .eq. mat_id(iel)) then
          nels_cpore(i) = nels_cpore(i) + 1
        endif
      end do
    endif
    ! contact
    elseif (etype(iel) .eq. 2) then
  !loop gauss pts
  do ip=1,nip(2)
    call surfShapeFun(surfpoints(ip,:),surfFun)
    g_ipCoord(:,g_ip) = matmul(surfFun,coord)
    g_numip(ip,iel) = g_ip
    g_ipMat(g_ip) = mat_id(iel)
    g_ipType(g_ip) = etype(iel)
    g_ip = g_ip + 1
  end do
endif
end do
! build node-element connectivity array
maxshare = maxval(nshare) + 1
allocate (g_nconn(maxshare,nn))
call node_connect (g_num,g_nconn)
! build elem-elem connectivity array

```

```

! get max no. elements connected to another element
call parse_elconnect (g_num,g_nconn,max_elconn)
! allocate and fill
allocate (g_elconn(max_elconn,neltot))
call el_connect(g_num,g_nconn,g_elconn)
! build gauss pt spatial bucket array
if (ncpore .ne. 0) then
  r_bkt = maxval(prad_m) + 3.*maxval(prad_std)
else
  r_bkt = 1.
endif
call parse_ipbucket1(g_elconn,g_numip,g_ipcoord,r_bkt,max_ipbkt)
allocate (g_ipbkt1(max_ipbkt,niptot))
call ipbucket1(g_elconn,g_numip,g_ipcoord,r_bkt,g_ipbkt1)
!-----gauss pt output file set-up-----
! write point data headers for gauss pt results files
write (22, '(a,tr1,i7,tr1,a)') 'POINTS', niptot, 'float'
! write gauss pt coordinates to '.ipr' file
call write_globArray(g_ipCoord,ftype,22,ndim,v_scheme,fname(1:status),'.ipr')
! write cell 'connectivity' and type data for vtk unstructured grid format
allocate (iarray(2,niptot))
iarray(1,:) = 1
do ip=1,niptot
  iarray(2,ip) = ip - 1
enddo
write (22, '(/,a,tr1,i,tr1,i)') "CELLS", niptot, 2*niptot
call write_globArray(iarray,ftype,22,fname(1:status),'.ipr')
write (22, '(/,a,tr1,i)') "CELL_TYPES", niptot
call write_globArray(iarray(1,:),ftype,22,fname(1:status),'.ipr')
deallocate (iarray)
! write material id's for gauss pts
write (22, '(/,a,tr1,i)') "CELL_DATA", niptot
write (22, '(a)') "SCALARS Materials int 1"
write (22, '(a)') "LOOKUP_TABLE default "
call write_globArray(g_ipMat,ftype,22,fname(1:status),'.ipr')
! headers for results data sections of nodal and gauss pt results files
! write header for 'nodal results data' section of '.res' file
write (20, '(/,a,tr1,i7)') "POINT_DATA", nn
! write header for 'data' section of '.ipr' file
write (22, '(/,a,tr1,i7)') "POINT_DATA", niptot
! Output gauss pt volumes as first dataset
write (22, '(a)') "SCALARS Volumes float 1"
write (22, '(a)') "LOOKUP_TABLE default "
call write_globArray(g_ipvol,ftype,22,fname(1:status),'.ipr')
!-----Restrains (nodal freedom array)-----
! read in nodal freedom array (i.e. boundary constraints) and generate
! nodal freedom no.s
! array, nf; note: 1 => free and 0 => restrained
nf=1
read(11,*) word1
read(11,*) nr ; if(nr>0) read(11,*) (k,nf(:,k),i=1,nr)
! 'formnf' increments and substitutes the value of a counter every time it
! finds a 'free' nodal dof
call formnf(nf)
! no. eqts = maximum nodal freedom value
neq=maxval(nf)
!
! allocate load, and displacement arrays
! note: 0:neq allocation used to prevent 0 values from nf causing out-of-bounds
! ref; i.e. only 1:neq used for calculations
allocate (loads(0:neq),apload(0:neq),apl_inc(0:neq),lastdisp(0:neq), &
  cnvdisp(0:neq),totdisp(0:neq),incdisp(0:neq),tot_inc(0:neq), &
  bdylds(0:neq),cnvlds(0:neq),resld(0:neq),lastresid(0:neq), &
  ls_resid(0:neq),resdisp(0:neq),ls_incdisp(0:neq))

```

```

loads = .0; apload = .0; apl_inc = .0; bdylds = .0; cnvlds = .0
cnvdisp = .0; lastdisp = .0; incdisp = .0; totdisp = .0; lastresid = .0
! solver strategy set-up
read (11,*) word1, strategy
read (11,*) word1, solver
select case (solver)
  case ('sky')
    allocate (kdiag(neq),skyRhs(neq,1))
    kdiag = 0
! case ('itr')
! allocate (diag_precon(0:neq))
! diag_precon = .0
end select
!-----
!-----Assembly strategy set-up requirements-----
!-----
! loop elements to find nband,kdiag and store steering vectors
nband=0
do iel=1,neltot
  num=g_num(:,iel)
  call num_to_g(num,nf,g);
  g_g(:,iel) = g
  ! check bandwidth and reset if necessary
  if(nband < bandwidth(g)) nband = bandwidth(g)
  ! calculate max bandwidth for skyline storage for this element
  ! and store in kdiag
  if (solver .eq. 'sky') call fkdiag(kdiag,g)
end do
! allocate stiffness arrays based on storage scheme
select case (solver)
  case ('ban')
    ! band storage
    allocate (g_k(neq*(nband+1)))
  case ('sky')
    ! skyline storage
    ! use bandwidths in kdiag to calculate positions of diagonal stiffness matrix
    ! entries and update kdiag to store these positions
    kdiag(1) = 1
    do i=2,neq
      kdiag(i) = kdiag(i) + kdiag(i-1)
    end do
    allocate (g_k(kdiag(neq)))
! case ('itr')
! iterative solver using general non zero storage by rows
! find no. non-zero entries
! call fknz(g_g,nz)
! allocate (g_k(nz),g_i(nz+1),g_j(nz))
end select
! initialise global stiffness to zero
g_k= .0
! display problem size
print *, "There are",neq," equations and the half-bandwidth is",nband
write (21,'(a,i7)') "There are",neq," equations and the half-bandwidth is",&
nband
!-----
!-----Read in residual stress -----
!-----generate porosity -----
!-----initiate damage -----
!-----
! read/generate damage and/or pores and write to '.ipr' file
g_ipDam = .0 ; g_dam = .0 ; g_ipPrDam = .0 ; crackstat = 0 ; totcrack = 0
totdam = .0 ; g_ipPoros = .0 ; g_poros = .0 ; poros = .0
g_ipSigma = .0 ; g_ipPrSig = .0 ; g_ipEpInit = .0
!check if any porous materials are present
if (ncpore .ne. 0) then
! generate random pore distribution to return array of porosities
call random_pores (cpore_m,cpore_std,prad_m,prad_std,g_ipvol,g_ipMat,&

```

```

g_ipbkt1,g_ipcoord,mat_cpore,g_ipPoros,avPoros)
! read initial stresses to be used in assigning initial damages
if (init_stress .eq. 1) then
! open file and read nodal stresses
open (30,file='init_sig.txt',status='old',action='read')
read (30,*) k
do i=1,k
  read (30,*) j, g_stress(:,j)
enddo
! loop elements to read initial stress and calculate initial damage
do iel=1,neltot
  do imat=1,ncpore
    if (mat_id(iel) .eq. mat_cpore(imat)) then
      numip = g_numip(:,iel)
      num = g_num(:,iel)
      ! store element gauss pt porosites (and nodal initial stresses
      ! (if present))
      do ip=1,nip(1)
        poros_el(:,ip) = g_ipPoros(:,numip(ip))
        if (init_stress .eq. 1) then
          sigma_el(:,ip) = g_stress(:,num(ip))
        endif
      enddo
      ! interpolate to give gauss pt stresses (if present)
      if (init_stress .eq. 1) then
        e = prop(1,mat_id(iel)) ; nu = prop(2,mat_id(iel))
        do ip=1,nip(1)
          call hex8ShapeFun (solpoints(ip,:),solFun)
          do j=1,nst(1)
            sigma(j) = dot_product(solFun(1:nod(1)),sigma_el(j,:))
          enddo
          poros = g_ipPoros(:,numip(ip))
          do j=1,ndam
            if (mat_id(iel) .eq. mat_dam(j)) maxsig = uts(j)
          enddo
          call poreStress(maxsig,poros(1),v_scheme,sigma)
          g_ipSigma(:,numip(ip)) = sigma
          sigma_el(:,ip) = sigma
          ! default to isotropic 'dee' for equivalent strain calc
          call isoDee(solDee,e,nu)
          ! invert to give compliance
          call gen_invert(solDee,solDee)
          ! calculate shrinkage damage if a pore is present and material
          ! can damage
          if (ndam .gt. 0) then
            do i=1,ndam
              if (poros(1) .gt. .0 .and. mat_id(iel) .eq. mat_dam(i)) then
                call shrinkage_dam (sigma,poros,uts(i),notch,v_scheme,&
& ncrack,ipDam)
                g_ipDam(:,numip(ip)) = ipDam
                totdam(i) = totdam(i) + ipdam(1) + ipdam(2) + ipdam(3)
                totcrack(i) = totcrack(i) + ncrack
                crackstat(numip(ip)) = ncrack
                lbound = 0.9999 ; optype = 1 ; eptot = .0
                if (poros(1) .eq. 1.) then
                  ! don't calculate equivalent strain if pore occupies
                  ! entire vol
                  solDee = .0
                else
                  ! treat compliance with same projection operator as
                  ! stiffness
                  call damdee(solDee,e,nu,ipDam,eptot,ncrack,lbound,&
& optype)
                endif
              endif
            enddo
          endif
          g_ipEpInit(:,numip(ip)) = matmul(solDee,sigma)

```

```

! calculate and store principal values
  call vprival (sigma,v_scheme,g_ipPrSig(:,numip(ip)))
  call vprival (ipDam,v_scheme,g_ipPrDam(:,numip(ip)))
enddo
endif
enddo
endif
! copy and average values at nodes
g_poros = .0 ; g_stress = .0 ; g_prSig = .0 ; g_prDam = .0
do iel=1,neltot
  do imat=1,ncpore
    if (mat_id(iel) .eq. mat_cpore(imat)) then
      numip = g_numip(:,iel)
      num = g_num(:,iel)
      do inod=1,nod(1)
        g_poros(:,num(inod)) = g_poros(:,num(inod)) &
          + (1./nshare(num(inod)))*g_ipPoros(:,numip(inod))
        if (init_stress .eq. 1) then
          g_dam(:,num(inod)) = g_dam(:,num(inod)) &
            + (1./nshare(num(inod)))*g_ipDam(:,numip(inod))
          g_stress(:,num(inod)) = g_stress(:,num(inod)) &
            + (1./nshare(num(inod)))*g_ipSigma(:,numip(inod))
          g_prSig(:,num(inod)) = g_PrSig(:,num(inod)) &
            + (1./nshare(num(inod)))*g_ipPrSig(:,numip(inod))
          g_prDam(:,num(inod)) = g_prDam(:,num(inod)) &
            + (1./nshare(num(inod)))*g_ipPrDam(:,numip(inod))
        endif
      end do
    endif
  enddo
enddo
! write total volume and porosity data to '.tbl' file
13 format (a,tr4,a,tr5,a,tr5,a,tr5,a)
14 format (f3.0,tr6,f8.1,tr4,f7.2,tr4,f8.1,tr4,f8.1)
write (23,13) " mat_id","volume","%_pores",' totcracks',' totdam'
do imat = 1,np_types
  re-zero table before filling
  sigvol_tab = .0
  sigvol_tab(1,1) = imat
  sigvol_tab(1,2) = totvol(imat)
  do j=1,ncpore
    if (imat .eq. mat_cpore(j)) then
      sigvol_tab(1,3) = avPoros(j)*100
    endif
  enddo
  do j=1,ndam
    if (imat .eq. mat_dam(j)) then
      sigvol_tab(1,4) = totcrack(j)
      sigvol_tab(1,5) = totdam(j)
    endif
  enddo
  write (23,14) sigvol_tab(1,1:5)
enddo
! write porosities to '.res' and '.ipr' files
write (20,'(a,TR1,a,TR1,a)') "TENSORS","Porosity","float"
write (22,'(a,TR1,a,TR1,a)') "TENSORS","Porosity","float"
call write_globArray(g_Poros,ftype,20,ndim,v_scheme,fname(1:status),' .res')
call write_globArray(g_ipPoros,ftype,22,ndim,v_scheme,fname(1:status),&
  & '.ipr')
if (init_stress .eq. 1) then
! write initial stresses and damages to '.res' and '.ipr' files
write (20,'(a,TR1,a,TR1,a)') "TENSORS","init_sig","float"
write (22,'(a,TR1,a,TR1,a)') "TENSORS","init_sig","float"
call write_globArray(g_stress,ftype,20,ndim,v_scheme,fname(1:status),&
  & '.res')
call write_globArray(g_ipSigma,ftype,22,ndim,v_scheme,fname(1:status),&
  & '.ipr')

```

```

& '.ipr')
write (20,'(a,TR1,a,TR1,a)') "TENSORS","init_dam","float"
write (22,'(a,TR1,a,TR1,a)') "TENSORS","init_dam","float"
call write_globArray(g_dam,ftype,20,ndim,v_scheme,fname(1:status),' .res')
call write_globArray(g_ipDam,ftype,22,ndim,v_scheme,fname(1:status),&
  & '.ipr')
write (20,'(a,TR1,a,TR1,a)') "VECTORS","init_prsig","float"
write (22,'(a,TR1,a,TR1,a)') "VECTORS","init_prsig","float"
call write_globArray (g_PrSig,ftype,20,ndim,v_scheme,fname(1:status),&
  & '.res')
call write_globArray (g_ipPrSig,ftype,22,ndim,v_scheme,fname(1:status),&
  & '.ipr')
write (20,'(a,TR1,a,TR1,a)') "VECTORS","init_prdam","float"
write (22,'(a,TR1,a,TR1,a)') "VECTORS","init_prdam","float"
call write_globArray (g_PrDam,ftype,20,ndim,v_scheme,fname(1:status),&
  & '.res')
call write_globArray (g_ipPrDam,ftype,22,ndim,v_scheme,fname(1:status),&
  & '.ipr')
endif
endif
!-----
! initialise time, loadstep counters and convergence controls
time = .0
astep = 1 ; iter(0) = 1 ;
ostep = 1 ; ipstep = 1
! check 'gen' file for tolerance controls, load increments and max iterations
read (11,*) word1, c_val
select case (c_val)
case (0)
  dconv = .false.
  mlconv = .false.
case (1)
  dconv = .true.
  mlconv = .false.
  read(11,*) word1,dtoler
case (2)
  dconv = .false.
  mlconv = .true.
  read(11,*) word1,ltoler
case (3)
  dconv = .true.
  mlconv = .true.
  read(11,*) word1,dtoler
  read(11,*) word1,ltoler
end select
read (11,*) word1, load_incs
read (11,*) word1, maxiter
read (11,*) word1, maxtime
!-----
!-----apply loads-----
! message
write (*,'(/,tr1,a,/)' ) "Applying loads and boundary conditions..."
write(21,'(/,tr1,a,/)' ) "Applying loads and boundary conditions..."
! read loading type
! 1 => cyclic ramped loading/unloading
! 2 => ramp loading for first step and hold constant for subsequent steps
read(11,*) word1, loading
! applied forces
read(11,*) word1
read(11,*) loaded_nodes
if(loaded_nodes .ne. 0) then
  read (11,*)(k,apload(nf(:,k)),i=1,loaded_nodes)
endif
! applied displacements
read(11,*) word1
read(11,*) fixed_nodes
if(fixed_nodes/=0)then

```

```

allocate(node(fixed_nodes),sense(fixed_nodes),value(fixed_nodes),&
         no(fixed_nodes))
read(11,*)(node(i),sense(i),value(i),i=1,fixed_nodes)
do i=1,fixed_nodes
  no(i)=nf(sense(i),node(i))
end do
end if
! applied surface pressures
! read no. faces and no. nodes per face
read(11,*) word1
read(11,*) npres, nodPres
if (npres .ne. 0) then
  allocate (Pval(nodPres),nvec(ndim),lcoord(nodPres,ndim),&
           & elforce(nodof,nodPres))
  if (elatypes .ne. 2) allocate (force(nodof))
  ! if no contact elements present then must explicitly allocate
  ! shape function and sampling points, etc. arrays for a surface
  ! note: assume same no. gauss pts. as nodes making face
  if (elatypes .eq. 1 .and. etype(1) .ne. 2) then
    allocate (surfpnts(nodPres,ndim),surfFun(nodPres),&
             surfwts(nodPres),ntot(nodof,nodPres*nodof))
  endif
  elforce = .0
  do i=1,npres
! loop nodes to generate num and coord arrays
    do j=1,nodPres
      read(11,*) num(j), Pval(j)
      coord(j,:) = g_coord(: , num(j))
    end do
! loop gauss pts to calc load contribution
    do ip=1,nodPres
      ! interpolate gauss pt pressure from nodal values
      call surfSampleGaussPtsLocal (nodPres,surfpnts,surfwts)
      call surfShapeFun(surfpnts(ip,:),surfFun)
      ipPres = dot_product(surfFun,Pval)
      ! calc jacobian, rotation matrix, normal vector, and area
      call surfJac(surfpnts(ip,:),coord(1:nodPres,:),actrot,jac2D)
      nvec = actrot(:,3)
      call gen_det'(jac2D,det)
      ! force vector = (pressure*area)*(normal vector)
      ! '-1' premultiplier because ansys uses negative value for tensile
      ! pressure
      force = (-1.*ipPres*det)*nvec
      elforce(:,ip) = force
    end do
! extrapolate to nodes
    call surfExtrapSample (nodPres,surfpnts,surfwts)
    do inod=1,nodPres
      call surfShapeFun (surfpnts(inod,:),surfFun)
      do j=1,nodof
        force(j) = dot_product(surfFun,elforce(j,:))
      end do
      ! add to global loads vector
      apload(nf(:,num(inod))) = apload(nf(:,num(inod))) + force
    end do
  end do
endif
!-----start analysis-----
! write headers for solution convergene summary file
write (24,'(a)')&
  & " astep l_inc iter(0) tdnorm idnorm lnorm rlnorm lcrit"
! start time step loop
!-----
converged = .true.
load_loop = .true.
timesteps: do while (time .le. maxtime)
! initialise iteration counter and load increment size

```

```

iter(0) = 1 ; iter(-1) = 5 ; totiter = 1 ; l_inc_r(-1) = .0 ; bisects = 0
! load increment loop
l_inc = 0
load_increments:do
! set total displacement to last converged displacement
totdisp = cnvdisp
reset total displacement increment
tot_inc = .0
! set break condition for load increment loop
if (load_loop .eq. .false.) exit
! set load increment counter and reset iteration counter if starting from
! a converged solution
if (converged) then
  l_inc = l_inc + 1
  iter(0) = 1
endif
message
print *, "Beginning load increment", l_inc
write (21,'(a,i3)') "Beginning load increment", l_inc
! set load scaling factor and scale applied forces for current increment
if (l_inc .eq. 1 .and. iter(0) .eq. 1 .and. converged) then
  set load increment position based on direction of loading
  l_inc_r(0) = l_inc
! bisect load increment if previous increment size failed to converge
elseif (iter(0) .eq. 1 .and. converged .eq. .false.) then
  assume linear analysis if maxiter is 1 and exit
  if (maxiter .eq. 1) then
    cnvdisp = cnvdisp + tot_inc
    exit
  else stop if bisection limit exceeded
  elseif (bisects .gt. 3) then
    print *, "Maximum no. bisections exceeded---analysis stopped"
    write (21,'(a)') "Maximum no. bisections exceeded---analysis stopped"
    stop
  endif
  print *, "Bisecting load increment"
  write (21,'(a)') "Bisecting load increment"
  bisect load increment
  l_inc_r(0) = l_inc_r(-1) + 0.5*delta_l_inc
  bisects = bisects + 1
! set displacements to last converged displacements unless constant load
! is being held---in such case use bisection to give extra iterations
if (loading .eq. 1) then
  totdisp = cnvdisp
elseif (loading .eq. 2 .and. l_inc_fact .eq. .1 .and. astep .eq. 1) then
  totdisp = cnvdisp
elseif (loading .eq. 2 .and. astep .gt. 1) then
  totdisp = totdisp
endif
! increase load increment size if analysis showing good convergence
elseif (iter(0) .eq. 1 .and. iter(-1) .le. 3) then
  l_inc_r(0) = l_inc_r(-1) + 1.25*delta_l_inc
! default behaviour is to increment by last increment
else
  l_inc_r(0) = l_inc_r(-1) + delta_l_inc
endif
! calc load increment size for current increment
delta_l_inc = l_inc_r(0) - l_inc_r(-1)
! calc load scaling factor
if (loading .eq. 1 .or. (loading .eq. 2 .and. astep .eq. 1)) then
  l_inc_fact = l_inc_r(0)/load_incs
elseif (loading .eq. 0) then
  l_inc_fact = 1. - l_inc_r(0)/load_incs
elseif (loading .eq. 2 .and. astep .ne. 1) then
  l_inc_fact = 1.
endif
! prevent load from exceeding applied load
if (l_inc_fact .gt. 1.) then

```

```

    l_inc_fact = 1.
    elseif (l_inc_fact .lt. .1 .and. astep .ne. 1) then
    l_inc_fact = .1
endif
! scale applied load for load increment
apl_inc = l_inc_fact * aload
! write scale factor to output
print *, "Proportion of applied load is ", l_inc_fact
write (21,'(a,f5.3)') "Proportion of applied load is ", l_inc_fact
! calculate Euclidean norm and set convergence criterion if required
lnorm_ap = (dot_product(apl_inc,apl_inc))*0.5
lconv = mlconv
if (c_val .eq. 2 .or. lconv) then
    l_crit = ltoler*lnorm_ap
    print *, "Load norm is ", lnorm_ap
    write(21,'(a,e12.6)') "Load norm is ", lnorm_ap
    print *, "Load convergence criterion is ", l_crit
    write(21,'(a,e12.6)') "Load convergence criterion is ", l_crit
    if (lnorm_ap .lt. 1d-8) then
        l_crit = 0.05
    endif
endif
! start Newton-Raphson convergence iterations-----
!
converged = .false.
iterations: do while ((converged .eq. .false.) .and. (iter(0) &
    & .le. maxiter))
! calculate residual load vector here if first iteration
if (iter(0) .eq. 1) then
    bdylds = .0 ; g_ipSigma = .0
    call bodyloads (g_g,g_num,g_numip,g_coord,prop,cnvdisp,tot_inc,&
        mat_id,mat_dam,mat_cpore,etype,gaptol,g_ipDam,g_ipPoros,&
        crackstat,bdylds,g_ipSigma)
! compute residual load vector and norm for current step
resld = apl_inc - bdylds
rlnorm = (dot_product(resld,resld))*0.5
endif
if (iter(0) .eq. 1 .and. astep .eq. 1) lastresid = apl_inc
!-----Tangent stiffness formation-----
!
message
print *, "Beginning element stiffness integration and assembly..."
write (21,'(a)') &
    "Beginning element stiffness integration and assembly..."
!
call form_K_tan (g_g,g_num,g_numip,g_coord,prop,totdisp,incdisp,&
    mat_id,mat_dam,mat_cpore,etype,gaptol,g_ipDam,g_ipPoros,&
    crackstat,solver,g_k,kdiag,kdiag)
! use penalty approach for applied displacements
if (fixed_nodes/=0) then
    if (solver .eq. 'ban') then
        g_k(no)=g_k(no) + 1.e20
        loads(no) = g_k(no) * value
    endif
endif
!-----equation solution-----
!
message
210 format (tr1,a,i3,a,i3,a,i3)
211 format (a,i4,a,i4,a,i4,a,/)
212 format (a,i3,a,i3,a/)
240 format (i4,tr2,i4,tr2,i4,tr1,e12.6,tr1,e12.6,tr1,e12.6,tr1,&
    & e12.6,tr1,e12.6)
print 210,"Solving equations for analysis step", astep, ", load_inc ", &
    & l_inc, ", iteration ", iter(0)
write(21,210) &
    "Solving equations for analysis step", astep, ", load_inc ", &
    & l_inc, ", iteration ", iter(0)

```

```

! calculate displacement increment from Kt*incdisp = resld
! i.e. by invoking relevant solver on residual
!
select case (solver)
case ('ban')
    print *, "Performing Gaussian reduction on global stiffness"
    write(21,'(a)') "Performing Gaussian reduction on global stiffness"
    call banred(g_k,neq)
    print *, "Performing back-substitution"
    write(21,'(a)') "Performing back-substitution"
    call bacsub(g_k,resld)
    incdisp = resld
case ('sky')
    skyRhs (:,1) = resld(1:neq)
    t_solve_in = .0
    call cpu_time(t_solve_in)
    call skysolve (g_k,kdiag,skyRhs,21)
    call cpu_time(t_solve_out) ; t_solve = t_solve_out - t_solve_in
    print *, "Time to solve system was ",t_solve
    write(21,'(a,f)') "Time to solve system was ",t_solve
    incdisp(1:neq) = skyRhs(:,1)
end select
!
message
print *, "Equation solution finished for iteration", iter(0)
write(21,'(a,i3)') "Equation solution finished for iteration ", iter(0)
! calculate some displacement norms
idnorm = (dot_product(incdisp,incdisp))*0.5
if (astep .eq. 1 .and. load_incs .eq. 1) idnorm0 = idnorm
tdnorm = (dot_product(totdisp,totdisp))*0.5
! if 1st iteration set displacement tolerance
if (iter(0) .eq. 1) then
    if (astep .eq. 1 .and. c_val .ge. 1) then
        d_crit = dtoler*idnorm
    endif
! ensure rdnorm > d_crit for 1st iteration
rdnorm = idnorm
else
    resldisp = totdisp + incdisp - lastdisp
    rdnorm = dot_product(incdisp,incdisp)
endif
print *, "Displacement increment norm is ", rdnorm
write(21,'(a,e12.6)') "Displacement increment norm is ", rdnorm
if (c_val .ge. 1) then
    print *, "Criterion is ", d_crit
    write(21,'(a,e12.6)') "Criterion is ", d_crit
endif
!-----recover internal bodyloads-----
!
bdylds = .0 ; g_ipSigma = .0
call bodyloads (g_g,g_num,g_numip,g_coord,prop,totdisp,incdisp,&
    mat_id,mat_dam,mat_cpore,etype,gaptol,g_ipDam,g_ipPoros,&
    crackstat,bdylds,g_ipSigma)
! compute residual load vector and norm for current step
resld = apl_inc - bdylds
rlnorm = (dot_product(resld,resld))*0.5
lnorm = (dot_product(bdylds,bdylds))*0.5
!-----perform line-search-----
!
if (rlnorm .gt. l_crit) then
    call line_search (apl_inc,lastresid,resld,rlnorm,totdisp,incdisp,&
        g_g,g_num,g_numip,g_coord,prop,mat_id,mat_dam,mat_cpore,etype,&
        gaptol,g_ipDam,g_ipPoros,crackstat,lsearch,25)
endif
! update last residual and calculate current residual norm
rlnorm = (dot_product(resld,resld))*0.5
cnv_ratio = rlnorm/last_rlnorm
! write messages

```



```

print *, "Line search parameter is ", lsearch
write(21,(a,f6.4)) "Line search parameter is ", lsearch
print *, "Residual load convergence norm is ", rlnorm
write(21,(a,f12.6)) "Residual load convergence norm is ", rlnorm
print *, "Criterion is ", l_crit
write(21,(a,f12.6)) "Criterion is ", l_crit
!
write convergence info to file
write (24,240) astep, l_inc, iter(0), tdnorm, idnorm, lnorm, rlnorm,&
& l_crit
!
update total displacement vector
totdisp = totdisp + incdisp
!
update total displacement increment
tot_inc = tot_inc + incdisp
!
idnorm = (dot_product(tot_inc,tot_inc))*0.5
store total displacement and residual
lastdisp = totdisp
lastresid = resld
last_rlnorm = rlnorm
!
reset displacement increment to zero
incdisp = .0
!-----check for convergence-----
if (lconv .and. dconv) then
  if (rlnorm .lt. l_crit .and. rdnorm .lt. d_crit) then
    converged = .true.
  else
    converged = .false.
  endif
elseif (lconv) then
  if (rlnorm .lt. l_crit) then
    converged = .true.
  else
    converged = .false.
  endif
elseif (dconv) then
  if (rdnorm .lt. dtoler) then
    converged = .true.
  else
    converged = .false.
  endif
endif
iter(-1) = iter(0)
if (converged .and. l_inc_fact .lt. 1 .and. loading .eq. 1) then
  print 212, " Load increment",l_inc," has converged after ", iter(0),&
  " iterations"
  write(21,212) "Load increment",l_inc,&
  " has converged after ", iter(0)," iterations"
  l_inc_r(-1) = l_inc_r(0)
  bisects = 0
  cnvdisp = cnvdisp + tot_inc
  cnvlds = bdylds
  g_ipSigcnv = g_ipSigma
  load_loop = .true.
  exit
elseif (converged .and. l_inc_fact .gt. 0.1 .and. loading .eq. 0) then
  print 212, " Load increment",l_inc," has converged after ", iter(0),&
  " iterations"
  write(21,212) "Load increment",l_inc,&
  " has converged after ", iter(0)," iterations"
  l_inc_r(-1) = l_inc_r(0)
  bisects = 0
  cnvdisp = cnvdisp + tot_inc
  cnvlds = bdylds
  g_ipSigcnv = g_ipSigma
  load_loop = .true.
  exit
elseif (converged .and. l_inc_fact .lt. 1 .and. loading .eq. 2) then
  print 212, " Load increment",l_inc," has converged after ", iter(0),&
  " iterations"
  write(21,212) "Load increment",l_inc,&
  " has converged after ", iter(0)," iterations"
  l_inc_r(-1) = l_inc_r(0)
  bisects = 0
  cnvdisp = cnvdisp + tot_inc
  cnvlds = bdylds
  g_ipSigcnv = g_ipSigma
  load_loop = .true.
  exit
elseif (converged .and. l_inc_fact .eq. 1.) then
  print 211," Analysis step ",astep," has converged after ", l_inc,&
  " load increments and", totiter," iterations"
  write(21,211) "Analysis step ",astep," has converged after ",l_inc,&
  " load increments and", totiter," iterations"
  l_inc_r(-1) = l_inc_r(0)
  bisects = 0
  cnvdisp = cnvdisp + tot_inc
  cnvlds = bdylds
  g_ipSigcnv = g_ipSigma
  load_loop = .false.
  l_inc = 0
  exit
elseif (converged .and. l_inc_fact .eq. 0.1 .and. loading .eq. 0) then
  print 211," Analysis step ",astep," has converged after ", l_inc,&
  " load increments and", totiter," iterations"
  write(21,211) "Analysis step ",astep," has converged after ",l_inc,&
  " load increments and", totiter," iterations"
  l_inc_r(-1) = l_inc_r(0)
  bisects = 0
  cnvdisp = cnvdisp + tot_inc
  cnvlds = bdylds
  g_ipSigcnv = g_ipSigma
  load_loop = .false.
  l_inc = 0
  exit
elseif (iter(0) .ge. maxiter .and. converged .eq. .false.) then
  cnv_err = rlnorm - l_crit
  if (cnv_err .lt. 0.5*l_crit .and. cnv_ratio .lt. 0.2) then
    !
    exhibiting convergent behaviour so keep iterating
    iter(0) = iter(0) + 1
    !
  else
    !
    not close enough to convergence to warrant further iterations
    iter(0) = 1
    totdisp = cnvdisp
    g_ipSigma = g_ipSigcnv
    exit
  endif
elseif
  iter(0) = iter(0) + 1
  endif
  totiter = totiter + 1
  enddo iterations
enddo load_increments
!-----nonconverged termination-----
if not converged after maxiter iterations then stop execution
if (converged .eq. .false.) then
  print *, "Analysis step ",astep," failed to converge after ", iter(0)-1,&
  " iterations---terminating analysis!"
  print *, "Total elapsed time is ", time
  write(21,(a,i3,a,i3)) &
  & "Analysis step ",astep," failed to converge after &
  ", iter(0)-1," iterations---terminating analysis!"
  write(21,(a,f20.7)) "Total elapsed time is", time
  !
  assume that if maxiter is 1 then a linear analysis has been performed
  !
  i.e. continue in spite of failed convergence, otherwise terminate
  if (maxiter .ne. 1) stop
  endif
!-----calc gauss pt stresses and strains, and map to nodes-----
!
reset global stress arrays
g_ipEpot = .0 ; g_eptot = .0 ; g_ipSigma = .0 ; g_ipPrSig = .0
g_stress = 0. ; g_PrSig = .0
!
calculate effective stresses
t_visc = 604800. + time/5.
call calc_stress (g,g_num,g_numip,g_coord,prop,totdisp,incdisp,t_visc,&

```

```

mat_id,mat_dam,mat_cpore,mat_visc,etype,v_scheme,gaptol,&
g_ipDam,g_ipPoros,crackstat,g_ipEpnit,g_ipEptot,&
g_ipSigma)
! loop elements
do iel=1,neltot
num = g_num(:,iel) ; coord = transpose(g_coord(:,num))
numip = g_numip(:,iel)
g = g_g(:,iel)
! loop gauss pts
! solids
if (etype(iel) .eq. 1) then
do ip=1,nip(1)
! retrieve stress tensor
sigma = g_ipSigma(:,numip(ip))
sigma_el(:,ip) = sigma
! calculate and store principal values
call vprinvl (sigma,v_scheme,g_ipPrSig(:,numip(ip)))
pr_el(:,ip) = g_ipPrSig(:,numip(ip))
end do
extrapolate/copy to nodes
select case (ipout)
extrapolation
case ('extrap')
call hex8ExtrapSample (nod(1),solpoints,solwts)
do inod=1,nod(1)
call hex8ShapeFun (solpoints(inod,:),solFun)
do j=1,nst(1)
stress(j) = dot_product(solFun(1:nod(1)),sigma_el(j,:))
if (j .le. ndim) pr_nod(j) = dot_product(solFun(1:nod(1)),&
& pr_el(j,:))
end do
g_stress(:,num(inod)) = g_stress(:,num(inod)) &
+ (1./nshare(num(inod)))*stress
g_PrSig(:,num(inod)) = g_PrSig(:,num(inod)) &
+ (1./nshare(num(inod)))*pr_nod
end do
copy gauss pt values to nodes
case ('copyip')
do inod=1,nod(1)
g_stress(:,num(inod)) = g_stress(:,num(inod)) &
+ (1./nshare(num(inod)))*sigma_el(:,inod)
g_PrSig(:,num(inod)) = g_PrSig(:,num(inod)) &
+ (1./nshare(num(inod)))*pr_el(:,inod)
g_eptot(:,num(inod)) = g_eptot(:,num(inod)) &
+ (1./nshare(num(inod)))*g_ipEptot(:,numip(inod))
end do
end select
endif
end do
!-----calculate timestep-----
! only calc timestep and damage if model is being loaded (unloading will result
! in very long tstep)
if (loading .eq. 1 .or. loading .eq. 2) then
! message
print *, "Estimating new timestep"
write(21,'(a)') "Estimating new timestep"
!
max_inc = 200000000
tstep_min = max_inc
tstep_min2 = tstep_min
! reset timestep failure indicator array
tstep_fail = 0
! loop elements
damage_1:do iel=1,neltot
if (etype(iel) .eq. 1) then
numip = g_numip(:,iel)
do imat=1,ndam
if (mat_id(iel) .eq. mat_dam(imat)) then
loop gauss pts
do ip=1,nip(1)
actrot = 0. ; ncrack = crackstat(numip(ip))
solDam = g_ipRefDam(:,numip(ip))
sigma = g_ipSigma(:,numip(ip))
call formsytens_12(solDam,refdam)
! only calculate damage if at least one direction remains
! undamaged
!
if (ncrack .lt. 3) then
and if failure in less than min tstep is not flagged
call damgrowth(refdam,ncrack,'murphy',uts(imat),&
& sn_slope(imat),sigma,tstep,newdam)

```

