Interactive three-dimensional boundary element stress analysis of components in aircraft structures

T.M. Foster[a,*], M.S. Mohamed[b], J. Trevelyan[a], G. Coates[a], S.H. Spence[c], S.K. Walker[d]

[a]*School of Engineering and Computing Sciences, Durham University, Durham, DH1 3LE, UK*
[b]*Institute for Infrastructure and Environment, Heriot-Watt University, Edinburgh, EH14 4AS, UK*
[c]*Industrial Sponsor, BAE Systems (Operations) Limited, Samlesbury Aerodrome, Balderstone, Lancashire, BB2 7LF, UK*
[d]*Industrial Sponsor, Jesmond Engineering Limited, Brough Business Centre, Skillings Lane, Brough, East Yorkshire, HU15 1EN, UK*

Abstract

Computer aided design of mechanical components is an iterative process that often involves multiple stress analysis runs; this can be time consuming and expensive. Significant efficiency improvements can be made by increasing interactivity at the conceptual design stage. One approach is through real-time re-analysis of models with continuously updating geometry. Thus each run can benefit from an existing mesh and governing boundary element matrix that are similar to the updated geometry.

For small problems, amenable to real-time analysis, re-integration accounts for the majority of the re-analysis time. This paper assesses how efficiency can be achieved during re-integration through both algorithmic and hardware based methods. For models with fewer than 10,000 degrees of freedom, the proposed algorithm performs up to five times faster than a standard integration scheme. An additional six times speed is achieved on eight cores over the serial implementation. By combining this work with previously addressed meshing and solution schemes, real-time re-analysis becomes a reality for small three-dimensional problems. Significant acceleration of larger systems is also achieved.

This work demonstrates the viability of application in the aerospace industry where rapid validation of a range of similar models is an essential tool for optimising aircraft designs.

Keywords: Boundary Element Method (BEM), Interactive, Re-analysis, Re-integration

1. Introduction

Real-time analysis of two-dimensional models is now a reality [1]. However, real-time stress analysis of three-dimensional models presents numerous additional challenges. Various schemes based on the finite element method that aim to provide interactivity have previously been presented [2, 3, 4, 5]. Meier *et al.* [6] review a range of deformable models that utilise both finite element and boundary element (BE) techniques. In this paper we address only BE implementations. The boundary element method (BEM) is a natural method to use where re-meshing is involved as changes need to be applied only to elements on the surface of the model, this enables the propagation of modifications to the mesh to be more easily contained within a small region. Wang *et al.* [7] present a BEM based scheme for interactive analysis for surgical simulation. However, this makes use of an extensive library of pre-computed solutions. In this work we aim to completely re-generate the boundary element system in real-time during each update iteration.

The primary aim of this work is to build a new stress analysis software package that is capable of accurately re-analysing three-dimensional computer models of mechanical components in real-time as the

*Corresponding author. Tel.: +44 (0)191 3342487; fax: +44 (0)191 3342408
Email addresses: timothy.foster@durham.ac.uk (T.M. Foster), m.s.mohamed@hw.ac.uk (M.S. Mohamed), jon.trevelyan@durham.ac.uk (J. Trevelyan), graham.coates@durham.ac.uk (G. Coates), stuart.spence@baesystems.com (S.H. Spence), simon.walker@jesmondengineering.com (S.K. Walker)

geometry is updated by the user. In this work, 'Real-time' refers to the re-analysis taking place within the refresh rate of the media on which the model is viewed. It is intended that this software should be used at the conceptual design stage to aid the engineer in optimising the design of components. Key to this is the identification of the interaction of feature stress concentrations. This makes it possible to design components more appropriately for their expected loading from the earliest design stages and will remove the need to make costly adjustments later in the design process if the geometry were found to be unsuitable. As the software is for conceptual purposes it should be intuitive to use and does not need to be able to handle the full range of complex geometry associated with the finished products but will often focus on sub regions of components.

The acceleration of boundary element algorithms has been the subject of a considerable volume of research over the last two decades, with most attention paid to the Fast Multipole Method (FMM) and Adaptive Cross Approximation (ACA) approaches. However, these approaches (in particular FMM) involve additional overheads that make them less attractive for the small problems that are the focus of the current work. The authors have studied ACA as a candidate approach for the real-time reanalysis, but find that it becomes cost-effective only for problem sizes larger than 5000 degrees of freedom. Although we show here that this is larger than the problem size offering real-time feedback, it is anticipated that it could very well be the most promising strategy quite soon.

To realise the goals of the current work in three dimensions, new boundary element techniques which incorporate a new re-meshing scheme and a modified re-integration algorithm, have been developed [8]. These new algorithms enable re-use of unchanged model geometry to limit propagation of changes through the mesh and thereby accelerate the reconstruction of the boundary element system of equations. The reconstruction of these equations is the subject of this paper as, for the small problems assessed in this work, the system construction time takes a significantly higher proportion of the total re-analysis time than re-solving the system; this can be seen in Figures 22 and 23. The solution is carried out using a generalised minimal residual (GMRES) solver with full approximate LU preconditioning as developed by Foster *et al.* [8].

This work is being carried out with a view to applications in aerospace. The structures assessed in the aerospace industry cover a broad range of applications such as helicopters, civil airliners, missiles, training aircraft, military transport aircraft and compact fast jets. Whilst each of these structures may be subject to significantly different loading regimes and different life requirements, the fundamental critical design features are often of a broadly similar nature and an accurate assessment of their stress concentrations is equally vitally important in calculating fatigue life.

The two-dimensional BE software Concept Analyst [1], which makes use of many of the ideas presented in this paper but with a restriction to planar problems, has been used in the aerospace industry for several years where it has been found powerful in terms of its ability to rapidly assess a number of the types of critical features commonly encountered in aerospace structures. Extensive industrial testing has been carried out on the two-dimensional BE software, comparing results with those of both published sources and of fine-mesh finite element analyses. These comparisons have shown the BE software to be accurate, reliable, and extremely fast in comparison with other techniques. However, there have remained a number of key design features that could not be analysed with the two-dimensional software due to their three-dimensional nature, including problems such as a countersunk hole and interacting fillets. The current work to extend the capability of the software into the three-dimensional domain promises to open up several of these types of features to analysis whilst maintaining the established benefits of re-analysis based rapid BE analysis seen in two dimensions.

2. The Boundary Element Method

The BEM is a standard method of analysis in the solution of partial differential equations, and is the subject of numerous texts including Becker [9]. This section contains a brief overview of the principal steps involved. We consider the problem of finding displacements and stresses in a linear elastic material comprising a domain $\Omega \subset \mathbb{R}^3$, having boundary $\partial\Omega = \Gamma$. We seek to solve the equations of linear elasticity

subject to boundary conditions:

$$u_i(q) = \bar{u}, \qquad q \in \Gamma_u \tag{1}$$

$$t_i(q) = \bar{t}, \qquad q \in \Gamma_t \tag{2}$$

where u_i and t_i are displacement and traction components, $\bar{u}$ and $\bar{t}$ are prescribed displacement and traction boundary conditions, and $\Gamma = \Gamma_u \bigcup \Gamma_t$. In practice, the use of different boundary condition types in different coordinate directions at the same location is common, so that division of Γ into separate Neumann and Dirichlet boundaries in this fashion is purely symbolic. A boundary integral equation (BIE) can be formulated for displacements at a source point, $p \in \Gamma$, due to tractions and displacements on Γ:

$$c_{ij}(p)u_i(p) + \int_\Gamma T_{ij}(p,q)u_i(q)d\Gamma(q) = \int_\Gamma U_{ij}(p,q)t_i(q)d\Gamma(q) \tag{3}$$

where $c(p)$ is a term introduced as a result of the limits applied to allow the strongly singular integral containing the traction kernel to be evaluated, so that the integral on the left hand side of equation (3) is evaluated in the Cauchy Principal Value sense. T_{ij} and U_{ij} form 3×3 matrices, $[T]$ and $[U]$, for each source-field point pairing and refer respectively to the traction and displacement kernels, given by:

