INTERNATIONAL JOURNAL FOR NUMERICAL AND ANALYTICAL METHODS IN GEOMECHANICS
Int. J. Numer. Anal. Meth. Geomech., 2002; **26**:181–216 (DOI: 10.1002/nag.198)

Upper bound limit analysis using linear finite elements and non-linear programming

A. V. Lyamin[‡] and S. W. Sloan*[,†,§]

Department of Civil, Surveying and Environmental Engineering, University of Newcastle, NSW 2308, Australia

SUMMARY

A new method for computing rigorous upper bounds on the limit loads for one-, two- and three-dimensional continua is described. The formulation is based on linear finite elements, permits kinematically admissible velocity discontinuities at all interelement boundaries, and furnishes a kinematically admissible velocity field by solving a non-linear programming problem. In the latter, the objective function corresponds to the dissipated power (which is minimized) and the unknowns are subject to linear equality constraints as well as linear and non-linear inequality constraints.

Provided the yield surface is convex, the optimization problem generated by the upper bound method is also convex and can be solved efficiently by applying a two-stage, quasi-Newton scheme to the corresponding Kuhn–Tucker optimality conditions. A key advantage of this strategy is that its iteration count is largely independent of the mesh size. Since the formulation permits non-linear constraints on the unknowns, no linearization of the yield surface is necessary and the modelling of three-dimensional geometries presents no special difficulties.

The utility of the proposed upper bound method is illustrated by applying it to a number of two- and three-dimensional boundary value problems. For a variety of two-dimensional cases, the new scheme is up to two orders of magnitude faster than an equivalent linear programming scheme which uses yield surface linearization. Copyright © 2001 John Wiley & Sons, Ltd.

KEY WORDS: limit analysis; upper bound; kinematic; finite element; nonlinear programming

1. INTRODUCTION

The upper bound theorem states that the power dissipated by any kinematically admissible velocity field can be equated to the power dissipated by the external loads to give a rigorous upper bound on the true limit load. A kinematically admissible velocity field is one which satisfies compatibility, the flow rule and the velocity boundary conditions. The upper bound theorem assumes that the deformations are small at incipient collapse and that the material can be modelled with sufficient accuracy using perfect plasticity and an associated flow rule. Because an upper bound calculation considers only velocity modes and energy dissipation, the

*Correspondence to. S. W. Sloan, Department of Civil, Surveying and Environmental Engineering, University of Newcastle, NSW 2308, Australia.
[†]E-mail: scott.sloan@newcastle.edu.au
[‡]Research Academic
[§]Professor

Received 12 February 2001
Revised 5 September 2001

corresponding stress distribution (if one is computed) need not be in equilibrium. Such a distribution, however, must satisfy the yield criterion.

For problems involving complicated geometries, inhomogeneous materials, or complex loading, it is difficult to construct a kinematically admissible velocity field analytically and numerical methods are usually necessary. The most common numerical formulation of the upper bound theorem is based on a finite element discretization of the continuum. This leads to a standard non-linear optimization problem where the objective function corresponds to the dissipated power and the unknowns are subject to a highly sparse set of equality and inequality constraints. If linear finite elements are used in conjunction with a polyhedral approximation to the yield surface, the resulting optimization problem is a classical linear program and can be solved using simplex, active set, or interior point algorithms. Although this type of approach has been widely adopted in two-dimensions (see, e.g. References [1–3]), it is difficult to implement in three dimensions because the linearization of the yield surface becomes non-trivial. Moreover, even if a suitable linearization can be performed, the process will usually generate a very large number of inequality constraints, thereby increasing the cost of solution.

This paper describes the steps involved in a general formulation of the upper bound theorem using linear finite elements and non-linear programming. In two dimensions, each node has two unknown nodal velocities, each triangular element has three unknown stresses, and each interelement velocity discontinuity has four unknown variables. In three dimensions, each node has three unknown velocities, each tetrahedral element has six unknown stresses, and each planar interelement discontinuity has 12 unknowns. Velocity discontinuities are included in the formulation because they greatly enhance its ability to predict the true limit load, especially for frictional materials.

In the final optimization problem, the objective function corresponds to the power dissipated by plastic deformation in the continuum as well as the power dissipated by plastic sliding along the velocity discontinuities. After imposing the flow rule conditions in the continuum, the flow rule conditions in the discontinuities, the velocity boundary conditions, the yield conditions on the stresses, and any loading constraints, the unknowns must satisfy a set of linear equalities as well as a set of linear and non-linear inequalities. It is shown that the solution to this upper bound optimization problem can be found very efficiently by using a two-stage, quasi-Newton algorithm to solve the corresponding Kuhn–Tucker optimality conditions.