```

do ip=1,nip(1)
ncrack = crackstat(numip(ip))
if (ncrack .ne. 3) then
sigma = g_ipSigma(:,numip(ip))
SolDam = g_ipDam(:,numip(ip))
call tstep_bisect(sigma,uts(imat),sn_slope(imat),SolDam,&
& v_scheme,ncrack,'murphy',tstep)
! check for minimum timestep
!
if (tstep .lt. tstep_min) then
first replace 2nd shortest time step
tstep_min2 = tstep_min
update minimum time step to new value
tstep_min = tstep
endif
endif
end do
endif
end do damage_1
! select minimum tstep unless time exceeds test time
if ((time + tstep_min) .le. maxtime) then
tstep = tstep_min
elseif (time .eq. maxtime) then
tstep = 1
else
tstep = maxtime - time
endif
!-----calculate damage at end of block-----
! message
print *, "Updating damage"
write(21,'(a)') "Updating damage"
initiate = .false.
lastcrack = sum(crackstat)
! loop twice in case min tstep is not enough to initiate a crack
g_ipRefDam = g_ipDam
damage_2:do i=1,2
! break condition if any cracks have initiated
if (initiate) then
exit
elseif (initiate .eq. .false. .and. i .ne. 1) then
use 2nd shortest timestep
if ((time + tstep_min) .le. maxtime) then
tstep = tstep_min2
elseif (time .eq. maxtime) then
tstep = 1
else
tstep = maxtime - time
endif
endif
! rezero nodal damages and total damage and crack variables
g_dam = .0 ; g_PrDam = .0 ; totdam = .0 ; totcrack = 0
! loop elements
do iel=1,neltot
if (etype(iel) .eq. 1) then
numip = g_numip(:,iel)
do imat=1,ndam
if (mat_id(iel) .eq. mat_dam(imat)) then
loop gauss pts
do ip=1,nip(1)
actrot = 0. ; ncrack = crackstat(numip(ip))
solDam = g_ipRefDam(:,numip(ip))
sigma = g_ipSigma(:,numip(ip))
call formsytens_12(solDam,refdam)
! only calculate damage if at least one direction remains
! undamaged
!
if (ncrack .lt. 3) then
and if failure in less than min tstep is not flagged
call damgrowth(refdam,ncrack,'murphy',uts(imat),&
& sn_slope(imat),sigma,tstep,newdam)

```

```

!
! check if any values have ruptured and explicitly set to
! exactly one if so ; also update no. cracks for pt.
! call r2prinval(newdam,prindam)
! ncrack = 0
! do j=1,ndim
!   if (prindam(j) .gt. 0.95) then
!     prindam(j) = 1
!     ncrack = ncrack + 1
!   endif
! enddo
! update no. cracks at gauss pt and for material
! crackstat(numip(ip)) = ncrack
! totcrack(imat) = totcrack(imat) + ncrack
! update total damage (i.e. sum of principal damages)
! totdam(imat) = totdam(imat) + sum(prindam)
! form tensor of principal damages and rotate back to global
! csys
! prdamtens = 0. ; prdamtens(1,1) = prindam(1)
! prdamtens(2,2) = prindam(2) ; prdamtens(3,3) = prindam(3)
! call r2rot_t(newdam,prinrot)
! newdam = matmul(transpose(prinrot), &
!   & matmul(prdamtens, prinrot))
!
! else
!   every direction cracked so damage is unchanged
!   newdam = rfdam
!   still have to update total cracks and damage as they are
!   zeroed at every iteration
!   totcrack(imat) = totcrack(imat) + 3
!   update total damage (i.e. sum of principal damages)
!   totdam(imat) = totdam(imat) + 3
! endif
! update global gauss pt and local element damage vector
! call formvoigt_12(newdam,solDam)
! dam_el(:,ip) = solDam
! g_ipDam(:,numip(ip)) = solDam
! calculate and store principal values
! call vprinval (solDam,v_scheme,g_ipPrDam(:,numip(ip)))
! pr_el(:,ip) = g_ipPrDam(:,numip(ip))
! enddo
! copy to nodes
! num = g_num(:,iel)
! do inod=1,nod(1)
!   g_dam(:,num(inod)) = g_dam(:,num(inod)) &
!     + (1./nshare(num(inod)))*dam_el(:,inod)
!   g_PrDam(:,num(inod)) = g_PrDam(:,num(inod)) &
!     + (1./nshare(num(inod)))*pr_el(:,inod)
! end do
! enddo
! end do
! end do
! enddo
! ncrack = sum(crackstat)
! if (ncrack .gt. lastcrack) initiate = .true.
! enddo damage_2
!
! message
! print *, "New timestep is", tstep
! write(21,'(a,f20.7)') "New timestep is", tstep
! endif
!-----output results-----
!
! message
! print *, "Updating results output files"
! write(21,'(a)') "Updating results output files"
!
! set formatting width of astep for start of step
! call numwidth(astepl,width)
!
! force results output if displacement incrmnt discontinuity has occurred
! if (time .ge. (time + tstep)) then
!   ostep = astep
!   ipstep = astep
!
! endif
!
! Output for beginning of current step-----
!
! Stress-volume data and results summary
! re-zero table before filling
! sigvol_tab = .0
!
! main body of table
! sigvol_tab(1,4)= time
! sigvol_tab(1,5)= time + tstep
! sigvol_tab(1,6)= astep
! sigvol_tab(1,7)= totiter
! call stressvol (g_ipSigma,v_scheme,g_ipvol,g_ipMat,sigMin,sigMax,sigvol_tab)
! format statements used for different parts of table
! 10 format (a,tr4,a,tr3,a,tr15,a,tr10,a,tr10,a,tr2,a,tr2,a)
! 11 format (f3.0,tr7,f5.1,tr1,f15.6,tr1,f18.6,tr1,f18.6,tr3,f4.0,tr4,f4.0,&
!   tr4,f6.0,f15.2)
! 12 format (f3.0,tr7,f5.1,tr1,f15.6)
! write table headings
! if (astepl .eq. 1) then
!   write (23,10) " mat_id","stress","volume","tstart","t_end","astep",&
!     & "iters","ncracks" totdam"
! else
!   write (23,*)
! endif
! endif
! initialise output, material and porosity counters, and start main table loop
! lastmat = 0
! do i=1,ubound(sigvol_tab,1)
!   write full line if material id changes
!   if (sigvol_tab(i,1) .ne. lastmat) then
!     do j=1,ndam
!       if (sigvol_tab(i,1) .eq. mat_dam(j)) then
!         sigvol_tab(i,8) = totcrack(j)
!         sigvol_tab(i,9) = totdam(j)
!       endif
!     enddo
!     write (23,11) sigvol_tab (i,:)
!   else
!     write (23,12) sigvol_tab (i,1:3)
!   endif
!   lastmat = sigvol_tab(i,1)
! enddo
!
! nodal results listing
! lists displacement components as:
! node1 node2 node3 ... node1 node2 node3...
! ux ux ux uy uy uy uz uz uz
! if (astepl .eq. 1) write (26,*) nlis_idarr, nlis_idarr, nlis_idarr
! do i=1,listnodes
!   k = nlis_idarr(i)
!   nlis_arr(i) = totdisp(nf(1,k))
!   j = i + listnodes
!   nlis_arr(j) = totdisp(nf(2,k))
!   nlis_arr(j + listnodes) = totdisp(nf(3,k))
! enddo
! write (26,*) nlis_arr
!
! vtk point data sections
! 102 format (f15.6,tr1,f15.6,tr1,f15.6)
! if (astepl .eq. ostep) then
!   write nodal stresses to '.res' file
!   write (20,'(a,TR1,a,i<width>,TR1,a)') "TENSORS", "S",astepl-1, "float"
!   & ".res")
!   call write_globArray (g_stress,ftype,20,ndim,v_scheme,fname(1:status),&
!     & ".res")
!   write nodal principal stresses to '.res' file
!   write (20,'(a,TR1,a,i<width>,TR1,a)') "VECTORS", "prS",astepl-1, "float"
!   call write_globArray (g_PrSig,ftype,20,ndim,v_scheme,fname(1:status),&
!     & ".res")
!   write nodal strains to '.res' file
!   write (20,'(a,TR1,a,i<width>,TR1,a)') "TENSORS", "eptot",astepl-1, "float"
!   call write_globArray (g_eptot,ftype,20,ndim,v_scheme,fname(1:status),&
!     & ".res")
!

```

```

! gauss pt stresses to '.ipr' file
  if (astep .eq. ipstep) then
    write (22,'(a,TR1,a,i<width>,TR1,a)') "TENSORS", "S",astep-1, "float"
    call write_globArray (g_ipSigma,ftype,22,ndim,v_scheme,fname(1:status),&
      & '.ipr')
    write (22,'(a,TR1,a,i<width>,TR1,a)') "VECTORS", "prS",astep-1, "float"
    call write_globArray (g_ipPrSig,ftype,22,ndim,v_scheme,fname(1:status),&
      & '.ipr')
  endif
! write nodal displacements to '.res' file
write (20,'(a,TR1,a4,i<width>,TR1,a)') "VECTORS", "disp",astep-1, "float"
select case (ftype)
case ('ascii')
do i=1,nn
  write (20,102) totdisp(nf(:,i))
end do
case ('binar')
fname = fname(1:status)//'.res'
close (20)
open (20,file=fname,position='append',form='binary',status='old',&
  action='write')
do i=1,nn
  sgl_vec = totdisp(nf(:,i))
  write (20) sgl_vec
end do
close (20)
open (20,file=fname,position='append',status='old',action='write')
write (20,*)
end select
! Output for present time-----
call numwidth(astep,width)
  if (ndam .ne. 0) then
    write nodal damages to '.res' file
    write (20,'(a,TR1,a3,i<width>,TR1,a)') "TENSORS", "D",astep, "float"
    call write_globArray (g_dam,ftype,20,ndim,v_scheme,fname(1:status),&
      & '.res')
    write nodal principal damages to '.res' file
    write (20,'(a,TR1,a4,i<width>,TR1,a)') "VECTORS", "prD",astep, "float"
    call write_globArray (g_PrDam,ftype,20,ndim,v_scheme,fname(1:status),&
      & '.res')
  !
  ! gauss pt damage tensors to '.ipr' file
  if (astep .eq. ipstep) then
    write (22,'(a,TR1,a3,i<width>,TR1,a)') "TENSORS", "D",astep, "float"
    call write_globArray (g_ipDam,ftype,22,ndim,v_scheme,fname(1:status),&
      & '.ipr')
    write (22,'(a,TR1,a3,i<width>,TR1,a)') "VECTORS", "prD",astep, "float"
    call write_globArray (g_ipPrDam,ftype,22,ndim,v_scheme,&
      & fname(1:status),'.ipr')
    ipstep = ipstep + ipout_inc
  endif
endif
!
! set next analysis step for output
if (loading .eq. 0 .or. loading .eq. 2) then
  ostep = ostep + nout_inc
elseif (loading .eq. 1) then
  ostep = ostep + 1
endif
!
! update time and step counter for loop and print to output-----
time = time + tstep
astep = astep + 1
iter(-1) = iter(0) ; iter(0) = 1
load_loop = .true.
if (loading .eq. 1) then
  loading = 0

```

```

elseif (loading .eq. 0) then
  loading = 1
endif
endif
write(*,'(tr1,a,f20.7,/)') "Total elapsed time is", time
write(21,'(tr1,a,f20.7,/)') "Total elapsed time is", time
! force exit if displacement increment discontinuity has occurred
if (idnorm .ge. 1.75*idnorm0 .and. load_incs .eq. 1) then
  print *, "Displacement increment exceeded allowable limit",&
    & "---probable specimen failure"
  write(21,'(a)') "Displacement increment exceeded allowable limit"&
    & "---probable specimen failure"
  exit
endif
! close timestep loop
enddo timesteps
! message
print *, "Analysis complete --- program terminated"
write(21,'(a)') "Analysis complete --- program terminated"
end program cdm_fem

module closedPorous
contains
subroutine random_pores (xm,xstd,rm,rstd,g_ipvol,g_ipMat,g_ipbkt,g_ipCoord,&
  mat_cpore,g_ipPoros,avPoros)
! subroutine to randomly generate pores according to a normal distribution
! requires IMSL Stat libraries
!
! algorithm is based on
! (i) generating a random standard normal distribution which
! is then scaled to the mean and standard deviation desired (results in
! 'tvec')
! (ii) a pore is said to exist at some entry of tvec if its value falls
! within
! some tolerance of the mean
! (iii) the ratio of porous pts to total pts is then calculated and checked
! against the desired mean and the tolerance is scaled up or down until the
! percentage porosity falls within one standard deviation of the desired
! mean
!
! variables-----
! in-----
! xm = mean porosity of distribution
! xstd = standard deviation of porosity of distribution
! rm = mean pore radius of distribution
! rstd = standard deviation of pore radius of distribution
! g_ipvol = on input contains array of volumes for each point of material
! on output contains the void ratio for each point
! g_ipMat = global array of gauss pt material id's
! g_ipCoord = global array of gauss pt coordinates
! out-----
! g_ipPoros = global array of gauss pt porosities
! internal-----
! tvec = trial vector produced by generator containing real values
! rvec = vector produced by generator containing pore radii
! pvol = vector containing pore volumes
! pvol_xs = excess volume of pore (a pore with vol > the vol attributed to
! the pt at its centre has excess vol: pvol(pt) - vol(pt))
! n = size of vec (i.e. no. points to be sampled for pore existence)
! npores = no. pores generated (real valued to prevent rounding down or
! up)
! cpores = array of closed pores (1 => present; 0 => not present)
! this is derived from the real valued vector produced by random
! number generator
! p2g = array transforming pore material numbering to global numbering
! g2p = array transforming global numbering to pore material numbering
! numbkt = global gauss pt no.s for a given gauss pt bucket
! s = coordinates of source pt in spatial search through a bucket

```

```

! t      = coords if target pt in spatial search through a bucket
! dvec   = distance vector between s and t
! dist   = distance between source and target pts
! mindist = minimum distance for a given search (used when no pts fall
!         within desired distance)
! min_id  = id of pt at min distance from source pt
! tot_pvol = total volume of pores in active material for material loop
! totvol  = total volume of active material for material loop
! testm   = test of mean of sample
! testmlast = previous testm (used for bisection iterations)
! scaletol = scale factor used to increase the tolerance
! lastscale = scale factor for previous 2 iterations
! i       = loop counter
! iter    = iteration counter
! ncomp   = no. independent components in porosity tensor
! ndim    = no. coordinate dimensions
! max_bkt = max no. of pts in a gauss pt bucket
! ibkt    = counter for loops through buckets
! npts    = no. pts within pore radius
!-----
! call IMSL libs
use numerical_libraries
implicit none
integer,intent(in)::mat_cpore(:),g_ipMat(:),g_ipbkt(:,:)
real(8),intent(in)::xm(:),xstd(:),rm(:),rstd(:),g_ipvol(:),&
& g_ipCoord(:,:)
real(8),intent(out)::g_ipPoros(:),avPoros(:)
real(8),allocatable::vol(:),tvec(:),rvec(:),pvol(:),s(:),t(:),&
dvec(:)
integer,allocatable::cpores(:),p2g(:),g2p(:),numbkt(:),pbkt(:)
real(8)::testm,testmlast,scaletol,pi,lastscale(-2:-1),tot_pvol,totvol,&
dist,mindist,pvol_xs,r_p,r_ip
integer::npmat,ipore,imat,niptot,ip,i,iter,ptot,ncomp,max_bkt,ibkt,ndim,&
& npts,min_id,iseed
logical::converged
! assign a value for pi for use in pore volume calcs
pi = 3.14159
npmat = size(mat_cpore)
niptot = size(g_ipvol)
ncomp = ubound(g_ipPoros,1)
max_bkt = ubound(g_ipbkt,1)
ndim = ubound(g_ipCoord,1)
allocate (poros(ncomp),numbkt(max_bkt),pbkt(max_bkt),s(ndim),t(ndim),&
& dvec(ndim))
! loop through mesh to determine how many pts are allowed to have porosity
! and what their volumes are; then allocate relevant arrays
do imat=1,npmat
! loop gauss pts
ptot = 0
do ip=1,niptot
if (g_ipMat(ip) .eq. mat_cpore(imat)) then
ptot = ptot + 1
endif
enddo
allocate (cpores(ptot),vol(ptot),tvec(ptot),rvec(ptot), &
pvol(ptot),p2g(ptot),g2p(niptot))
! loop again to retrieve gauss pt volumes of porous materials
ipore = 1 ; g2p = 0
do ip=1,niptot
if (g_ipMat(ip) .eq. mat_cpore(imat)) then
vol(ipore) = g_ipvol(ip)
p2g(ipore) = ip
g2p(ip) = ipore
ipore = ipore + 1
endif
enddo
! trial vector of pore existence-----

```

```

! set seed
call rnget(iseed)
call rnset(iseed)
generate standard normally distributed numbers
call drnnoa (ptot, tvec)
! scale to actual standard deviation
tvec = xstd(imat)*tvec
offset to actual mean
tvec = tvec + xm(imat)
! trial vector of pore radius-----
! generate standard normally distributed numbers
set seed
call rnget(iseed)
call rnset(iseed)
call drnnoa (ptot, rvec)
! scale to actual standard deviation
rvec = rstd(imat)*rvec
! offset to actual mean
rvec = rvec + rm(imat)
! ensure all +ive radii
do ip=1,ptot
rvec(ip) = dabs(rvec(ip))
enddo
! Initialise to no. pores and loop through tvec to determine presence
! of pores. Retrieve pore radius and calc void ratio etc. and finally
! check total porosity until trial distribution falls within one
! standard deviation of mean note: scaletol is also randomly generated
! since setting a fixed initial value for every run tends to bias the
! distribution towards one side; the means from different simulation
! do not then match the desired means and standard deviations
! generate random number between 0 and 1
set seed
call rnget(iseed)
call rnset(iseed)
call drnun(1,s(1))
scaletol = 2*s(1)
testm = .0 ; lastscale = .0
iter = 1
do while (testm .lt. (xm(imat) - xstd(imat)) .or. &
testm .gt. (xm(imat) + xstd(imat)) .or. iter .eq. 1)
! break condition
if (iter .ge. 1000) then
print *, "Random pore generation did not converge within 1000 &
stop & iterations"
endif
! generate pore vector according to tolerance
cpores = 0
do ip=1,ptot
if ((xm(imat) - scaletol*xstd(imat)) .le. tvec(ip) &
.and. tvec(ip) .le. (xm(imat) + scaletol*xstd(imat))) then
cpores(ip) = 1
endif
enddo
! calculate individual void volumes
pvol = .0
do ip=1,ptot
! calculate pore volume if gauss pt is flagged for a pore and if
! pore radius
! is +ive
if (cpores(ip) .ne. 0 .and. pvol(ip) .eq. 0) then
pvol(ip) = 4./3.*pi*rvec(ip)**3.
! don't allow pore vol > gauss point vol
if (pvol(ip) .gt. vol(ip)) then
! find all gauss pts within pore radius of current gauss pt
numbkt = g_ipbkt(:,p2g(ip))
pbkt = 0 ; npts = 0
! calc excess volume
pvol_xs = pvol(ip) - vol(ip)
! set source pt

```

```

s = g_ipCoord(:,p2g(ip))
do i=1,max_bkt
! exit loop if zero entry in bucket array
if (numbkt(i) .eq. 0) exit
! check whether pt is porous material, does not already
! have a pore
if (g_ipMat(numbkt(i)) .eq. mat_cpore(imat)) then
if (cpores(g2p(numbkt(i))) .eq. 0) then
! set target pt and calculate distance
t = g_ipCoord(:,numbkt(i))
dvec = t - s
dist = dot_product(dvec,dvec)**0.5
! check if target falls within pore radius
if (dist .le. rvec(ip)) then
npts = npts + 1
pbkt(npts) = numbkt(i)
endif
! check for pt within min distance of source pt
if (i .eq. 1) then
mindist = dist
min_id = numbkt(i)
elseif (dist .lt. mindist) then
mindist = dist
min_id = numbkt(i)
endif
endif
endif
enddo
! set pore volumes of surrounding pts
if (npts .eq. 0 .and. min_id .ne. 0) then
if (g2p(min_id) .gt. 0) pvol(g2p(min_id)) = pvol_xs
else
do i=1,npts
if (pbkt(i) .ne. 0 .and. g2p(pbkt(i)) .gt. 0) then
if (pvol_xs/npts .gt. vol(g2p(pbkt(i)))) then
pvol(g2p(pbkt(i))) = vol(g2p(pbkt(i)))
else
pvol(g2p(pbkt(i))) = pvol_xs/npts
endif
endif
enddo
endif
! set pore volume of pore centre pt equal to entire pt
! volume
pvol(ip) = vol(ip)
endif
endif
end do
! test mean total porosity and exit loop if criterion is satisfied
tot_pvol = sum(pvol)
totvol = sum(vol)
testm = tot_pvol/totvol
if (testm .gt. (xm(imat) - 2*xstd(imat)) .and. &
testm .lt. (xm(imat) + 2*xstd(imat))) then
converged = .true.
exit
elseif (sum(cpores) .eq. ptot) then
converged = .false.
exit
endif
! increase/decrease tolerance if necessary using bisection
if (iter == 1) testmlast = testm
if (testm .lt. (xm(imat) - xstd(imat)) .and. &
testmlast .lt. (xm(imat) - xstd(imat))) then
! both below -std => too many rejected => increase tol
lastscale(-1) = scaletol
scaletol = scaletol + 0.5*dabs(lastscale(-2) - scaletol)
elseif (testm .gt. (xm(imat) + xstd(imat)) .and. &
testmlast .gt. (xm(imat) + xstd(imat))) then
! both above +std => too many accepted => decrease tol
lastscale(-1) = scaletol

```

```

scaletol = scaletol - 0.5*dabs(lastscale(-2) - scaletol)
elseif (testm .lt. (xm(imat) - xstd(imat)) .and. &
testmlast .gt. (xm(imat) + xstd(imat))) then
! current mean below -std and last above +std
! => too many rejected => increase tol
lastscale(-2) = scaletol
scaletol = scaletol + 0.5*dabs(lastscale(-1) - scaletol)
elseif (testm .gt. (xm(imat) + xstd(imat)) .and. &
testmlast .lt. (xm(imat) - xstd(imat))) then
! current mean above +std and last below -std
! => too many accepted => decrease tol
lastscale(-2) = scaletol
scaletol = scaletol - 0.5*dabs(lastscale(-1) - scaletol)
else
return
endif
if (iter .ne. 1) testmlast = testm
iter = iter + 1
end do
! scale pore volumes if non-converged exit from loop and mean is too
! low/high
! this can occur when the mean pore volume is significantly .lt. or
! .gt. each gauss pt volume
if (converged .eq. .false.) then
! start by setting a scale factor
if (testm .lt. (xm(imat) - xstd(imat))) then
scaletol = 2.
elseif (testm .gt. (xm(imat) - xstd(imat))) then
scaletol = 0.5
endif
! loop until the desired mean is achieved
iter = 1
do
! loop porous gauss pts
do ip=1,ptot
! scale pore vol
pvol(ip) = scaletol*pvol(ip)
! don't allow pores .gt. ip vol
if (pvol(ip) .gt. vol(ip)) pvol(ip) = vol(ip)
enddo
! test mean total porosity and exit loop if criterion is
! satisfied
tot_pvol = sum(pvol)
totvol = sum(vol)
testm = tot_pvol/totvol
if (testm .gt. (xm(imat) - xstd(imat)) .and. &
testm .lt. (xm(imat) + xstd(imat))) then
converged = .true.
exit
endif
! increase/decrease tolerance if necessary using bisection
if (iter == 1) testmlast = testm
if (testm .lt. (xm(imat) - xstd(imat)) .and. &
testmlast .lt. (xm(imat) - xstd(imat))) then
! both below -std => pores too small => increase scaletol
lastscale(-1) = scaletol
scaletol = scaletol + 0.5*dabs(lastscale(-2) - scaletol)
elseif (testm .gt. (xm(imat) + xstd(imat)) .and. &
testmlast .gt. (xm(imat) + xstd(imat))) then
! both above +std => pores too big => decrease scaletol
lastscale(-1) = scaletol
scaletol = scaletol - 0.5*dabs(lastscale(-2) - scaletol)
elseif (testm .lt. (xm(imat) - xstd(imat)) .and. &
testmlast .gt. (xm(imat) + xstd(imat))) then
! current mean below -std and last above +std
! => pores too small => increase scaletol
lastscale(-2) = scaletol
scaletol = scaletol + 0.5*dabs(lastscale(-1) - scaletol)
elseif (testm .gt. (xm(imat) + xstd(imat)) .and. &
testmlast .lt. (xm(imat) - xstd(imat))) then
! current mean above +std and last below -std
! => pores too big => decrease scaletol

```

```

        lastscale(-2) = scaletol
        scaletol = scaletol - 0.5*dabs(lastscale(-1) - scaletol)
    endif
enddo
endif
! loop gauss pts to store porosities
do ip=1,ptot
! zero poros
poros = .0
! set porosity according to void ratio
if (pvol(ip) .ne. .0) then
    r_p = ((3.*pvol(ip))/(4.*pi)**(1./3.))
    r_ip = ((3.*vol(ip))/(4.*pi)**(1./3.))
    poros(1:3) = pvol(ip)/vol(ip)
    if (poros(1) .gt. 1.) poros(1:3) = 1.
endif
! store in element and global arrays
g_ipPoros(:,p2g(ip)) = poros
enddo
! deallocate arrays to enable reallocation for next porous material
deallocate (cpores,vol,tvec,rvec,pvol,p2g,g2p)
! store average porosity
avPoros(imat) = testm
enddo
return
end subroutine random_pores

subroutine cporeDee(dee,e,v,vr)
! returns the elastic stiffness matrix of a material with closed pore
! microstructure
! Note: models porosity as isotropic damage without closure capability
! in-----
! e      = young's modulus for isotropic material
! v      = poisson's ration
! vr     = void ratio (i.e. (vol pore)/(vol region))
! out-----
! dee   = stiffness matrix in global coordinate system (must be returned by
!         user)
! internal variables-----
! nst=   = no. stress/strain components
!         3->plane stress
!         4->axisymmetry or plane strain elastoplasticity
!         6->general three dimensional
! pore_eff = rank4 pore-effect tensor (matrix form)
! -----
use linelastic ; use tensor_ops
!
implicit none
real(8),intent(in)::e,v,vr
real(8),intent(out)::dee(:, :)
real(8),allocatable::pore_eff(:, :)
integer::nst,dim
!
! assign dimensionality
nst = ubound(dee,1)
select case (nst)
case (6)
    dim = 3
case (3)
    dim = 3
case (2)
    dim = 2
end select
allocate (pore_eff(nst,nst))
! -----
! exit condition: if no pore exists then return elastic 'dee'
call isoDee(dee,e,v)
if (vr .eq. .0) then
    return

```

```

endif
! -----
! construct pore-effect matrix (analogue of damage effect operator)
! only contains diagonal components so can zero and fill
pore_eff = 0.
if (vr .lt. 1.) then
! normal components
    pore_eff(1,1) = (1.-vr)
    pore_eff(2,2) = (1.-vr)
    if (dim .eq. 3) then
        pore_eff(3,3) = (1.-vr)
    elseif (dim .eq. 2) then
! 2D shear component
        pore_eff(3,3) = (((1.-vr)**0.5)*((1.-vr)**0.5))
    endif
! shear components
    if (dim .eq. 3) then
        pore_eff(4,4) = ((1.-vr)**0.5)*((1.-vr)**0.5)
        pore_eff(5,5) = ((1.-vr)**0.5)*((1.-vr)**0.5)
        pore_eff(6,6) = ((1.-vr)**0.5)*((1.-vr)**0.5)
    endif
endif
! -----
! calculate effective stiffness in crack csys
!
dee = matmul(pore_eff,matmul(dee,transpose(pore_eff)))
return
end subroutine cporeDee

subroutine poreStress(uts,vr,vscheme,stress)
! returns the elastic stiffness matrix of a material with closed pore
! microstructure
! Note: models porosity as isotropic damage
! in-----
! uts   = ultimate tensile stress for material
! vr    = void ratio (i.e. (vol pores)/(vol region))
! vscheme = voigt storage scheme
! in/out-----
! stress = stress tensor (voigt form)
! internal variables-----
! nst    = no. stress/strain components
!         3->plane stress
!         4->axisymmetry or plane strain elastoplasticity
!         6->general three dimensional
! s_tens = stress tensor (matrix form)
! pore_eff = rank4 pore-effect tensor (matrix form)
! -----
use tensor_ops
implicit none
real(8),intent(in)::uts,vr
integer,intent(in)::vscheme
real(8),intent(inout)::stress(:, :)
real(8),allocatable::pore_eff(:,:),tensor(:,:),prval(:),rot(:, :)
integer::nst,dim,i
!
! assign dimensionality
nst = size(stress,1)
select case (nst)
case (6)
    dim = 3
case (4)
    dim = 2
case (3)
    dim = 3
case (2)
    dim = 2
end select
allocate (pore_eff(nst,nst),tensor(dim,dim),prval(dim),rot(dim,dim))

```

```

! default to 0 pore effective operator (to be used for the case of porosity
! equal to 1)
pore_eff = .0
! construct pore-effect matrix (analogue of damage effect operator)
! only contains diagonal components so can zero and fill
! write out in full form in case a future version needs to use non-
! isotropic pores
if (vr .lt. 1.) then
! normal components
pore_eff(1,1)= 1./(1.-vr)
pore_eff(2,2) = 1./(1.-vr)
if(dim .eq. 3) then
pore_eff(3,3) = 1./(1.-vr)
elseif (dim .eq. 2) then
pore_eff(3,3) = 1./(((1.-vr)**0.5)*((1.-vr)**0.5))
endif
! shear components
if (dim .eq. 3) then
pore_eff(4,4) = 1./(((1.-vr)**0.5)*((1.-vr)**0.5))
pore_eff(5,5) = 1./(((1.-vr)**0.5)*((1.-vr)**0.5))
pore_eff(6,6) = 1./(((1.-vr)**0.5)*((1.-vr)**0.5))
endif
endif
! calculate effective stress
stress = matmul(pore_eff, stress)
! check if principal stresses exceed ultimate tensile strength-----
! convert stress vector to tensor and find principal values and rotation matrix
if (vscheme .eq. 12) then
call formsyntens_12 (stress, tensor)
elseif (vscheme .eq. 23) then
call formsyntens_23 (stress, tensor)
endif
call r2prinall(tensor, prval, rot)
tensor = 0
do i=1,dim
if (prval(i) .gt. uts) then
tensor(i,i) = uts
else
tensor(i,i) = prval(i)
endif
enddo
! rotate stress back to global csys and convert to voigt storage
tensor = matmul(transpose(rot), matmul(tensor, rot))
if (vscheme .eq. 12) then
call formvoigt_12 (tensor, stress)
elseif (vscheme .eq. 23) then
call formvoigt_23 (tensor, stress)
endif
return
end subroutine poreStress

subroutine poreStrain(vr, vscheme, strain)
! returns the elastic stiffness matrix of a material with closed pore
! microstructure
! Note: models porosity as isotropic damage
!
! in-----
! vr = void ratio (i.e. (vol pores)/(vol region))
! vscheme = voigt storage scheme
!
! in/out-----
! strain = strain tensor (voigt form)
!
! internal variables-----
! nst = no. stress/strain components
! 3->plane stress
! 4->axisymmetry or plane strain
! 6->general three dimensional
! pore_eff = rank4 pore-effect tensor (matrix form)

```

```

!-----
use tensor_ops
implicit none
real(8), intent(in)::vr
integer, intent(in)::vscheme
real(8), intent(inout)::strain(:)
real(8), allocatable::pore_eff(:, :)
integer::nst, dim, i
! assign dimensionality
nst = size(strain, 1)
select case (nst)
case (6)
dim = 3
case (4)
dim = 2
case (3)
dim = 2
end select
allocate (pore_eff(nst, nst))
! default to 0 pore effective operator (to be used for the case of porosity
! equal to 1)
pore_eff = .0
! construct pore-effect matrix (analogue of damage effect operator)
! only contains diagonal components so can zero and fill
! write out in full form in case a future version needs to use non-
! isotropic pores
if (vr .lt. 1.) then
! normal components
pore_eff(1,1)= (1.-vr)
pore_eff(2,2) = (1.-vr)
if(dim .eq. 3) then
pore_eff(3,3) = (1.-vr)
elseif (dim .eq. 2) then
pore_eff(3,3) = ((1.-vr)**0.5)*((1.-vr)**0.5)
endif
! shear components
if (dim .eq. 3) then
pore_eff(4,4) = ((1.-vr)**0.5)*((1.-vr)**0.5)
pore_eff(5,5) = ((1.-vr)**0.5)*((1.-vr)**0.5)
pore_eff(6,6) = ((1.-vr)**0.5)*((1.-vr)**0.5)
endif
endif
! calculate effective strain
strain = matmul(pore_eff, strain)
return
end subroutine poreStrain
end module closedPorous

module elContactSurf
! module of common operations for contact analysis
contains
subroutine surfSampleGaussPtsLocal (nip, s, wt)
! calc local 2D coords of gauss pts in natural nodal surface csys
!
! in-----
! nip = no. gauss pts
!
! out-----
! s = array of sampling pts
! wt = weighting for each point
!
! internal-----
! root3 = temp variable used to prevent recalculation of dsqrt(3.)
!
implicit none
! declare variables
integer, intent(in)::nip

```