$$T_{ij} = \frac{-1}{8\pi(1-\nu)r^2}\frac{\partial r}{\partial n}\left[(1-2\nu)\delta_{ij} + 3\frac{\partial r}{\partial x_i}\frac{\partial r}{\partial x_j}\right] + \frac{1-2\nu}{8\pi(1-\nu)r^2}\left[\frac{\partial r}{\partial x_j}n_i - \frac{\partial r}{\partial x_i}n_j\right] \tag{4}$$

$$U_{ij} = \frac{1}{16\pi(1-\nu)r}\left[(3-4\nu)\delta_{ij} + \frac{\partial r}{\partial x_i}\frac{\partial r}{\partial x_j}\right] \tag{5}$$

where $r = |q-p|$, n_i and n_j are components of the outward normal, n, at q, δ_{ij} is the Kronecker delta, μ is the shear modulus and ν is the Poisson's ratio of the material. The subscripts define directional components, so that T_{ij} and U_{ij} refer to a traction or displacement in the Cartesian direction i at field point q, caused by a unit load in direction j at the source point p. To solve the system numerically the boundary of the object must first be discretised into elements, forming a surface mesh. The discretised BIE can now be re-written in the local parametric element coordinates (ξ, η):

$$c_{ij}(p)u_i(p) + \sum_{elem\ e}\int_{-1}^{1}\int_{-1}^{1} T_{ij}(p,q)N_k(\xi,\eta)J(\xi,\eta)d\xi d\eta\ u_i^{ke} = \sum_{elem\ e}\int_{-1}^{1}\int_{-1}^{1} U_{ij}(p,q)N_k(\xi,\eta)J(\xi,\eta)d\xi d\eta\ t_i^{ke} \tag{6}$$

where t_i^{ke} and u_i^{ke} are the tractions and displacements acting in the i direction at node k of element e. The vector N contains the value of each shape function at the current integration point and J is the Jacobian, which transforms the differential variables at the current point from the local into the global coordinate system. The vectors u and t are now the nodal displacements and tractions respectively. For reasons of computational performance, we use the collocation form of the BEM requiring collocation of equation (6) at a sufficient number of points, p, that for convenience coincide with the nodal positions. Performing the integrations given in equation (6) a system of equations can be derived. These are given in matrix form as:

$$[H]\{u\} = [G]\{t\} \tag{7}$$

where $[H]$ and $[G]$ contain terms resulting from the evaluation of the double integrals in (6). After applying boundary conditions, equation (7) can be rewritten in the form:

$$[A]\{x\} = \{b\} \tag{8}$$

This system can now be solved as a set of linear equations to find the unknown tractions and displacements contained in $\{x\}$. Internal stresses may be found by declaring p at the point of interest, substituting the now fully defined values of displacement and traction at the nodes into equation (3) and summing boundary integrals to yield $u(p)$.

3. Integration strategy

3.1. Initial integration

To improve the efficiency of boundary element integration, the literature suggests customising the integration scheme across each element depending on the distance between the field element, e, and the current source node, p, r [10, 11, 12]. If r is large the variation in the stresses across e caused by a load at p will be small and hence a low order integration scheme can be applied. As r decreases, higher order schemes must be introduced, with specialist schemes used to deal with singular and near-singular integrals, where p lies on e. As the scheme is specific to the current element-node pairing, the geometric data associated with the element: shape functions, Gauss point locations, weights and normals, must be re-computed for every node in the model. Integration can then be carried out and the $[H]$ and $[G]$ sub-matrices corresponding to the current element-node pairing constructed. With the application of boundary conditions these are assembled to form the $[A]$ matrix and $\{b\}$ vector given in (8). The strongly singular terms on the diagonal of $[A]$ taken from $[H]$ are calculated by applying rigid body motion, the weakly singular terms from $[G]$ are already computed to sufficient accuracy using the singular integration scheme. The complete system is then passed to the linear solver.

The number of Gauss points, m, required to integrate over each dimension of an element is a function of the size of the element, L, and the minimum distance between the element and the source node, R. In three dimensions, L is the maximum dimension of the element in the ξ and η directions and m must be computed independently for each dimension, producing a grid of Gauss points.

Lachat and Watson [10] developed an algorithm for finding m for quadrilateral elements. This was later generalised by Mustoe [11]. The original algorithm uses an iterative scheme to find the maximum ratio R/L; Gao and Davies [12] propose the following approximation to improve efficiency:

$$m = \frac{p' \ln\left(\frac{e}{2}\right)}{2 \ln\left(\frac{L}{4R}\right)} \tag{9}$$

where e is the prescribed tolerance of the relative integration error and

$$p' = \sqrt{\frac{2}{3}p + \frac{2}{5}} \tag{10}$$

where p is the order of the singularity. The number of Gauss points suggested by the scheme is plotted in Figure 1. It should be noted that if $R/L < 0.25$ the integration order grows to infinity. If this is the case the element must be subdivided to ensure accurate integration.

An alternative scheme, based on numerical tests for surface integrals, is proposed by Bu and Davies [13]. This is generalised by Gao and Davies [12] and can be summarised as:

$$m = -0.1p' \ln\left(\frac{e}{2}\right)\left(\left(\frac{8L}{3R}\right)^{\frac{3}{4}} + 1\right) \tag{11}$$

In the current work a reusable intrinsic sample point (RISP) integration scheme is applied [14]. The RISP algorithm uses a discrete number of integration schemes, therefore the shape functions and derivatives for each integration scheme need only be calculated once for each element type.

In the current work, R is taken to be the distance from the centroid of the field element to the source node and L as the average side length of the element. Equations (9) and (11) can be rearranged to find the maximum ratio R/L for a given number of Gauss points. From (9):

$$\frac{R}{L} = \frac{1}{4}\left(\frac{e}{2}\right)^{-\frac{p'}{2m}} \tag{12}$$

From (11):

$$\frac{R}{L} = \frac{8}{3}\left(\frac{-10m}{p' - \ln(e/2)} - 1\right)^{-\frac{4}{3}} \tag{13}$$

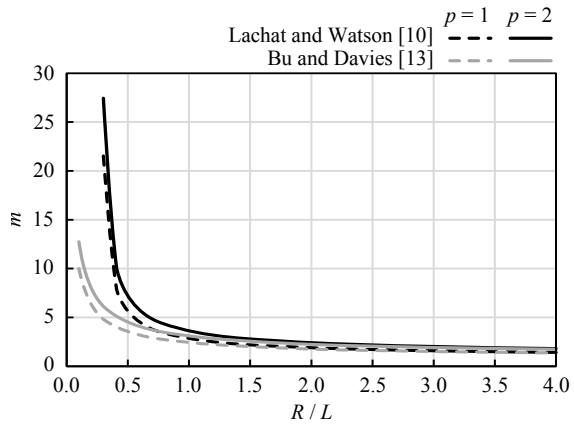

Figure 1: Integration order.

This ratio can be used in conjunction with the RISP integration scheme to directly determine the appropriate number of Gauss points to use across the entire element. If e is taken to be 0.001, this allows Table 1 to be constructed, where $\hat{m}$ is the total number of Gauss points on the element. This scheme can now be applied to both quadrilateral and triangular elements. If the source node lies on the field element the integral is singular and is dealt with using the scheme suggested by Kane [14].

Table 1: Total number of Gauss points, $\hat{m}$, for $e = 0.001$.

Condition		$\hat{m}$	
Lachat and Watson	Bu and Davies	Tri	Quad
Singularity at corner node		32	64
Singularity at mid-side node		36	96
$R/L < 1.6$	$R/L < 1.9$	13	16
$1.6 \leq R/L < 3.3$	$1.9 \leq R/L < 3.7$	7	9
$3.3 \leq R/L < 17.5$	$3.7 \leq R/L < 13.9$	4	4
$17.5 \leq R/L$	$13.9 \leq R/L$	1	1

Using a discrete number of integration schemes leads to a reduction in memory requirements and, as the models encountered in this work generally contain fewer than 2000 elements, it is possible to store all the geometric data associated with the integration in memory. The Gauss point locations, associated normals and Jacobians for all integration schemes are therefore only computed once for each element in the model. This data is stored so that, where possible it can be re-used in the re-analysis. The singular integration schemes are formulated such that they can be applied using the same algorithm as the standard integrals.