The motivation for this work stems from the development of efficient non-linear programming schemes for mixed limit analysis formulations by Zouain *et al*. [4]. These procedures obviate the need to linearize the yield constraints and thus can be used for any type of yield function. The solution technique itself is based on the two-stage feasible point algorithm originating from the work of Herskovits [5]. The new algorithm for solving the upper bound optimization problem is described in detail and its efficiency is illustrated by a number of examples. Timing comparisons for a variety of two-dimensional problems suggest that the new formulation is up to two orders of magnitude faster than an equivalent linear programming formulation. Indeed, a key feature of the new solution scheme is that its iteration count is typically constant, regardless of the mesh refinement.

2. DISCRETE FORMULATION OF UPPER BOUND THEOREM

Consider a body with a volume V and surface area A, as shown in Figure 1. Let $\mathbf{t}$ and $\mathbf{q}$ denote, respectively, a set of fixed tractions acting on the surface area A_t and a set of unknown tractions

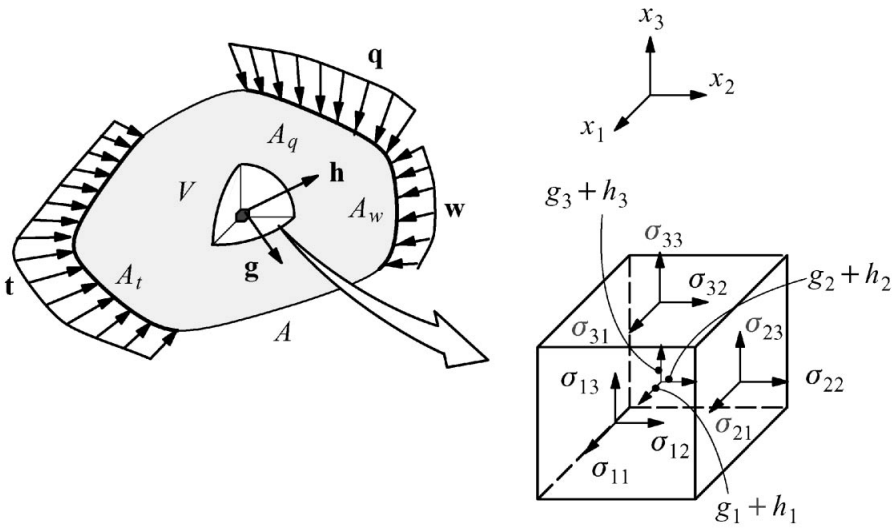


Figure 1. A body subject to a system of surface tractions and body forces.

acting on the surface area A_q. Similarly, let **g** and **h** be a system of fixed and unknown body forces which act, respectively, on the volume V. Under these conditions, the objective of an upper bound calculation is to find a velocity distribution **u** which satisfies compatibility, the flow rule, the velocity boundary conditions **w** on the surface area A_w, and minimizes the dissipated power defined by

$$W^{\text{int}} = \int_V \boldsymbol{\sigma}\dot{\boldsymbol{\varepsilon}}\,\mathrm{d}V$$

where $\boldsymbol{\sigma}$ denotes the stresses and $\dot{\boldsymbol{\varepsilon}}$ denotes the plastic strain rates. Once W^{int} has been minimized to some value $W^{\text{int}}_{\min}$, an upper bound on the true collapse load can be found by equating it to the power dissipated by the external loads according to

$$W^{\text{int}}_{\min} = W^{\text{ext}} = \int_{A_t} \mathbf{t}^{\mathrm{T}}\mathbf{u}\,\mathrm{d}A + \int_{A_q} \mathbf{q}^{\mathrm{T}}\mathbf{u}\,\mathrm{d}A + \int_V \mathbf{g}^{\mathrm{T}}\mathbf{u}\,\mathrm{d}V + \int_V \mathbf{h}^{\mathrm{T}}\mathbf{u}\,\mathrm{d}V$$

As this type of problem is difficult to solve analytically for cases with complex geometries, inhomogeneous material properties, or complicated loading patterns, we seek a numerical formulation which can model the velocity field in a general manner. The most appropriate method for this task is the finite element method. There are a number of compelling reasons for choosing linear finite elements, as opposed to higher-order finite elements, for an upper bound formulation. In summary, these are:

- Linear finite elements generate only linear equality constraints on the nodal velocities in the final optimization problem. This simplifies the design of the non-linear programming solver and allows us to devise a very effective solution strategy.