```

real(8),intent(out)::s(:,:),wt(:)
real(8)::root3
! select sampling according to no. gauss pts
select case(nip)
case(1)
! centroid only
s(1,1) = .0 ; s(1,2) = .0
wt(1) = 4.
case(4)
! 2x2 quadrature
! set local sampling points and weighting
root3 = dsqrt(3.)
s(1,1) = -1./root3; s(1,2) = -1./root3
s(2,1) = -1./root3; s(2,2) = 1./root3
s(3,1) = 1./root3; s(3,2) = 1./root3
s(4,1) = 1./root3; s(4,2) = -1./root3
wt = 1.0
end select
return
end subroutine surfSampleGaussPtsLocal
subroutine surfExtrapSample (nip,s,wt)
! calc local 2D coords of nodes in natural gauss surface csys
! in-----
! nip = no. gauss pts
! out-----
! s = array of sampling pts
! wt = weighting for each point
! internal-----
! root3 = temp variable used to prevent recalculation of dsqrt(3.)
implicit none
! declare variables
integer, intent(in)::nip
real(8),intent(out)::s(:,:),wt(:)
real(8)::root3
! select sampling according to no. gauss pts
select case(nip)
case(1)
! centroid only
s(1,1) = .0 ; s(1,2) = .0
wt(1) = 4.
case(4)
! 2x2 quadrature
! set local sampling points and weighting
root3 = dsqrt(3.)
s(1,1) = -root3; s(1,2) = -root3
s(2,1) = -root3; s(2,2) = root3
s(3,1) = root3; s(3,2) = root3
s(4,1) = root3; s(4,2) = -root3
wt = 1.0
end select
return
end subroutine surfExtrapSample
subroutine surfShapeFun (s,fun)
! calc 2D shape functions for quadrilateral surface
! in-----
! s = sampling point on surface
! out-----
! fun = array of shape functions
! internal-----
! xi,eta,zeta = natural coordinates of surface
implicit none
! declare variables

```

```

real(8),intent(in)::s(:)
real(8),intent(out)::fun(:)
real(8)::xi,eta
! assign gauss pt local coords
xi=s(1); eta=s(2)
! calculate shape functions
fun=(.25*(1.-xi)*(1.-eta),.25*(1.-xi)*(1.+eta),&
     .25*(1.+xi)*(1.-eta),.25*(1.+xi)*(1.-eta)/)
return
end subroutine surfShapeFun
subroutine surfShapeFunArr (fun,ntot)
! assemble total shape function array
! (used in calc of bee matrix and relative displacement vector)
! in-----
! fun = vector of shape functions
! out-----
! ntot = array of shape functions
! internal-----
! ntop = array of shape fns for top face
! nbot = ditto bottom
! l = length of ntot
! i,j = counters
implicit none
! declare variables
real(8),intent(in)::fun(:)
real(8),intent(out)::ntot(:,:)
real(8),allocatable::ntop(:,:),nbot(:,:)
integer::i,j,l
l = ubound(ntot,2)
select case (l)
case (24)
! 8 corner node quadriateral contact element
allocate (ntop(3,1/2),nbot(3,1/2))
case (12)
! 4 corner node quadrilateral single surface element or surface load
allocate (ntop(3,1),nbot(3,1))
end select
! fill shape function arrays
ntop = .0 ; nbot = .0 ; ntot = .0
j = 1
do i=1,3
ntop(i,j) = fun(1)
ntop(i,j+3) = fun(2)
ntop(i,j+6) = fun(3)
ntop(i,j+9) = fun(4)
j = j + 1
end do
nbot = -1.*ntop
ntot(:,1:1/2) = ntop ; ntot(:,(1/2)+1:1) = nbot
return
end subroutine surfShapeFunArr
subroutine surfShapeDer (s,nod,der)
! calculate 3d linear surface shape derivatives of a given point on the surface
! in-----
! s = sampling point on surface
! nod = no. nodes on surface
! out-----
! der = array of shape derivatives
! internal-----
! xi,eta,zeta = natural coordinates of surface
implicit none
! declare variables
real(8),intent(in)::s(:) ; integer,intent(in)::nod

```

```

real(8),intent(out)::der(:, :)
real(8)::xi, eta
! assign gauss pt local coords
xi=s(1); eta=s(2)
!
select case (nod)
case (4)
! calculate shape derivatives
! row 1
  der(1,1)= -.25*(1.-eta) ; der(1,2)= -.25*(1.+eta)
  der(1,3)= .25*(1.+eta) ; der(1,4)= .25*(1.-eta)
! row 2
  der(2,1)= -.25*(1.-xi) ; der(2,2)= .25*(1.-xi)
  der(2,3)= .25*(1.+xi) ; der(2,4)= -.25*(1.+xi)
end select
return
end subroutine surfShapeDer
subroutine surfJac (s,surfcoord,rot,jac)
! calc 3D jacobian of a surface
!
! in-----
! s           = coords of sampling pt
! surfcoord  = array of all nodal coords
!
! out-----
! jac        = jacobian matrix
!
! internal-----
! der        = array of shape function derivatives
! nod        = no. of nodes on surface
! lcoord     = surfcoord rotated to local csys
!
implicit none
! declare variables
real(8),intent(in)::s(:),surfcoord(:, :)
real(8),intent(out)::rot(:, :),jac(:, :)
real(8),allocatable::der(:, :),lcoord(:, :)
integer::nod,ndim
!
nod = ubound(surfcoord,1)
ndim = ubound(surfcoord,2)
allocate (der(2,nod),lcoord(nod,ndim))
! calculate shape derivatives
call surfShapeDer(s,nod,der)
! calculate rotation matrix
call surfRot(s,nod,surfcoord,rot)
! rotate coordinates to local csys (should make coeffs of 3rd dimension zero)
lcoord = transpose(matmul(transpose(rot),transpose(surfcoord)))
! calculate jacobian
jac = matmul(der,lcoord(:,1:2))
return
end subroutine surfJac
subroutine surfRot(s,nod,surfcoord,rot)
! calculate surface normal and tangent vectors using cross products of first
! two rows of jacobian and assemble into rotation matrix
!
! in-----
! s           = coords of sampling pt
! nod        = no. of nodes on surface
! surfcoord  = array of all nodal coords
!
! out-----
! rot        = rotation matrix - tangent and normal vectors stored in
!              each column
!
! internal-----
! jac        = 3D surface jacobian matrix
! n          = surface normal
! s1         = first tangent vector
! s2         = second (mutually orthogonal to n and s1) tangent vector

```

```

! gbv          = global base vector (used when calculating s1 from n)
!
! use tensor_ops ; implicit none
!
! declare variables
real(8),intent(in)::s(:),surfcoord(:, :) ; integer,intent(in)::nod
real(8),intent(out)::rot(:, :)
real(8)::der(2,4),n(3),s1(3),s2(3),gbv(3)
! calc shape derivatives
call surfShapeDer (s,nod,der)
! n
! (note:use s1 and s2 as dummy variables for rows 1 and 2 of jac)
s1(1) = dot_product(der(1,:),surfcoord(:,1))
s1(2) = dot_product(der(1,:),surfcoord(:,2))
s1(3) = dot_product(der(1,:),surfcoord(:,3))
s2(1) = dot_product(der(2,:),surfcoord(:,1))
s2(2) = dot_product(der(2,:),surfcoord(:,2))
s2(3) = dot_product(der(2,:),surfcoord(:,3))
call cross_product(s1,s2,n)
! normalise
n = n/dsqrt(dot_product(n,n))
! s1
! select appropriate form of global base vector in case n is in line with any
! n = (+/-1,0,0)
if (n(1) .eq. 1 .or. n(1) .eq. -1.) then
  gbv = (/0,1.,.0/)
! n = (0,+/-1,0)
elseif (n(2) .eq. 1 .or. n(2) .eq. -1.) then
  gbv = (/0.,0,1./)
! n = (0,+/-1,0)
elseif (n(3) .eq. 1 .or. n(3) .eq. -1.) then
  gbv = (/0,1.,.0/)
else
  gbv = (/1.,.0.,0/)
endif
call cross_product(gbv,n,s1)
s1 = s1/dsqrt(dot_product(s1,s1))
! s2 (uniquely defined as cross product of other two vectors)
call cross_product(n,s1,s2)
s2 = s2/dsqrt(dot_product(s2,s2))
! fill array of surface base vectors
rot(:,1) = s1
rot(:,2) = s2
rot(:,3) = n
return
end subroutine surfRot
subroutine surfBee (s,coord,det,bee)
! calc 'bee' matrix for a gauss pt
! note: contact bee matrix is used to relate gap vector at gauss pt
! to element nodal displacement vector
!
! in-----
! s           = sampling pt
! coord       = array of nodal coordinates
!
! out-----
! det        = determinant of Jacobian matrix (needed for numerical
!              integration)
! bee        = bee matrix
!
! internal-----
! fun        = shape functions
! ntot       = array of shape functions
! rot        = rotation matrix of surface normal and tangent vectors
!
use tensor_ops
! declare variables
implicit none
real(8),intent(in)::s(:),coord(:, :)

```

```

real(8),intent(out)::det,bee(:,.)
real(8)::rot(3,3)
real(8),allocatable::fun(:,),ntot(:,),jac2D(:,.)
integer::ndim,nod

ndim = ubound(coord,2)
nod = ubound(coord,1)
allocate (fun(nod),ntot(ndim,ndim*nod),jac2D(ndim-1,ndim-1))

! calculate shape functions and assemble shape function array
call surfShapeFun(s,fun)
call surfShapeFunArr(fun,ntot)
! calc jacobian and rotation matrix
call surfJac(s,coord(1:nod/2,:),rot,jac2D)
! get 2x2 determinant
call gen_det(jac2D,det)
! calc 'bee' matrix
bee = matmul(transpose(rot),ntot)
return
end subroutine surfBee

subroutine surfDebondDee (gap,gaptol,ks,kn,mu,dee)
! calculate interfacial stiffness based on relative displacement vector
!
! in-----
! gap = relative displacement vector
! ks = shear stiffness
! kn = normal stiffness
! mu = coefficient of friction
!
! out-----
! dee = dee (elasticity matrix) matrix
!
! internal-----
! fn = normal force
! rs = resultant shear force
! fy = yield force for slip
!
implicit none
! variable declartion
real(8),intent(in)::gap(:,),gaptol,ks,kn,mu
real(8),intent(out)::dee(:,.)
real(8)::fn,rs,fy

! initialise dee to zero (default is open)
dee = .0

! if open then return to calling routine
if (gap(3) .ge. gaptol) then
return
endif

! calc normal force
fn = kn*gap(3)
! calc limiting friction force
fy = dabs(mu*fn)
! calc resultant shear force
rs = dsqrt((ks*gap(1))**2. + (ks*gap(2))**2.)

! if normal traction is zero or compressive then assume closed and set normal
! stiffness to kn
if (fn .le. 0.0) then
dee(3,3) = kn
else
return
endif

! check for slip and set relevant stiffnesses
if (rs .le. fy) then
dee(1,1) = ks
dee(2,2) = ks
elseif (rs .gt. fy) then
if (gap(1) .ne. .0 .and. gap(2) .ne. .0) then
dee(1,1) = fy**2./dsqrt(gap(1)**2. + gap(2)**2.)
else
dee(1,1) = .0
endif
dee(2,2) = dee(1,1)

```

```

endif
return
end subroutine surfDebondDee

subroutine surfBondDee (ks,kn,dee)
! calculate interfacial stiffness based on relative displacement vector
!
! in-----
! gap = relative displacement vector
! ks = shear stiffness
! kn = normal stiffness
! mu = coefficient of friction
!
! out-----
! dee = dee (elasticity matrix) matrix
!
! internal-----
! fn = normal force
! rs = resultant shear force
! fy = yield force for slip
!
implicit none
! variable declartion
real(8),intent(in)::ks,kn
real(8),intent(out)::dee(:,.)

! initialise dee to zero for off diagonal entries
dee = .0

! set normal stiffness to kn
dee(3,3) = kn

! set tangential stiffnesses
dee(1,1) = ks
dee(2,2) = ks

return
end subroutine surfBondDee

end module elContactSurf

module damage
contains
subroutine shrinkage_dam (sigma,poros,uts,notch,vscheme,nrcrack,initdam)
! initiate damage due to shrinkage stress
!
! in-----
! sigma = shrinkage stress tensor (voigt form)
! poros = porosity tensor (voigt form)
! uts = ultimate tensile strength of non-porous material
! notch = notch parameter to reduce critical stress at pore surface
! vscheme = voigt storage scheme
!
! out-----
! nrcrack = no. of cracks initiated
! initdam = initiated damage tensor (voigt form)
use tensor_ops
real(8),intent(in)::sigma(:,),poros(:,),uts,notch
real(8),intent(out)::initdam(:)
integer,intent(in)::vscheme
integer,intent(out)::nrcrack
real(8),allocatable::prsig(:,),stress(:,),rot(:,),newdam(:,.)
real(8)::sig_fail
integer::ndim,nst,i

! find stress components no. and dimensions
nst = size(sigma)
select case (nst)
case (6)
ndim = 3
case (4)
ndim = 2
case (3)
ndim = 2
end select

! allocate necessary arrays
allocate (prsig(ndim),stress(ndim,ndim),rot(ndim,ndim),newdam(ndim,ndim))

```

```

! convert stress vector to tensor and find principal values and rotation matrix
if (vscheme .eq. 12) then
  call formsyntens_12 (sigma,stress)
elseif (vscheme .eq. 23) then
  call formsyntens_23 (sigma,stress)
endif
call r2prinall(stress,prsig,rot)
! set adapt ultimate tensile strength to account for lowered strength at the
! edge of a pore due to matrix-bead structure
sig_fail = notch*uts
! calculate initial damage in principal stress csys
newdam = .0
ncrack = 0
do i=1,ndim
  if (prsig(i) .gt. .0) then
    newdam(i,i) = prsig(i)/sig_fail
  else
    newdam(i,i) = .0
  endif
  ! apply a rupture criterion
  if (newdam(i,i) .ge. 0.95) then
    newdam(i,i) = 1.
    ncrack = ncrack + 1
  endif
enddo
! rotate damage back to global csys and convert to voigt storage
newdam = matmul(transpose(rot),matmul(newdam,rot))
initdam = .0
if (vscheme .eq. 12) then
  call formvoigt_12 (newdam,initdam)
elseif (vscheme .eq. 23) then
  call formvoigt_23 (newdam,initdam)
endif
return
end subroutine shrinkage_dam
subroutine damgrowth(refdam,ncrack,dam_model,a,b,sigma_ip,cycles_ref,newdam)
! forms complete damage matrix for given stress tensor and time step
!
! in-----
! refdam = damage tensor at beginning of loading block
! ncrack = no. cracks present for this point at reference time
! dam_model = damage growth model used
! a,b = S-N curve constants
! sigma_ip = voigt form of stress tensor for gauss pt
! cycles = no. of cycles for loading block
!
! out-----
! newdam = updated damage at end of loading block
!
! internal-----
! prin_stress = principal stresses
! stress = stress tensor
! actrot = rotation matrix: reference csys <-> principal stress,
!         i.e. active damage, csys
! act_dam = damage tensor rotated to active damage csys
! cycles_ref = equivalent no. of cycles req'd to induce damage in given active
!             direction for stress in that direction
! prindam = array of principal damages
! dambasis = array of principal damage base vectors (stored in columns)
! cycles_fail = failure life at a given stress level (from S-N curve)
! alpha = exponent for power type growth equation
! i,j = counters
! nst = no. stress components
! dim = no. dimensions (computed from nst)
use tensor_ops
implicit none
character*(*) intent(in)::dam_model
real(8),intent(in)::refdam(:,,:), sigma_ip(:,),a,b,cycles
real(8),intent(out)::newdam(:,,:)

```

```

real(8),allocatable::prin_stress(:,),stress(:,,:),actrot(:,,:),act_dam(:,,:), &
  prindam(:,), dambasis(:,,:)
real(8) cycles_fail,cycles_ref,alpha
integer::ncrack,i,j,nst,dim
nst = ubound(sigma_ip,1)
! check if ncrack = 3 (i.e. all directions cracked) and return if so
if (ncrack .eq. 3) then
  newdam = refdam
  return
endif
! set dimensionality
select case (nst)
case(1); dim = 1 ! 1D case
case(3); dim = 2 ! 2D case
case(6); dim = 3 ! 3D case
end select
allocate (prin_stress(dim), stress(dim,dim), actrot(dim,dim), &
  act_dam(dim,dim), prindam(dim), dambasis(dim,dim))
! convert stress vector to tensor and find principal values
call formsyntens_12 (sigma_ip,stress)
call r2prinall(stress,prin_stress,actrot)
! limit principal stresses to ultimate tensile strength
! - prevents excessively short time steps
newdam = .0
do i=1,dim
  if (prin_stress(i) .gt. a) then
    newdam(i,i) = a
  else
    newdam(i,i) = prin_stress(i)
  endif
enddo
! rotate stress back to global csys
stress = matmul(transpose(actrot),matmul(newdam,actrot))
newdam = .0
! choose damage model
select case (dam_model)
case('murphy')
  if (ncrack .lt. 2) then
    ! until two crack planes exist there is no unique damage csys and growth
    ! should be governed by principal stresses.
    ! Rotate refdam to active (principal stress) csys
    act_dam = matmul(actrot,matmul(refdam,transpose(actrot)))
    ! apply damage growth in principal stress directions
    do i=1,dim
      ! check if plane has cracked already
      if (act_dam(i,i) .lt. 1.) then
        calculate no. ref cycles req'd to induce this damage
        if (prin_stress(i) .gt. .0) then
          calculate growth
          call murphydam(act_dam(i,i),prin_stress(i),a,b,cycles,cycles_ref)
        endif
      endif
    enddo
  elseif (ncrack .eq. 2) then
    ! if 2 planes cracked already its better to rotate to the 3rd damage rather
    ! than stress direction as this direction is now uniquely defined by the
    ! other directions start by calculating principal damages and rotation
    ! matrix
    call r2prinbasis (refdam,prindam,dambasis)
    call r2rot_b(dambasis,actrot)
    ! rotate stresses and damages to damage csys
    ! (in theory undamaged component should be in position (3,3))
    stress = matmul(actrot,matmul(stress,transpose(actrot)))
    act_dam = matmul(actrot,matmul(refdam,transpose(actrot)))
    ! calculate growth for tensile stress
    if (stress(3,3) .gt. .0) then
      call murphydam(act_dam(3,3),stress(3,3),a,b,cycles,cycles_ref)
    else
      act_dam(3,3) = act_dam(3,3)
    endif
  endselect

```

```

    else
    ! all directions are cracked so return
    newdam = refdam
    return
    endif
end select
! rotate updated damage tensor back to global csys
newdam = matmul(transpose(actrot),matmul(act_dam,actrot))
return
end subroutine damgrowth

subroutine murphydam(dam,stress,a,b,cycles,cycles_tot)
! calculates damage for a given direction based on Murphy's power type model
! in-----
! stress      = applied stress
! a,b         = S-N curve constants
! cycles      = no. cycles at applied stress
! cycles_tot  = total elapsed no. cycles from previous load blocks
! in/out-----
! dam         = scalar damage value
implicit none
real(8),intent(in)::stress,a,b,cycles, cycles_tot
real(8),intent(inout)::dam
real(8) cycles_fail, alpha, cycles_eq
call sn_life(a,b,stress,cycles_fail)
call murphyalpha(stress,alpha)
! calculate equivalent elapsed cycles
cycles_eq = cycles_fail*dam**(1/alpha)
! allow damage growth only if stress is tensile
if (stress .gt. 0.1) then
    dam = ((cycles_eq + cycles)/cycles_fail)**alpha
else
    dam = dam
endif
! limit the maximum damage possible to avoid excessive large damage growth from
! crashing the tstep_bisect subroutine
if (dam .gt. 2.) dam = 2.
return
end subroutine murphydam

subroutine murphyalpha(stress,alpha)
! calculates damage for a given direction based on Murphy's Power Law model
implicit none
real(8),intent(in)::stress ; real(8),intent(out)::alpha
real(8) beta, gamma
! beta and gamma are material parameters for power law exponent
beta = 2.73; gamma = 5.6;
! limit maximum value of alpha
if (stress .gt. 55.) then
    alpha = (55. - gamma)/beta
elseif (stress .gt. (beta+gamma) .and. stress .le. 55.) then
    alpha = (stress - gamma)/beta
else
    alpha = 1.
endif
return
end subroutine murphyalpha

subroutine sn_life(a,b,stress,cycles_fail)
! calculates cycles to failure at a given stress according to SN data
! in-----
! a,b         = constants for S-N curve (i.e. S = a - b*log10(Nf))
! stress      = value of stress component
! out-----
! cycles_fail = no. cycles to failure at applied stress
implicit none

```

```

real(8),intent(in)::a,b,stress; real(8),intent(out)::cycles_fail
! calculate Nf
if (stress .gt. .0) then
    cycles_fail = 10**((a-stress)/b)
else
! return a very large value if stress is negative
    cycles_fail = 100000**((a-dabs(stress))/b)
endif
return
end subroutine sn_life

subroutine tstep_bisect(stress,a,b,dam,v_scheme,ncrack,dam_model,tstep)
! calculate time required to induce crack in at least 1 direction
! in-----
! stress      = voigt form of stress tensor
! a,b         = S-N curve constants (a = uts, b = slope of S-N curve)
! v_scheme    = voigt storage scheme variable (i.e. 12 or 23 format)
! ncrack      = no. cracks present for this point at reference time
! dam_model   = damage growth model used
! tequiv      = equivalent time req'd to cause pre-existing damage at current
!              stress
! out-----
! tstep       = timestep req'd to induce failure in one direction
! inout-----
! dam         = on input contains voigt form of damage tensor upto present
!              on output gives damage tensor for final predicted tstep (should
!              be ruptured in at least one direction)
! internal-----
! i           = principal direction flagged for rupture (depends on no. of
!              existinc cracks, ncrack)
! prindam     = principal damage values predicted from current iteration
! prindvec    = principal damage vectors
! prindamlast = principal damage values for start of current iteration
! sigma       = rank2 stress tensor
! prinstress  = principal stress values
! remlife     = remaining life values for each principal stress direction
! alpha       = damage growth exponent
! alphamax    = maximum damage growth exponent
! cycles_fail = no. cycles to failure at applied stress (from S-N curve)
! refdam      = damage tensor at start of timestep
! actdam      = refdam rotated to principal stress (active damage) csys
! actrot      = rotation matrix from global->active csys
! newdam      = trial damage tensor at a given iteration of tstep
! minstep     = minimum value of remlife array
! maxstep     = maximum value of remlife array
! min_id      = array subscript of minimum component of an array
! max_id      = array subscript of maximum component of an array
use tensor_ops
implicit none
character*(*) ,intent(in)::dam_model
real(8),intent(in)::stress(:),a,b
real(8),intent(inout)::dam(:)
real(8),intent(out)::tstep
real(8)::tequiv,toltest,minstep,maxstep,cycles_fail,understep,overstep,&
    alpha,dot1,dot2,dot11,dot21
real(8),allocatable::prindam(:),prindvec(:,:),crack(:,:),actdvec(:,:),&
    sigma(:,:),prinstress(:),actrot(:,:),remlife(:,:),&
    refdam(:,:),actdam(:,:),newdam(:,:)
integer,intent(in)::v_scheme, ncrack; integer::i,j,k,l,nst,dim,min_id,max_id
! Determine and set dimensionality of problem
nst = ubound(stress,1)
select case (nst)
case(1); dim = 1 ! 1D case
case(3); dim = 2 ! 2D case
case(6); dim = 3 ! 3D case
end select

```

```

allocate (prindam(dim),prindvec(dim,dim),crack(dim,dim),actdvec(dim,dim),&
sigma(dim,dim),prinstress(dim),remlife(dim),refdam(dim,dim),&
actdam(dim,dim),actrot(dim,dim),newdam(dim,dim))
! form symmetric tensor
select case(v_scheme)
case (12)
call formsytens_12(stress,sigma)
call formsytens_12(dam,refdam)
case (23)
call formsytens_23(stress,sigma)
call formsytens_23(dam,refdam)
end select
! calculate principal stresses, rotation matrix, and principal damage for
! active damage csys
call r2prinbasis(sigma,prinstress,actdvec)
actrot = transpose(actdvec) !call r2rot_t(sigma,actrot)
call r2prinbasis (refdam,prindam,prindvec)
! initialise variables
newdam = 0. ; actdam = 0. ; crack = .0
select case (ncrack)
case(0) ! target direction is '1' and all directions free to damage
i = 1
case(1) ! target direction is '2' and direction '1' is fixed
i = 2 ; crack(:,1) = prindvec(:,1)
case(2) ! target direction is '3' and directions '1'and '2' are fixed and,
! by right hand rule, so is direction '3'
i = 3 ; crack = prindvec
case(3) ! cracked in all three directions so return with large tstep
tstep = 1d9 ; return
end select
! check that all principal stresses are not very low or compressive and return
! to calling program with very large tstep if they all are
if (prinstress(1) .le. .01 .and. prinstress(2) .le. .01 &
.and. prinstress(3) .le. .01) then
tstep = 1d9
return
endif
! rotate to active csys and calculate equivalent reference time and
! remaining life estimate for each principal stress direction
! absolute values reqd when very numerically small damages are rotated and
! become negative
actdam = matmul(actrot,matmul(refdam,transpose(actrot)))
!alphamax = 0.
do j=1,dim
call sn_life(a,b,prinstress(j),cycles_fail)
call murphyalpha(prinstress(j),alpha)
! split expression over two lines for readability
tequiv = cycles_fail*(dabs(actdam(j,j))**(1/alpha))
remlife(j) = cycles_fail - tequiv
enddo
! get minimum remaining life estimate for tstep
! (maximum used to identify intermediate value only in case that zero is
! returned)
call minvalue(remlife,minstep,min_id)
call maxvalue(remlife,maxstep,max_id)
if (minstep == 0.) then
do k=1,dim
dot1 = dot_product(crack(:,1),actdvec(:,k))
dot2 = dot_product(crack(:,2),actdvec(:,k))
if (k .ne. min_id .and. k .ne. max_id .and. dot1 .lt. 0.01 .and. &
dot2 .lt. 0.01) then
dir 'k' has non-zero remaining life and not in any existing crack
!
direction
minstep = remlife(k)
min_id = k
exit
endif
endif

```

```

end do
endif
tstep = minstep
! initialise tstep for undershoot to zero for bisection loop
understep = .0
! calculate damage at trial step and test for rupture
call damgrowth(refdam,ncrack,dam_model,a,b,stress,tstep,newdam)
call r2prinbasis (newdam,prindam,prindvec)
dot11 = dot_product(crack(:,1),prindvec(:,1))
!
! calculate tstep to cause overshoot and store last tstep before overshoot
if (prindam(1) .gt. 1. .and. dot11 .lt. 0.1) then
! trial step already causes failure
overstep = tstep
else
! increase tstep until overshoot
do k=1,100
exit condition
if (prindam(1) .ge. 2.) exit
!
double tstep
tstep = 2*tstep
!
calc. damage growth and principal values
call damgrowth(refdam,ncrack,dam_model,a,b,stress,tstep,newdam)
call r2prinbasis (newdam,prindam,prindvec)
dot1 = dot_product(crack(:,1),prindvec(:,1))
dot2 = dot_product(crack(:,2),prindvec(:,1))
dot11 = dot_product(crack(:,1),prindvec(:,1))
!
check for over- or undershoot
if (prindam(1) .gt. 1.1 .and. dot11 .lt. 0.1) then
!
damage direction has overshoot
overstep = tstep
exit
elseif (prindam(1) .le. 1. .and. prindam(i) .lt. 1. .and. &
dot1 .lt. 0.1 .and. dot2 .lt. 0.1) then
understep = tstep
endif
end do
endif
! if the loop ran the full 100 iterations then assume no overshoot and try
! some multiple of tstep as the overshoot
if (k .ge. 100) then
overstep = 1d9
endif
! start main bisection loop
do k=1,100
! set break condition
if (toltest .gt. 0.01 .or. prindam(1) .gt. 1.) then
tstep = overstep
exit
endif
! calculate tstep
tstep = understep + 0.5*(overstep - understep)
! calculate damage at trial step
call damgrowth(refdam,ncrack,dam_model,a,b,stress,tstep,newdam)
call r2prinbasis (newdam,prindam,prindvec)
dot1 = dot_product(crack(:,1),prindvec(:,1))
dot2 = dot_product(crack(:,2),prindvec(:,1))
dot11 = dot_product(crack(:,1),prindvec(:,1))
!
overshoot
if (prindam(1) .gt. 1. .and. dot11 .lt. 0.01 .and. tstep .lt. overstep) &
overstep = tstep
!
undershoot
if (prindam(1) .le. 1. .and. prindam(i) .lt. 1. .and. dot1 .lt. 0.01 &
.and. dot2 .lt. 0.01 .and. tstep .gt. understep) understep = tstep
! test for loop control damage criterion and increment counter
toltest = dabs(1 - prindam(i))
enddo
! assume that if maximum iterations have elapsed that tstep was not found

```

```

! and assign a very large step
if (k .ge. 100 .or. overstep .eq. 1d9 .or. tstep .lt. 0.1) tstep = 1d9
return
end subroutine tstep_bisect

subroutine damdee(dee,e,v,d,v,eptot,nrcrack,lbound,optype)
! returns the elastic stiffness matrix of a user defined material
! current implementation for orthogonally cracking material (bone cement)
! requires damage to be coupled in principal damage csys
! therefore
! (i) rotate stiffness matrix to damage csys
! (ii) couple damage into stiffness
! (iii) rotate back to global csys
!
! special requirements:
! (i) 4th order rotations in matrix form
! (ii) 4th order projections of crack normals in matrix form
! - required for tracking crack strain and implementing closure
!
! note: works with voigt form of damage tensor instead of established crack
! vectors this is to allow future implementation of coupling before
! prindam(i,i)=1
!
! nst=3->plane stress; nst=4->axisymmetry or plane strain elastoplasticity
! nst=6->general three dimensional
!
!-----
in
e = young's modulus for isotropic material
v = poisson's ration
d_v = damage tensor stored as vector (i.e. {d11,d22,d33,d12,d23,d31})
eptot = total strain (.12,23,31 voigt form) of gauss pt
nrcrack = no cracks for this point
optype = operation type: 0 => used for calculating stress for unconverged
        solution 1 => " " " " " converged
!-----
inout
dee = stiffness matrix in global coordinate system
      (must be passed in as undamaged stiffness and returned by user)
!-----
local variables-----
!-----
! prdam = vector of principal values
! crk = matrix of eigenvectors (stored as rows) for each principal damage
! direction
! dam_eff = rank 4 damage effect tensor represented as voigt matrix
! dee_eff = effective stiffness matrix
! dee_0 = undamaged stiffness matrix
! epcrk = vector of normal crack strains
! eptmp = temp voigt strain vector used for projected voigt strain
! r = rank 2 rotation matrix
! p1,p2,p3 = matrices of rank 4 projection operator for each crack
! t_e = rotation matrix for voigt strain/stiffness
! i.e: d = transpose(t_e).d'.t_e
! t_s = rotation matrix for voigt stress/stiffness
! i.e. d' = t_s.d.transpose(t_s)
!
! i = counter
! nst = no. stress-strain relations
! a = damage coupling component in 1st principal damage direction
! b = damage coupling component in 2nd principal damage direction
! c = damage coupling component in 3rd principal damage direction
! sp = softening parameter used for convergence stability
!-----
use tensor_ops ; use linelastic
!
implicit none
real(8),intent(in)::e,v,d_v(:,),eptot(:,),lbound
integer,intent(in)::nrcrack,optype
real(8),intent(inout)::dee(:,)
real(8),allocatable::prdam(:,),crk(:,),r2dam(:,),&

```

```

dam_eff(:,),dee_eff(:,),dee_0(:,),&
epcrk(:,),eptmp(:,),&
r(:,),t_e(:,), t_s(:,),&
p1(:,), p2(:,), p3(:,),&
dee1(:,),dee2(:,),dee3(:,)

real(8)::a,b,c,sp
integer::i,nst,dim,vscheme; nst = ubound(dee,1)
allocate (dam_eff(nst,nst),dee_eff(nst,nst),dee_0(nst,nst),eptmp(nst),&
          p1(nst,nst),p2(nst,nst),p3(nst,nst),t_e(nst,nst),t_s(nst,nst),&
          dee1(nst,nst),dee2(nst,nst),dee3(nst,nst))
!
! assign dimensionality
select case (nst)
! 3d
case (6)
dim = 3
case default
print *, "Only 3d case is supported at this time"
end select
allocate (prdam(dim),crk(dim,dim),r2dam(dim,dim),epcrk(dim),r(dim,dim))
! assign voigt storage scheme
vscheme = 12
!-----
! exit condition: if no damage exists then return elastic 'dee'
if (nrcrack .eq. 0) then
return
endif
!-----
! check if any directions have cracks in them and set relevant damage values
! and matrix for rank 4 projection
call vprinall (d_v,vscheme,prdam,r)
call r4rot_12 (r,t_s,t_e)
! direction 1
if (prdam(1) .gt. lbound) then
call r4proj_12 (crk(:,1),p1)
eptmp = matmul(p1,eptot)
epcrk(1) = eptmp(1) + eptmp(2) + eptmp(3)
a = 1.
else
p1 = 0. ; a = 0.
endif
! direction 2
if (prdam(2) .gt. lbound) then
call r4proj_12 (crk(:,2),p2)
eptmp = matmul(p2,eptot)
epcrk(2) = eptmp(1) + eptmp(2) + eptmp(3)
b = 1.
else
p2 = 0. ; b = 0.
endif
! direction 3
if (dim .eq. 3) then
if (prdam(3) .gt. lbound) then
call r4proj_12 (crk(:,3),p3)
eptmp = matmul(p3,eptot)
epcrk(3) = eptmp(1) + eptmp(2) + eptmp(3)
c = 1.
else
p3 = 0. ; c = 0.
endif
endif
endif
!-----
! set undamaged stiffness matrix
dee_0 = dee
!
! construct damage effect matrix in crack csys
! only contains diagonal components so can zero and fill
dam_eff = 0.
! normal components
dam_eff(1,1) = (1.-a)

```

```

dam_eff(2,2) = (1.-b)
if(dim .eq. 3) then
  dam_eff(3,3) = (1.-c)
elseif(dim .eq. 2) then
  if(a .lt. 1. .and. b .lt. 1.) then
    dam_eff(3,3) = ((1.-a)**0.5)*((1.-b)**0.5)
  else
    dam_eff(3,3) = .0
  endif
endif
! shear components
if(dim .eq. 3) then
  if(a .lt. 1. .and. b .lt. 1.) then
    dam_eff(4,4) = ((1.-a)**0.5)*((1.-b)**0.5)
  else
    dam_eff(4,4) = .0
  endif
  if(b .lt. 1. .and. c .lt. 1.) then
    dam_eff(5,5) = ((1.-b)**0.5)*((1.-c)**0.5)
  else
    dam_eff(5,5) = .0
  endif
  if(a .lt. 1. .and. c .lt. 1.) then
    dam_eff(6,6) = ((1.-a)**0.5)*((1.-c)**0.5)
  else
    dam_eff(6,6) = .0
  endif
endif
! calculate effective stiffness in crack csys
dee_eff = matmul(dam_eff,matmul(dee_0,transpose(dam_eff)))
! rotate effective stiffness back to global csys
! dee_eff = transpose(t_e).dee_eff'.t_e
dee_eff = matmul(transpose(t_e),matmul(dee_eff,t_e))
-----
! check for crack activation/deactivation-----
! this is achieved by decomposing the strain tensor into a strain tensor for
! each crack the trace of each decomposition corresponds to normal crack strain
! for that particular crack; i.e. e_crk(i) = trace(p(i):etot) (tensor notation)
! if +ive then use the projection operator to implement the effective stiffness
! for that crack; i.e. dee_eff_i = transpose(p(i)).dee_eff.p(i)
! the activation/deactivation is applied by summing the contribution for each
! crack
! first check crack strains and convert p(i) to loss of stiffness contribution
! if req'd
!
! crack 1
if(ncrack .eq. 1) then
  if(epcrk(1) .le. .0) then
    sp = 1.0
  elseif(epcrk(1) .gt. .0 .and. epcrk(1) .le. 25d-6 .and. optype .eq. 0) then
    sp = 1. + ((0.5-1.)/(25d-6 - .0))*(epcrk(1)-.0)
  elseif(epcrk(1) .gt. 25d6 .and. epcrk(1) .le. 50d-6 .and. optype .eq. 0) &
    & then
    sp = 0.5 + ((0.1-0.5)/(50d-6 - 25d-6))*(epcrk(1)-25d-6)
  elseif(epcrk(1) .gt. 50d6 .and. epcrk(1) .le. 100d-6 .and. optype .eq. 0) &
    & then
    sp = 0.1 + ((0.01-0.1)/(100d-6 - 50d-6))*(epcrk(1)-50d-6)
  else
    sp = .0
  endif
  dee1 = sp*matmul(transpose(p1),matmul((dee_0 - dee_eff),p1))
else
  dee1 = .0
endif
! crack 2
if(ncrack .eq. 2) then
  if(epcrk(2) .le. .0) then
    sp = 1.0
  elseif(epcrk(2) .gt. .0 .and. epcrk(2) .le. 25d-6 .and. optype .eq. 0) then
    sp = 1. + ((0.5-1.)/(25d-6 - .0))*(epcrk(2)-.0)

```