It is not normally necessary to explicitly calculate the full $[H]$ and $[G]$ matrices; the linear system (8) may be constructed directly. However, in the implementation proposed in this work, the full $[H]$ and $[G]$ sub-matrices associated with each element-node pairing are stored so that, where possible, they may be re-used to reconstruct $[A]$ and $\{b\}$ when the model is updated. As the $[H]$ and $[G]$ sub-matrices overlap when assembled into $[A]$ and $\{b\}$, it is not practicable to store only $[A]$ and $\{b\}$.

3.2. Suppression of integrals

The majority of the traction or displacement boundary conditions applied to a model have zero magnitude. These will appear as zeros in the corresponding cells in the $\{t\}$ and $\{u\}$ vectors found in equation (7). During construction of equation (8), these zero values multiply some of the integral equations. It is therefore not necessary to compute these integrals, thereby reducing the total integration time.

3.3. Re-integration

The re-meshing algorithm used in this work aims to limit propagation of geometric changes through the mesh to those areas close to the updated geometry. This means that large parts of the mathematical system are unchanged. Re-integration of the system can therefore be accelerated by re-using any unchanged $[H]$ and $[G]$ sub-matrices. This is done by following the flowchart given in Figure 2.

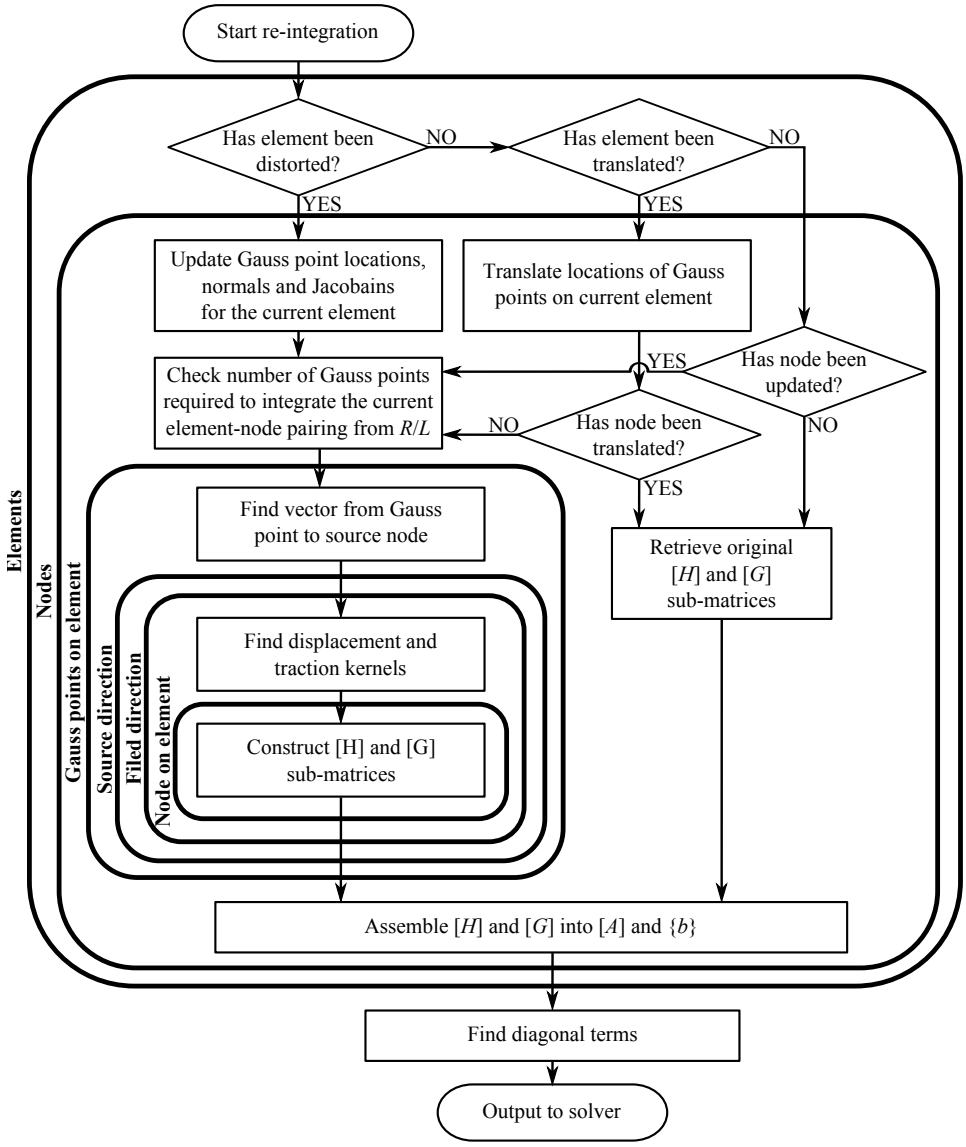

Figure 2: Flowchart of re-integration algorithm.

Each updated element may be be treated in one of two different ways: it may be translated or distorted. Where the source node and field element have the same translation applied, the associated $[H]$ and $[G]$ sub-matrices will not require updating. However, if an element is classified as distorted, all the associated geometric data and sub-matrices will require recalculation. At present rotated elements are classed as distorted. However, if a group of elements are rotated together, the associated $[H]$ and $[G]$ sub-matrices will not change, as with the translation case.

Where elements have been distorted, the ratio R/L will have changed and it may be necessary to update

the integration scheme. In Section 5.3 the merits of maintaining the same integration scheme for all model updates or updating the scheme as the model is distorted will be discussed. An additional refinement of the integration scheme is also introduced, whereby elements of poor quality, Q, will be re-integrated using a higher order integration scheme to reduce the potential error associated with the distortion. This scheme will be applied according to Table 2. A full description of the quality measures used in this work can be found in [8].

Table 2: Integration refinement scheme for poor quality elements.

	Current $\hat{m}$ (Triangular/Quadrilateral)			
	1/1	4/4	7/9	13/16
$Q > 0.4$	1/1	4/4	7/9	13/16
$0.4 \geq Q > 0.3$	4/4	7/9	13/16	13/16
$0.3 \geq Q > 0.2$	7/9	13/16	13/16	13/16
$0.2 \geq Q$	13/16	13/16	13/16	13/16

It should be noted that the outer pair of loops in Figure 2 are interchangeable. The differences between carrying out integration by looping primarily over the field elements or the source nodes are discussed in section 5.5.

For models undergoing more severe levels of distortion it will become difficult to maintain accuracy using a constant model size. In such cases a full re-mesh may be performed from time to time as suggested by an error indicator. Such a scheme, and fuller details of the re-meshing algorithm used by the authors, are available in [8].

4. Computational strategy

This section covers areas of the implementation that are not affected by the algorithm structure but are influenced by the computer architecture.

4.1. Precision

Double precision computing allows more precise storage and computation of data. However, it requires twice as much memory and more processor time than single precision. It is generally accepted, and often assumed, that double precision is needed for accurate BE computation. However, it has not been possible to find research that validates this fact. For this reason both single and double precision implementations of the algorithm will be assessed in this work.

4.2. Architecture

In this work we use the term system architecture to refer to the width of the data path to the processor and hence the length of the word that can be used to address the computer memory. Longer words allow more memory locations to be addressed. There are two primary processor architectures in use: 64- and 32-bit. The 32-bit architecture only allows access to 4 gigabytes of computer memory; by utilising 64-bits this can be extended to 16×10^9 gigabytes. However, with 64-bit architecture the same data occupies more space in memory than with a 32-bit architecture due to the longer pointers and additional alignment padding. This can also have implications for efficient utilisation of the processor cache, which could require more memory read/writes. If more than 4GB of memory is required a 64-bit system must be used.

In this work both 64- and 32-bit implementations will be compared for models that are small enough to use a 32-bit architecture.

4.3. Parallelisation

The first algorithm for parallelisation of the BEM appeared in 1984 in a work by Symm [15]. Davies [16] comprehensively reviews the early work on parallelising the method. Many authors, including Kane [17] and Baltz and Ingber [18], use block partitioning to distribute different parts of the $[A]$ matrix over multiple processors. However the overlap between these blocks, where elements share a node, adds an extra level of complexity. Erhart [19] and Kamiya *et al.* [20, 21] use domain decomposition to solve the BE problem. This is easily parallelised as each domain can be allocated to a different processor. However, additional overheads are incurred in assessing the performance of each processor for optimal distribution of the computation. Smith and Margetts [22] developed an element-by-element parallelisation strategy, in which boundary element matrices were distributed over multiple processors and used directly in the iterative solver BiCGStab(l), precluding the need to store and reduce a large linear system.