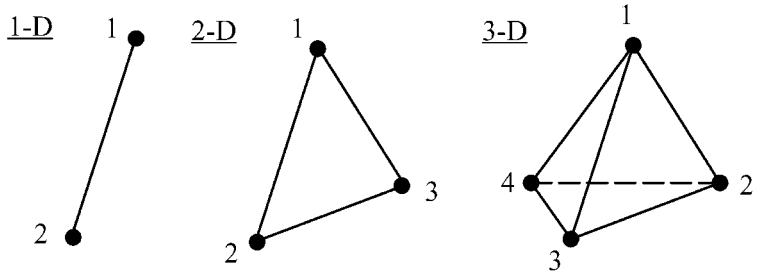
- It is easier to incorporate kinematically admissible velocity discontinuities at interelement boundaries for linear elements. These discontinuities, which permit large velocity jumps to occur over an infinitesimal distance, allow accurate estimates of the collapse load to be computed without using excessive numbers of elements. This feature, to a large extent, compensates for the low order of linear elements, especially for frictional materials. A further advantage of velocity discontinuities, first reported by Sloan and Kleeman [3], is that they prevent the well-known phenomenon of 'locking' for incompressible plasticity problems.
- Many problems in mechanics have boundaries which can be modelled, with sufficient accuracy, using a piecewise linear approximation. For cases involving loaded curved boundaries, it is possible to use a higher-order approximation for the geometry when computing the equivalent power dissipated by the external loads. This decreases the error introduced by the use of a piecewise linear boundary.
- All the mesh generation, assembly and solution software can be designed to be independent of the dimensionality of the problem. This means that the same code can deal with one-, two- and three-dimensional geometries and greatly simplifies the software.

2.1. *Linear finite elements*

In our numerical formulation of the upper bound theorem, simplex finite elements are used to discretize the continuum as shown in Figure 2. Unlike the usual form of the finite element method, each node is unique to a particular element and more than one node can share the same co-ordinates (Figure 3).

In an effort to provide the best possible solution, kinematically admissible velocity discontinuities are permitted at all interelement boundaries. If D is the dimensionality of the problem (where $D = 1, 2, 3$) then there are $D + 1$ nodes in the element and each node is associated with a D-dimensional vector of velocity variables

$$\mathbf{u}^l = \{u_i^l\}^{\mathrm{T}};\ i = 1, \ldots, D \quad \Leftrightarrow \left\{u_1^l \ldots u_D^l\right\}^{\mathrm{T}}$$



$\mathbf{u}^l = \left\{u_1^l\right\}^{\mathrm{T}}$ $\quad$ $\mathbf{u}^l = \left\{u_1^l\ u_2^l\right\}^{\mathrm{T}}$ $\quad$ $\mathbf{u}^l = \left\{u_1^l\ u_2^l\ u_3^l\right\}^{\mathrm{T}}$

$\boldsymbol{\sigma}^e = \left\{\sigma_{11}^e\right\}^{\mathrm{T}}$ $\quad$ $\boldsymbol{\sigma}^e = \left\{\sigma_{11}^e\ \sigma_{22}^e\ \sigma_{12}^e\right\}^{\mathrm{T}}$ $\quad$ $\boldsymbol{\sigma}^e = \left\{\sigma_{11}^e\ \sigma_{22}^e\ \sigma_{33}^e\ \sigma_{12}^e\ \sigma_{23}^e\ \sigma_{13}^e\right\}^{\mathrm{T}}$

Figure 2. Simplex finite elements used in upper bound formulation.

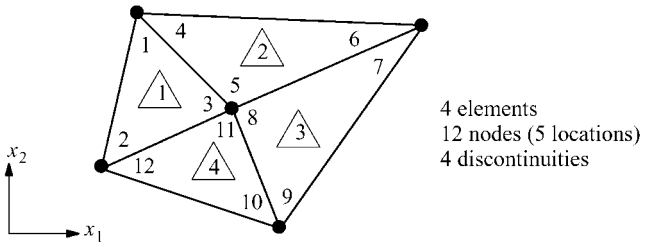


Figure 3. Mesh for upper bound analysis.