```

  elseif(epcrk(2) .gt. 25d6 .and. epcrk(2) .le. 50d-6 .and. optype .eq. 0) &
    & then
    sp = 0.5 + ((0.1-0.5)/(50d-6 - 25d-6))*(epcrk(2)-25d-6)
  elseif(epcrk(2) .gt. 50d6 .and. epcrk(2) .le. 100d-6 .and. optype .eq. 0) &
    & then
    sp = 0.1 + ((0.01-0.1)/(100d-6 - 50d-6))*(epcrk(2)-50d-6)
  else
    sp = .0
  endif
  dee2 = sp*matmul(transpose(p2),matmul((dee_0 - dee_eff),p2))
else
  dee2 = .0
endif
! crack 3
if(ncrack .eq. 3) then
  if(epcrk(3) .le. .0) then
    sp = 1.0
  elseif(epcrk(3) .gt. .0 .and. epcrk(3) .le. 25d-6 .and. optype .eq. 0) then
    sp = 1. + ((0.5-1.)/(25d-6 - .0))*(epcrk(3)-.0)
  elseif(epcrk(3) .gt. 25d6 .and. epcrk(3) .le. 50d-6 .and. optype .eq. 0) &
    & then
    sp = 0.5 + ((0.1-0.5)/(50d-6 - 25d-6))*(epcrk(3)-25d-6)
  elseif(epcrk(3) .gt. 50d6 .and. epcrk(3) .le. 100d-6 .and. optype .eq. 0) &
    & then
    sp = 0.1 + ((0.01-0.1)/(100d-6 - 50d-6))*(epcrk(3)-50d-6)
  else
    sp = .0
  endif
  dee3 = sp*matmul(transpose(p3),matmul((dee_0 - dee_eff),p3))
else
  dee3 = .0
endif
! sum to give active effective stiffness-----
dee = dee_eff + dee1 + dee2 + dee3
return
end subroutine damdee

subroutine dam_stress (dam,vscheme,sigma)
! calculate stress required to produce equivalent strain in undamaged material
! based on an average stress already computed using an energy equivalence model
! in-----
! dam = voigt vector of damages
! vscheme = voigt storage scheme
! inout-----
! sigma = stress vector
use tensor_ops ; use linelastic
implicit none
real(8),intent(in)::dam(:)
integer,intent(in)::vscheme
real(8),intent(inout)::sigma(:)
real(8),allocatable::prdam(:),dam_eff(:,,:),rot(:,,:),t_e(:,,:),t_s(:,,:)
real(8)::a,b,c
integer::dim,nst
nst = size(dam)
allocate (dam_eff(nst,nst),t_e(nst,nst),t_s(nst,nst))
! assign dimensionality
select case (nst)
! 3d
case (6)
  dim = 3
case default
  print *, "Only 3d case is supported at this time"
end select
allocate (prdam(dim),rot(dim,dim))
call vprinall (dam,vscheme,prdam,rot)
a = prdam(1)
b = prdam(2)
if (dim .eq. 3) c = prdam(3)
if (vscheme .eq. 12) then

```



```

    call r4rot_12 (rot,t_s,t_e)
elseif (vscheme .eq. 23) then
    call r4rot_23 (rot,t_s,t_e)
endif
! construct damage effect matrix in crack csys
! only contains diagonal components so can zero and fill
! also calculate inverse directly since matrix is diagonal
dam_eff = 0.
! normal components
dam_eff(1,1) = 1./((1.-a))
dam_eff(2,2) = 1./((1.-b))
if (dim .eq. 3) then
    dam_eff(3,3) = 1./((1.-c))
elseif (dim .eq. 2) then
    if (a .lt. 1. .and. b .lt. 1.) then
        dam_eff(3,3) = 1./(((1.-a)**0.5)*((1.-b)**0.5))
    else
        dam_eff(3,3) = .0
    endif
endif
! shear components
if (dim .eq. 3) then
    if (a .lt. 1. .and. b .lt. 1.) then
        dam_eff(4,4) = 1./(((1.-a)**0.5)*((1.-b)**0.5))
    else
        dam_eff(4,4) = .0
    endif
    if (b .lt. 1. .and. c .lt. 1.) then
        dam_eff(5,5) = 1./(((1.-b)**0.5)*((1.-c)**0.5))
    else
        dam_eff(5,5) = .0
    endif
    if (a .lt. 1. .and. c .lt. 1.) then
        dam_eff(6,6) = 1./(((1.-a)**0.5)*((1.-c)**0.5))
    else
        dam_eff(6,6) = .0
    endif
endif
! rotate damage effect operator back to global csys
dam_eff = matmul(transpose(t_e),matmul(dam_eff,t_e))
! calculate stress
sigma = matmul(dam_eff,sigma)
return
end subroutine dam_stress
end module damage

module elements
! module containing common tasks required in FE solution process
! e.g. nodal restoring loads and tangent stiffness calculation
contains
subroutine form_K_tan (g_g_num,g_numip,g_coord,prop,totdisp,incdisp,&
    mat_id,mat_dam,mat_cpore,etype,gaptol,g_ipDam,g_ipPoros,&
    crackstat,solver,g_k,g_ka,g_kb)
! form global tangent stiffness matrix
! g_k = global stiffness matrix stored as 1-d array
! g_ka = 1st locator array (used for all schemes; e.g. kdiag for skyline)
! g_kb = 2nd locator array (used for compressed storage schemes)
! user supplied elements (hex8 = 1, contact = 2)
use hex8_xtr ; use elcontactsurf
! user materials
use linelastic ; use damage ; use closedPoros
! matrix-tensor manipulation and solver front-ends
use tensor_ops ; use solvers
implicit none
real(8),intent(in)::g_coord(:,,:),prop(:,,:),totdisp(0:),incdisp(0:),&
    gaptol,g_ipDam(:,,:),g_ipPoros(:,,:)
integer,intent(in)::g_g(:,,:),g_num(:,,:),g_numip(:,,:),g_ka(:,),g_kb(:,),&

```

```

    mat_id(:,),etype(:,),mat_dam(:,),mat_cpore(:,),crackstat(:,)
character*(*) ,intent(in)::solver
real(8),intent(out)::g_k(0:)
real(8),allocatable::coord(:,),solpoints(:,),surfpnts(:,),solwts(:,)&
    & surfwts(:,),surfFun(:,),eld_tot(:,),eld_inc(:,),gap_tot(:,),gap_elas(:,)&
    & gap_plas(:,),eptot(:,),epinc(:,),gap_inc(:,),solDeriv(:,),solDer(:,),&
    & jac(:,),jac2d(:,),ntot(:,),SolBee(:,),contBee(:,),solDee(:,),&
    & surfDee(:,),solDam(:,),poros(:,),actrot(:,),ke(:,)
integer,allocatable::g(:,),num(:,),numip(:,),nip(:,),nst(:,)
integer::neq,ndim,neltot,nod,nodof,ndof,ntypes,i,iel,ip,imat,ncpore,ndam,&
    & ncrack,optype
real(8)::e,nu,lbound,ks,kn,mu,bonding,det
! assess problem size and make necessary allocations
neq = size(totdisp) - 1
ndim = ubound(g_coord,1) ; neltot = ubound(g_g,2) ; nod = ubound(g_num,1)
ndof = ubound(g_g,1) ; nodof = ndof/nod ; ntypes = maxval(etype)
ncpore = size(mat_cpore) ; ndam = size(mat_dam)
allocate (nip(ntypes),nst(ntypes))
nip(1) = 8
nst(1) = ubound(g_ipDam,1)
allocate (g(ndof),num(nod),numip(nip(1)),coord(nod,ndim),&
    eld_tot(ndof),eld_inc(ndof),eptot(nst(1)),epinc(nst(1)),&
    solpoints(nip(1),ndim),solwts(nip(1)),jac(ndim,ndim),&
    solDee(nst(1),nst(1)),solDam(nst(1)),poros(nst(1)),ke(ndof,ndof))
! allocate element arrays depending on whether extra displacement shapes
! are used: 1 => extra shapes; 0 => standard formulation
if (shapes .eq. 1) then
    allocate (solDer(ndim,nod+ndim),solDeriv(ndim,nod+ndim),&
        solBee(nst(1),ndof+ndim*ndof))
else
    allocate (solDer(ndim,nod),solDeriv(ndim,nod),solBee(nst(1),ndof))
endif
! allocate arrays for contact elements if present
if (ntypes .eq. 2) then
    nip(2) = 4
    nst(2) = nodof
    allocate (gap_tot(nodof),gap_elas(nodof),gap_plas(nodof),gap_inc(nodof),&
        surfwts(nip(2)),surfFun(nod/2),ntot(nodof,ndof),surfpnts(nip(2),ndim),&
        actrot(ndim,ndim),jac2D(ndim-1,ndim-1),contBee(nst(2),ndof),&
        surfDee(nst(2),nst(2)))
endif
! fill array of gaussian integration sampling points
call hex8GaussSample(nip(1),solpoints,solwts)
if (ntypes .eq. 2) then
    call surfSampleGaussPtsLocal(nip(2),surfpnts,surfwts)
endif
! loop elements
! zero any required arrays to be filled
g_k = .0
call hex8GaussSample (nip(1),solpoints,solwts)
do iel=1,neltot
! retrieve element node and gauss pt no.s
num = g_num(:,iel) ; numip = g_numip(:,iel)
! retrieve nodal coordinates
coord = transpose(g_coord(:, num))
! retrieve element steering vector
g=g_g(:,iel)
! retrieve elem nodal disps
eld_tot = totdisp(g)
! default stiffness to zero
ke = .0
! loop gauss pts
if (etype(iel) .eq. 1) then
    material; (props are: e, nu)
    e = prop(1,mat_id(iel)) ; nu = prop(2,mat_id(iel))

```

```

do ip=1,nip(1)
! get shape derivatives wrt local elem csys
call hex8ShapeDer(solpoints(ip,:),solDer)
! calculate jacobian, and its determinant and inverse
jac=matmul(solDer(:,1:nod),coord)
call gen_det(jac,det)
! invert jacobian
call gen_invert(jac, jac)
! calculate shape deriv wrt global ref csys
solDeriv = matmul(jac,solDer)
! calc strain-displacement matrix 'solBee'
call hex8DerivToBee (solDeriv,solBee)
! calc 'solDee' matrix for either linear isotropic or damaged
! default to isotropic elastic
call isoDee(solDee,e,nu)
! check if porous 'dee' req'd
do i=1,ncpore
  if (mat_id(iel) .eq. mat_cpore(i)) then
    poros = g_ipPoros(:,numip(ip))
    call cporeDee(solDee,e,nu,poros(1))
    exit
  endif
end do
! check if damage 'dee' req'd
do i=1,ndam
  if (mat_id(iel) .eq. mat_dam(i)) then
    ncrack = crackstat(numip(ip))
    eptot = matmul(solBee(:,1:ndof),eld_tot)
    SolDam = g_ipDam(:,numip(ip))
    lbound = .9999 ; otype = 0
    call damDee(solDee,e,nu,SolDam,eptot,ncrack,lbound,otype)
    exit
  endif
end do
! fill in relevant portion of element stiffness matrix
ke = ke + &
matmul(matmul(transpose(solBee),solDee),solBee)*det*solwts(ip)
end do
elseif (etype(iel) .eq. 2) then
do ip=1,nip(2)
! calculate shape functions and assemble shape function array
call surfShapeFun(surfpoints(ip,:),surfFun)
! calc jacobian and rotation matrix
call surfJac(surfpoints(ip,:),coord(1:nod/2,:),actrot,jac2D)
! get 2x2 determinant
call gen_det(jac2D,det)
! fill shape function array
call surfShapeFunArr(surfFun,ntot)
! calc 'bee' matrix
contBee = matmul(transpose(actrot),ntot)
! calc gap displacement vector
gap_tot = matmul(contBee,eld_tot)
! calc force-displacement 'dee' matrix (props are: ks, kn, mu)
ks = prop(1,mat_id(iel)) ; kn = prop(2,mat_id(iel))
mu = prop(3,mat_id(iel)) ; bonding = prop(4,mat_id(iel))
if (bonding .eq. 0) then
  call surfDebondDee(gap_tot,gaptol,ks,kn,mu,surfDee)
elseif (bonding .eq. 1) then
  call surfBondDee(ks,kn,surfDee)
else
  call surfBondDee(ks,kn,surfDee)
endif
! calc contribution to stiffness matrix
ke = ke + &
matmul(transpose(contBee),matmul(surfDee,contBee))*det*surfwts(ip)
end do
endif
! assemble into global stiffness
select case (solver)
! case ('itr')

```

```

! iterative solver
! store stiffness matrix
! call formkgnr (g_k,g_ka,g_kb,ke,g)
case ('ban')
! band storage
  call formkv (g_k,ke,g,neq)
case ('sky')
! skyline storage
  call fsparv (g_k,ke,g,g_ka)
end select
end do
return
end subroutine form_K_tan
subroutine bodyloads (g_g,g_num,g_numip,g_coord,prop,totdisp,incdisp,&
  mat_id,mat_dam,mat_cpore,etype,gaptol,g_ipDam,g_ipPoros,&
  crackstat,bdyls,g_ipSigma)
! calculate nodal restoring loads from current displacements for comparison
! with externally applied loads
! user supplied elements (hex8 = 1, contact = 2)
use hex8_xtr ; use elcontactsurf
! user materials
use linelastic ; use damage ; use closedPorous
! matrix and tensor manipulation front-end
use tensor_ops
implicit none
real(8),intent(in)::g_coord(:,,:),prop(:,,:),totdisp(0:),incdisp(0:),&
  gaptol,g_ipDam(:,,:),g_ipPoros(:,,:)
integer,intent(in)::g_g(:,,:),g_num(:,,:),g_numip(:,,:),mat_id(:,,:),etype(:,)&
  mat_dam(:,),mat_cpore(:,),crackstat(:)
real(8),intent(out)::bdyls(0:),g_ipSigma(:,,:)
real(8),allocatable::coord(:,,:),solpoints(:,,:),surfpnts(:,,:),solwts(:,)&
  & surfwts(:,),surfFun(:,),eld_tot(:,),eld_inc(:,),gap_tot(:,),gap_elas(:,)&
  & gap_plas(:,),eptot(:,),epinc(:,),force(:,),f_inc(:,),gap_inc(:,), &
  & solDeriv(:,),solDer(:,),jac(:,),jac2d(:,),ntot(:,),SolBee(:,),&
  & contBee(:,),solDee(:,),surfDee(:,),sigma(:,),sig_inc(:,),ipload(:,)&
  & eload(:,),solDam(:,),poros(:,),actrot(:,):)
integer,allocatable::g(:),num(:),numip(:),nip(:),nst(:)
integer::ndim,neltot,nod,nodof,ndof,ntypes,iel,ip,imat,ncpore,ndam,ncrack,&
  & otype
real(8)::e,nu,lbound,ks,kn,mu,bonding,det,fs,theta,fy
! assess problem size and make necessary allocations
ndim = ubound(g_coord,1) ; neltot = ubound(g_g,2) ; nod = ubound(g_num,1)
ndof = ubound(g_g,1) ; nodof = ndof/nod ; ntypes = maxval(etype)
ncpore = size(mat_cpore) ; ndam = size(mat_dam)
allocate (nip(ntypes),nst(ntypes))
nip(1) = 8
nst(1) = ubound(g_ipSigma,1)
allocate (g(ndof),num(nod),numip(nip(1)),coord(nod,ndim),&
  eld_tot(ndof),eld_inc(ndof),eptot(nst(1)),epinc(nst(1)),&
  solpoints(nip(1),ndim),solwts(nip(1)),jac(ndim,ndim),&
  solDee(nst(1),nst(1)),solDam(nst(1)),poros(nst(1)),sigma(nst(1)),&
  sig_inc(nst(1)),ipload(ndof),eload(ndof))
! allocate element arrays depending on whether extra displacement shapes are
! used: 1 => extra shapes; 0 => standard formulation
if (shapes .eq. 1) then
  allocate (solDer(ndim,nod+ndim),solDeriv(ndim,nod+ndim),&
    solBee(nst(1),ndof+ndim*ndof))
else
  allocate (solDer(ndim,nod),solDeriv(ndim,nod),solBee(nst(1),ndof))
endif
! allocate arrays for contact elements if present
if (ntypes .eq. 2) then
  nip(2) = 4
  nst(2) = nodof
  allocate (gap_tot(nodof),gap_elas(nodof),gap_plas(nodof),actrot(ndim,ndim),&

```

```

surfwts(nip(2)),surfFun(nod/2),ntot(nodof,ndof),surfpoints(nip(2),ndim),&
jac2D(ndim-1,ndim-1),surfDee(nst(2),nst(2)),contBee(nst(2),ndof),&
force(nodof),f_inc(nodof),gap_inc(nodof))
endif
! fill array of gaussian integration sampling points
call hex8GaussSample(nip(1),solpoints,solwts)
if (ntypes .eq. 2) then
  call surfSampleGaussPtsLocal(nip(2),surfpoints,surfwts)
endif
! loop elements
do iel=1,neltot
  num = g_num(:,iel) ; numip = g_numip(:,iel)
  coord = transpose(g_coord(:,num))
  g = g_g(:,iel)
  ipload = .0 ; eload = .0
! loop gauss pts:
  if (etype(iel) .eq. 1) then
! set total displacements for end of increment
    eld_tot = totdisp(g) + incdisp(g)
    e = prop(1,mat_id(iel)) ; nu = prop(2,mat_id(iel))
    do ip=1,nip(1)
      call formHex8Bee (solpoints(ip,:),coord,det,solBee)
      eptot = matmul(solBee(:,1:ndof),eld_tot)
      default to isotropic elastic 'dee'
      call isoDee(solDee,e,nu)
      check if pore is present and couple only if void ratio equal to 1
      do imat=1,ncpore
        if (mat_id(iel) .eq. mat_cpore(imat)) then
          poros = g_ipPoros(:,numip(ip))
          call cporeDee(solDee,e,nu,poros(1))
        endif
      end do
      check if damage 'dee' req'd
      do imat=1,ndam
        if (mat_id(iel) .eq. mat_dam(imat)) then
          ncrack = crackstat(numip(ip))
          SolDam = g_ipDam(:,numip(ip))
          lbound = .9999 ; optype = 0
          call damdee(solDee,e,nu,SolDam,eptot,ncrack,lbound,optype)
        endif
      end do
      calculate total stress
      sigma = matmul(solDee,eptot)
! calculate nodal contribution for current gauss pt
      ipload = matmul(transpose(solBee),sigma)
! add to nodal load vector for element
      eload = eload + ipload*det*solwts(ip)
! store stress in global array
      g_ipSigma(:,numip(ip)) = sigma
    end do
  elseif (etype(iel) .eq. 2) then
! set total displacements for beginning and end of increment
    eld_tot = totdisp(g) + incdisp(g)
    ks = prop(1,mat_id(iel)) ; kn = prop(2,mat_id(iel))
    mu = prop(3,mat_id(iel)) ; bonding = prop(4,mat_id(iel))
! loop gauss pts
    do ip=1,nip(2)
      call surfShapeFun(surfpoints(ip,:),surfFun)
      call surfJac(surfpoints(ip,:),coord(1:nod/2,:),actrot,jac2D)
      call gen_det(jac2D,det)
      call surfShapeFunArr(surfFun,ntot)
      contBee = matmul(transpose(actrot),ntot)
      gap_tot = matmul(contBee,eld_tot)
      if (bonding .eq. 0) then
        call surfDebondDee(gap_tot,gaptol,ks,kn,mu,surfDee)
      elseif (bonding .eq. 1) then
        call surfBondDee(ks,kn,surfDee)
      endif
      calculate total force
      force = matmul(surfDee,gap_tot)
! calculate nodal contribution for current gauss pt
      ipload = matmul(transpose(contBee),force)
! add to nodal load vector for element
      eload = eload + ipload*det*surfwts(ip)
! store some gauss pt results (requires 'dee' for end of increment)
      fs = dsqrt(force(1)**2. + force(2)**2.)
      fy = dabs(mu*f_inc(3))
      1->3: force components
      g_ipSigma(1:3,numip(ip)) = force/det
! 4: sticking/sliding status
      if (gap_tot(3) .gt. gaptol) then
        open
          g_ipSigma(4,numip(ip)) = .0
        elseif ((surfDee(1,1) .eq. ks .and. gap_tot(3) .le. gaptol) &
          .or. bonding .eq. 1.) then
          closed and sticking
          g_ipSigma(4,numip(ip)) = 1.
        elseif (surfDee(1,1) .lt. ks .and. gap_tot(3) .le. gaptol) then
          closed and sliding
          g_ipSigma(4,numip(ip)) = 2.
        endif
! 5: store penetration
        g_ipSigma(5,numip(ip)) = gap_tot(3)
! 6: calculate and store irreversible displacement
        gap_elas = .0
        gap_elas(1) = 1./ks * force(1)
        gap_plas(1) = gap_tot(1) - gap_elas(1)
        if (ndim .gt. 2) then
          gap_elas(2) = 1./ks * force(2)
          gap_plas(2) = gap_tot(2) - gap_elas(2)
        calc resultant
          gap_plas(1) = dsqrt(gap_plas(1)**2. + gap_plas(2)**2.)
        endif
        g_ipSigma(6,numip(ip)) = gap_plas(1)
      enddo
! update global internal nodal loads vector
      bdylds(g) = bdylds(g) + eload
    end do
    bdylds(0) = .0
  return
end subroutine bodyloads
subroutine calc_stress (g_g,g_num,g_numip,g_coord,prop,totdisp,incdisp,time,&
  mat_id,mat_dam,mat_cpore,mat_visc,etype,vscheme,gaptol,&
  g_ipPoros,g_ipDam,crackstat,g_ipEpInit,g_ipEptot,&
  g_ipSigma)
! calculate effective stress based on average stress recovered from porous
! and damaged materials
! user supplied elements (hex8 = 1, contact = 2)
use hex8_xtr ; use elcontactsurf
! user materials
use linelastic ; use damage ; use closedPorous ; use viscoelastic
! matrix and tensor manipulation front-end
use tensor_ops
implicit none
real(8),intent(in)::g_coord(:,,:),prop(:,,:),totdisp(0:),incdisp(0:),time,&
  gaptol,g_ipDam(:,,:),g_ipPoros(:,,:),g_ipEpInit(:,,:)
integer,intent(in)::g_g(:,,:),g_num(:,,:),g_numip(:,,:),mat_id(:,,:),etype(:,)
  mat_dam(:,),mat_cpore(:,),mat_visc(:,),vscheme,crackstat(:,)
real(8),intent(out)::g_ipEptot(:,,:),g_ipSigma(:,,:)
real(8),allocatable::coord(:,,:),solpoints(:,,:),surfpoints(:,,:),solwts(:,)
  & surfwts(:,),surfFun(:,),eld_tot(:,),eld_inc(:,),gap_tot(:,),gap_elas(:,)
  & gap_plas(:,),eptot(:,),epeff(:,),epini(:,),force(:,),f_inc(:,),gap_inc(:,)

```

```

      & solDeriv(:, :), solDer(:, :), jac(:, :), jac2d(:, :), ntot(:, :), SolBee(:, :), &
      & contBee(:, :), solDee(:, :), surfDee(:, :), sigma(:, :), sig_ini(:, :), solDam(:, :), &
      & poros(:, :), actrot(:, :))
integer, allocatable: g(:), num(:), numip(:), nip(:), nst(:)
integer: ndim, neltot, nod, nodof, ndof, ntypes, iel, ip, imat, ncpore, ndam, nvisc, &
      & ncrack, optype
real(8): e, e_inf, tau, nu, lbound, ks, kn, mu, bonding, det, fs, theta, fy
ndim = ubound(g_coord, 1) ; neltot = ubound(g_g, 2) ; nod = ubound(g_num, 1)
ndof = ubound(g-g, 1) ; nodof = ndof/nod ; ntypes = maxval(etype)
ncpore = size(mat_cpore) ; ndam = size(mat_dam) ; nvisc = size(mat_visc)
allocate (nip(ntypes), nst(ntypes))
nip(1) = 8
nst(1) = ubound(g_ipSigma, 1)
allocate (g(ndof), num(nod), numip(nip(1)), coord(nod, ndim), &
      eld_tot(ndof), eld_inc(ndof), eptot(nst(1)), epeff(nst(1)), epini(nst(1)), &
      solpoints(nip(1), ndim), solvts(nip(1)), jac(ndim, ndim), &
      solDee(nst(1), nst(1)), solDam(nst(1)), poros(nst(1)), &
      sigma(nst(1)), sig_ini(nst(1)))
! allocate element arrays depending on whether extra displacement shapes are
! used: 1 => extra shapes; 0 => standard formulation
if (shapes .eq. 1) then
  allocate (solDer(ndim, nod+ndim), solDeriv(ndim, nod+ndim), &
      solBee(nst(1), ndof+ndim*nodof))
else
  allocate (solDer(ndim, nod), solDeriv(ndim, nod), solBee(nst(1), ndof))
endif
if (ntypes .eq. 2) then
  nip(2) = 4
  nst(2) = nodof
  allocate (gap_tot(nodof), gap_elas(nodof), gap_plas(nodof), actrot(ndim, ndim), &
      surfvts(nip(2)), surfFun(nod/2), ntot(nodof, ndof), surfpoints(nip(2), ndim), &
      jac2D(ndim-1, ndim-1), surfDee(nst(2), nst(2)), contBee(nst(2), ndof), &
      force(nodof), f_inc(nodof), gap_inc(nodof))
endif
call hex8GaussSample(nip(1), solpoints, solvts)
if (ntypes .eq. 2) then
  call surfSampleGaussPtsLocal(nip(2), surfpoints, surfvts)
endif
! loop elements
do iel=1, neltot
  num = g_num(:, iel)
  numip = g_numip(:, iel)
  coord = transpose(g_coord(:, num))
  g = g_g(:, iel)
! set total displacements for end of increment
  eld_tot = totdisp(g) + incdisp(g)
! loop gauss pts:
  if (etype(iel) .eq. 1) then
    e = prop(1, mat_id(iel)) ; nu = prop(2, mat_id(iel))
    if (nvisc .ne. 0) then
      e_inf = prop(3, mat_id(iel))
      tau = prop(4, mat_id(iel))
    endif
    do ip=1, nip(1)
! calculate Bee matrix
      call formHex8Bee (solpoints(ip, :), coord, det, solBee)
! calculate and store total strain
      eptot = matmul(solBee(:, 1:ndof), eld_tot)
      epeff = eptot
      g_ipEptot(:, numip(ip)) = eptot
! default to isotropic elastic 'dee'
      call isoDee(solDee, e, nu)
! check if pore is present
      do imat=1, ncpore
        if (mat_id(iel) .eq. mat_cpore(imat)) then

```

```

      poros = g_ipPoros(:, numip(ip))
      call poreStrain(poros(1), vscheme, epeff)
! calc effective stiffness
      if (poros(1) .eq. 1.) then
        solDee = .0
      !else
      ! call cporeDee(solDee, e, nu, poros(1))
      endif
      exit
    endif
  enddo
! check if damage stress req'd
  do imat=1, ndam
    if (mat_id(iel) .eq. mat_dam(imat)) then
      solDam = g_ipDam(:, numip(ip))
      ncrack = crackstat(numip(ip))
      lbound = 0.9999 ; optype = 1
! ensure that crack planes lose complete stiffness (i.e. can't support
! stress)
      if (ncrack .ne. 0) call damdee(solDee, e, nu, SolDam, eptot, ncrack, &
          & lbound, optype)
      exit
    endif
  enddo
! calculate stress
  sigma = matmul(solDee, epeff)
! calculate relaxed initial stress
  sig_ini = .0
  if (nvisc .ne. 0) then
    do imat=1, nvisc
      if (mat_id(iel) .eq. mat_visc(imat)) then
        epini = g_ipEInit(:, numip(ip))
        call relax_mod (e, e_inf, nu, tau, time, solDee)
        sig_ini = matmul(solDee, epini)
      endif
    enddo
  endif
! store stress in global arrays
  g_ipSigma(:, numip(ip)) = sigma + sig_ini
enddo
elseif (etype(iel) .eq. 2) then
  ks = prop(1, mat_id(iel)) ; kn = prop(2, mat_id(iel))
  mu = prop(3, mat_id(iel)) ; bonding = prop(4, mat_id(iel))
! loop gauss pts
  do ip=1, nip(2)
! calculate contact Bee matrix and gap vector
    call surfBee (surfpoints(ip, :), coord, det, contBee)
    gap_tot = matmul(contBee, eld_tot)
! assign relevant constitutive stiffness matrix
    if (bonding .eq. 0) then
      call surfDebondDee(gap_tot, gaptol, ks, kn, mu, surfDee)
    elseif (bonding .eq. 1) then
      call surfBondDee(ks, kn, surfDee)
    endif
! calculate total force
    force = matmul(surfDee, gap_tot)
! store
    g_ipSigma(1:3, numip(ip)) = force(1:3)
  enddo
endif
enddo
return
end subroutine calc_stress
end module elements

module hex8_xtr
! module of required operations for 8 node hexahedral element with extra shape
! functions
! specify shapes (identifier for extra displacement shapes) as public
public::shapes

```

```

contains
subroutine hex8GaussSample (nip,s,wt)
! calc local 3D coords of gauss pts in nodal natural csys
!-----
! in-----
! nip      = no. gauss pts
!-----
! out-----
! s        = array of sampling pts
! wt       = weighting for each point
!-----
! internal-----
! root3    = temp variable used to prevent recalculation of dsqrt(3.)
!-----
implicit none
! declare variables
integer, intent(in)::nip
real(8),intent(out)::s(:,.),wt(:)
real(8)::root3
root3 = 1./dsqrt(3.)
! select sampling according to no. gauss pts
select case(nip)
case(1)
! centroid only
s(1,1) = .0 ; wt(1) = 8.
case(8)
s(1,1)=-root3;s(1,2)=-root3;s(1,3)=-root3
s(2,1)=-root3;s(2,2)=-root3;s(2,3)= root3
s(3,1)= root3;s(3,2)=-root3;s(3,3)= root3
s(4,1)= root3;s(4,2)=-root3;s(4,3)=-root3
s(5,1)=-root3;s(5,2)= root3;s(5,3)=-root3
s(6,1)=-root3;s(6,2)= root3;s(6,3)= root3
s(7,1)= root3;s(7,2)= root3;s(7,3)= root3
s(8,1)= root3;s(8,2)= root3;s(8,3)=-root3
wt = 1.0
end select
return
end subroutine hex8GaussSample
subroutine hex8ExtrapSample (nip,s,wt)
! calc local 3D coords of node pts in gauss natural csys
!-----
! in-----
! nip      = no. gauss pts
!-----
! out-----
! s        = array of sampling pts
! wt       = weighting for each point
!-----
! internal-----
! root3    = temp variable used to prevent recalculation of dsqrt(3.)
!-----
implicit none
! declare variables
integer, intent(in)::nip
real(8),intent(out)::s(:,.),wt(:)
real(8)::root3
root3 = dsqrt(3.)
! select sampling according to no. gauss pts
select case(nip)
case(1)
! centroid only
s(1,1) = .0 ; wt(1) = 8.
case(8)
s(1,1)=-root3;s(1,2)=-root3;s(1,3)=-root3
s(2,1)=-root3;s(2,2)=-root3;s(2,3)= root3
s(3,1)= root3;s(3,2)=-root3;s(3,3)= root3
s(4,1)= root3;s(4,2)=-root3;s(4,3)=-root3

```

```

s(5,1)=-root3;s(5,2)= root3;s(5,3)=-root3
s(6,1)=-root3;s(6,2)= root3;s(6,3)= root3
s(7,1)= root3;s(7,2)= root3;s(7,3)= root3
s(8,1)= root3;s(8,2)= root3;s(8,3)=-root3
wt = 1.0
end select
return
end subroutine hex8ExtrapSample
subroutine hex8ShapeFun (s,fun)
! calc 3D shape functions for quadrilateral hex8face
!-----
! in-----
! s        = sampling point
!-----
! out-----
! fun      = array of shape functions
!-----
! internal-----
! xi,eta,zeta = natural coordinates
!-----
implicit none
! declare variables
real(8),intent(in)::s(:)
real(8),intent(out)::fun(:)
real(8)::xi,eta,zeta
integer::len
! len = size(fun)
! assign gauss pt local coords
xi = s(1); eta = s(2); zeta = s(3)
! calculate shape functions
select case (len)
case (8)
! standard formulation
shapes = 0
fun=(/.125*(1.-xi)*(1.-eta)*(1.-zeta),.125*(1.-xi)*(1.-eta)*(1.+zeta),&
.125*(1.+xi)*(1.-eta)*(1.-zeta),.125*(1.+xi)*(1.-eta)*(1.+zeta),&
.125*(1.-xi)*(1.+eta)*(1.-zeta),.125*(1.-xi)*(1.+eta)*(1.+zeta),&
.125*(1.+xi)*(1.+eta)*(1.-zeta),.125*(1.+xi)*(1.+eta)*(1.+zeta)/)
case (11)
! extra shape functions (displacement shapes)
shapes = 1
fun=(/.125*(1.-xi)*(1.-eta)*(1.-zeta),.125*(1.-xi)*(1.-eta)*(1.+zeta),&
.125*(1.+xi)*(1.-eta)*(1.-zeta),.125*(1.+xi)*(1.-eta)*(1.+zeta),&
.125*(1.-xi)*(1.+eta)*(1.-zeta),.125*(1.-xi)*(1.+eta)*(1.+zeta),&
.125*(1.+xi)*(1.+eta)*(1.-zeta),.125*(1.+xi)*(1.+eta)*(1.+zeta),&
(1.-xi**2.), (1.-eta**2.), (1.-zeta**2.)/)
end select
return
end subroutine hex8ShapeFun
subroutine hex8ShapeFunArr (fun,ntot)
! assemble total shape function array
! (used in calc of bee matrix and relative displacement vector)
!-----
! in-----
! fun      = vector of shape functions
!-----
! out-----
! ntot     = array of shape functions
!-----
! internal-----
! l        = length of ntot
! i,j      = counters
!-----
implicit none
! declare variables
real(8),intent(in)::fun(:)
real(8),intent(out)::ntot(:,.)
integer::i,j,len
len = ubound(fun,2)
ntot = .0
! fill shape function arrays

```