The majority of the literature associated with parallelising the BEM deals with applying the method across multiple distributed memory devices. A typical aerospace design office contains a large number of high specification, multi-core PCs. It is therefore logical to exploit this processing capability that enables analysts to carry out analyses on their own machine. In this work we therefore parallelise the method on a single shared memory machine. This is a simple process, aided by the way the integration algorithm described in this work has been constructed.

If the outer loop of the integration scheme is over the elements, as described in Figure 2, and the algorithm is parallelised by sending each element to a different processor, assembly of $[A]$ and $\{b\}$ must be carried out after the integration has been completed. As the full $[H]$ and $[G]$ matrices are being stored for re-use during each subsequent re-analysis this is a trivial matter. The assembly of $[A]$ and $\{b\}$ may be independently parallelised by generating each row of the system on a different processor. If the outer loop of the algorithm is over the nodes, the construction of $[A]$ and $\{b\}$ may be carried out inside the main loop. It will be assessed as to whether this is beneficial for the computational efficiency of the algorithm.

5. Assessment of the acceleration techniques

This section contains an overview of the tests carried out on the different analysis acceleration techniques. Each technique is applied individually to the basic re-integration scheme so that the merits of applying each scheme can be independently assessed. The most beneficial schemes are then combined for the final set of tests.

This section is organised as follows: Section 5.1 introduces the test model. Section 5.2 discusses using the same integration scheme over the entire model before a variable integration scheme is introduced in Section 5.3. The benefits of suppressing integrals which multiply zero values are summarised in Section 5.4. Section 5.5 contains a brief discussion on applying the outer loop over either the nodes or the elements. The effect of using single instead of double precision variables is discussed in Section 5.6 and the merits of using 64-bit computing are considered in Section 5.7. The findings of all these tests are summarised in Section 5.8 and the best combination of schemes is discussed. These are then combined before the parallel implementation of the algorithm is assessed in Section 5.9.

5.1. Test models

To initially assess the speed and accuracy of the proposed solvers, three test models have been considered to cover a range of stress conditions:

- Thick walled cylinder with an internal pressure (TWC). The internal radius has been reduced as the number of updates applied to the model, i, increases.
- Cantilever under bending (CTL). The length of the cantilever has been extended as i increases.
- Plate with a hole under uniaxial tension (PWH). This gives a more complex stress field than the uniform field found in the thick walled cylinder. The hole has been moved along the plate as i increases.

Each model has been analysed using several different meshes with varying degrees of refinement and different step sizes of geometric perturbation. The results given throughout this assessment summarise the results of analysis carried out over all of these test cases.

Throughout this paper, $\hat{m}$ is the total number of Gauss points across each element, n is the total number of nodes and n_u the number of updated nodes. The speed up factor, S, is calculated from $S = t/t_u$ where t is the initial integration time and t_u is the time required to update the parts of the system that have changed. An error measure, ε_x, is used to compare the accuracy of the scheme to the benchmark:

$$\varepsilon_x = \frac{\|\{x\} - \{\hat{x}\}\|}{\|\{\hat{x}\}\|} \tag{14}$$

where $\|\cdot\|$ denotes the L_2-norm, $\{\hat{x}\}$ the benchmark solution vector and $\{x\}$ the BEM approximation to $\{\hat{x}\}$.

With the exception of Section 5.2, the benchmark models to which each acceleration scheme has been independently applied use the re-integration algorithm described in Figure 2. This is applied with 13 Gauss points for all non-singular integrals across triangular elements and 16 Gauss points for all non-singular integrals across quadrilateral elements. For singular integrals the scheme given in Section 3.1 is applied. The algorithm is implemented using double precision on a single processor with a 32-bit architecture. This set of conditions has been chosen so that ε_x reflects the increase in error of applying the chosen acceleration scheme.

In Section 5.2, the benchmark models use a refined version of each mesh where each element has been divided into four parts. This is used with $\hat{m} = 28$ for standard integrals and the normal singular integration scheme to ensure a high degree of accuracy in $\{\hat{x}\}$ so that the effect of reducing $\hat{m}$ can be accurately assessed.

The algorithms discussed in this paper have been written and compiled using Visual C++ 2010, with the compiler options optimised for speed. The timings given in the following sections are for a single shared memory machine utilising Intel Xeon X5570 2.9GHz processors. Parallelisation has been carried out using the OpenMP libraries.

5.2. Re-integration

Figure 3 summarises the error, ε_x, if the same number of Gauss points, $\hat{m}$, is used to integrate over each element to evaluate all regular boundary integrals. In Figure 3, $\hat{m}$ refers to the number of Gauss points used to integrate over each triangular element.

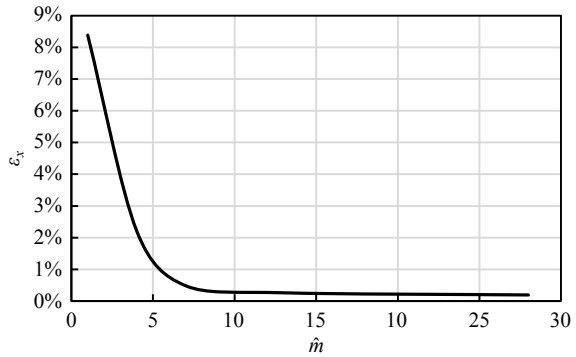

Figure 3: Accuracy using a fixed integration scheme across all elements.

Figure 3 shows that the error increases rapidly if fewer than 5 Gauss points are used across each element. Less than 0.3% reduction in the error can be achieved by using more than 7 Gauss points. As the system construction time increases quadratically with the number of Gauss points this indicates that, for maximum efficiency whilst retaining a suitable accuracy for a fixed integration scheme, the standard 7 Gauss point scheme should be applied to all triangular elements; this corresponds to a 9 point scheme across quadrilateral elements. The similarity between the shape of the curves in Figures 3 and 1 should be noted.

If only the updated node-elements pairings are re-integrated during re-analysis the speed up achieved is shown in Figure 4. Due to the overheads involved the speed up factor, $S < 1$ if almost all of the elements in the model were updated. However, the re-meshing algorithm will not permit this many elements to be updated in one iteration.

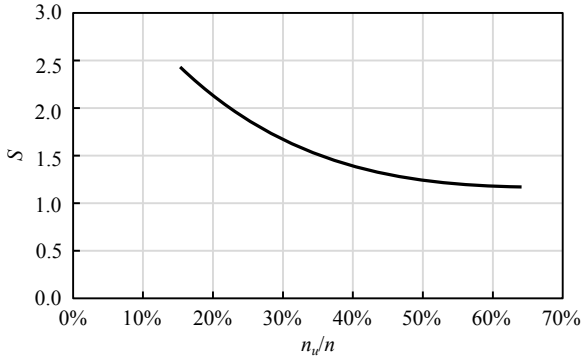

Figure 4: Speed up factor achieved using partial re-integration

5.3. Integration scheme

Figure 5 compares the speed up, S, achieved using the Bu and Davies [13] and Lachat and Watson [10] integration schemes. The solid lines show the effect of updating the integration scheme, based on R/L, across each element. The dashed lines show the effect of re-applying the scheme calculated during the initial analysis. The dotted lines show the effect of refining the integration scheme based on both R/L and the quality, Q, of the element. Q has been calculated using the circle ratio defined by Foster *et al.* [8].

The Bu and Davies [13] scheme typically runs around 3% faster than the Lachat and Watson [10] scheme for problems where $n > 1000$ as it uses fewer Gauss points. This coarser scheme leads to a small increase of 0.01% in the error, ε_x. For problems where $n < 1000$ the Lachat and Watson scheme performs fastest. This is because R/L is generally small in small models and very few elements are integrated using a single integration point. The main speed advantage in Bu and Davies' scheme is gained when $R/L > 13.9$ and this is normally lost when $n < 1000$.

If the integration scheme is not updated, an additional speed up can be achieved. However, this is negligible and is at the expense of a much greater variability in the error, ε_x, especially as the change to the geometry becomes large. For this reason $\hat{m}$ should be updated periodically, if not after every update iteration.