These, together with a $D(D+1)/2$-dimensional vector of elemental stresses

$$\boldsymbol{\sigma}^e = \{\sigma_{ij}^e\}^{\mathrm{T}};\ i = 1, \ldots, D,\ j = i, \ldots, D$$

or

$$\boldsymbol{\sigma}^e = \{\sigma_{11}^e \ldots \sigma_{DD}^e\ \sigma_{12}^e \ldots \sigma_{D-1,D}^e \ldots \sigma_{1D}^e\}^{\mathrm{T}}$$

and a vector of discontinuity variables, $\mathbf{v}^{\mathrm{d}}$, are taken as the problem variables. For convenience, the global vector of problem variables will be referred to as $\mathbf{x} = (\mathbf{u}, \mathbf{v}, \boldsymbol{\sigma})$.

The variation of the velocities throughout each element may be written as

$$\mathbf{u} = \sum_{l=1}^{D+1} N_l \mathbf{u}^l$$

where N_l are linear shape functions. More formally, the shape functions N_l can be written as

$$N_l = \frac{1}{|\mathbf{C}|} \sum_{k=0}^{D} (-1)^{l+k+1} |\mathbf{C}_{(l)(k)}| x_k \tag{1}$$

where x_k are the co-ordinates of the point k at which the shape functions are to be computed (with the convention that $x_0 = 1$), $\mathbf{C}$ is a $(D+1) \times (D+1)$ matrix formed from the element nodal co-ordinates according to

$$\mathbf{C} = \begin{vmatrix} 1 & x_1^1 & \ldots, & x_D^1 \\ 1 & x_1^2 & \ldots, & x_D^2 \\ \ldots & \ldots & \ldots, & \ldots \\ 1 & x_1^{D+1} & \ldots, & x_D^{D+1} \end{vmatrix} \quad \text{or} \quad \mathbf{C} = \begin{vmatrix} x_0^1 & x_1^1 & \ldots, & x_D^1 \\ x_0^2 & x_1^2 & \ldots, & x_D^2 \\ \ldots & \ldots & \ldots, & \ldots \\ x_0^{D+1} & x_1^{D+1} & \ldots, & x_D^{D+1} \end{vmatrix} \tag{2}$$

and $\mathbf{C}_{(l)(k)}$ is a $D \times D$ submatrix of $\mathbf{C}$ obtained by deleting the lth row and the kth column of $\mathbf{C}$ (the determinant $|\mathbf{C}_{(l)(k)}|$ is the *minor* of the entry c_{lk} of $\mathbf{C}$). In the expressions (2), the superscripts are row numbers and correspond to the local node number of the element, while the subscripts are column numbers and designate the co-ordinate index. Elements in the first column of $\mathbf{C}$ are, by convention, allocated the subscript 0 and a value of unity. After grouping terms,

Equation (1) can be written in the more compact form

$$N_l = \sum_{k=0}^{D} a_{lk} x_k, \quad a_{lk} = a(l,k) = (-1)^{l+k+1} \frac{|\mathbf{C}_{(l)(k)}|}{|\mathbf{C}|} \tag{3}$$

2.2. Plastic flow rule constraints for continuum

To be kinematically admissible, and thus provide a rigorous upper bound on the true collapse load, the velocity field at any point in the body must satisfy the set of constraints imposed by an associated flow rule

$$\dot{\boldsymbol{\varepsilon}} = \mathbf{B}\mathbf{u} = \lambda \nabla f(\boldsymbol{\sigma}), \quad \lambda \geqslant 0, \quad \lambda f(\boldsymbol{\sigma}) = 0 \tag{4}$$

where $\dot{\boldsymbol{\varepsilon}}$ is a $D(D+1)/2$-dimensional vector of strain rates, $\mathbf{B}$ is the strain rate–velocity compatibility operator, f is the yield function, λ is a non-negative plastic multiplier rate, and $\nabla = \{\partial/\partial\sigma_{11} \ldots \partial/\partial\sigma_{DD}\, \partial/\partial\sigma_{12} \ldots \partial/\partial\sigma_{D-1,D} \ldots \partial/\partial\sigma_{1D}\}^{\mathrm{T}}$. Conditions (4) imply that the plastic strains are normal to the yield surface. They also imply there is no plastic deformation in elements whose stresses lie inside the yield surface.