```

j = 1
do i=1,3
  shapes = 0
  ntot(i,j) = fun(1)
  ntot(i,j+3) = fun(2)
  ntot(i,j+6) = fun(3)
  ntot(i,j+9) = fun(4)
  ntot(i,j+12) = fun(5)
  ntot(i,j+15) = fun(6)
  ntot(i,j+18) = fun(7)
  ntot(i,j+21) = fun(8)
  if (len .eq. 33) then
    shapes = 1
    ntot(i,j+24) = fun(9)
    ntot(i,j+27) = fun(10)
    ntot(i,j+30) = fun(11)
  endif
  j = j + 1
end do
return
end subroutine hex8ShapeFunArr
subroutine hex8ShapeDer (s,der)
! calculate 3d linear shape derivatives at a given point
! in-----
! s = sampling point
! out-----
! der = array of shape derivatives
! internal-----
! xi,eta,zeta = natural coordinates of hex8ace
implicit none
! declare variables
real(8),intent(in)::s(:)
real(8),intent(out)::der(:,,:)
real(8)::xi,eta,zeta
integer::len
! len = ubound(der,2)
! assign gauss pt local coords
xi=s(1); eta=s(2); zeta= s(3)
! default to no extra displacement shapes
shapes = 0
! calculate shape derivatives for first 8 shape functions
! row 1
der(1,1)= -.125*(1.-eta)*(1.-zeta) ; der(1,2)= -.125*(1.-eta)*(1.+zeta)
der(1,3)= .125*(1.-eta)*(1.-zeta) ; der(1,4)= .125*(1.-eta)*(1.-zeta)
der(1,5)= -.125*(1.+eta)*(1.-zeta) ; der(1,6)= -.125*(1.+eta)*(1.+zeta)
der(1,7)= .125*(1.+eta)*(1.-zeta) ; der(1,8)= .125*(1.+eta)*(1.-zeta)
! row 2
der(2,1)= -.125*(1.-xi)*(1.-zeta) ; der(2,2)= -.125*(1.-xi)*(1.+zeta)
der(2,3)= -.125*(1.+xi)*(1.-zeta) ; der(2,4)= -.125*(1.+xi)*(1.-zeta)
der(2,5)= .125*(1.-xi)*(1.-zeta) ; der(2,6)= .125*(1.-xi)*(1.+zeta)
der(2,7)= .125*(1.+xi)*(1.-zeta) ; der(2,8)= .125*(1.+xi)*(1.-zeta)
! row 3
der(3,1)= -.125*(1.-xi)*(1.-eta) ; der(3,2)= .125*(1.-xi)*(1.-eta)
der(3,3)= .125*(1.+xi)*(1.-eta) ; der(3,4)= -.125*(1.+xi)*(1.-eta)
der(3,5)= -.125*(1.-xi)*(1.+eta) ; der(3,6)= .125*(1.-xi)*(1.+eta)
der(3,7)= .125*(1.+xi)*(1.+eta) ; der(3,8)= -.125*(1.+xi)*(1.+eta)
! calculate extra shape derivatives if necessary
if (len .eq. 11) then
  shapes = 1
  ! row 1
  der(1,9) = -2.*xi; der(1,10) = .0; der(1,11) = .0
  ! row 2
  der(2,9) = .0; der(2,10) = -2.*eta; der(2,11) = .0
  ! row 3
  der(3,9) = .0; der(3,10) = .0; der(3,11) = -2.*zeta
endif

```

```

endif
return
end subroutine
subroutine hex8DerivToBee (deriv,bee)
! calc 'bee' (strain-displacement) matrix for a gauss pt from deriv matrix
! in-----
! deriv = matrix of shape derivatives wrt. global csys
! out-----
! bee = bee matrix
! internal-----
! x,y,z = dummy variables used to make notation concise
! k,l,m,n = counters
! len = length of bee (used to differentiate between standard
! and extra shape function (hierarchical) formulations
!
implicit none
! declare variables
real(8),intent(in)::deriv(:,,:)
real(8),intent(out)::bee(:,,:)
real(8)::x,y,z
integer::k,l,m,n,len
! get length of deriv
! 8 => standard
! 11 => hierarchical
len = ubound(deriv,2)
! initialise bee to zero components and then fill non zero entries
bee = .0
do m=1,len
  n=3+m; k=m-1; l=k-1
  x=deriv(1,m); y=deriv(2,m); z=deriv(3,m)
  bee(1,1)=x; bee(4,k)=x; bee(6,n)=x
  bee(2,k)=y; bee(4,1)=y; bee(5,n)=y
  bee(3,n)=z; bee(5,k)=z; bee(6,1)=z
end do
return
end subroutine hex8DerivToBee
subroutine formHex8Bee (s,coord,det,solBee)
! form strain-displacement matrix from coordinates for a single gauss pt
! in-----
! s = sampling pt
! coord = array of nodal coordinates for element
! out-----
! solBee = strain-displacement matrix
! det = determinant of Jacobian matrix
use tensor_ops
real(8),intent(in)::s(:),coord(:,,:)
real(8),intent(out)::det,solBee(:,,:)
real(8),allocatable::solDer(:,,:),jac(:,,:),SolDer(:,,:)
integer::ndim,nod,ndof,nshapes
ndim = ubound(coord,2) ; nod = ubound(coord,1)
ndof = ubound(solBee,2) ; nshapes = ndof/ndim
allocate (solDer(ndim,nshapes),jac(ndim,ndim),solDer(ndim,nshapes))
! calc shape derivatives wrt natural csys
call hex8ShapeDer(s,solDer)
! form jacobian and calculate determinant and inverse
jac=matmul(solDer(:,1:nod),coord)
call gen_det(jac,det)
call gen_invert(jac, jac)
! calc shape derivatives wrt global csys
solDer = matmul(jac,solDer)
! form Bee matrix from components of global csys shape derivative matrix
call hex8DerivToBee(solDer,solBee)
return
end subroutine

```

```

end module hex8_xtr
module linelastic
contains
subroutine isoDee(dee,e,v)
! returns the isotropic elastic dee matrix for given ih
! ih=3,plane strain; =4,axisymmetry or plane strain elastoplasticity
! =6 , three dimensional
implicit none
real(8),intent(in)::e,v; real(8),intent(out)::dee(:, :)
! local variables
real(8)::v1,v2,c,vv; integer :: i,ih; dee=0.0 ; ih = ubound(dee,1)
v1 = 1. - v; c = e/((1.+v)*(1.-2.*v))
select case (ih)
case(3)
  dee(1,1)=v1*c; dee(2,2)=v1*c; dee(1,2)=v*c; dee(2,1)=v*c
  dee(3,3)=.5*c*(1.-2.*v)
case(4)
  dee(1,1)=v1*c; dee(2,2)=v1*c; dee(4,4)=v1*c
  dee(3,3)=.5*c*(1.-2.*v) ; dee(1,2)=v*c; dee(2,1)=v*c
  dee(1,4)=v*c; dee(4,1)=v*c; dee(2,4)=v*c; dee(4,2)=v*c
case(6)
  v2=v/(1.-v); vv=(1.-2.*v)/(1.-v)*.5
  do i=1,3; dee(i,i)=1.;end do; do i=4,6; dee(i,i)=vv; end do
  dee(1,2)=v2; dee(2,1)=v2; dee(1,3)=v2; dee(3,1)=v2
  dee(2,3)=v2; dee(3,2)=v2
  dee = dee*e/(2.*(1.+v)*vv)
case default
  print*, 'wrong size for dee matrix'
end select
return
end subroutine isoDee
end module linelastic

module viscoelastic
! module for viscoelastic materials
! (very simplified at this time)
contains
subroutine relax_mod (e_0,e_inf,nu,tau,t,dee_t)
! returns the isotropic relaxation viscoelastic stress-strain matrix
use linelastic
implicit none
real(8),intent(in)::e_0,e_inf,nu,tau,t
real(8),intent(inout)::dee_t(:, :)
integer::nst
real(8),allocatable::dee_0(:, :),dee_inf(:, :)
nst = ubound(dee_t,1)
allocate (dee_0(nst,nst),dee_inf(nst,nst))
! assume input dee is unrelaxed
dee_0 = dee_t
! stiffness at infinite time
dee_inf = (e_inf/e_0)*dee_0
! at req'd time
dee_t = dee_inf + (dee_0 - dee_inf)*dexp(-1*(t/tau))
return
end subroutine relax_mod
end module

module post_ops
! module of common operations on results of analysis (interim and/or final)
! e.g. get max principal vectors or values etc.
interface write_globArray
! generic name's explicit interface for procedures that write global arrays
! to file

```

```

module procedure write_gmat_r, write_gvec_r, write_gmat_i, write_gvec_i
end interface write_globArray
contains
subroutine write_gmat_r (globmat,rectype,iounit,ndim,vscheme,prefix,suffix)
! write 2-dimensional real valued global array to output file
! e.g. global stress array
! in-----
! globmat = global matrix to be written to output file
! rectype = record type (ascii or binary)
! iounit = file io unit number
! ndim = no. spatial dimensions
! vscheme = storage scheme if voigt form of tensors are being used
! prefix = file name (including last used suffix which may be required to
! change based on value of suffix)
! suffix = file suffix
! internal-----
! fname = name of output file
! l_pfix = length of prefix
! ncomp = no. components for particular entry in global array
! assumes entries stored as columns (hence ncomp = no. rows)
! sgl_arr = single precision array used for output quantities in case binary
! record type (to save file size and post-processing effort)
! dbl_arr = double precision array
! i = loop counter
! tensor operations module req'd for procedure to convert voigt storage to full
! symmetric tensor
use tensor_ops
implicit none
real(8),intent(in)::globmat(:, :)
integer,intent(in)::iounit,ndim,vscheme
character*(*),intent(in)::rectype,prefix,suffix
real(4),allocatable::sgl_arr(:, :)
real(8),allocatable::dbl_arr(:, :)
integer::l_pfix,ncomp,i,fmt_lab
character*24::fname
l_pfix = len(prefix)
ncomp = ubound(globmat,1)
select case (rectype)
case ('ascii')
! set formatting based on no. dimensions and write
103 format (f15.6,tr1,f15.6,tr1,f15.6)
102 format (f15.6,tr1,f15.6)
! find out what type of variables are being stored to determine if a loop
! through the array is req'd
if ((ndim .eq. 2 .and. ncomp .eq. 3) .or. &
(ndim .eq. 3 .and. ncomp .eq. 6)) then
! tensor is stored in upper triangular form so loop through entries and
! write symmetric tensor to output
allocate (dbl_arr(ndim,ndim))
do i=1,ubound(globmat,2)
! form symmetric tensor based on voigt storage scheme
if (vscheme .eq. 12) then
call formsytens_12 (globmat(:,i),dbl_arr)
elseif (vscheme .eq. 23) then
call formsytens_23 (globmat(:,i),dbl_arr)
endif
endif
if (ndim .eq. 3) then
write (iounit,103) dbl_arr
elseif (ndim .eq. 2) then
write (iounit,102) dbl_arr
endif
enddo
else
! non-tensor variable so just write
if (ndim .eq. 3) then

```

```

        write (iounit,103) globmat
    elseif (ndim .eq. 2) then
        write (iounit,102) globmat
    endif
endif
case ('binar')
! set name of file
fname = prefix//suffix
! close file (in case it was ascii before this operation) and re-open as
! binary file
close (iounit)
open (iounit,file=fname,position='append',status='old',form='binary',&
      action='write')
! find out what type of variables are being stored to determine if a loop
! through the array is req'd
if ((ndim .eq. 2 .and. ncomp .eq. 3) .or. &
    (ndim .eq. 3 .and. ncomp .eq. 6)) then
! tensor is stored in upper triangular form so loop through entries and
! write symmetric tensor to output
allocate (sgl_arr(ndim,ndim),dbl_arr(ndim,ndim))
do i=1,ubound(globmat,2)
! form symmetric tensor based on voigt storage scheme
if (vscheme .eq. 12) then
    call formsyntens_12 (globmat(:,i),dbl_arr)
elseif (vscheme .eq. 23) then
    call formsyntens_23 (globmat(:,i),dbl_arr)
endif
sgl_arr = dbl_arr
write (iounit) sgl_arr
enddo
else
allocate (sgl_arr(ncomp,ubound(globmat,2)))
sgl_arr = globmat
write (iounit) sgl_arr
endif
! close and re-open as ascii file, and insert blank line
close (iounit)
open (iounit,file=fname,position='append',status='old',action='write')
write (iounit,*)
end select
return
end subroutine write_gmat_r
subroutine write_gvec_r (globvec,rectype,iounit,prefix,suffix)
! write 1-dimensional real valued global array to output file
! e.g. global vector of pore radii
! in-----
! globvec = global matrix to be written to output file
! rectype = record type (ascii or binary)
! iounit = file io unit number
! fname = file name (including last used suffix which may be required to
! change based on value of suffix)
! suffix = file suffix
! internal-----
! sgl_arr = single precision array used for output quantities in case binary
! record type (to save file size and post-processing effort)
implicit none
real(8),intent(in)::globvec(:)
integer,intent(in)::iounit
character*(*) ,intent(in)::rectype,prefix,suffix
real(4),allocatable::sgl_arr(:)
character*24::fname
select case (rectype)
case ('ascii')
100 format (f15.6)
write (iounit,100) globvec
case ('binar')
! set name of file and allocate single precision output array
fname = prefix//suffix
allocate (sgl_arr(ubound(globmat,1),ubound(globmat,2)))
! close file (in case it was ascii before this operation) and re-open as
! binary file
close (iounit)
open (iounit,file=fname,position='append',status='old',form='binary',&
      action='write')
sgl_arr = globmat
write (iounit) sgl_arr
! close and re-open as ascii file, and insert blank line
close (iounit)
open (iounit,file=fname,position='append',status='old',action='write')
write (iounit,*)
end select
return
end subroutine write_gvec_i (globvec,rectype,iounit,prefix,suffix)
! write 1-dimensional integer valued global array to output file
! e.g. global vector of element type no.s
! in-----
! globvec = global vector to be written to output file

```

```

! set name of file and allocate single precision output array
fname = prefix//suffix
allocate (sgl_arr(size(globvec)))
! close file (in case it was ascii before this operation) and re-open as
! binary file
close (iounit)
open (iounit,file=fname,position='append',status='old',form='binary',&
      action='write')
sgl_arr = globvec
write (iounit) sgl_arr
! close and re-open as ascii file, and insert blank line
close (iounit)
open (iounit,file=fname,position='append',status='old',action='write')
write (iounit,*)
end select
return
end subroutine write_gvec_r
subroutine write_gmat_i (globmat,rectype,iounit,prefix,suffix)
! write 2-dimensional integer valued global array to output file
! e.g. global array of element node no.s
! in-----
! globmat = global matrix to be written to output file
! rectype = record type (ascii or binary)
! iounit = file io unit number
! fname = file name (including last used suffix which may be required to
! change based on value of suffix)
! suffix = file suffix
! l_pfix = length of file prefix
! internal-----
! sgl_arr = single precision array used for output quantities in case binary
! record type (to save file size and post-processing effort)
implicit none
integer,intent(in)::globmat(:,)
integer,intent(in)::iounit
character*(*) ,intent(in)::rectype,prefix,suffix
integer,allocatable::sgl_arr(:,)
character*24::fname
select case (rectype)
case ('ascii')
100 format (i,10)
write (iounit,100) globmat
case ('binar')
! set name of file and allocate single precision output array
fname = prefix//suffix
allocate (sgl_arr(ubound(globmat,1),ubound(globmat,2)))
! close file (in case it was ascii before this operation) and re-open as
! binary file
close (iounit)
open (iounit,file=fname,position='append',status='old',form='binary',&
      action='write')
sgl_arr = globmat
write (iounit) sgl_arr
! close and re-open as ascii file, and insert blank line
close (iounit)
open (iounit,file=fname,position='append',status='old',action='write')
write (iounit,*)
end select
return
end subroutine write_gmat_i
subroutine write_gvec_i (globvec,rectype,iounit,prefix,suffix)
! write 1-dimensional integer valued global array to output file
! e.g. global vector of element type no.s
! in-----
! globvec = global vector to be written to output file

```



```

! rectype      = record type (ascii or binary)
! iounit       = file io unit number
! fname        = file name (including last used suffix which may be required to
!               change based on value of suffix)
! suffix       = file suffix
! l_prefix     = length of file prefix
! internal-----
! sgl_arr      = single precision array used for output quantities in case binary
!               record type (to save file size and post-processing effort)

implicit none
integer,intent(in)::globvec(:)
integer,intent(in)::iounit
character*(*) ,intent(in)::rectype,prefix,suffix
integer,allocatable::sgl_arr(:)
character*24::fname

select case (rectype)
case ('ascii')
  100 format (i)
  write (iounit,100) globvec
case ('binar')
! set name of file and allocate single precision output array
  fname = prefix//suffix
  allocate (sgl_arr(size(globvec)))
! close file (in case it was ascii before this operation) and re-open as
! binary file
  close (iounit)
  open (iounit,file=fname,position='append',status='old',form='binary',&
        action='write')
  sgl_arr = globvec
  write (iounit) sgl_arr
! close and re-open as ascii file, and insert blank line
  close (iounit)
  open (iounit,file=fname,position='append',status='old',action='write')
  write (iounit,*)
end select
return
end subroutine write_gvec_i

subroutine rw_vtkCells (g_num,iunit,ounit,rectype,prefix,suffix)
! read and write vtkCells array from and to vtk unstructured_grid file
! vtkCells corresponds to global array of element node numbers in finite
! element program
! in-----
! iunit      = file unit no. for input, i.e. reading
! ounit      = file unit no. for output, i.e. writing
! rectype    = record type (binary or ascii)
! prefix     = filename prefix
! suffix     = filename suffix
! out-----
! g_num      = global array of element node numbers
! internal-----
! nels       = no. elements
! iel        = loop counter for pass through elements
! cell_pts   = dummy variable for no. nodes for a given element (this is used
!               by vtk when reading unstructured grid data but is not required
!               for this code since the g_num array is allocated based on the
!               element type with the highest no. nodes per element; unused
!               are filled as zero, which can be achieved by zeroing the array
!               before calling this subroutine)
! fname      = string variable for filename

implicit none
integer,intent(in)::iunit,ounit
integer,intent(out)::g_num(:, :)
character*(*) ,intent(in)::rectype,prefix,suffix
integer::nels,iel,cell_pts
character*24::fname

```

```

! query g_num size for no. elements
nels = ubound(g_num,2)
select case (rectype)
case ('ascii')
! set default format for 8 node brick; i.e. 9 integers (1 for cell_pts
! dummy variable (=8 for 8 node brick) and the 8 node no.s)
  100 format (i,tr1,i,tr1,i,tr1,i,tr1,i,tr1,i,tr1,i,tr1,i)
! loop through elements and read, fill g_num, and write vtkCells
  do iel=1,nels
    read (iunit,*) cell_pts,g_num(:,iel)
    write (ounit,100) cell_pts, g_num(:,iel)
  end do
case ('binar')
! set name of file
  fname = prefix//suffix
! close file (in case it was ascii before this operation) and re-open as
! binary file
  close (ounit)
  open (ounit,file=fname,position='append',form='binary',status='old',&
        action='write')
! loop through elements and read, fill g_num, and write vtkCells
  do iel=1,nels
    read (iunit,*) cell_pts,g_num(:,iel)
    write (ounit) cell_pts, g_num(:,iel)
  end do
! close and re-open as ascii file, and insert blank line
  close (ounit)
  open (ounit,file=fname,position='append',status='old',action='write')
  write (ounit,*)
end select
return
end subroutine rw_vtkCells

subroutine numwidth (int,w)
! subroutine to return width for formatting an integer to remove trailing space
! when labelling variable names in output (only works to 9.9e8 at the moment)
integer,intent(in)::int
integer,intent(out)::w
if ((int .gt. 9) .and. (int .le. 99)) then
  w = 2
elseif ((int .gt. 99) .and. (int .le. 999)) then
  w = 3
elseif ((int .gt. 999) .and. (int .le. 9999)) then
  w = 4
elseif ((int .gt. 9999) .and. (int .le. 99999)) then
  w = 5
elseif ((int .gt. 99999) .and. (int .le. 999999)) then
  w = 6
elseif ((int .gt. 999999) .and. (int .le. 9999999)) then
  w = 7
elseif ((int .gt. 9999999) .and. (int .le. 99999999)) then
  w = 8
elseif ((int .gt. 99999999) .and. (int .le. 999999999)) then
  w = 9
else
  w = 1
endif
return
end subroutine numwidth

subroutine stressvol (g_ipStress,vscheme,g_ipVol,g_ipMat,sMin,sMax,sv_tab)
! returns the volumes of individual materials experiencing prescribed ranges of
! maximum principal stress
! in-----
! g_ipStress = global array of gauss pt stress tensors (voigt form)
! vscheme    = storage scheme for voigt form
!             '12' => ...,12,23,31
!             '23' => ...,23,31,12
! g_ipVol    = global array of gauss pt volumes
! g_ipMat    = global array of gauss pt material id's

```

```

! sMin      = minimum stress requested by user
! sMax      = maximum stress requested by user
!
! out-----
! sv_tab    = array representing tabular data for stress-volume data
!
! internal----
! prS      = array of principal stresses for a given point
! min,max  = min and max of a given interval under consideration
! nst      = no. stress components
! nip,ip   = no. gauss pts and gauss pt counter for loops
! range    = no. stress ranges
! i        = loop counter
! row      = row counter for table
!
! format of output array
! | mat id | stress range | volume | misc |
!
! req'd libraries---
! use tensor_ops
!
! declare variables
! implicit none
! real(8),intent(in)::g_ipStress(:,:),g_ipVol(:,)
! real(8),intent(out)::sv_tab(:,)
! real(8),allocatable::prS(:)
! real(8)::min,max
! integer,intent(in)::vscheme,sMin,sMax,g_ipMat(:,)
! integer::nst,dim,nip,ip,range,i,nmat,row
!
! find dimensionality, etc.
! nst = ubound(g_ipStress,1) ; nip = size(g_ipVol)
! select case (nst)
!   case (6)
!     dim = 3
!   case (3)
!     dim = 2
! end select
! allocate(prS(dim))
!
! set stress range
! range = sMax - sMin
!
! no. materials (extra interval (i.e. '+ 1') for values above sMax)
! nmat = ubound(sv_tab,1) ; nmat = nmat/(range + 1)
!
! fill 'mat id' column
! row = 1
! do i=1,nmat
!   sv_tab(row:row+range,1) = i
!   row = row + range + 1
! end do
!
! loop gauss pts
! do ip=1,nip
!   call vprival (g_ipStress(:,ip),vscheme,prS)
!   min = sMin ; max = min + 1
!   do i=1,range+1
!     !update row counter
!     row = (range+1)*g_ipMat(ip) - (range+1) + i
!     sv_tab(row,2) = sMax - (range+1) + i
!     if (prS(1) .ge. min .and. prS(1) .lt. max) then
!       stress within interval
!       sv_tab(row,3) = sv_tab(row,3) + g_ipVol(ip)
!       exit
!     elseif (prS(1) .gt. sMax) then
!       stress above max desired stress
!       row = (range+1)*g_ipMat(ip)
!       sv_tab(row,3) = sv_tab(row,3) + g_ipVol(ip)
!       exit
!     endif
!     min = min + 1. ; max = max + 1.
!   end do
! end do
! return

```

```

end subroutine stressvol
end module post_ops

module pre_ops
!
! module for some common operations in generating the finite element model
! prior to solution stage of analysis
!
! contains
!
! subroutine formnf(nf)
! reform array of nodal freedoms for entire mesh
!
! in/out-----
! nf = array of nodal freedoms
!
! internal-----
! i,j,m = counters
!
! implicit none
! integer,intent(in out)::nf(:,)
! integer:: i,j,m
!
! m=0
! do j=1,ubound(nf,2)
!   do i=1,ubound(nf,1)
!     ! everytime a nonzero value is found (i.e. '1' which implies a 'free' dof)
!     ! the dummy index m is incremented and substituted for the current array
!     ! position
!     if(nf(i,j)/=0) then
!       m=m+1; nf(i,j)=m
!     end if
!   end do
! end do
! return
! end subroutine formnf
!
! subroutine num_to_g(num,nf,g)
! Node to freedom number conversion
! finds the steering vector from nodal connectivity vector and nodal freedoms
!
! in-----
! num = nodal connectivity vector for element
! nf = nodal freedoms vector
!
! out-----
! g = element steering vector
!
! internal-----
! nod = no. of nodes
! nodof = no. dof per node
! i,k = counters
!
! implicit none
! integer,intent(in)::num(:,),nf(:,) ; integer,intent(out):: g(:,)
! integer::i,k,nod,nodof
!
! find no. nodes and no. dof/node
! nod=ubound(num,1) ; nodof=ubound(nf,1)
!
! do i=1,nod
!   k = i*nodof
!   g(k-nodof+1:k) = nf( : , num(i) )
! end do
! return
! end subroutine num_to_g
!
! subroutine node_connect (g_num,g_nconn)
! Returns array of elements connected to each node
!
! in-----
! g_num = global array of element node numbers
!
! out-----
! g_nconn = global array of node-element connectivities
!
! internal-----

```

```

! nels      = no. elements
! nn        = no. nodes
! nod       = no. nodes per element
! num       = array of current element node numbers
! indexor   = array used to index to next unused position in connectivity array
implicit none
integer,intent(in)::g_num(:,:)
integer,intent(out)::g_nconn(:,:)
integer::nels,nn,maxshare,nod,iel,inod
integer,allocatable::num(:),indexor(:)
! get size of arrays and make allocations
nels = ubound(g_num,2)
nn = ubound(g_nconn,2)
maxshare = ubound(g_nconn,1)
nod = ubound(g_num,1)
allocate (num(nod),indexor(nn))
! initialise indexing array and start main loop
indexor = 0 ; g_nconn = 0
do iel=1,nels
  num = g_num(:,iel)
  do inod=1,nod
    indexor(num(inod)) = indexor(num(inod)) + 1
    g_nconn(indexor(num(inod)),num(inod)) = iel
  enddo
enddo
return
end subroutine node_connect

subroutine parse_elconnect (g_num,g_nconn,max_elconn)
! Parse element connectivity to find max no. elements connected to another
!-----
! g_num      = global array of element node numbers
! g_nconn    = global array of node-element connectivities
!-----
! out-----
! max_elconn = maximum element-element connectivity in mesh
!-----
! internal-----
! nels      = no. elements
! nn        = no. nodes
! nod       = no. nodes per element
! max_nconn = max no. elements shared by a node
! num       = array of current element node numbers
! elnum     = array of connectivities with duplicate entries
! k         = index indicating no. non-repeated array entries for an element
! inc       = variable to assign whether non-repeated index is incremented
! iel,i,j   = loop counters
implicit none
integer,intent(in)::g_num(:,:),g_nconn(:,:)
integer,intent(out)::max_elconn
integer::nels,nn,nod,max_nconn,iel,inod,i,j,k
integer,allocatable::num(:),elnum(:)
logical::inc
! get size of arrays and make allocations
nels = ubound(g_num,2)
nn = ubound(g_nconn,2)
nod = ubound(g_num,1)
max_nconn = ubound(g_nconn,1)
allocate (num(nod),elnum(max_nconn*nod))
! start main loop
do iel=1,nels
  num = g_num(:,iel)
  k=1
  do inod=1,nod
    elnum(k:k+max_nconn-1) = g_nconn(:,num(inod))
    k = k + max_nconn
  enddo
  k = 1
! loop from entry 2 -> end and increment k for non-repeated entries

```

```

do i=2,max_nconn*nod
! default 'inc' to true
inc = .true.
! loop through preceding entries and check if current entry already present
do j=1,i-1
  if (elnum(i) .eq. elnum(j)) then
! repeated entry so exit and do not increment
inc = .false.
exit
endif
enddo
if (inc .eq. .true. .and. elnum(i) .ne. 0 .and. elnum(i) .ne. iel) k = k+1
enddo
! check if k exceeds existing maximum
if (k .gt. max_elconn) max_elconn = k
enddo
return
end subroutine parse_elconnect

subroutine el_connect (g_num,g_nconn,g_elconn)
! Returns array of elements connected to each element
!-----
! in-----
! g_num      = global array of element node numbers
! g_nconn    = global array of node-element connectivities
!-----
! out-----
! g_elconn   = global array of element-element connectivities
!-----
! internal-----
! nels      = no. elements
! nn        = no. nodes
! nod       = no. nodes per element
! num       = array of current element node numbers
! elnum     = array of connectivities with duplicate entries
! connector  = array of connectivities without duplicate entries
! k         = index indicating no. non-repeated array entries for an element
! store     = variable to assign whether non-repeated entry is stored
! iel,i,j   = loop counters
implicit none
integer,intent(in)::g_num(:,:),g_nconn(:,:)
integer,intent(out)::g_elconn(:,:)
integer::nels,nn,nod,max_nconn,max_elconn,iel,inod,i,j,k
integer,allocatable::num(:),elnum(:),connector(:)
logical::store
! get size of arrays and make allocations
nels = ubound(g_num,2)
nn = ubound(g_nconn,2)
nod = ubound(g_num,1)
max_nconn = ubound(g_nconn,1)
max_elconn = ubound(g_elconn,1)
allocate (num(nod),elnum(max_nconn*nod),connector(max_elconn))
! start main loop
do iel=1,nels
  num = g_num(:,iel)
  k = 1
  elnum = 0; connector = 0
  do inod=1,nod
    elnum(k:k+max_nconn-1) = g_nconn(:,num(inod))
    k = k + max_nconn
  enddo
  k = 1
! loop from entry 2 -> end and store any non-repeated entries
do i=1,max_nconn*nod
! default 'inc' to true
store = .true.
! loop through preceding entries and check if current entry already present
do j=1,i-1
  if (elnum(i) .eq. elnum(j)) then
! repeated entry so exit and do not increment
store = .false.
exit
endif
enddo
enddo

```

```

if (store .eq. .true. .and. elnum(i) .ne. 0 .and. elnum(i) .ne. iel) then
! store entry and increment index counter
connector(k) = elnum(i)
k = k + 1
endif
enddo
! store in global array
g_elconn(:,iel) = connector
enddo
return
end subroutine el_connect

subroutine parse_ipbucket1(g_elconn,g_numip,g_ipcoord,r_bkt,max_ipbkt)
! Parse element connectivity to find max no. gauss pts within a specified
! radius of each gauss pt - stops at one level of connectivity element search
! tree
! in-----
! g_elconn = global array of element-element connectivities
! g_numip = global array of element gauss pt numbers
! g_ipcoord = global array of gauss pt coordinates
! r_bkt = radius of bucket
! out-----
! max_ipbkt = maximum no. gauss pts in a bucket
! internal-----
! nels = no. elements
! nip_el = no. gauss pts per element
! ndim = no. of coordinate dimensions
! numip = array of current element gauss pt numbers
! elconn = array of current element connectivity bucket (i.e. surrounding
! elements
! k = index indicating no. non-repeated array entries for an element
! inc = variable to assign whether non-repeated index is incremented
! iel,ip = outer loop counters
! i,j = inner loop counters
! iel_t = target element id
! s_coord = coords of source gauss pt (i.e. centre of bucket)
! t_coord = coords of target gauss pt (i.e. gauss pt that is being checked
! for intersection with bucket around source pt
! dvec = distance vector
! dnorm = euclidean distance norm between s_coord and t_coord

implicit none
integer,intent(in)::g_elconn(:,,:),g_numip(:,:)
real(8),intent(in)::g_ipcoord(:,,:),r_bkt
integer,intent(out)::max_ipbkt
integer::nels,nip_el,max_elconn,ndim,k,iel,iel_t,ip,i,j
real(8)::dnorm
real(8),allocatable::s_coord(:),t_coord(:),dvec(:)
integer,allocatable::numip(:),elconn(:)
logical::inc

! get size of arrays and make allocations
nels = ubound(g_elconn,2)
max_elconn = ubound(g_elconn,1)
nip_el = ubound(g_numip,1)
ndim = ubound(g_ipcoord,1)
allocate (s_coord(ndim),t_coord(ndim),dvec(ndim),numip(nip_el),&
elconn(max_elconn))

! main loop
do iel=1,nels
elconn = g_elconn(:,iel)
! loop gauss pts for bucket centres
do ip=1,nip_el
! re-zero k and retrieve source coordinate if ip no. not zero
k = 0
if (g_numip(ip,iel) .ne. 0) then
s_coord = g_ipcoord(:,g_numip(ip,iel))
! check gauss pts in current element for intersection with bucket
do i=1,nip_el
! calc distance if current gauss pt is not the source pt and not
! a zero entry

```

```

if (i .ne. ip .and. g_numip(i,iel) .ne. 0) then
t_coord = g_ipcoord(:,g_numip(i,iel))
dvec = t_coord - s_coord
dnorm = dot_product(dvec,dvec)**0.5
if (dnorm .le. r_bkt) k = k + 1
endif
enddo
! loop surrounding elements for targets
do i=1,max_elconn
iel_t = elconn(i)
! only operate on non-zero entries of connectivity array
if (iel_t .ne. 0) then
numip = g_numip(:,iel_t)
! loop gauss pts within current target element
do j=1,nip_el
! elements with less nodes than highest order element in mesh will
! have zero entries so check for this before calculating distance
if (numip(j) .ne. 0) then
t_coord = g_ipcoord(:,numip(j))
dvec = t_coord - s_coord
dnorm = dot_product(dvec,dvec)**0.5
if (dnorm .le. r_bkt) k = k + 1
endif
endif
enddo
endif
enddo
if (k .gt. max_ipbkt) max_ipbkt = k
endif
enddo
enddo
return
end subroutine parse_ipbucket1

subroutine ipbucket1(g_elconn,g_numip,g_ipcoord,r_bkt,g_ipbkt1)
! Generate bucket of gauss pts within a specified radius of each gauss pt in
! mesh
! in-----
! g_elconn = global array of element-element connectivities
! g_numip = global array of element gauss pt numbers
! g_ipcoord = global array of gauss pt coordinates
! r_bkt = radius of bucket
! out-----
! g_ipbkt1 = global array of level 1 gauss pt buckets
! internal-----
! nels = no. elements
! nip_el = no. gauss pts per element
! ndim = no. of coordinate dimensions
! max_ipbkt = maximum no. gauss pts in a bucket
! numip = array of current element gauss pt numbers
! elconn = array of current element connectivity bucket (i.e. surrounding
! elements
! ipbkt = array of gauss pts in current bucket
! k = index indicating no. non-repeated array entries for an element
! inc = variable to assign whether non-repeated index is incremented
! iel,ip = outer loop counters
! i,j = inner loop counters
! iel_t = target element id
! s_coord = coords of source gauss pt (i.e. centre of bucket)
! t_coord = coords of target gauss pt (i.e. gauss pt that is being checked
! for intersection with bucket around source pt
! dvec = distance vector
! dnorm = euclidean distance norm between s_coord and t_coord

implicit none
integer,intent(in)::g_elconn(:,,:),g_numip(:,:)
real(8),intent(in)::g_ipcoord(:,,:),r_bkt
integer,intent(out)::g_ipbkt1(:,:)
integer::nels,nip_el,niptot,max_elconn,max_ipbkt,ndim,k,iel,iel_t,ip,i,j
real(8)::dnorm
real(8),allocatable::s_coord(:),t_coord(:),dvec(:)
integer,allocatable::numip(:),elconn(:),ipbkt(:)
logical::inc

```