Using a higher order integration scheme over poor quality elements leads to a reduction in the error of fewer 0.01%. This is at the expense of a 4% decrease in S.

The Bu and Davies scheme has been adopted for use in the integration algorithm. This has been implemented without adjustment for poor quality elements and $\hat{m}$ is updated during each update iteration. Using this scheme the accuracy remains comparable to using a 13 Gauss point scheme across every element whilst the solve time is reduced to that of using 4 Gauss points.

5.4. Suppression of integrals

The speed up, S, achieved through suppressing calculation of integrals that are multiplied by zero traction boundary conditions is summarised in Figure 6. It should be noted that the integrals that multiply zero displacement boundary conditions are not suppressed, as these values are required in the rigid body motion summation which is used to calculate the diagonals of $[H]$. Good speed up is achieved, especially where n_u/n is small.

In Figure 6 it appears that if $n_u/n > 70\%$, $S < 1$. Due to the algorithm's implementation this can never occur so $S \not< 1$.

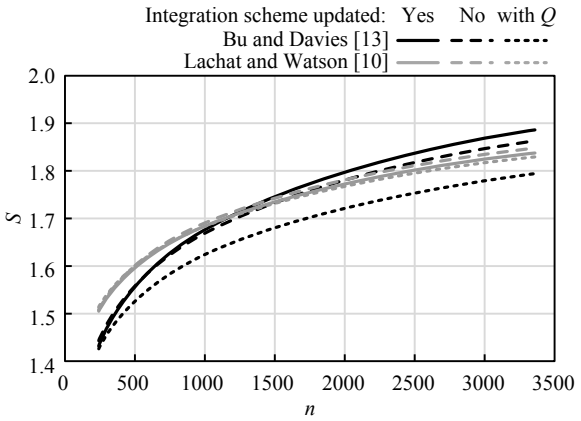

Figure 5: Speed up factor of variable integration schemes.

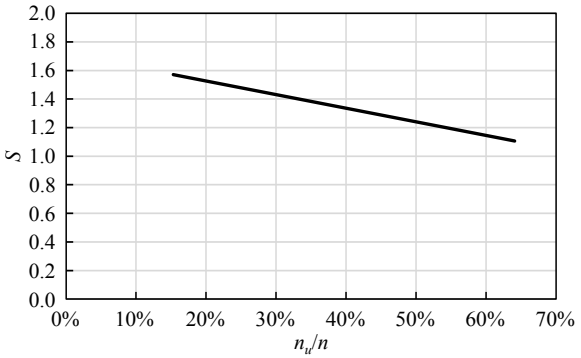

Figure 6: Speed up through suppression of integrals.

5.5. Outer integration loop

For the majority of models, $S \approx 1.05$ if the outer loop of the integration algorithm is taken over the nodes instead of over the elements although, for small models ($n < 500$), S can be as much as 1.35. However, if the Bu and Davies integration scheme is used with the outer integration loop over the nodes it actually performs, on average, 20% faster than integrating primarily over the elements in conjunction with the same scheme. Node order integration also parallelises more efficiently than element order integration. An outer integration loop over the nodes has therefore been adopted for use in the accelerated algorithm.

5.6. Precision

Using single precision variables is on average 20% faster than using double precision variables. However, the size of the L_2-norm error in $\{x\}$, ε_x, associated with the reduction in precision increases exponentially with the number of nodes in the model as shown in Figure 7. For the sizes of model encountered in this work single precision accuracy is sufficient. However, due to the exponential growth, double precision should be applied to larger problems. From Figure 7 the following formula can be constructed:

$$n \leq \frac{\ln(\bar{\varepsilon}_x) + 11.33}{0.0007} \tag{15}$$

where $\bar{\varepsilon}_x$ is the prescribed mean accuracy and n is the maximum number of nodes in the model. If this relationship is satisfied then $\varepsilon_x \leq \bar{\varepsilon}_x$.

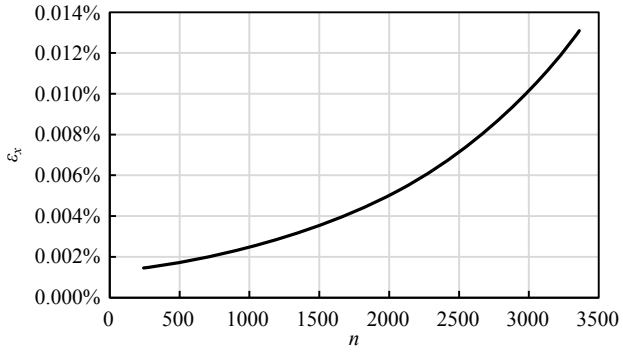

Figure 7: Increase in error using single precision.

It should be noted that using all single precision values in an iterative solver routine can reduce the convergence rate or prevent convergence. However, for the GMRES algorithm employed in this work, the solution accuracy is maintained and, if double precision variables are retained in the computation of matrix vector products, the reduction in convergence rate can be overcome.

5.7. Architecture

Figure 8 shows the speed up, S, achieved when using 64-bit processing instead of 32-bit processing. If $n > 900$, $S = 1.3$. However, for problems where $n < 450$, 32-bit processing is faster. Using the algorithm proposed in this work, models where $n > 1800$ must be computed using 64-bit processing as they require more than the maximum amount of memory that can be addressed using 32-bit processing. As $n > 450$ in the majority of problems, a 64-bit system architecture should be used wherever possible although some computer systems may only support 32-bits.

5.8. Summary

So far in this paper the acceleration schemes have been independently applied to the re-integration algorithm. This section combines them into a single algorithm.

Figures 9 and 10 summarise the results of applying each scheme independently. The vertical bars show the range of the data, the boxes the upper and lower quartiles and the bars the mean value of the speed

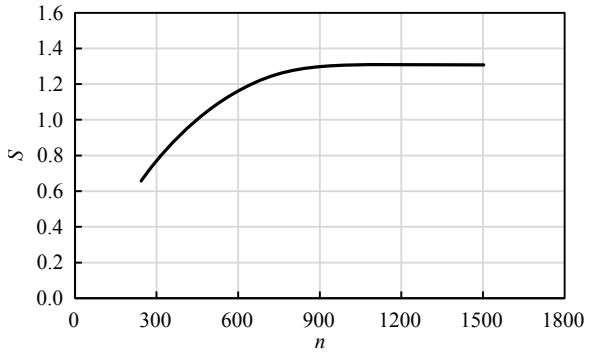

Figure 8: Speed up achieved using 64-bit over 32-bit computing.

up, S, or the error, ε_x. The reader is reminded that ε_x is the increase in error resulting from application of the acceleration schemes and not the absolute error. Figures 9 and 10 have been constructed using data gathered from a range of different models with different numbers of degrees of freedom. It should be noted that all significant relationships relating to changes in S and ε_x have already been portrayed in this work, the remaining speed gains or changes in error related to the different acceleration schemes are approximately constant across all models and are within the ranges shown in Figures 9 and 10. It should be noted that Figure 10 has been cropped at 0.14% to show the results clearly. The peak values not shown in Figure 10 are between 3.0% and 4.5%. However, as demonstrated by the low mean value these peak values occur in very few cases.

In Figure 9 the field labeled 'Re-integration' shows S when applying the partial re-integration scheme as opposed to carrying out a full re-integration over the entire model with $\hat{m}$ fixed at 13 or 16 for triangular and quadrilateral elements respectively. The speed gain for the rest of the fields is the additional acceleration that can be achieved in addition to using this scheme. The field labelled 'All' summarises S when applying all of the selected improvements (suppressed integrals; 64-bit architecture; node order integration; single precision; Bu and Davies RISP integration scheme; updating $\hat{m}$ with each re-analysis).