In terms of the notation of Section 2.1, the flow rule constraints (4) can be written for each finite element e as

$$\dot{\boldsymbol{\varepsilon}}^e = \mathbf{B}^e \mathbf{u}^e = \lambda^e \nabla f(\boldsymbol{\sigma}^e), \quad \lambda^e \geqslant 0, \quad \lambda^e f(\boldsymbol{\sigma}^e) = 0 \tag{5}$$

where $\mathbf{u}^e$ is a $D(D+1)$-dimensional vector of element nodal velocities, $\mathbf{B}^e$ is a $D(D+1)/2 \times D \times (D+1)$ compatibility matrix composed of $D+1$ nodal compatibility matrices according to

$$\mathbf{B}^e = [\mathbf{B}^1, \ldots, \mathbf{B}^l, \ldots, \mathbf{B}^{D+1}] \tag{6}$$

and

$$\mathbf{B}^{l\mathrm{T}} = \begin{bmatrix} \delta_{111} a(l, \phi_{111}) & \ldots, & \delta_{1DD} a(l, \phi_{1DD}) & \ldots, & \delta_{11D} a(l, \phi_{11D}) \\ \cdots & \ldots, & \cdots & \ldots, & \cdots \\ \delta_{D11} a(l, \phi_{D11}) & \ldots, & \delta_{DDD} a(l, \phi_{DDD}) & \ldots, & \delta_{D1D} a(l, \phi_{D1D}) \end{bmatrix} \tag{7}$$

with the coefficients $a(l, k)$ given by (3) and the two auxiliary index functions defined as

$$\delta(i,k,m) = \delta_{ikm} = \begin{cases} 1 & \text{if } i = k \text{ or } i = m \\ 0 & \text{otherwise} \end{cases}$$

$$\phi(i,k,m) = \phi_{ikm} = \begin{cases} m & \text{if } i = k \\ k & \text{if } i = m \\ 1 & \text{otherwise} \end{cases}$$

Let $\mathbf{B}$ denote the global compatibility matrix (where each element block $\mathbf{B}^e$ is multiplied by the element volume V^e), $\mathbf{u}$ the global velocity vector, $\boldsymbol{\sigma}$ the global stress vector, and $\boldsymbol{\lambda}$ the total set of plastic multiplier rates (scaled by the element volume V^e). We can then write the flow rule

constraints for the mesh in matrix form as

$$\mathbf{Bu} = \sum_{j=1}^{E} \lambda_j \nabla f_j(\boldsymbol{\sigma}) \tag{8}$$

$$\lambda_j \geqslant 0, \quad j = 1, \ldots, E \tag{9}$$

$$\lambda_j f_j(\boldsymbol{\sigma}) = 0, \quad j = 1, \ldots, E \tag{10}$$

where E is the number of elements. Thus, the continuum flow rule generates $E+ED(D+1)/2$ equality constraints and E inequality constraints on the problem variables.

2.3. Flow rule constraints for discontinuities

A three-dimensional velocity discontinuity is shown in Figure 4. These are introduced at all interelement boundaries, are defined by D pairs of adjacent nodes, and are of zero thickness. To be kinematically admissible, the normal and tangential velocity jumps across the discontinuity must satisfy the flow rule. For a Mohr–Coulomb yield criterion, this is of the form

$$\Delta u_{\mathrm{n}} = |\Delta u_{\mathrm{t}}| \tan \phi \tag{11}$$

where Δu_{n} and Δu_{t} are the normal and tangential velocity jumps, respectively, and ϕ is the material friction angle. For a dilatational yield criterion whose shape is non-linear in the meridional plane, the flow rule constraints are applied using a local Mohr–Coulomb approximation to the yield surface (Figure 5). Non-dilatational yield criteria, such as the Tresca and Von-Mises surfaces, with a cohesion c and $\phi = 0$ are simpler to model as they imply

$$\begin{Bmatrix} x_1' \\ \vdots \\ x_D' \end{Bmatrix} = \begin{bmatrix} \beta_{11} & \cdots & \beta_{1D} \\ \vdots & & \vdots \\ \beta_{D1} & \cdots & \beta_{DD} \end{bmatrix} \begin{Bmatrix} x_1 \\ \vdots \\ x_D \end{Bmatrix}$$