```

! get size of arrays and make allocations
nels = ubound(g_elconn,2)
max_elconn = ubound(g_elconn,1)
nip_el = ubound(g_numip,1)
ndim = ubound(g_ipcoord,1)
niptot = ubound(g_ipbkt1,2)
max_ipbkt = ubound(g_ipbkt1,1)
allocate (s_coord(ndim),t_coord(ndim),dvec(ndim),numip(nip_el),&
          elconn(max_elconn),ipbkt(max_ipbkt))

! default to zero
g_ipbkt1 = 0

! main loop
do iel=1,nels
  elconn = g_elconn(:,iel)
! loop gauss pts for bucket centres
do ip=1,nip_el
! re-zero ipbkt and k, and retrieve source coordinate
  ipbkt = 0
  k = 0
  if (g_numip(ip,iel) .ne. 0) then
    s_coord = g_ipcoord(:,g_numip(ip,iel))
! check gauss pts in current element for intersection with bucket
    do i=1,nip_el
! calc distance if current gauss pt is not the source pt
      if (i .ne. ip .and. g_numip(i,iel) .ne. 0) then
        t_coord = g_ipcoord(:,g_numip(i,iel))
        dvec = t_coord - s_coord
        dnorm = dot_product(dvec,dvec)**0.5
        if (dnorm .le. r_bkt) then
          k = k + 1
          ipbkt(k) = g_numip(i,iel)
        endif
      endif
    enddo
! loop surrounding elements for targets
do i=1,max_elconn
  iel_t = elconn(i)
! only operate on non-zero entries of connectivity array
  if (iel_t .ne. 0) then
    numip = g_numip(:,iel_t)
! loop gauss pts within current target element
    do j=1,nip_el
      if (numip(j) .ne. 0) then
        t_coord = g_ipcoord(:,numip(j))
        dvec = t_coord - s_coord
        dnorm = dot_product(dvec,dvec)**0.5
        if (dnorm .le. r_bkt) then
          k = k + 1
          ipbkt(k) = g_numip(j,iel_t)
        endif
      endif
    enddo
! store in global array
    g_ipbkt1(:,g_numip(ip,iel)) = ipbkt
  endif
enddo
enddo
return
end subroutine ipbucket1

subroutine vtkcell12fem90 (type_id, type_string, nod, nip)
! returns the fem90 element type string variable and no. of nodes and gauss pts
! per element from the vtk cell-type id
!
! in-----
! type_id      = vtk cell-type id
!
! out-----
! type_string  = element type name stored as string
! nod         = no. of nodes per element

```

```

! nip      = no. of gauss pts
!
! implicit none
integer,intent(in)::type_id ; character*(*) ,intent(out)::type_string
integer,intent(out)::nod, nip
!
! generate req'd values from type_id
select case (type_id)
case (12)
  type_string = 'hexahedron'
  nod = 8
  nip = 8
case (0)
  print *, " Element type not supported at this time"
  return
end select
return
end subroutine
end module pre_ops

module solution
contains
! module of common operations required for solution of finite element
! linear systems of equations (e.g. Newton-Raphson iteration and line search)
subroutine line_search (apl_inc,lastresid,resid,rlnorm,totdisp,incdisp,&
  g_g,g_num,g_numip,g_coord,prop,mat_id,mat_dam,mat_cpore,etype,&
  gaptol,g_ipDam,g_ipPoros,crackstat,lsearch,iounit)

use elements
implicit none
real(8),intent(in)::apl_inc(0:),lastresid(0:),totdisp(0:),g_coord(:,:),&
  prop(:,:),gaptol,g_ipDam(:,:),g_ipPoros(:,:)
integer,intent(in)::g_g(:,:),g_num(:,:),g_numip(:,:),etype(:),mat_id(:),&
  mat_dam(:),mat_cpore(:),crackstat(:),iounit
real(8),intent(inout)::incdisp(0:),resid(0:),rlnorm
real(8),intent(out)::lsearch
real(8),allocatable::ls_incdisp(:),ls_resid(:),resid_min(:),bdyls(:),&
  & g_ipSigma(:,:)
real(8)::ls_param(0:5),ls_norm(0:5),ls_deriv(0:2),deriv_0,deriv_1,ls_rlnorm,&
  & test(0:1),crit,deriv_min,param_min,min_norm,inc,ascend_pt,&
  & descend_pt,ascend_norm,descend_norm
integer::iter,totdof,nst,niptot,i
logical::intersect

totdof = size(totdisp) - 1
nst = ubound(g_ipDam,1)
niptot = ubound(g_ipDam,2)
allocate (ls_incdisp(0:totdof),ls_resid(0:totdof),resid_min(0:totdof),&
  bdyls(0:totdof),g_ipSigma(nst,niptot))

write (iounit, '(f12.6,/)' ) rlnorm
ls_param(0) = 0.999
ls_param(1) = 1.0
ls_param(2) = 0.5
ls_param(3) = 0.501
ls_param(4) = 0.0
ls_param(5) = 0.001
intersect = .false.

! calculate derivative for first position
! calculate the residual "work" norm
do iter=1,3
  do i=0,5
    if (ls_param(i) .ne. 1.) then
      ls_incdisp = ls_param(i)*incdisp ; bdyls = .0 ; g_ipSigma = .0
      call bodyloads (g_g,g_num,g_numip,g_coord,prop,totdisp,ls_incdisp,&
        & mat_id,mat_dam,mat_cpore,etype,gaptol,g_ipDam,&
        & g_ipPoros,crackstat,bdyls,g_ipSigma)
      ls_resid = apl_inc - bdyls
    endif
  enddo
enddo

```

```

else
  ls_resid = resid
endif
ls_norm(i) = dabs(dot_product(incdisp,ls_resid))
enddo
! derivative ~ slope of norm function over the interval
ls_deriv(0) = (ls_norm(1)-ls_norm(0))/(ls_param(1)-ls_param(0))
ls_deriv(1) = (ls_norm(3)-ls_norm(2))/(ls_param(3)-ls_param(2))
ls_deriv(2) = (ls_norm(5)-ls_norm(4))/(ls_param(5)-ls_param(4))
! store initial minimum
if (min(dabs(ls_deriv(0)),dabs(ls_deriv(1)),dabs(ls_deriv(1))) &
.eq. dabs(ls_deriv(0))) then
  deriv_min = ls_deriv(0)
  param_min = ls_param(0) + 0.5*(ls_param(1) - ls_param(0))
elseif (min(dabs(ls_deriv(0)),dabs(ls_deriv(1)),dabs(ls_deriv(1))) &
.eq. dabs(ls_deriv(1))) then
  deriv_min = ls_deriv(1)
  param_min = ls_param(2) + 0.5*(ls_param(3) - ls_param(2))
elseif (min(dabs(ls_deriv(0)),dabs(ls_deriv(1)),dabs(ls_deriv(1))) &
.eq. dabs(ls_deriv(2))) then
  deriv_min = ls_deriv(2)
  param_min = ls_param(4) + 0.5*(ls_param(5) - ls_param(4))
endif
! store derivative at zero for possible use in acceptance criterion
if (ls_param(0) .eq. .0) then
  deriv_0 = ls_deriv(0)
  crit = dabs(0.3*deriv_0)
endif
! write data to file
do i=0,5
  write (iounit,'(f12.6,tr1,e12.6)') ls_param(i), ls_norm(i)
enddo
! check derivatives to see between which two the minimum lies and calculate
! intersection if req'd
if (ls_deriv(0) .gt. 0 .and. ls_deriv(1) .lt. .0) then
! first store data for last used ascending and descending pts
  ascend_pt = ls_param(0) ; ascend_norm = ls_norm(0)
  descend_pt = ls_param(3) ; descend_norm = ls_norm(3)
  ls_param(0) = descend_pt + &
  descend_norm*(ascend_pt - descend_pt)/(ascend_norm + descend_norm)
  intersect = .true.
  exit
elseif (ls_deriv(1) .gt. 0 .and. ls_deriv(2) .lt. .0) then
  ascend_pt = ls_param(2) ; ascend_norm = ls_norm(2)
  descend_pt = ls_param(5) ; descend_norm = ls_norm(5)
  ls_param(0) = descend_pt + &
  descend_norm*(ascend_pt - descend_pt)/(ascend_norm + descend_norm)
  intersect = .true.
  exit
elseif (ls_deriv(0) .lt. .0 .and. ls_deriv(1) .lt. .0 .and. ls_deriv(2) &
& .lt. .0) then
  ls_param = ls_param + 1.
elseif (ls_deriv(0) .gt. .0 .and. ls_deriv(1) .gt. .0 .and. ls_deriv(2) &
& .gt. .0) then
  ls_param = ls_param - 1.
endif
enddo
! iterations to find a suitable interval with a leftmost descending pt and a
! rightmost ascending pt
do iter=1,5
! exit condition when an intersection pt has been found
if (intersect) exit
! calc new interval of line-search parameter for second position by solving
! for the x-axis intercept of slope-lines through the two trial pts
ls_param(2) = -1*ls_norm(0)/ls_deriv(0) + ls_param(0)
inc = 0.001*dabs(ls_param(2) - ls_param(1))
if (inc .gt. 0.001) inc = 0.001
ls_param(3) = ls_param(2) + inc

```

```

! calculate derivative for second position
do i=2,3
  if (ls_param(i) .ne. .0) then
    ls_incdisp = ls_param(i)*incdisp ; bdylds = .0 ; g_ipSigma = .0
    call bodyloads (g_g,g_num,g_numip,g_coord,prop,totdisp,ls_incdisp,&
      mat_id,mat_dam,mat_cpore,etype,gaptol,g_ipDam,g_ipPoros,&
      crackstat,bdylds,g_ipSigma)
    ls_resid = apl_inc - bdylds
  else
    ls_resid = lastresid
  endif
  ls_norm(i) = dabs(dot_product(incdisp,ls_resid))
  write (iounit,'(f12.6,tr1,e12.6)') ls_param(i), ls_norm(i)
enddo
ls_deriv(1) = (ls_norm(3)-ls_norm(2))/(ls_param(3)-ls_param(2))
! check if less than minimum derivative found previously
if (dabs(ls_deriv(1)) .lt. dabs(deriv_min)) then
  deriv_min = ls_deriv(1)
  if (deriv_min .lt. .0) then
    param_min = ls_param(3)
  else
    param_min = ls_param(2)
  endif
endif
! calculate new increment for derivative calcs based on gap between two
! intervals for current iteration
inc = 0.001*dabs(ls_param(2) - ls_param(1))
if (inc .gt. 0.001) inc = 0.001
! calculate ls_param @ intersection or else update first search interval
if (ls_deriv(0) .gt. 0 .and. ls_deriv(1) .lt. .0) then
  ascend_pt = ls_param(0) ; ascend_norm = ls_norm(0)
  descend_pt = ls_param(3) ; descend_norm = ls_norm(3)
  ls_param(0) = descend_pt + &
  descend_norm*(ascend_pt - descend_pt)/(ascend_norm + descend_norm)
  intersect = .true.
  exit
else
! calculate new position and interval for next iteration
  ls_param(0) = ls_param(2)
  ls_param(1) = ls_param(3)
! update ls_deriv(0)
  ls_deriv(0) = ls_deriv(1)
endif
enddo
! improve the line search parameter estimate if an intersectio pt has been
! found
if (intersect) then
  do iter=1,5
! calculate derivative at new pt
  inc = 0.001*dabs(ascend_pt - descend_pt)
  ls_param(1) = ls_param(0) + inc
  do i=0,1
    ls_incdisp = ls_param(i)*incdisp ; bdylds = .0 ; g_ipSigma = .0
    call bodyloads (g_g,g_num,g_numip,g_coord,prop,totdisp,ls_incdisp,&
      mat_id,mat_dam,mat_cpore,etype,gaptol,g_ipDam,g_ipPoros,&
      crackstat,bdylds,g_ipSigma)
    ls_resid = apl_inc - bdylds
    ls_norm(i) = dabs(dot_product(incdisp,ls_resid))
    write (iounit,'(f12.6,tr1,e12.6)') ls_param(i), ls_norm(i)
  enddo
  ls_deriv(0) = (ls_norm(1)-ls_norm(0))/(ls_param(1)-ls_param(0))
! check if less than minimum derivative found previously
if (dabs(ls_deriv(1)) .lt. dabs(deriv_min)) then
  deriv_min = ls_deriv(0)
  if (deriv_min .lt. .0) then
    param_min = ls_param(1)
  else
    param_min = ls_param(0)
  endif
endif

```

```

endif
! check if ascending or descending and calculate pt for second intersection
if (ls_deriv(0) .lt. .0) then
  descend_pt = ls_param(1)
  descend_norm = ls_norm(1)
elseif (ls_deriv(0) .gt. .0) then
  ascend_pt = ls_param(0)
  ascend_norm = ls_norm(0)
endif
ls_param(0) = descend_pt + &
descend_norm*(ascend_pt - descend_pt)/(ascend_norm + descend_norm)
enddo
lsearch = ls_param(0)
else
! default to pt with minimum derivative found from previous iterations
lsearch = param_min
endif
! calculate residual load norm at trial value
ls_incdisp = lsearch*incdisp ; bdylds = .0 ; g_ipSigma = .0
call bodyloads (g,g_num,g_numip,g_coord,prop,totdisp,ls_incdisp,&
mat_id,mat_dam,mat_cpore,etype,gaptol,g_ipDam,g_ipPoros,&
crackstat,bdyls,g_ipSigma)
ls_resid = apl_inc - bdylds
ls_rlnorm = dot_product(ls_resid,ls_resid)**0.5
! compare with Newton-Raphson residual
if (ls_rlnorm .lt. rlnorm) then
  resid = ls_resid
  incdisp = ls_incdisp
  rlnorm = ls_rlnorm
else
  ls_incdisp = param_min*incdisp ; bdylds = .0 ; g_ipSigma = .0
  if (lsearch .ne. param_min) then
    call bodyloads (g,g_num,g_numip,g_coord,prop,totdisp,ls_incdisp,&
mat_id,mat_dam,mat_cpore,etype,gaptol,g_ipDam,g_ipPoros,&
crackstat,bdyls,g_ipSigma)
    ls_resid = apl_inc - bdylds
    ls_rlnorm = dot_product(ls_resid,ls_resid)**0.5
  if (ls_rlnorm .lt. rlnorm) then
    lsearch = param_min
    incdisp = lsearch*incdisp
    rlnorm = ls_rlnorm
  else
    lsearch = 1.
  endif
endif
endif
! write resulting linesearch and residual norm to file
write (iounit,'/,f12.6') lsearch
write (iounit,'f12.6,/') rlnorm
return
end subroutine line_search
end module solution

module solvers
contains
! module of common operations required for solvers of linear systems
! of equations generated by finite element method
function bandwidth(g) result(nband)
! finds the element bandwidth from element steering vector
!
! in-----
! g = element steering vector
!
! out-----
! nband = bandwidth
!
implicit none ; integer :: nband
integer,intent(in)::g(:)

```

```

! nband= maxval(g,1,g>0)-minval(g,1,g>0)
!
end function bandwidth
subroutine fkdiag(kdiag,g)
! finds the maximum bandwidth for each freedom (row) in a skyline storage
! system
!
! in-----
! g = element steering vector
!
! out-----
! kdiag = array storing row lengths of stiffness matrix
!
! internal-----
! idof = dof counter
! iwpl = temporary bandwidth for main loop
! im = temporary bandwidth from inner loop
! i,j = loop counters
!
implicit none
integer,intent(in)::g(:); integer,intent(out)::kdiag(:)
integer::idof,i,iwpl,j,im
! find the no. dofs (i.e. entries) in g
idof=size(g)
!
! loop through dofs
do i = 1,idof
!
! iwpl=1
! if active freedom
! if(g(i)/=0) then
! loop through all dofs and calculate bandwidth between dof of inner loop
! and dof of outer loop
do j=1,idof
! calculate bandwidth + diagonal only if active dof
! if(g(j)/=0) then
! im=g(i)-g(j)+1
! if this bandwidth is higher than that of outer loop replace outer
! value with inner one
! if(im>iwpl) iwpl=im
end if
! substitute bandwidth into kdiag for this global dof if it is greater
! than current value
! if(iwpl > kdiag(g(i))) kdiag(g(i)) = iwpl
end do
end do
return
end subroutine fkdiag
subroutine parse_nz(g_g,nz)
! find no. non-zero entries in global stiffness matrix
!
! in-----
! g_g = global array of steering vectors
!
! out-----
! nz = no. non-zero entries
!
! internal-----
! nel = no. elements
! nel = element counter
! i = row counter
! j = column counter
implicit none
real(8),intent(in)::g_g(:,:)
real(8),intent(out)::nz
integer::i,j
nz = 0
return
end subroutine parse_nz
subroutine formkv(bk,km,g,n)
! form global stiffness matrix stored as a vector (upper triangle)
!

```

```

in-----
km      = element stiffness matrix
g       = element steering vector
n       = no. equations for global system

out-----
bk      = upper triangular banded global stiffness matrix
        store as a vector

internal-----
idof    = dof counter
icd     = dof counter
ival    = loop counters
i,j     = loop counters

implicit none
real(8),intent(in)::km(:,:);real(8),intent(out)::bk(:)
integer,intent(in)::g(:),n
integer::idof,i,j,icd,ival

! get no. dof (i.e. entries) in km
idof=size(km,1)

! loop through dofs for first array index
do i=1,idof
! check if dof is active
if(g(i)/=0) then
! loop through dofs for second array index
do j=1,idof
! check if dof is active
if(g(j)/=0) then
! calculate bandwidth + diagonal wrt dof in outer loop
icd = g(j) - g(i) + 1
! this dof occurs after outer loop dof if icd-1 is >=0
if(icd-1>=0) then
! if so, then icd-1 is position beyond dof in outer loop
! convert this posn to overall position in matrix
ival = n*(icd-1) + g(i)
! store relevant stiffness coefficient in global stiffness
bk(ival) = bk(ival) + km(i,j)
end if
end do
end do
end do
return
end subroutine formkv

subroutine fparv(bk,km,g,kdiag)
! assembly of element matrices into skyline global matrix
! note: this uses a 'profile-in' storage scheme.
! The elements in a row of the lower triangular part are stored
! starting from the first nonzero element in the row and moving
! right to the diagonal element. The data for each row are stored
! in consecutive locations, the rows are stored in order and
! there is no space between the rows.

in-----
km      = element stiffness matrix
g       = element steering vector
kdiag   = vector storing positions of diagonal entries

out-----
bk      = skyline global matrix stored as a vector

internal-----
idof    = dof counter
iw      = dof counter
ival    = loop counters
i,j     = loop counters
k       = dummy variable used for dof no. retrieved from g
        (used to make statements read more concisely)

implicit none
real(8),intent(in)::km(:,:); integer,intent(in)::g(:),kdiag(:)
real(8),intent(out)::bk(:)

```

```

integer::i,idof,k,j,iw,ival
!
! get no. dof (i.e. entries) in g
idof=ubound(g,1)
! loop through dofs (by row) for first array index
do i=1,idof
! check if dof is active
if(g(i) /= 0) then
! loop through dofs (by column) for second array index
do j=1,idof
! check if dof is active
if(g(j)/=0) then
! calculate row dof no. - col dof no.
iw = g(i) - g(j)
! check if outer loop dof is greater than inner loop dof
if(iw>0) then
! calculate position in global array
ival = kdiag(g(i)) - iw
! add new stiffness to any existing one
bk(ival) = bk(ival) + km(i,j)
end if
end do
end do
end do
return
end subroutine fparv

subroutine skysolve (au,audiag,bx,iounit)
! subroutine to solve symmetric linear systems of equations in skyline storage
! uses CXML libraries

in-----
! au      = lhs matrix in skyline storage (i.e. profile-in or diagonal-out)
! audiag  = vector storing locations of diagonal elements of au
!         may not be double precision integer so must be explicitly
!         converted
! iounit  = io unit for output of any message

in/out-----
! bx      = rhs of system (can be a 2d array storing several rhs vectors if
!         needed; usually only one rhs vector though)

! internal variables-----
! (used for the simple driver routine 'dskyd' which only needs one call to
! solve the system)

n       = order of 'a' i.e. rectangular form of 'au'
nau     = no. elements in au
iaudiag = double precision version of audiag
ldbxb   = leading dim of bx (i.e. no. dof in a given rhs vector in bx)
nbxb    = no. of rhs vectors stored in bx
niparam = iparam(1) = length of the array iparam (>= 100)
nrparam = iparam(2) = length of the array rparam (>= 100)
iwrk    = iparam(3) = size of integer workspace (>= 2n)
rwrk    = iparam(4) = size of real workspace (default to zero as not yet
!         implemented)
iounit  = iparam(5) = io unit for output of any message; must be assigned in
!         calling program
iolevel = iparam(6) = level of messaging (0,1,2) 0: error only
!         1: error + minimal
!         2: error + detailed
idefault = iparam(7) = flag for use of default values
!         1: user values
!         0: defaults
istore  = iparam(8) = storage scheme. 1: profile-in, 2: diagonal-out
ipvt    = iparam(9) = flag for stopping if abs(pivot) is smaller than
!         pvt_sml; 0: stop; 1: continue
!         2: continue + replace pvt term with pvt_new
ipvt_loc = iparam(10) = location of pivot smaller than pvt_sml
!         if = 0 then no such pivot exists
pvt_sml = rparam(1) = value below which to stop if abs(pivot) is smaller

```



```

!
!           than pvt_sml
! pvt_val = rparam(2) = value to replace small pivot with if it occurs and
!           if ipvt = 2
! iwkr   = initial size of integer work space (default 2n) but can change due
!           calls with other routines for factorisation of lhs etc.
! rwrk   = initial size of real work space; remains unchanged as not yet
!           implemented
! ierror = error flag (zero implies normal exit)
!
! include 'cxml_include.f90'
!
! implicit none
! real(8), intent(in)::au(:); integer, intent(in)::audiag(:), iounit
! real(8), intent(inout)::bx(:, :)
! integer(4)::n, nau, ldbx, nbx, niparam, nrparam, ierror
! real(8)::pvt_sml, pvt_val
! integer(4), allocatable::iparam(:), iwkr(:), iaudiag(:)
! real(8), allocatable::rparam(:), rwrk(:)
!
! get no. elements in au
! nau = size(au)
! leading dim of bx is also order of full rectangular a
! ldbx = ubound(bx,1)
! n = ldbx
! allocate (iaudiag(n))
! iaudiag = audiag
! get no. rhs vectors
! nbx = ubound(bx,2)
! assign iparam etc.
! niparam = 100 ; nrparam = 100
! allocate (iparam(niparam), rparam(nrparam), iwkr(2*n), rwrk(1))
! iparam = 0 ; rparam = .0
! iparam(1) = niparam ; iparam(2) = nrparam
! iparam(3) = 2*n ; iparam(4) = 0
! iparam(5) = iounit ; iparam(6) = 0
! iparam(7) = 1 ; iparam(8) = 1
! iparam(9) = 2
! rparam(1) = 1.0d-12
! rparam(2) = 1.0d-12
!
! call the simple driver routine for factorization and solution,
! with a single right hand side (rwrk is a dummy argument).
!
! call dsskyd (n, au, iaudiag, nau, bx, ldbx, nbx, iparam, rparam, iwkr, rwrk, ierror)
!
! STOP condition: ierror = -3001
! if (ierror .eq. -3001) then
!   print *, "Solver cannot solve system of equations"
!   print *, "Possible instability due to material failure: check results &
!     carefully!"
!   write (iounit, '(a)') , "Solver cannot solve system of equations"
!   write (iounit, '(a)') , "Possible instability due to material failure: &
!     check results carefully!"
!
!   stop
! endif
! return
! end subroutine skysolve
!
! subroutine banred(bk,n)
! ! gaussian reduction on upper triangle of a symmetric matrix stored in
! ! banded vector form -
! ! avoid using if possible as very slow - use skysolver if possible
!
! ! in-----
! ! n = no. equations
! !
! ! in/out-----
! ! bk = upper triangular matrix stored as a vector
! !
! implicit none
! real(8), intent(in out)::bk(:); integer, intent(in)::n
! integer::i, il1, kbl, j, ij, nkb, m, ni, nj, iw ; real(8)::sum

```

```

!
! iw = ubound(bk,1)/n-1
! do i=2,n
!   il1=i-1; kbl=il1+iw+1
!   if (kbl-n>0) kbl=n
!   do j=1, kbl
!     ij=(j-1)*n+i; sum=bk(ij); nkb=j-iw
!     if (nkb<=0) nkb=1
!     if (nkb-il1<=0) then
!       do m=nkb, il1
!         ni=(i-m)*n+m ; nj=(j-m)*n+m
!         sum=sum-bk(ni)*bk(nj)/bk(m)
!       end do
!     end if
!     bk(ij)=sum
!   end do
! end do
!
! return
! end subroutine banred
!
! subroutine bacsub(bk, loads)
! ! performs the complete gaussian backsubstitution
! !
! ! in-----
! ! bk = reduced form of upper triangle of symmetric matrix packed band form
! !
! ! in/out-----
! ! loads = rhs vector of linear system
! !
! implicit none
! real(8), intent(in)::bk(:); real(8), intent(in out)::loads(0:)
! integer::nkb, k, i, jn, jj, il, n, iw; real(8)::sum
! n = ubound(loads,1); iw = ubound(bk,1)/n - 1
! loads(1)=loads(1)/bk(1)
! do i=2,n
!   sum=loads(i); il=i-1 ; nkb=i-iw
!   if (nkb<=0) nkb=1
!   do k=nkb, il
!     jn=(i-k)*n+k; sum=sum-bk(jn)*loads(k)
!   end do
!   loads(i)=sum/bk(i)
! end do
! do jj=2,n
!   i=n-jj+1; sum=.0; il=i+1; nkb=i+iw
!   if (nkb-n>0) nkb=n
!   do k=i+1, nkb
!     jn=(k-i)*n+i ; sum=sum+bk(jn)*loads(k)
!   end do
!   loads(i)=loads(i)-sum/bk(i)
! end do
!
! return
! end subroutine bacsub
!
! end module solvers
!
! module tensor_ops
! ! module for common tensor operations req'd in FE calculations
!
! interface prin_tensor
! ! interface to subroutines that calculate principal values, vectors and
! ! rotation
! ! matrices (can be used for function overloading, i.e. calling several similar
! ! but slightly different subroutines using the same calling name but different
! ! arguments)
!
! module procedure vprinval, vprinall, r2prinval, r2prinall
! end interface prin_tensor
!
! contains
!
! subroutine formsytens_23 (vector, tensor)
! ! forms a symmetric tensor from its voigt (i.e. column vector) form
! ! with ordering of shear components as: 23, 31, 12
! implicit none
! real(8), intent(in)::vector(:); real(8), intent(out)::tensor(:, :)

```

```

integer::n_comp, dim
n_comp = ubound(vector,1)
select case(n_comp)
  case(3)
    tensor(1,1)=vector(1);tensor(2,2)=vector(2);
    tensor(2,1)=vector(3);tensor(1,2)=tensor(2,1)
  case(6)
    dim = 3
    tensor(1,1)=vector(1) ; tensor(2,2)=vector(2) ; tensor(3,3)=vector(3)
    tensor(2,3)=vector(4) ; tensor(3,1)=vector(5) ; tensor(1,2)=vector(6)
    tensor(2,1)=tensor(1,2) ; tensor(3,2)=tensor(2,3) ; tensor(1,3)=tensor(3,1)
end select
return
end subroutine formsymtens_23

subroutine formsymtens_12 (vector,tensor)
! forms a symmetric tensor from its voigt (i.e. column vector) form
! with ordering of shear components as: 12, 23, 31
implicit none
real(8),intent(in)::vector(:); real(8),intent(out)::tensor(:,:)
integer::n_comp, dim
n_comp = ubound(vector,1)
select case(n_comp)
  case(3)
    tensor(1,1)=vector(1);tensor(2,2)=vector(2);
    tensor(2,1)=vector(3);tensor(1,2)=tensor(2,1)
  case(6)
    dim = 3
    tensor(1,1)=vector(1) ; tensor(2,2)=vector(2) ; tensor(3,3)=vector(3)
    tensor(1,2)=vector(4) ; tensor(2,3)=vector(5) ; tensor(3,1)=vector(6)
    tensor(2,1)=tensor(1,2) ; tensor(3,2)=tensor(2,3) ; tensor(1,3)=tensor(3,1)
end select
return
end subroutine formsymtens_12

subroutine formvoigt_23 (tensor,vector)
! forms a symmetric tensor from its voigt (i.e. column vector) form
! with ordering of shear components as: 23, 31, 12
implicit none
real(8),intent(in)::tensor(:,:); real(8),intent(out)::vector(:)
integer::n_comp, dim
dim = ubound(tensor,1)
select case(dim)
  case(2)
    vector(1)=tensor(1,1);vector(2)=tensor(2,2);
    vector(3)=tensor(2,1)
  case(3)
    vector(1)=tensor(1,1);vector(2)=tensor(2,2);vector(3)=tensor(3,3)
    vector(4)=tensor(2,3);vector(5)=tensor(3,1);vector(6)=tensor(1,2)
end select
return
end subroutine formvoigt_23

subroutine formvoigt_12 (tensor,vector)
! forms a symmetric tensor from its voigt (i.e. column vector) form
! with ordering of shear components as: 12, 23, 31
implicit none
real(8),intent(in)::tensor(:,:); real(8),intent(out)::vector(:)
integer::n_comp, dim
dim = ubound(tensor,1)
select case(dim)
  case(2)
    vector(1)=tensor(1,1);vector(2)=tensor(2,2);
    vector(3)=tensor(2,1)
  case(3)
    vector(1)=tensor(1,1);vector(2)=tensor(2,2);vector(3)=tensor(3,3)
    vector(4)=tensor(1,2);vector(5)=tensor(2,3);vector(6)=tensor(3,1)
end select
return
end subroutine formvoigt_12

subroutine v12to23 (vector)
! converts a ...12,23,31 voigt representation of a symmetric rank 2 tensor to

```

```

! a ...23,31,12 representation
! this is req'd because Smith and Griffiths FE library uses 12,23,31 whereas
! 23,31,12 notation is more commonly found in anisotropic elasticity texts
implicit none
real(8),intent(inout)::vector(:)
real(8),allocatable::tmpvec(:)
integer::nst
nst = ubound(vector,1)
select case (nst)
  case(3)
    print *, "No need to convert from 12->23 voigt notation for 2d case"
    print *, "See subroutine 12to23 in tensor_ops module"
  case(6)
    ! position: 4 5 6
    ! out: 23, 31, 12
    ! in: 12, 23, 31
    allocate (tmpvec(nst))
    tmpvec(1) = vector(1)
    tmpvec(2) = vector(2)
    tmpvec(3) = vector(3)
    tmpvec(4) = vector(5)
    tmpvec(5) = vector(6)
    tmpvec(6) = vector(4)
end select
vector = tmpvec
return
end subroutine v12to23

subroutine r2rot_t (tensor,rot)
! returns the rotation matrix of direction cosines of a symmetric rank 2 tensor
! principal coordinate system wrt global cartesian system
! note: ensures ordering of resulting rotation fits classical e1 > e2 > e3
! (req'd because IMSL seems to order based on absolute rather than
! integer values)
!
! in-----
! tensor = tensor for which rotation matrix to principal csys is req'd
!
! out-----
! rot = rotation matrix
!
! internal-----
! eval = vector of eigenvalues
! evec = array of eigenvectors
! dim = dimension of tensor
!
use numerical_libraries
implicit none
real(8),intent(in)::tensor(:,:); real(8),intent(out)::rot(:,:)
real(8),allocatable::gbasis(:,:);eval(:),evec(:,:),dummy(:,:)
integer,allocatable::idvec(:)
integer::dim,i,j

! get dimension of tensor and allocate arrays
dim = ubound(tensor,1)
allocate (gbasis(dim,dim),eval(dim),evec(dim,dim),idvec(dim),dummy(dim,dim))

! construct global base vector array
call kronecker(dim,2,gbasis)

! calculate eigenvalues/vectors with eigenvectors returned as columns of evec
call devcsf(dim,tensor,dim,eval,evec,dim)

! get correct ordering and reorder evec (store in dummy array)
call orderPrinvals(eval,idvec)
dummy(:,1) = evec(:,idvec(1))
dummy(:,2) = evec(:,idvec(2))
dummy(:,3) = evec(:,idvec(3))

! calculate each component of rot; i.e. rot(i,j) = ei'.ej, where prime symbol
! => principal csys
do i=1,dim

```

```

do j=1,dim
! assumes base (eigen) vectors stored by column
rot(i,j)=dot_product(dummy(:,i),gbasis(:,j))
enddo
enddo
return
end subroutine r2rot_t

subroutine r2rot_v12 (vector,rot)
! returns the rotation matrix of direction cosines from voigt form of symmetric
! rank 2 tensor principal coordinate system wrt global cartesian system
! first finds eigenvectors and then evaluates direction cosines
use numerical_libraries
implicit none
real(8),intent(in)::vector(:); real(8),intent(out)::rot(:, :)
real(8),allocatable:: eval(:), evec(:, :), gbasis(:, :), tensor(:, :)
integer::n_comp,dim,lda,ldevec,i,j
n_comp = ubound(vector,1)
select case(n_comp)
case(3)
dim = 2
case(6)
dim = 3
end select
allocate (tensor(dim,dim),eval(dim),evec(dim,dim),gbasis(dim,dim))
call formsytens_12(vector,tensor)
lda=dim; ldevec=dim
call devcsf(dim,tensor,lda,eval,evec,ldevec)
call kronecker(dim,2,gbasis)
do i=1,dim
do j=1,dim
! assumes base (eigen) vectors stored by column
rot(i,j)=dot_product(evec(:,i),gbasis(:,j))
enddo
enddo
return
end subroutine r2rot_v12

subroutine r2rot_b (basis,rot)
! returns the rotation matrix of direction cosines of previously calculated
! basis wrt global cartesian system
use numerical_libraries
implicit none
real(8),intent(in)::basis(:, :); real(8),intent(out)::rot(:, :)
real(8),allocatable::g_basis(:, :)
integer::dim,i,j ; dim = ubound(basis,1)
allocate (g_basis(dim,dim))
call kronecker(dim,2,g_basis)
do i=1,dim
do j=1,dim
! assumes base (eigen) vectors stored by column
rot(i,j)=dot_product(basis(:,i),g_basis(:,j))
enddo
enddo
return
end subroutine r2rot_b

subroutine r4rot_23 (r,ts,te)
! returns the rank 4 rotation operators in form of matrices (voigt notation)
! (with ...23,31,12 ordering) from the rank 2 rotation matrix (of direction
! cosines) the returned matrices can be used to rotate continuum stiffness
! matrix between coordinate systems
! 'ts' can be used to rotate stress in voigt form: s' = ts.s
! 'te' can be used to rotate strain in voigt form: e' = te.e
! when rotating stiffness in matrix form: d' = ts.d.transpose(ts)
! d = transpose(te).d'.te
! generally this routine is only called for rotating stiffness since to call
! this the rank2 rotation matrix needs to have been calculated already
!
! note:
! (1) for matrix coefficients see F.G. Yuan, Anisotropic Elasticity:
! Application to Composite Fracture Mechanics, Lecture in Mechanics

```