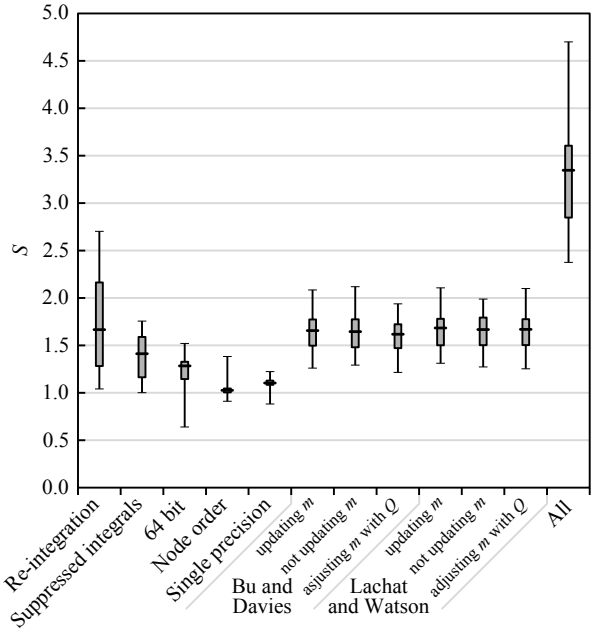

Figure 9: Summary of speed up.

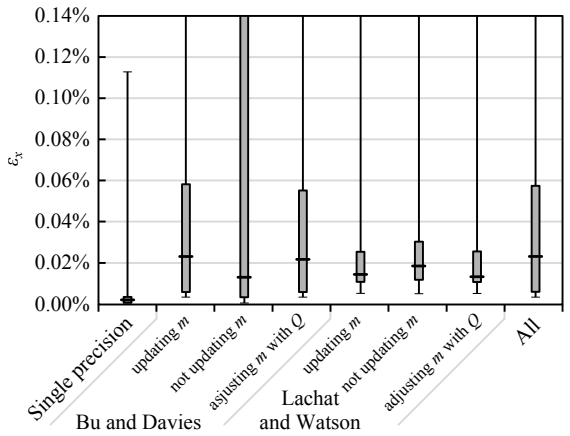

Figure 10: Errors associated with acceleration schemes.

Figure 11 shows t_u for the test cases described in Section 5.1 for the re-integration algorithm and acceleration techniques proposed in this work and the scheme used by Foster *et al.* in [8] which uses the same number of Gauss points for all integrations but does limit re-integration to updated elements only. The proposed algorithm is significantly faster, providing a typical speed up of $S = 5$.

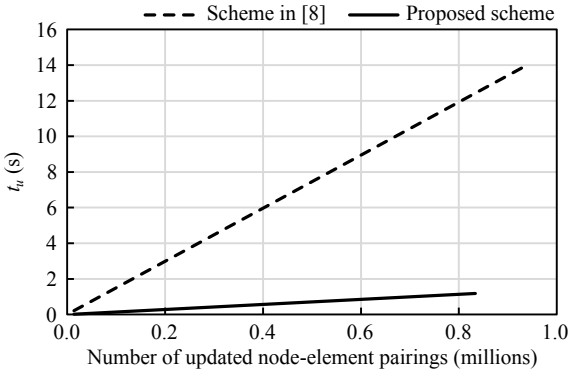

Figure 11: Absolute update time.

The percentage of time saved in the re-integration, t_u/t, is plotted in Figure 12 against the percentage of updated nodes, n_u/n, where n is the total number of nodes. This shows that the proposed scheme improves on the speed up achieved by Foster *et al.* [8]. The effect of overheads that resulted in $t_u > t$ when $n_u/n > 0.63$ in the Foster *et al.* [8] scheme has also been reduced. However the overheads still exist, which will result in $t_u > t$ as $n_u/n \to 1$.

5.9. Parallelisation

The average speed up, S, achieved on multiple processors is shown in Figure 13. For the re-integration, S is 15% smaller than for the initial integration. This is due to the overheads associated with parallelisation having a greater impact on the faster re-integration algorithm. Both schemes show good scalability which is independent of the percentage of the system that is updated.

6. Application to industrial problems

6.1. Case study

Two models are introduced here to demonstrate the benefits of the acceleration techniques used in this work when applied in an industrial context. These are a countersunk hole, shown in Figure 14, and a section

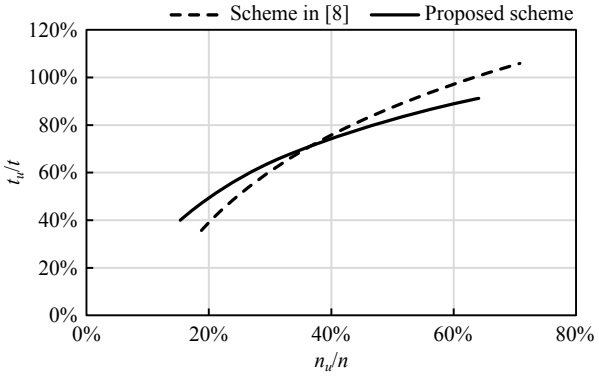

Figure 12: Normalised update time.

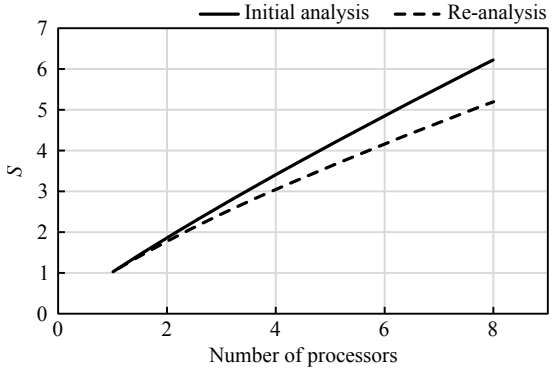

Figure 13: Integration speed gain for multiple processors.

from an aircraft rib shown in Figure 15.

Only a quarter of the countersunk hole has been meshed and analysed to reduce the complexity of the simulation as shown in Figure 14. Roller boundary conditions are applied to the planes of symmetry which bisect the hole and to the base of the model. A nominal load of 10MPa is applied as indicated. The countersunk hole has been modelled using 540 elements with 3066 degrees of freedom. For this model we consider the effect of changing the depth, d, of the countersink. Initially $d = 3$mm, the depth is then incrementally increased to establish how this affects the stress distribution.

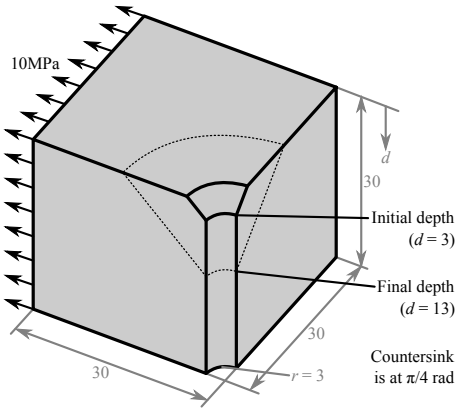

Figure 14: Case study 1: Countersunk hole. (Dimensions in mm)

A model of a section from an aircraft rib is shown in Figure 15. The roller boundary conditions applied to the model indicate lines of symmetry. A nominal load of 10MPa is applied as indicated. The rib section has been modelled using 1290 elements with 7500 degrees of freedom. To find the optimum radius, r, for the fillet indicated in Figure 15, r is initially set to 1.5mm which is the thickness of the step. The radius is then incrementally increased to establish how this affects the stress distribution.

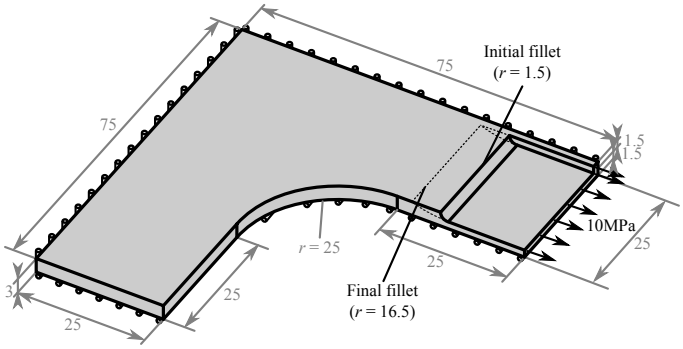

Figure 15: Case study 2: Aircraft rib section. (Dimensions in mm)

The results of applying the acceleration schemes to these models are summarised in this section. As with all BEM simulations, the timings and errors are mesh-dependent. The meshes presented in this work are generated using an algorithm for which a careful study of mesh density versus accuracy has been carried out with real-time simulations in mind. The algorithm allows a slightly lower level of accuracy during dynamic updating of the model in order to accelerate the analysis.

6.2. Results

The Von Mises stress, σ_{VM}, across the surface of the countersunk hole model is shown in Figure 16. It should be noted that the contour plots for both case studies have been created using a linear interpolation

of σ_{VM} at the corner nodes only (the elements are quadratic). This has been done to minimise the time required to re-display the updated contours. This is primarily responsible for the stress contours that appear influenced by the discretisation, especially apparent along the sides of the hole in Figure 16. It is therefore imperative that the contour results are treated with caution, especially in areas of high stress gradient. The peak stress is accurate to within 5% of the value given in Peterson [23]. A more accurate mesh could be applied to reduce this error however the reader is reminded that a compromise has been made between the accuracy of the results during re-analysis and the speed of the analysis. The mesh given in this example is exemplary of the amount of distortion that can be permitted whilst maintaining 5% accuracy during real-time re-analysis. Once a satisfactory design has been established by the analyst, a more detailed analysis with a finer mesh can be carried out, automatically, to ensure that the proposed model is viable.

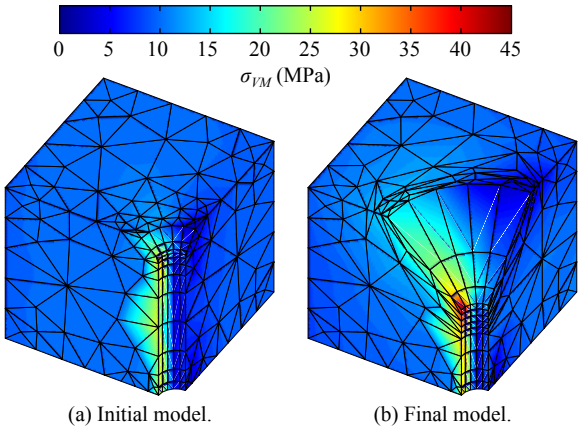

Figure 16: Case study 1: Stress results.

The stress concentration factor, K_t, is defined as the maximum Von Mises stress divided by the applied (reference) stress. The highest stress concentration is apparent at the base of the countersink and increases almost linearly with d with the exception of $6.5 < d < 7.5$, as shown in Figure 17, which is due, in part, to the reduction in mesh quality caused by the distortion. However, these results remain within 5% of Peterson [23], where the data has been generated through a series of finite element simulations. This suggests that, to minimise the stress, the countersink should be kept as shallow as possible which is the intuitive result. The remainder of the stresses in the model remain almost constant as d is updated.

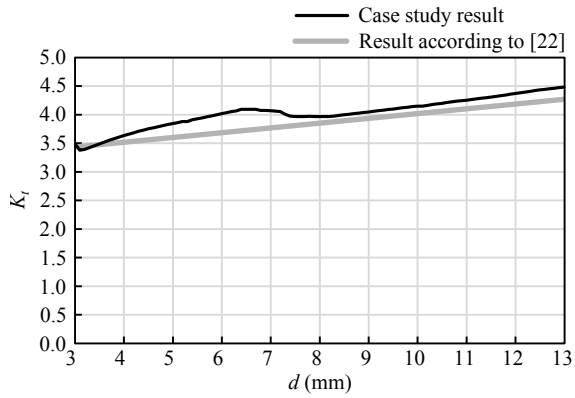

Figure 17: Case study 1: Stress concentration factor at the base of the countersink.

Figure 18 shows the increase in error, ε_x, caused by applying the acceleration schemes. As d gets larger, ε_x increases. This is caused by the increased distortion which results in poor quality elements which are

apparent around the updated geometry in Figure 16. A small improvement in accuracy can be achieved by using a more detailed integration scheme as suggested in Section 3.3. However this leads to a significant increase in the analysis time for little improvement due to the fact that, at this level of distortion, the shape functions, which are equivalent to the basis vectors in which we are seeking a solution, cease to be rich enough to match the solution as closely as for high quality elements. Local re-meshing is therefore required when the mesh quality decreases below a predefined threshold; the elements in this model have reached this threshold.

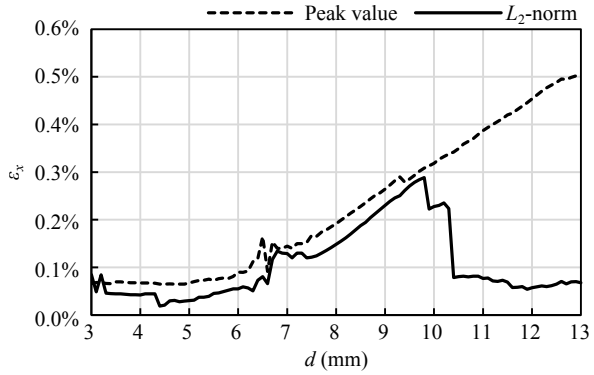

Figure 18: Case study 1: Increase in error ε_x.

For the aircraft rib section, σ_{VM} is shown in Figure 19. A stress concentration, $K_t = 1.67$, is apparent in the fillet on the initial model ($r = 1.5$mm). This reduces as r is increased to 16.5mm as shown in Figure 20. The results across the rest of the model remain almost constant. The stress concentration in this fillet should continue to reduce until $K_t = 1.0$. However, due to interactions with the other fillet in the model, K_t reaches a minimum of $K_t = 1.28$ when $r = 5.6$mm before increasing slightly. The aim of the analyst is to minimise K_t at the fillet; this would be apparent as the point at which the contours ceased to show a significant reduction in the stress.

For the aircraft rib section, ε_x is approximately constant for all values of r as shown in Figure 21.

6.3. Timings

The re-integration time, t_u, for the countersunk hole is shown in Figure 22. For this model $S = 4.7$, on average, when applying the acceleration schemes presented in this work. The algorithm is accelerated further through parallel processing, achieving an additional speed up of $S = 3.7$ on eight cores. Combining these two measures gives a total acceleration of $S = 17.4$. This reduces t_u to approach 0.13 seconds which, when combined with the fast GMRES solver suggested by Foster *et al.* [8], is sufficient to carry out a full re-analysis twice every second.

For the larger (in terms of number of degrees of freedom) aircraft rib section, t_u is shown in Figure 23. As this is a larger problem with a greater number of updated elements both the acceleration schemes and the parallelisation show greater improvement. Without parallelisation, $S = 7.8$ and an additional speed up of $S = 4.4$ is achieved on eight cores. This gives a total acceleration of $S = 34.3$, reducing t_u to 0.38 seconds, on average. This is not fast enough for real-time feedback; however, it is a significant improvement over the full re-integration scheme.

7. Conclusions

A new, efficient algorithm for accelerating the re-integration of the BIE when a geometric change is applied to a model has been introduced and shown to be effective both in serial and in parallel. By combining the work carried out in this paper with the meshing and solution schemes presented by Foster *et al.* [8], it is possible to perform real-time re-analysis for small three-dimensional problems and significantly

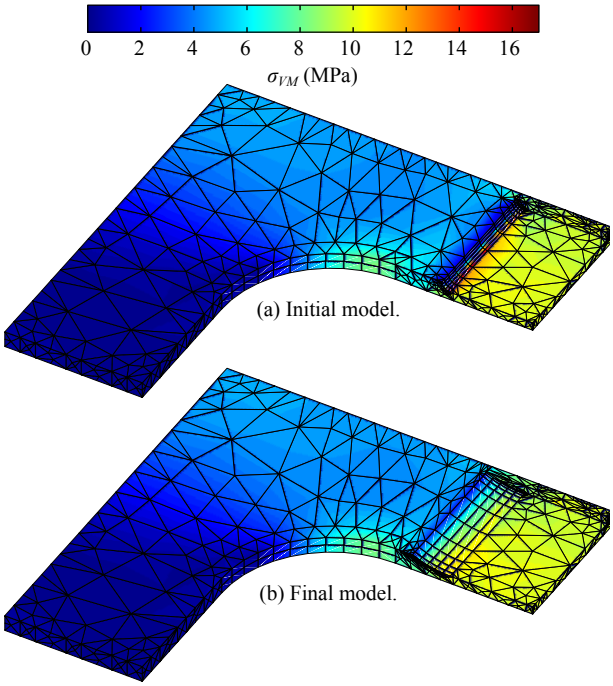

Figure 19: Case study 2: Stress results.