```

! of Materials Branch NASA Langley Research Centre, Hampton, VA 23681,
! July 28-31, 1998:
! 'ts' corresponds to his 'Q' and 'te' to 'inverse_transpose(Q)'

implicit none
real(8),intent(in)::r(:, :); real(8),intent(out)::te(:, :), ts(:, :)
integer::dim,i,j ; dim = ubound(r,1)
!
! fill rest of entries depending on dimensionality
select case (dim)
case (2)
! row 1
print *, "2D case not supported at this time"
te = 0.; ts = 0.
! row 2
! row 3
case (3)
! 'te'
! row 1
te(1,1)=r(1,1)**2.; te(1,2)=r(1,2)**2.; te(1,3)=r(1,3)**2.
te(1,4)=r(1,2)*r(1,3); te(1,5)=r(1,3)*r(1,1); te(1,6)=r(1,1)*r(1,2)
! row 2
te(2,1)=r(2,1)**2.; te(2,2)=r(2,2)**2.; te(2,3)=r(2,3)**2.
te(2,4)=r(2,2)*r(2,3); te(2,5)=r(2,3)*r(2,1); te(2,6)=r(2,1)*r(2,2)
! row 3
te(3,1)=r(3,1)**2.; te(3,2)=r(3,2)**2.; te(3,3)=r(3,3)**2.
te(3,4)=r(3,2)*r(3,3); te(3,5)=r(3,3)*r(3,1); te(3,6)=r(3,1)*r(3,2)
! row 4
te(4,1)=2.*r(2,1)*r(3,1); te(4,2)=2.*r(2,2)*r(3,2)
te(4,3)=2.*r(2,3)*r(3,3)
te(4,4)=r(2,2)*r(3,3)+r(2,3)*r(3,2); te(4,5)=r(2,3)*r(3,1)+r(2,1)*r(3,3)
te(4,6)=r(2,1)*r(3,2)+r(2,2)*r(3,1)
! row 5
te(5,1)=2.*r(3,1)*r(1,1); te(5,2)=2.*r(3,2)*r(1,2)
te(5,3)=2.*r(3,3)*r(1,3); te(5,4)=r(3,2)*r(1,3)+r(3,3)*r(1,2)
te(5,5)=r(3,3)*r(1,1)+r(3,1)*r(1,3); te(5,6)=r(3,1)*r(1,2)+r(3,2)*r(1,1)
! row 6
te(6,1)=2.*r(1,1)*r(2,1); te(6,2)=2.*r(1,2)*r(2,2)
te(6,3)=2.*r(1,3)*r(2,3); te(6,4)=r(1,2)*r(2,3)+r(1,3)*r(2,2)
te(6,5)=r(1,3)*r(2,1)+r(1,1)*r(2,3); te(6,6)=r(1,1)*r(2,2)+r(1,2)*r(2,1)
!
! 'ts'
! ts = te except that top right partition is multiplied x 2 and bottom right
! x 0.5
ts = te
!
! rescale top right partition
! row 1
ts(1,4)=2.*te(1,4); ts(1,5)=2.*te(1,5); ts(1,6)=2.*te(1,6)
! row 2
ts(2,4)=2.*te(2,4); ts(2,5)=2.*te(2,5); ts(2,6)=2.*te(2,6)
! row 3
ts(3,4)=2.*te(3,4); ts(3,5)=2.*te(3,5); ts(3,6)=2.*te(3,6)
!
! rescale bottom left partition
! row 4
ts(4,1)=0.5*te(1,4); ts(4,2)=0.5*te(4,2); ts(4,3)=0.5*te(4,3)
! row 5
ts(5,1)=0.5*te(1,5); ts(5,2)=0.5*te(5,2); ts(5,3)=0.5*te(5,3)
! row 6
ts(6,1)=0.5*te(1,6); ts(6,2)=0.5*te(6,2); ts(5,3)=0.5*te(6,3)
!
end select
return
end subroutine r4rot_23

subroutine r4rot_12 (r,ts,te)
! returns the rank 4 rotation operators in the form of matrices (voigt
! notation: ...12,23,31 form) from the rank 2 rotation matrix (of
! direction cosines). The returned matrices can be used to rotate continuum
! stiffness matrix between coordinate systems.
! 'ts' can be used to rotate stress in voigt form: s' = ts.s
! 'te' can be used to rotate strain in voigt form: e' = te.e
! when rotating stiffness in matrix form: d' = ts.d.transpose(ts)
! d = transpose(te).d'.te
! generally this routine is only called for rotating stiffness since to

```

```

! call this the rank2 rotation matrix needs to have been calculated already
!
! note:
! (i) for matrix coefficients see thesis (A.B. Lennon, 2002)
! (ii) this corresponds to a 11,22,33,12,23,31 form of voigt notation
!
implicit none
real(8),intent(in)::r(:,,:); real(8),intent(out)::te(:,,:), ts(:,,:)
integer::dim,i,j ; dim = ubound(r,1)
!
! fill rest of entries depending on dimensionality
select case (dim)
case (2)
! row 1
print *, "2D case not supported at this time"
te = 0. ; ts = 0.
! row 2
! row 3
case (3)
! 'ts'-stress rotation operator
! row 1
ts(1,1)=r(1,1)**2. ; ts(1,2)=r(1,2)**2. ; ts(1,3)=r(1,3)**2.
ts(1,4)=2.*r(1,1)*r(1,2) ; ts(1,5)=2.*r(1,2)*r(1,3)
ts(1,6)=2.*r(1,1)*r(1,3)
! row 2
ts(2,1)=r(2,1)**2. ; ts(2,2)=r(2,2)**2. ; ts(2,3)=r(2,3)**2.
ts(2,4)=2.*r(2,1)*r(2,2) ; ts(2,5)=2.*r(2,2)*r(2,3)
ts(2,6)=2.*r(2,1)*r(2,3)
! row 3
ts(3,1)=r(3,1)**2. ; ts(3,2)=r(3,2)**2. ; ts(3,3)=r(3,3)**2.
ts(3,4)=2.*r(3,1)*r(3,2) ; ts(3,5)=2.*r(3,2)*r(3,3)
ts(3,6)=2.*r(3,1)*r(3,3)
! row 4
ts(4,1)=r(1,1)*r(2,1) ; ts(4,2)=r(1,2)*r(2,2) ; ts(4,3)=r(1,3)*r(2,3)
ts(4,4)=r(1,2)*r(2,1)+r(1,1)*r(2,2) ; ts(4,5)=r(1,3)*r(2,2)+r(1,2)*r(2,3)
ts(4,6)=r(1,3)*r(2,1)+r(1,1)*r(2,3)
! row 5
ts(5,1)=r(2,1)*r(3,1) ; ts(5,2)=r(2,2)*r(3,2) ; ts(5,3)=r(2,3)*r(3,3)
ts(5,4)=r(2,2)*r(3,1)+r(2,1)*r(3,2) ; ts(5,5)=r(2,3)*r(3,2)+r(2,2)*r(3,3)
ts(5,6)=r(2,3)*r(3,1)+r(2,1)*r(3,3)
! row 6
ts(6,1)=r(1,1)*r(3,1) ; ts(6,2)=r(1,2)*r(3,2) ; ts(6,3)=r(1,3)*r(3,3)
ts(6,4)=r(1,2)*r(3,1)+r(1,1)*r(3,2) ; ts(6,5)=r(1,3)*r(3,2)+r(1,2)*r(3,3)
ts(6,6)=r(1,3)*r(3,1)+r(1,1)*r(3,3)
!
!-----
! 'te'-strain rotation operator
! te = ts except that top right partition is multiplied x 0.5 and bottom
! right x 2
te = ts
! rescale top right partition
! row 1
te(1,4)=0.5*ts(1,4) ; te(1,5)=0.5*ts(1,5) ; te(1,6)=0.5*ts(1,6)
! row 2
te(2,4)=0.5*ts(2,4) ; te(2,5)=0.5*ts(2,5) ; te(2,6)=0.5*ts(2,6)
! row 3
te(3,4)=0.5*ts(3,4) ; te(3,5)=0.5*ts(3,5) ; te(3,6)=0.5*ts(3,6)
!
! rescale bottom left partition
! row 4
te(4,1)=2.*ts(4,1) ; te(4,2)=2.*ts(4,2) ; te(4,3)=2.*ts(4,3)
! row 5
te(5,1)=2.*ts(5,1) ; te(5,2)=2.*ts(5,2) ; te(5,3)=2.*ts(5,3)
! row 6
te(6,1)=2.*ts(6,1) ; te(6,2)=2.*ts(6,2) ; te(6,3)=2.*ts(6,3)
!
end select
return
end subroutine r4rot_12
subroutine r4proj_12 (nvec,p)
! returns the rank 4 projection tensor from a unit normal vector (e.g.
! eigenvector of damage tensor)
! compatible with ...12,23,31 form of voigt notation as use by Smith and
! Griffiths

```

```

implicit none
real(8),intent(in)::nvec(:) ; real(8),intent(out)::p(:,,:)
real(8)::a,b,c
integer::dim
!
! find no. of dimensions for problem and implement req'd form of 'p'
dim = ubound(nvec,1)
select case (dim)
! 2d
case (2)
print*, "Not supported at this time"
p = 0.
case (3)
! assign variables a,b,c to each vector component
a = nvec(1) ; b = nvec(2) ; c = nvec(3)
! row 1
p(1,1)=a**4 ; p(1,2)=a**2*b**2 ; p(1,3)=a**2*c**2
p(1,4)=a**3*b ; p(1,5)=a**2*b*c ; p(1,6)=a**3*c
! row 2
p(2,1)=a**2*b**2 ; p(2,2)=b**4 ; p(2,3)=b**2*c**2
p(2,4)=a*b**3 ; p(2,5)=b**3*c ; p(2,6)=a*b**2*c
! row 3
p(3,1)=a**2*c**2 ; p(3,2)=b**2*c**2 ; p(3,3)=c**4
p(3,4)=a*b*c**2 ; p(3,5)=b*c**3 ; p(3,6)=a*c**3
! row 4
p(4,1)=2*a**3*b ; p(4,2)=2*a*b**3 ; p(4,3)=2*a*b*c**2
p(4,4)=2*a**2*b**2 ; p(4,5)=2*a*b**2*c ; p(4,6)=2*a**2*b*c
! row 5
p(5,1)=2*a**2*b*c ; p(5,2)=2*b**3*c ; p(5,3)=2*b*c**3
p(5,4)=2*a*b**2*c ; p(5,5)=2*b**2*c**2 ; p(5,6)=2*a*b*c**2
! row 6
p(6,1)=2*a**3*c ; p(6,2)=2*a*b**2*c ; p(6,3)=2*a*c**3
p(6,4)=2*a**2*b*c ; p(6,5)=2*a*b*c**2 ; p(6,6)=2*a**2*c**2
end select
return
end subroutine r4proj_12
subroutine kronecker (dim,rank,delta)
! returns the rank x Kronecker delta for a specified no. dimensions
implicit none
integer,intent(in)::dim, rank ; real(8),intent(out)::delta(:,,:)
integer::i,j
select case (rank)
case(2)
do i=1,dim
do j=1,dim
if (i.eq.j) then
delta(i,j)=1
else
delta(i,j)=0
endif
enddo
enddo
end select
return
end subroutine kronecker
subroutine prinbasis_12 (voigt_tensor,prin_val,prin_uvec)
! returns principal values and base vectors of voigt form of rank 2 tensor
use numerical_libraries
implicit none
real(8),intent(in)::voigt_tensor(:) !voigt form of tensor, e.g. 6x1 stress
! vector
real(8),intent(out)::prin_val(:),prin_uvec(:,,:) !principal values and matrix
! of unit vectors
real(8),allocatable::tensor(:,,:)
integer::i,j,n_comp,dim !counters, no. components of vector, and
! dimensionality, i.e. 2D or 3D
n_comp = ubound(voigt_tensor,1)
select case(n_comp)
case(3)
dim = 2 ; allocate (tensor(dim,dim))
case(6)

```

```

    dim = 3 ; allocate (tensor(dim,dim))
end select
call formsytmens_12 (voigt_tensor,tensor)
call devcsf(dim,tensor,dim,prin_val,prin_uvec,dim)
return
end subroutine prinbasis_12
subroutine vprinbasis (voigt_tensor,v_scheme,prin_val,prin_uvec)
! returns principal values and base vectors of voigt form of rank 2 tensor
in-----
! voigt_tensor      = voigt form of tensor, e.g. 6x1 stress vector
! v_scheme          = voigt storage scheme variable (i.e. 12 or 23)
out-----
! prin_val         = principal values
! prin_uvec        = matrix of unit eigenvectors (stored in columns)
internal-----
! tensor           = array of rank2 tensor form
! i,j,n_comp,dim   = counters, no. components of vector, and
!                   dimensionality, i.e. 2D or 3D
!
use numerical_libraries
implicit none
real(8),intent(in)::voigt_tensor(:)
integer,intent(in)::v_scheme
real(8),intent(out)::prin_val(:),prin_uvec(:,:)
real(8),allocatable::tensor(:,:)
integer::i,j,n_comp,dim
n_comp = ubound(voigt_tensor,1)
select case(n_comp)
  case(3)
    dim = 2 ; allocate (tensor(dim,dim))
  case(6)
    dim = 3 ; allocate (tensor(dim,dim))
end select
! call appropriate tensor formation method for storage scheme
if (v_scheme .eq. 12) then
  call formsytmens_12 (voigt_tensor,tensor)
elseif (v_scheme .eq. 23) then
  call formsytmens_23 (voigt_tensor,tensor)
else
  call formsytmens_12 (voigt_tensor,tensor)
endif
! find eigenvalues and basis of eigenvectors
call devcsf(dim,tensor,dim,prin_val,prin_uvec,dim)
return
end subroutine vprinbasis
subroutine r2prinbasis (tensor,prin_val,prin_uvec)
! returns principal values and base vectors of voigt form of rank 2 tensor
in-----
! voigt_tensor      = voigt form of tensor, e.g. 6x1 stress vector
! v_scheme          = voigt storage scheme variable (i.e. 12 or 23)
out-----
! prin_val         = principal values
! prin_uvec        = matrix of unit eigenvectors (stored in columns)
internal-----
! tensor           = array of rank2 tensor form
! dim              = dimensionality, i.e. 2D or 3D
!
use numerical_libraries
implicit none
real(8),intent(in)::tensor(:,:)
real(8),intent(out)::prin_val(:),prin_uvec(:,:)
real(8),allocatable::dummy(:,:)
integer,allocatable::idvec(:)
integer::dim

```

```

! set dimensionality
dim = ubound(tensor,1)
allocate (idvec(dim),dummy(dim,dim))
! find eigenvalues and basis of eigenvectors
call devcsf(dim,tensor,dim,prin_val,prin_uvec,dim)
! make sure ordering is based on integer, not absolute, values
call orderPrinvals(prin_val,idvec)
dummy(:,1) = prin_uvec(:,idvec(1))
dummy(:,2) = prin_uvec(:,idvec(2))
if (dim .eq. 3) dummy(:,3) = prin_uvec(:,idvec(3))
prin_uvec = dummy
return
end subroutine r2prinbasis
subroutine vprinval (voigt_tensor,v_scheme,prin_val)
! returns principal values only of voigt form of rank 2 tensor
in-----
! voigt_tensor      = voigt form of tensor, e.g. 6x1 stress vector
! v_scheme          = voigt storage scheme variable (i.e. 12 or 23)
out-----
! prin_val         = principal values
internal-----
! tensor           = array of rank2 tensor form
! n_comp,dim       = no. components of vector, and dimensionality
!
use numerical_libraries
implicit none
real(8),intent(in)::voigt_tensor(:)
integer,intent(in)::v_scheme
real(8),intent(out)::prin_val(:)
real(8),allocatable::tensor(:,:)
integer,allocatable::idvec(:)
integer::n_comp,dim
n_comp = ubound(voigt_tensor,1)
select case(n_comp)
  case(3)
    dim = 2
  case(6)
    dim = 3
end select
allocate (tensor(dim,dim),idvec(dim))
! call appropriate tensor formation method for storage scheme
if (v_scheme .eq. 12) then
  call formsytmens_12 (voigt_tensor,tensor)
elseif (v_scheme .eq. 23) then
  call formsytmens_23 (voigt_tensor,tensor)
else
  call formsytmens_12 (voigt_tensor,tensor)
endif
! find eigenvalues and basis of eigenvectors
call devlsf(dim,tensor,dim,prin_val)
! make sure ordering is based on integer, not absolute, values
call orderPrinvals(prin_val,idvec)
return
end subroutine vprinval
subroutine r2prinval (tensor,prin_val)
! returns principal values only of rank 2 tensor
in-----
! tensor           = tensor, e.g. 3x3 stress vector
out-----
! prin_val         = principal values

```

```

! internal-----
! i,j,dim      = counters, and dimensionality, i.e. 2D or 3D
!
! use numerical_libraries
! implicit none
! real(8),intent(in)::tensor(:,:)
! real(8),intent(out)::prin_val(:,)
! integer::i,j,n_comp,dim
! integer,allocatable::idvec(:)
!
! set dimensionality
! dim = ubound(tensor,1)
! allocate (idvec(dim))
!
! find eigenvalues
! call devlsf(dim,tensor,dim,prin_val)
!
! make sure ordering is based on integer, not absolute, values
! call orderPrinvals(prin_val,idvec)
!
! return
! end subroutine r2prinval
!
! returns principal values only of voigt form of rank 2 tensor
!
! in-----
! voigt_tensor = voigt form of tensor, e.g. 6x1 stress vector
! v_scheme     = voigt storage scheme variable (i.e. 12 or 23)
!
! out-----
! prin_val     = principal values
! rot          = rotation matrix for principal csys
!
! internal-----
! tensor       = array of rank2 tensor form
! n_comp,dim   = no. components of vector, and dimensionality
!
! use numerical_libraries
! implicit none
! real(8),intent(in)::voigt_tensor(:)
! integer,intent(in)::v_scheme
! real(8),intent(out)::prin_val(:,),rot(:,)
! real(8),allocatable::tensor(:,),evec(:,),gbasis(:,)
! integer,allocatable::idvec(:)
! integer::dim,i,j
! dim = ubound(rot,1)
! allocate (tensor(dim,dim),idvec(dim),evec(dim,dim),gbasis(dim,dim))
!
! call appropriate tensor formation method for storage scheme
! if (v_scheme .eq. 12) then
!   call formsytens_12 (voigt_tensor,tensor)
! elseif (v_scheme .eq. 23) then
!   call formsytens_23 (voigt_tensor,tensor)
! else
!   call formsytens_12 (voigt_tensor,tensor)
! endif
!
! find eigenvalues and basis of eigenvectors
! call devcsf(dim,tensor,dim,prin_val,evec,dim)
!
! make sure ordering is based on integer, not absolute, values
! call orderPrinvals(prin_val,idvec)
! tensor(:,1) = evec(:,idvec(1))
! tensor(:,2) = evec(:,idvec(2))
! tensor(:,3) = evec(:,idvec(3))
!
! construct global base vector array
! call kronecker(dim,2,gbasis)
!
! calculate each component of rot; i.e. rot(i,j) = ei'.ej, where prime symbol
! => principal csys
! do i=1,dim
!   do j=1,dim
!     assumes base (eigen) vectors stored by column

```

```

      rot(i,j)=dot_product(tensor(:,i),gbasis(:,j))
    enddo
  enddo
! return
! end subroutine vprinall
!
! subroutine r2prinall (tensor,prinvals,prinrot)
! returns the principal values and rotation matrix of direction cosines
! of a symmetric rank 2 tensor principal coordinate system wrt global
! cartesian system
! note: ensures ordering of resulting rotation fits classical e1 > e2 > e3
! (req'd because IMSL seems to order based on absolute rather than
! integer values)
!
! in-----
! tensor       = tensor for which rotation matrix to principal csys is req'd
!
! out-----
! rot          = rotation matrix
!
! internal-----
! eval         = vector of eigenvalues
! evec         = array of eigenvectors
! dim          = dimension of tensor
!
! use numerical_libraries
! implicit none
! real(8),intent(in)::tensor(:,)
! real(8),intent(out)::prinvals(:,),prinrot(:,)
! real(8),allocatable:: gbasis(:,),evec(:,),dummy(:,)
! integer,allocatable::idvec(:)
! integer::dim,i,j
!
! get dimension of tensor and allocate arrays
! dim = ubound(tensor,1)
! allocate (gbasis(dim,dim),evec(dim,dim),idvec(dim),dummy(dim,dim))
!
! construct global base vector array
! call kronecker(dim,2,gbasis)
!
! calculate eigenvalues/vectors with eigenvectors returned as columns of evec
! call devcsf(dim,tensor,dim,prinvals,evec,dim)
!
! get correct ordering and reorder evec (store in dummy array)
! call orderPrinvals(prinvals,idvec)
! dummy(:,1) = evec(:,idvec(1))
! dummy(:,2) = evec(:,idvec(2))
! dummy(:,3) = evec(:,idvec(3))
!
! calculate each component of rot; i.e. rot(i,j) = ei'.ej, where prime symbol
! => principal csys
! do i=1,dim
!   do j=1,dim
!     assumes base (eigen) vectors stored by column
!     prinrot(i,j)=dot_product(dummy(:,i),gbasis(:,j))
!   enddo
! enddo
! return
! end subroutine r2prinall
!
! subroutine prinvalue_12 (voigt_tensor,prin_val)
! returns principal values only of voigt form of rank 2 tensor
!
! in-----
! voigt_tensor = voigt form of tensor, e.g. 6x1 stress vector
!
! out-----
! prin_val     = principal values
!
! internal-----
! tensor       = symmetric tensor constructed from voigt form
! ncomp        = no. components of voigt form of tensor
! dim          = no. dimensions
! idvec        = array of array subscripts from reordering based on integer value

```

```

use numerical_libraries
implicit none
real(8),intent(in)::voigt_tensor(:)
real(8),intent(out)::prin_val(:)
real(8),allocatable::tensor(:,:)
integer::i,j,n_comp,dim
integer,allocatable::idvec(:)

! get no. components and set dimensionality
n_comp = ubound(voigt_tensor,1)
select case(n_comp)
  case(3)
    dim = 2 ; allocate (tensor(dim,dim))
  case(6)
    dim = 3 ; allocate (tensor(dim,dim))
end select

! form symmetric tensor from voigt vector in 12,23,31 format
call formsytens_12(voigt_tensor,tensor)

! find eigenvalues
call devlsf(dim,tensor,dim,prin_val)

! make sure ordering is based on integer, not absolute, values
call orderPrinvals(prin_val,idvec)
return
end subroutine prinvalue_12

subroutine maxprinvec(v_tens,v_store,vec)
! returns maximum principal vector of a tensor
!
! in-----
! v_tens = voigt form of tensor
! v_store = voigt storage scheme variable i.e. 12 or 23 format
!
! out-----
! vec = max principal vector of v_tens
!
! internal-----
! evals = array of eigenvalues
! basis = eigenbasis array (i.e. matrix of eigenvectors stored in columns)
! max_id = array subscript of maximum component
! maxpval = max principal value
!
implicit none
real(8),intent(in)::v_tens(:) ; integer,intent(in)::v_store
real(8),intent(out)::vec(:) ; real(8),allocatable::evals(:), basis(:,:)
real(8)::maxpval ; integer::dim, max_id

dim = ubound(v_tens,1)
select case (dim)
  case(6)
    dim = 3
  case(3)
    dim = 2
end select
allocate(evals(dim),basis(dim,dim))

call vprinbasis (v_tens,v_store,evals,basis)
call maxvalue(evals,maxpval,max_id)
vec = maxpval*basis(:,max_id)
return
end subroutine maxprinvec

subroutine gen_invert(a, ainv)
! invert general real matrix
use numerical_libraries

implicit none
real(8), intent(in)::a(:,:) ; real(8), intent(out)::ainv(:,:)
integer lda, ldainv, n, i, j, nout

! set dimensionality
lda = ubound(a,1) ; ldainv = lda ; n = lda

! compute inverse
call dling (n, a, lda, ainv, ldainv)
return
end subroutine gen_invert

subroutine gen_det (a,det)
! find determinant of a general real matrix
! if not 2x2 or 3x3: first LU factors the matrix and then
! computes the determinant (requires IMSL libraries)
use numerical_libraries
implicit none
real(8), intent(in)::a(:,:) ; real(8), intent(out)::det
real(8), allocatable::factor(:,:) ; real(8) det1, det2
integer, allocatable::ipvt(:) ; integer lda, ldfac, n, nout

! set dimensionality
lda = ubound(a,1) ; ldfac = lda ; n = lda
if (lda .eq. 2) then
  det = a(1,1)*a(2,2) - a(2,1)*a(1,2)
elseif (lda .eq. 3) then
  det= a(1,1)*a(2,2) * a(3,3) - a(3,2) * a(2,3)
  det= det - a(1,2)*a(2,1)*a(3,3) - a(3,1)*a(2,3)
  det= det + a(1,3)*a(2,1)*a(3,2) - a(3,1)*a(2,2)
elseif (lda .gt. 3) then
  allocate (factor(ldfac,ldfac),ipvt(n))
  ! compute LU factorisation
  call dlfrg (n, a, lda, factor, ldfac, ipvt)
  ! compute the determinant
  call dlfrg (n, factor, ldfac, ipvt, det1, det2)
  det = det1*10**det2
endif
return
end subroutine gen_det

subroutine cross_product (v1,v2,vout)
! calculate cross product of two 3D vectors
!
! in-----
! v1 = vector 1
! v2 = vector 2
!
! out-----
! vout = output vector of cross product calculation
!
! internal-----
! dim = dimensionality of vector
!
implicit none
real(8),intent(in)::v1(:),v2(:)
real(8),intent(out)::vout(:)

vout(1) = v1(2)*v2(3) - v1(3)*v2(2)
vout(2) = v1(3)*v2(1) - v1(1)*v2(3)
vout(3) = v1(1)*v2(2) - v1(2)*v2(1)
return
end subroutine cross_product

subroutine minvalue(a,minimum,min_id)
! returns minimum of a dim=3 vector (e.g. eigenvalues of 3x3 matrix)
!
! a = array (dim = 3) (intent: in)
! minimum = minvalue (intent: out)
! min_id = array subscript of minimum component
!
implicit none
real(8),intent(in)::a(:)
real(8),intent(out)::minimum
integer::dim, min_id

dim = ubound(a,1)
select case (dim)
  case (3)
    if ((a(1) .le. a(2)) .and. (a(2) .le. a(3))) then
      minimum = a(1) ; min_id = 1
    end if
end select

```

```

elseif ((a(2) .le. a(3)) .and. (a(3) .le. a(1))) then
  minimum = a(2) ; min_id = 2
elseif ((a(3) .le. a(1)) .and. (a(1) .le. a(2))) then
  minimum = a(3) ; min_id = 3
elseif ((a(3) .le. a(2)) .and. (a(2) .le. a(1))) then
  minimum = a(3) ; min_id = 3
elseif ((a(2) .le. a(1)) .and. (a(1) .le. a(3))) then
  minimum = a(2) ; min_id = 2
elseif ((a(1) .le. a(3)) .and. (a(3) .le. a(2))) then
  minimum = a(1) ; min_id = 1
endif
end select
return
end subroutine minvalue
subroutine maxvalue(a,maximum,max_id)
! returns maximum of a 2 or 3d vector (e.g. eigenvalues of 3x3 matrix)
!
! in-----
! a      = array (dim = 2 or 3)
!
! out-----
! maximum = maxvalue
!
! internal--
! max_id = array subscript of maximum component
! dim    = dimensionality of problem
!
implicit none
real(8),intent(in)::a(:)
real(8),intent(out)::maximum
integer::dim, max_id
!
dim = ubound(a,1)
select case (dim)
case (3)
  if ((a(1) .ge. a(2)) .and. (a(2) .ge. a(3))) then
    maximum = a(1) ; max_id = 1
  elseif ((a(2) .ge. a(3)) .and. (a(3) .ge. a(1))) then
    maximum = a(2) ; max_id = 2
  elseif ((a(3) .ge. a(1)) .and. (a(1) .ge. a(2))) then
    maximum = a(3) ; max_id = 3
  elseif ((a(3) .ge. a(2)) .and. (a(2) .ge. a(1))) then
    maximum = a(3) ; max_id = 3
  elseif ((a(2) .ge. a(1)) .and. (a(1) .ge. a(3))) then
    maximum = a(2) ; max_id = 2
  elseif ((a(1) .ge. a(3)) .and. (a(3) .ge. a(2))) then
    maximum = a(1) ; max_id = 1
  endif
end select

```

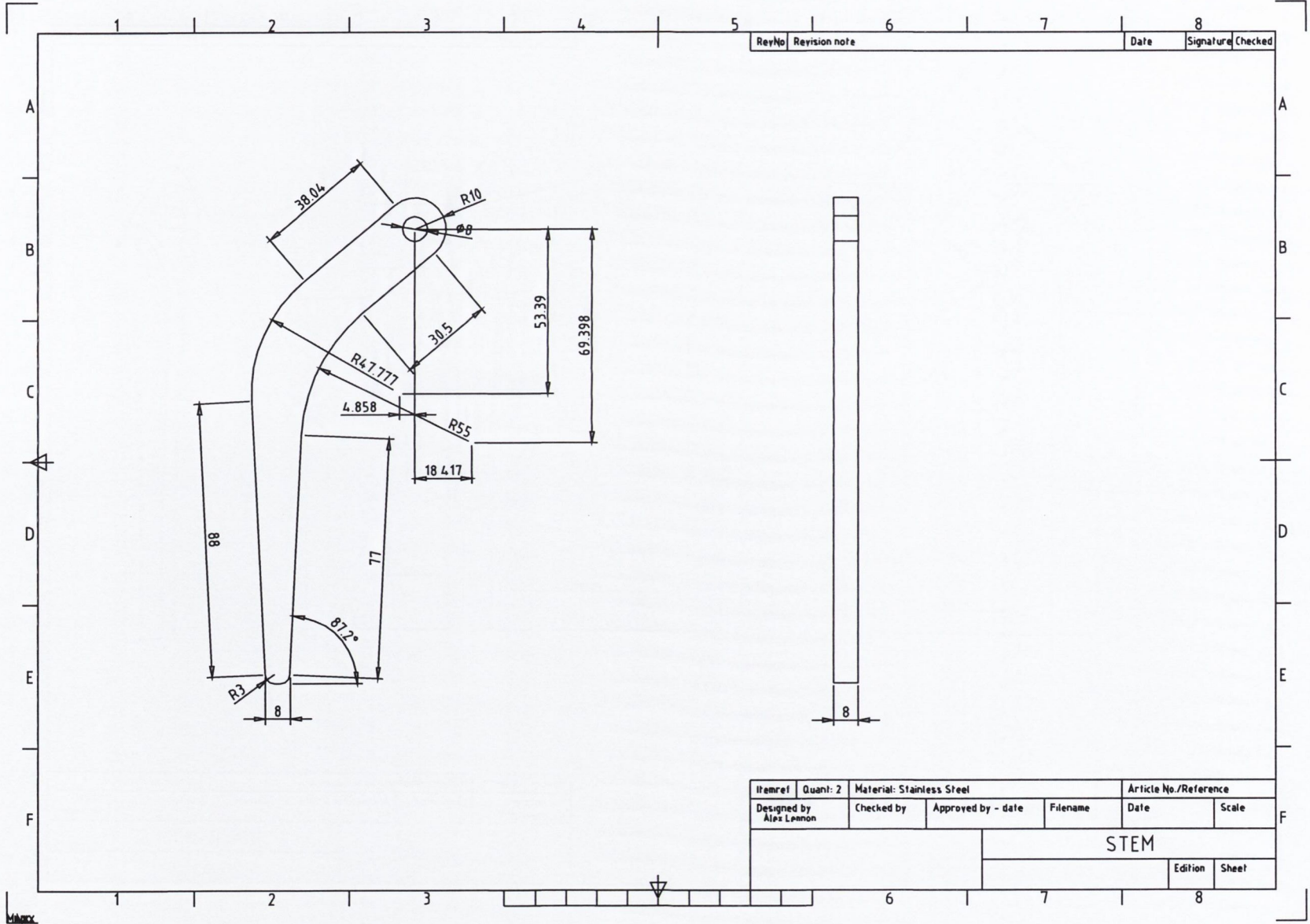
```