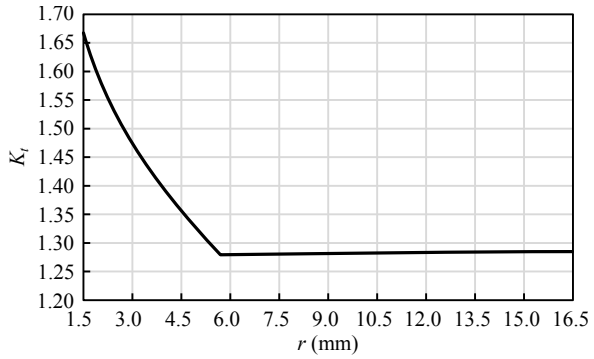

Figure 20: Case study 2: Stress concentration factor at the fillet.

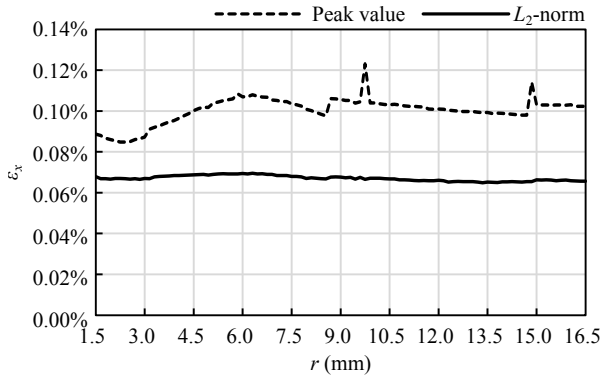

Figure 21: Case study 2: Increase in error ε_x.

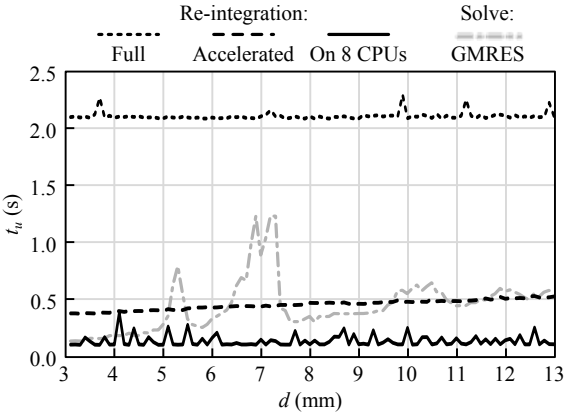

Figure 22: Case study 1: Re-integration timings.

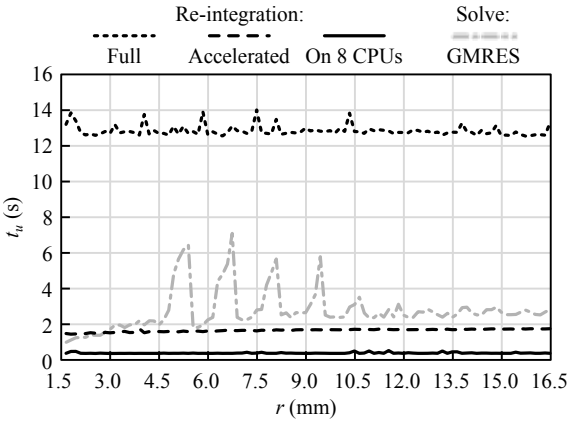

Figure 23: Case study 2: Re-integration timings.

accelerate re-analysis of larger systems. However, in order to achieve the acceleration, compromises must be made in terms of the accuracy of the interim solutions. These affect both degree of refinement of the mesh and the integration scheme. This work therefore poses the question: What level of solution accuracy is acceptable whilst facilitating real-time visual feedback during dynamic updating of a model? The answer to this question depends on the engineer using this technology, and in particular his/her appetite to accelerate the simulation at the expense of accuracy. Engineers tending to favour speed over accuracy are recommended to run a more accurate simulation using an appropriate mesh once the real-time computations are complete and a new design geometry arrived at.

The authors believe that the future of computational mechanics will involve the evolution of simulations alongside model geometry. Engineers will be naturally presented with key durability information as they construct a design and will therefore be able to adjust the design, based on these considerations, as it develops. The work carried out in this paper contributes towards this vision.

Acknowledgements

This research is supported by the UK Engineering and Physical Sciences Research Council (EPSRC) under grant EP/H000046/1.

References

[1] Concept Analyst Ltd., URL: www.conceptanalyst.com (2014).
[2] M. Ryken, J. Vance, Applying virtual reality techniques to the interactive stress analysis of a tractor lift arm, Finite Elements in Analysis and Design 35 (2000) 141–155.
[3] L. Margetts, C. Smethurst, R. Ford, Interactive finite element analysis, in: NAFEMS World Congress, Malta, 2005.
[4] S. Terdalkar, J. Rencis, Graphically driven interactive finite element stress analysis for machine elements in the early design stage, Finite Elements in Analysis and Design 42 (2006) 884–899.
[5] G. Joldes, A. Wittek, K. Miller, Real-time nonlinear finite element computations on GPU – Application to neurosurgical simulation, Computational Methods in Applied Mechanics and Engineering 199 (2010) 3305–3314.
[6] U. Meier, O. López, C. Monserrat, M. Juan, M. Alcañiz, Real-time deformable models for surgery simulation: A survey, Computer Methods and Programs in Biomedicine 77 (2005) 183–197.
[7] P. Wang, A. Becker, I. Jones, A. Glover, S. Benford, C. Greenhalgh, M. Vloeberghs, Virtual reality simulation of surgery with haptic feedback based on the boundary element method, Computers and Structures 85 (2007) 331–339.
[8] T. Foster, M. Mohamed, J. Trevelyan, G. Coates, Rapid re-meshing and re-solution of three-dimensional boundary element problems for interactive stress analysis, Engineering Analysis with Boundary Elements 36 (2012) 1331–1343.
[9] A. Becker, The Boundary Element Method in Engineering, McGraw-Hill International, 1992.
[10] J. Lachat, J. Watson, Effective numerical treatment of boundary integral equations: A formulation for three-dimensional elastostatics, International Journal for Numerical Methods in Engineering 10 (1976) 991–1005.
[11] G. Mustoe, Developments in Boundary Element Methods, Elsevier, London, 1984, Ch. Advanced integration schemes over boundary elements and volume cells for two- and three-dimensional non-linear analysis.
[12] X. Gao, T. Davies, Adaptive integration in elasto-plastic boundary element analysis, Journal of the Chinese Institute of Engineers 23(3) (2000) 349–356.
[13] S. Bu, T. Davies, Effective evaluation of non-singular integrals in 3D BEM, Advances in Engineering Software 23 (1995) 121–128.
[14] J. Kane, Boundary Element Analysis in Engineering Continuum Mechanics, Prentice-Hall, 1994.
[15] G. Symm, Boundary elements on a distributed array processor, Engineering Analysis 1 (1984) 162–165.
[16] A. Davies, Parallel implementations of the boundary element method, Computers and Mathematics with Applications 31(6) (1996) 33–40.
[17] J. Kane, Boundary element analysis on vector and parallel computers, Computing Systems in Engineering 5(3) (1994) 239–252.
[18] B. Baltz, M. Ingber, A parallel implementation of the boundary element method for heat conduction analysis in heterogeneous media, Engineering Analysis with Boundary Elements 19 (1997) 3–11.
[19] K. Erhart, E. Divo, A. Kassab, A parallel domain decomposition boundary element method approach for the solution of large-scale transient heat conduction problems, Engineering Analysis with Boundary Elements 30 (2006) 553–563.
[20] N. Kamiya, H. Iwase, E. Kita, Parallel implementation of boundary element method with domain decomposition, Engineering Analysis with Boundary Elements 18 (1996) 209–216.
[21] N. Kamiya, H. Iwase, E. Kita, Performance evaluation of parallel boundary element analysis by domain decomposition method, Engineering Analysis with Boundary Elements 18 (1996) 217–222.
[22] I. M. Smith, L. Margetts, Parallel boundary element analysis of tunnels, in: ECCOMAS Thematic Conference on Computatioanl Methods in Tunnelling (EURO:TUN 2007), 2007.
[23] W. Pilkey, D. Pilkey, Peterson's Stress Concentration Factors (Third Edition), John Wiley & Sons, 2008.