return
end subroutine maxvalue
subroutine orderPrinvals(a,ivec)
! reorders a 2 or 3d vector (e.g. eigenvalues of 3x3 matrix)
! such that the most positive is in the first position and the
! most negative is in the last
!
! in/out-----
! a      = array (dim = 2 or 3)
!
! out-----
! ivec   = vector of ids for reordered state
!
! internal-----
! maximum = maxvalue
! max_id  = array subscript of maximum component
!
! dim     = dimensionality of problem
!
implicit none
real(8),intent(inout)::a(:)
integer,intent(out)::ivec(:)
real(8)::maximum,minimum
integer::dim, i, max_id, min_id
!
dim = ubound(a,1)
select case (dim)
case (2)
  call maxvalue (a,maximum,max_id)
  call minvalue (a,minimum,min_id)
  a(1) = maximum
  a(2) = minimum
case (3)
  call maxvalue (a,maximum,max_id)
  call minvalue (a,minimum,min_id)
  do i=1,dim
    if (i .ne. max_id .and. i .ne. min_id) then
      a(1) = maximum
      ivec(1) = max_id
      a(2) = a(i)
      ivec(2) = i
      a(3) = minimum
      ivec(3) = min_id
    endif
  end do
end select
return
end subroutine orderPrinvals
end module tensor_ops

```

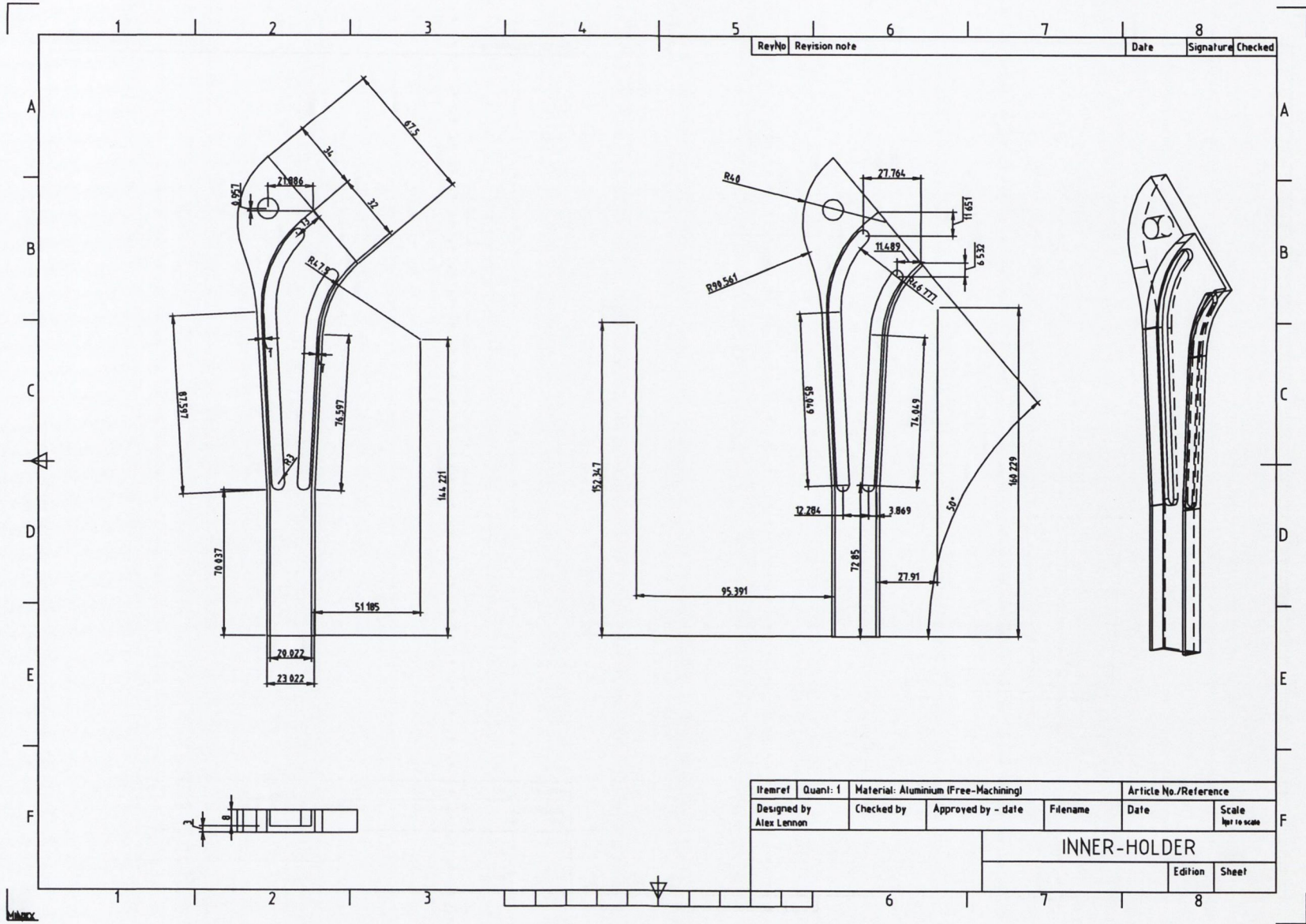

Appendix D

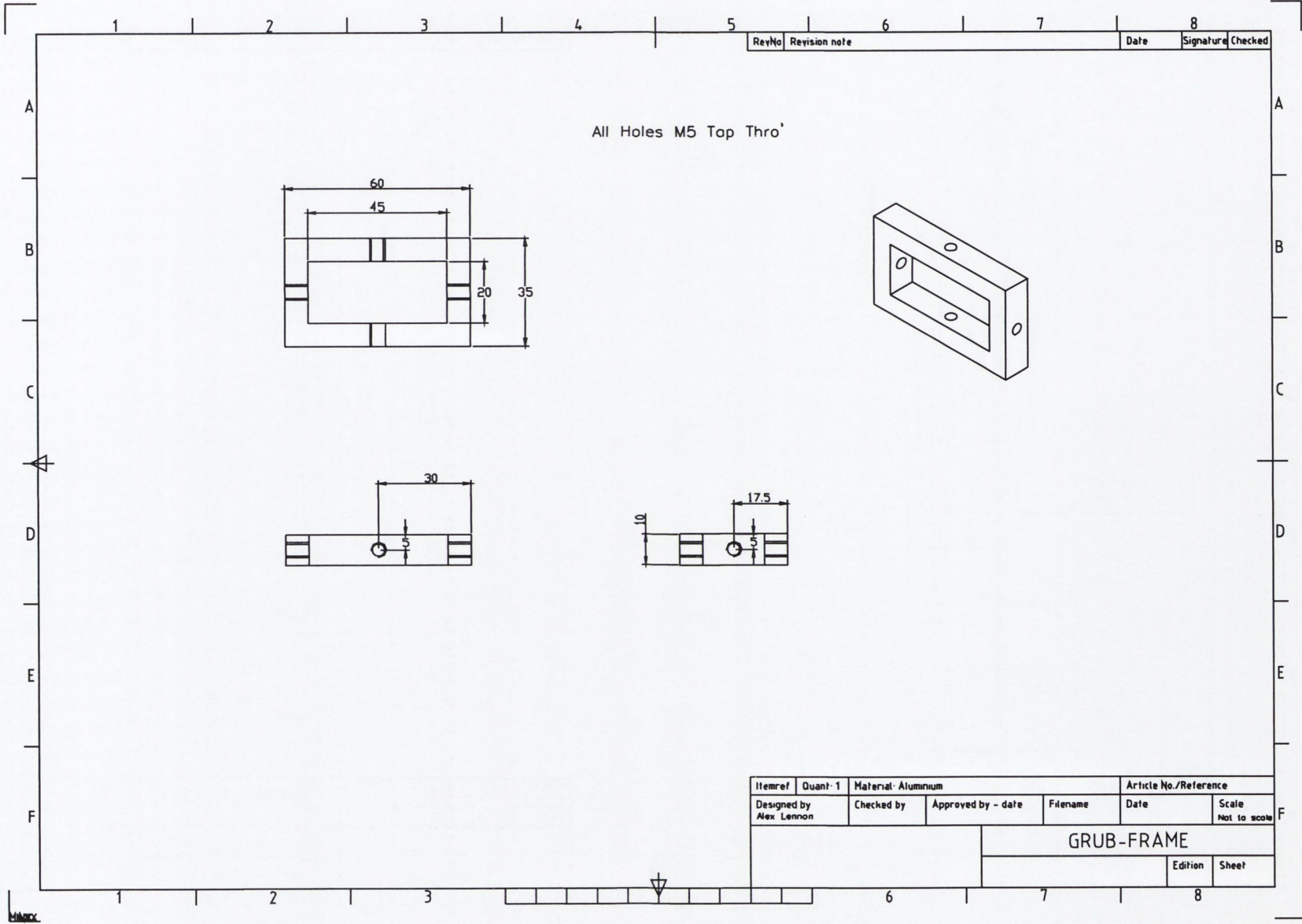
Drawings



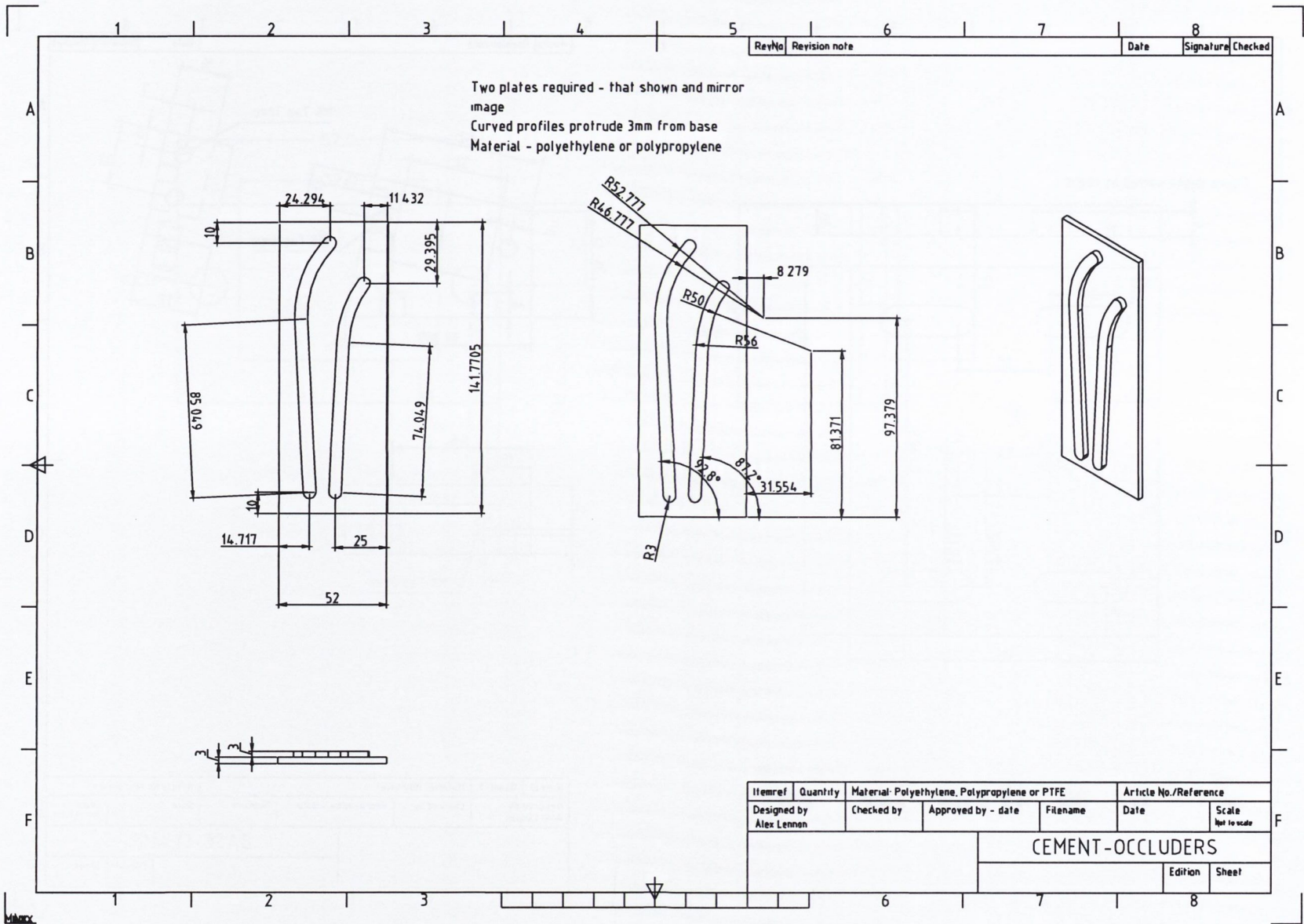
RevNo	Revision note	Date	Signature	Checked
-------	---------------	------	-----------	---------

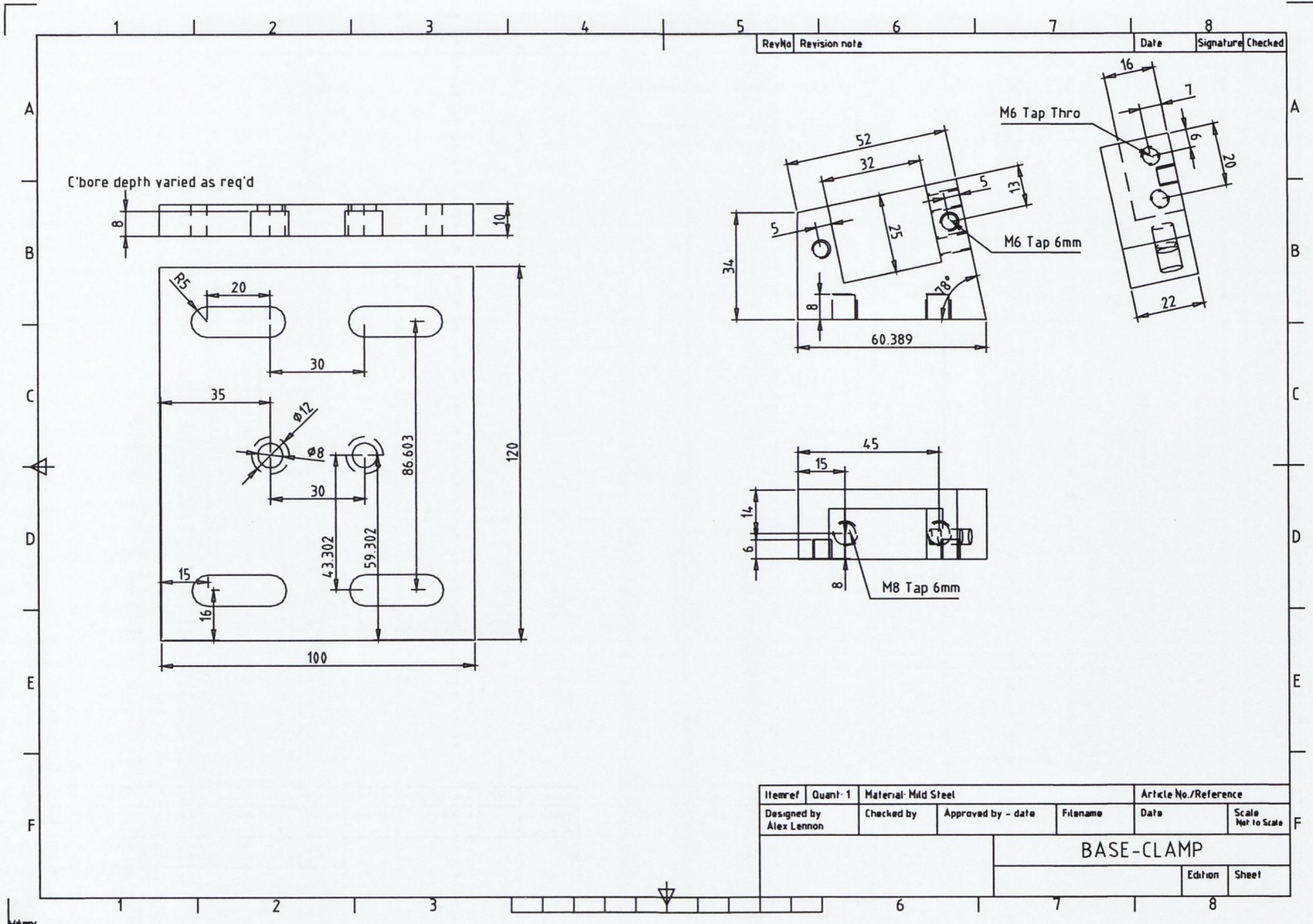
Itemref	Quant: 2	Material: Stainless Steel			Article No./Reference	
Designed by Alex Lannon	Checked by	Approved by - date	Filename	Date	Scale	
				STEM		





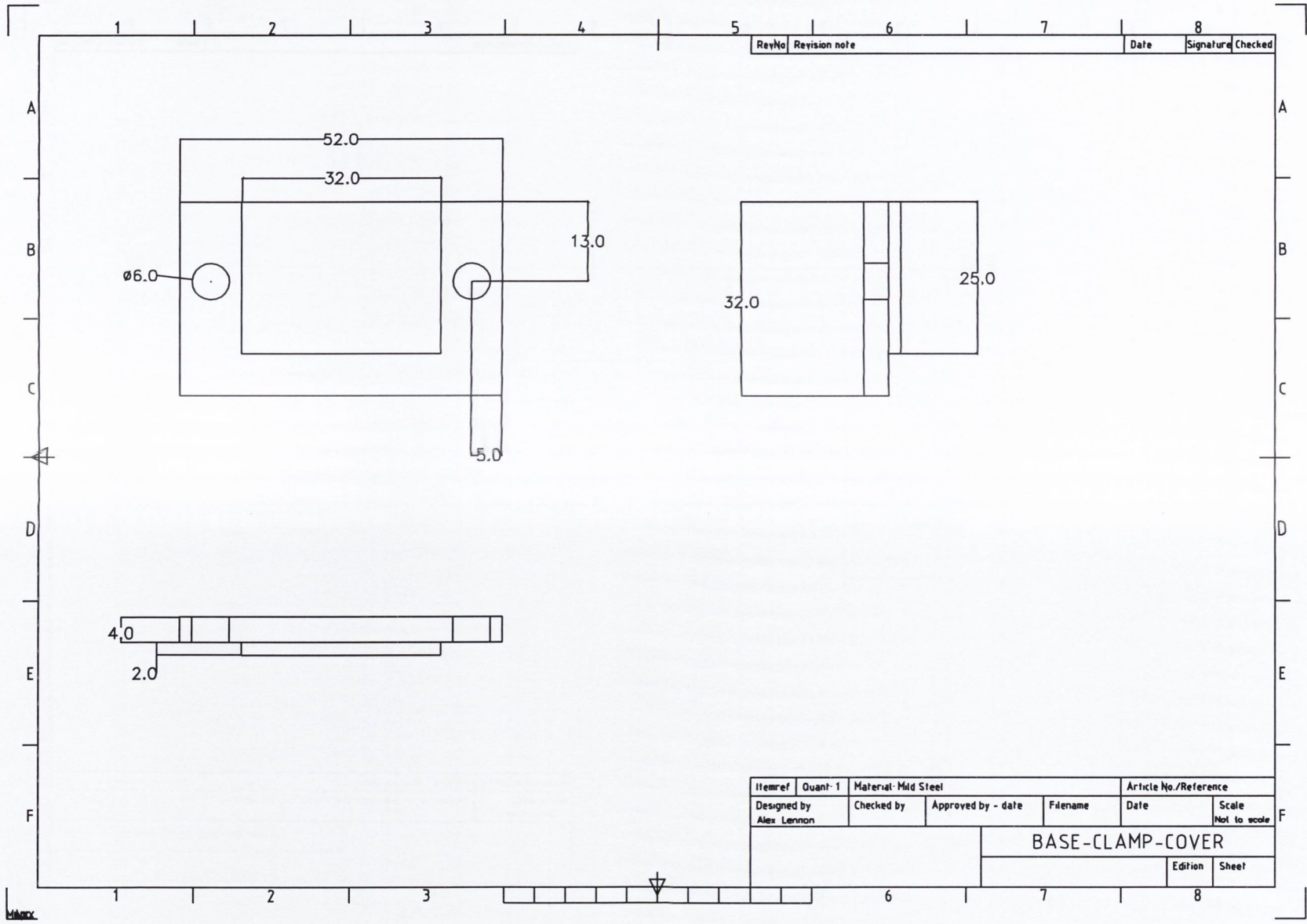
Itemref	Quant: 1	Material: Aluminum			Article No./Reference	
Designed by Alex Lennon	Checked by	Approved by - date	Filename	Date	Scale Not to scale	
				GRUB-FRAME		
				Edition	Sheet	





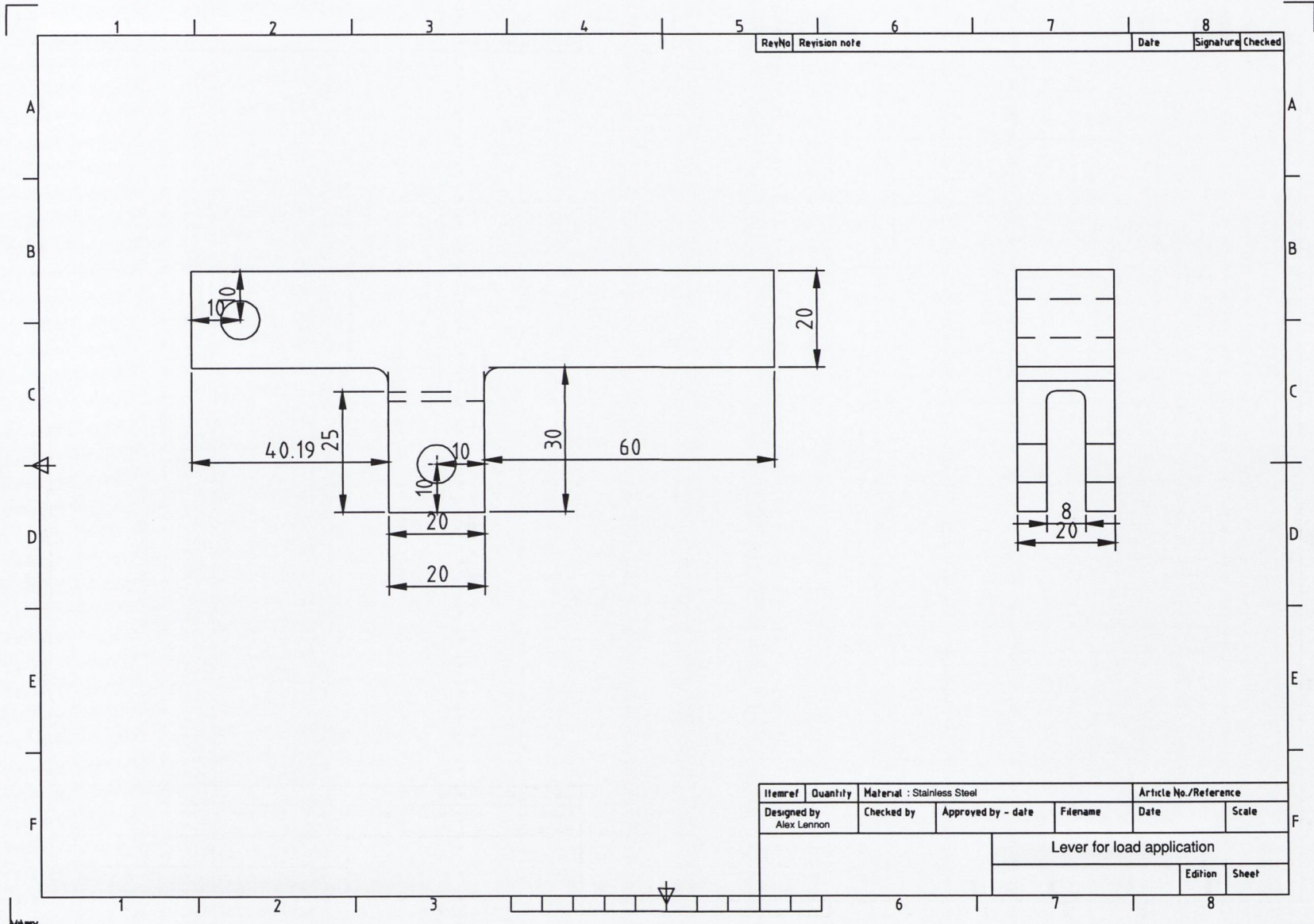
RevNo	Revision note	Date	Signature	Checked

Itemref	Quanti	Material	Article No./Reference		
	1	Mild Steel			
Designed by Alex Lennon	Checked by	Approved by - date	Filename	Date	Scale Net to Scale
			BASE-CLAMP		



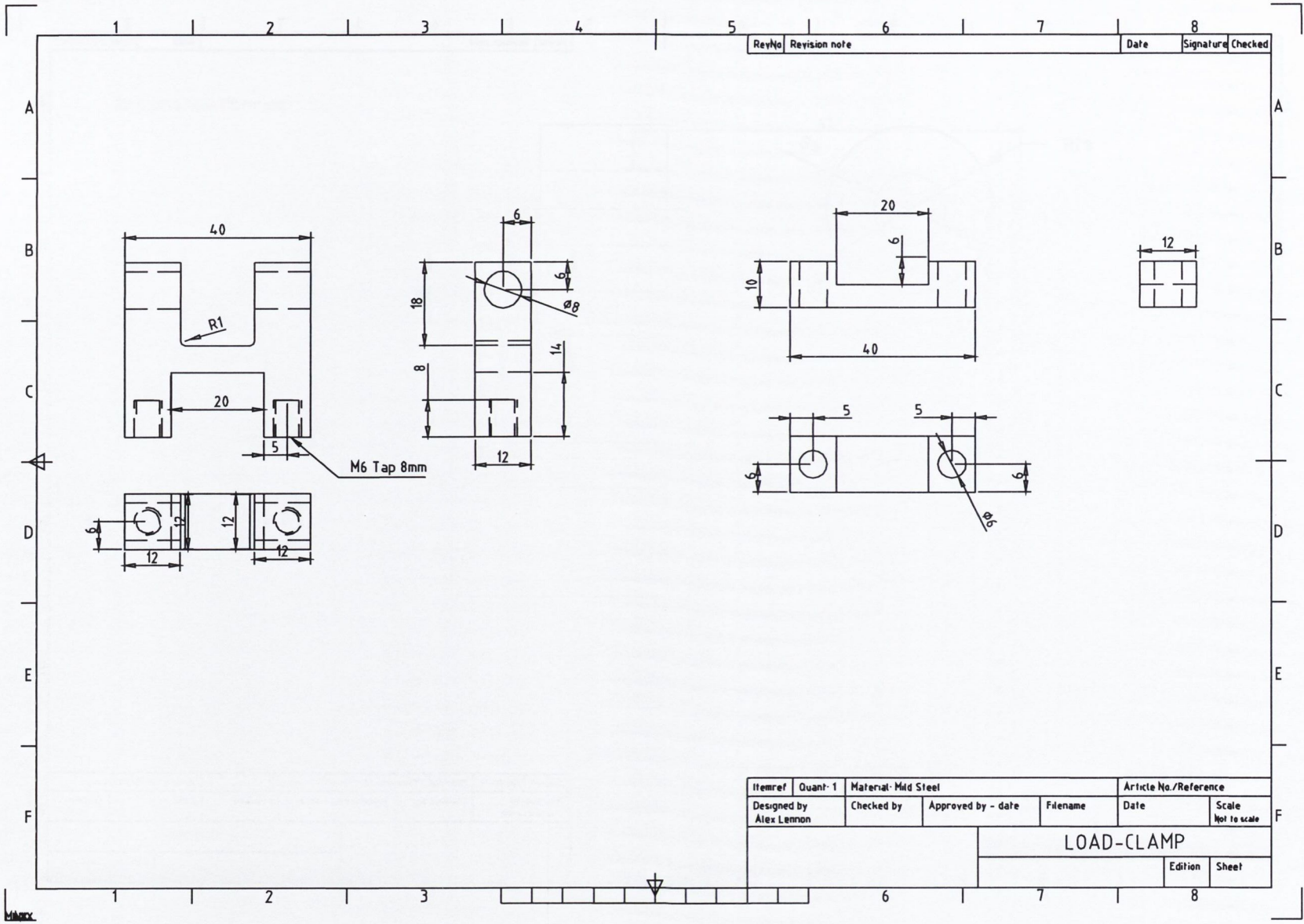
RevNo	Revision note	Date	Signature	Checked

Itemref	Quant	Material	Article No./Reference		
	1	Mid Steel			
Designed by	Checked by	Approved by - date	Filename	Date	Scale
Alex Lennon					Not to scale
			BASE-CLAMP-COVER		
				Edition	Sheet



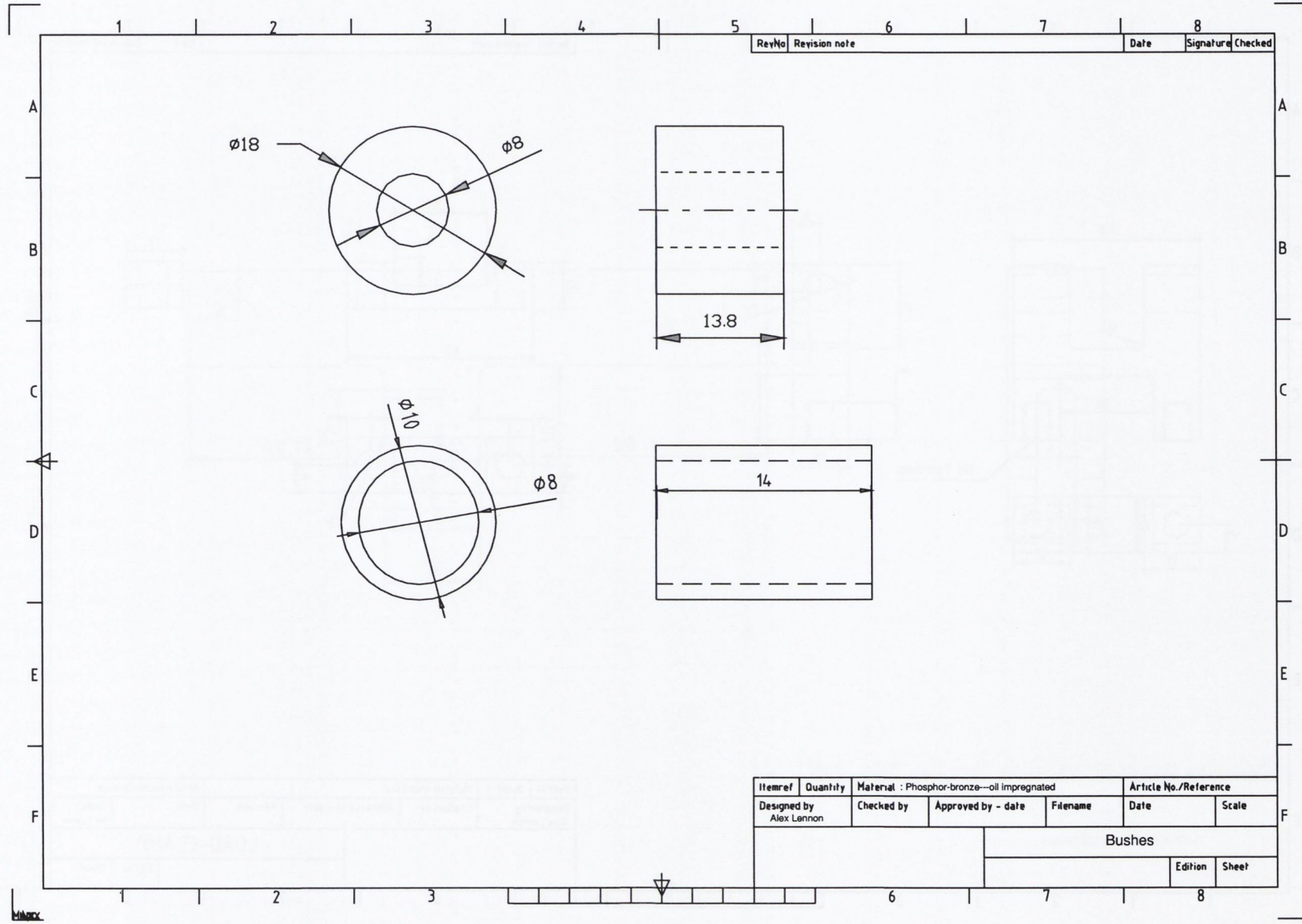
RevNo	Revision note	Date	Signature	Checked

Itemref	Quantity	Material : Stainless Steel		Article No./Reference		
Designed by Alex Lennon	Checked by	Approved by - date	Filename	Date	Scale	
			Lever for load application			
				Edition	Sheet	



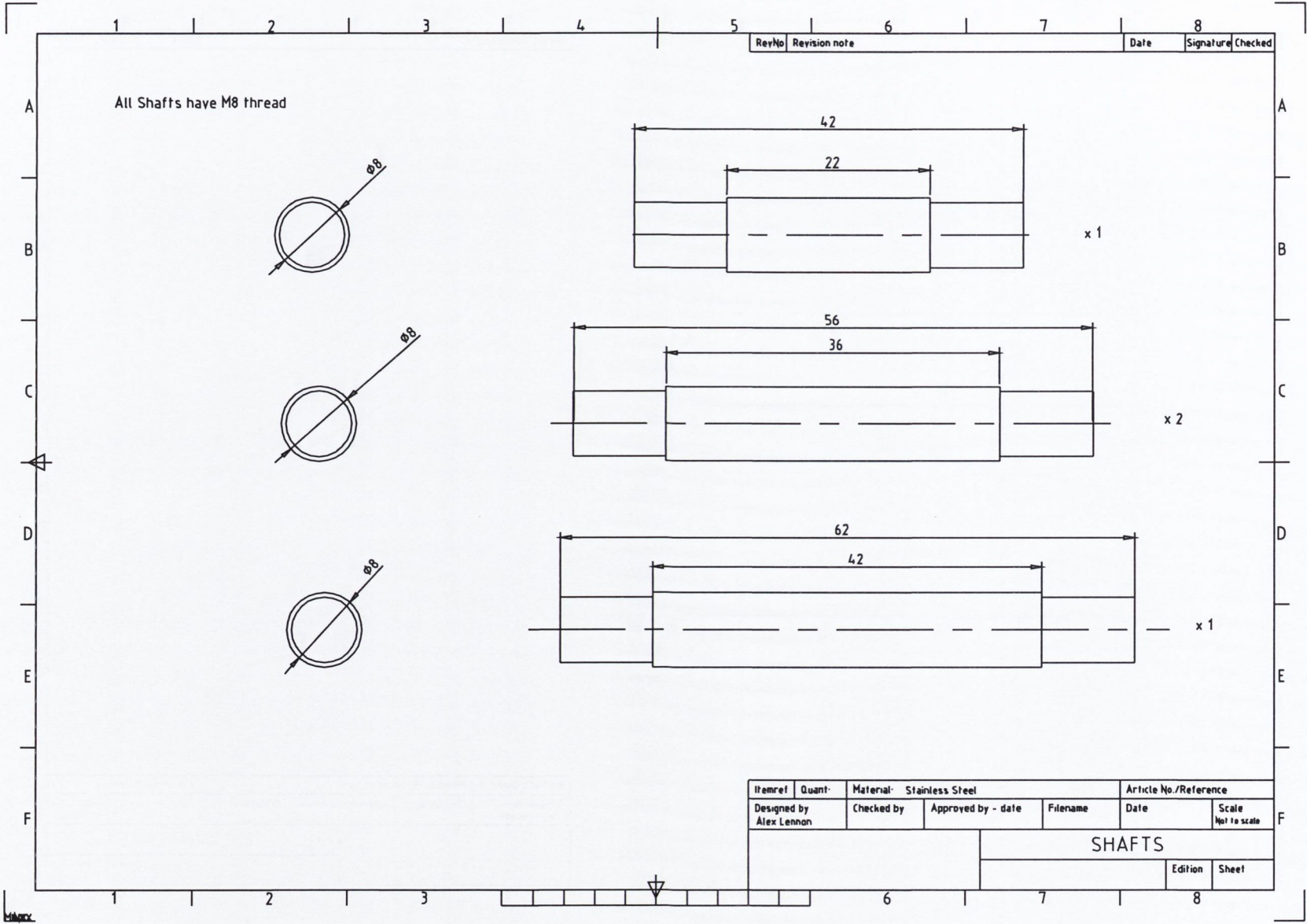
RevNo	Revision note	Date	Signature	Checked
-------	---------------	------	-----------	---------

Itemref	Quant: 1	Material: Mild Steel			Article No./Reference	
Designed by Alex Lennon	Checked by	Approved by - date	Filename	Date	Scale Not to scale	
				LOAD-CLAMP		
				Edition	Sheet	



RevNo	Revision note	Date	Signature	Checked

Itemref	Quantity	Material : Phosphor-bronze--oil impregnated			Article No./Reference	
Designed by Alex Lannon	Checked by	Approved by - date	Filename	Date	Scale	
				Bushes		
					Edition	Sheet



RevNo	Revision note	Date	Signature	Checked
-------	---------------	------	-----------	---------

Itemref	Quant	Material: Stainless Steel	Article No./Reference	
Designed by Alex Lennon	Checked by	Approved by - date	Filename	Date
			SHAFTS	
			Edition	Sheet