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_{11} & \cdots & \beta_{1D} \\ \vdots & & \vdots \\ \beta_{D1} & \cdots & \beta_{DD} \end{bmatrix}$$

$$\Delta \mathbf{u}' = \boldsymbol{\beta} \Delta \mathbf{u}$$

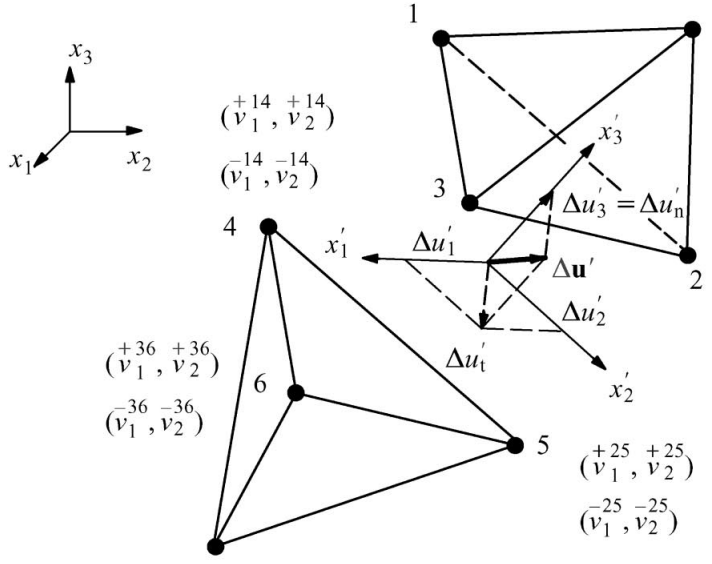


Figure 4. Three-dimensional velocity discontinuity.

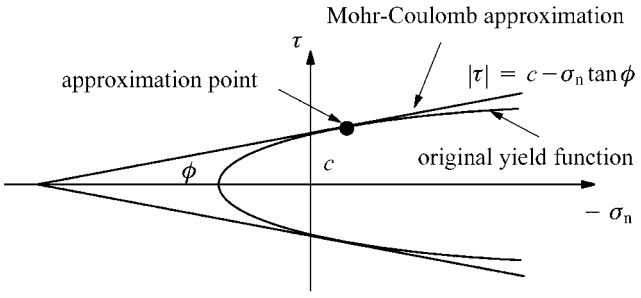


Figure 5. Mohr–Coulomb approximation of a general yield function.

that $\Delta u_{\mathrm{n}} = 0$. The absolute value on the right-hand side of Equation (11) is necessary because, for a non-zero friction angle, dilation occurs regardless of the direction of tangential shearing. This feature makes it difficult to derive a set of flow rule constraints which are everywhere differentiable. Our method for implementing kinematically admissible velocity discontinuities is a D-dimensional generalization of the formulation proposed originally by Sloan and Kleeman [3]. Each nodal pair (l, m) on a discontinuity is associated with $D-1$ pairs of non-negative variables $\overset{+}{\mathbf{v}}{}^{lm}$ and $\overset{-}{\mathbf{v}}{}^{lm}$, as shown in Figure 4, and thus gives rise to $2(D-1)$ additional unknowns. The tangential components $\Delta\mathbf{u}'^{lm}_{\mathrm{t}}$ of the velocity jump $\Delta\mathbf{u}'^{lm}$ at each nodal pair are defined as the difference between these two sets of non-negative variables according to

$$\Delta\mathbf{u}'^{lm}_{\mathrm{t}} = \overset{+}{\mathbf{v}}{}^{lm} - \overset{-}{\mathbf{v}}{}^{lm} \tag{12}$$

where

$$\overset{+}{\mathbf{v}}{}^{lm} = \left\{ \overset{+}{v}{}^{lm}_{1}, \ldots, \overset{+}{v}{}^{lm}_{D-1} \right\}^{\mathrm{T}}$$

$$\overset{-}{\mathbf{v}}{}^{lm} = \left\{ \overset{-}{v}{}^{lm}_{1}, \ldots, \overset{-}{v}{}^{lm}_{D-1} \right\}^{\mathrm{T}}$$

$$\Delta\mathbf{u}'^{lm}_{\mathrm{t}} = \left\{ \Delta u'^{lm}_{1}, \ldots, \Delta u'^{lm}_{D-1} \right\}^{\mathrm{T}}$$

$$\Delta\mathbf{u}'^{lm} = \left\{ \{\Delta\mathbf{u}'^{lm}_{\mathrm{t}}\}^{\mathrm{T}}, \Delta u'^{lm}_{D} \right\}^{\mathrm{T}}$$

The vector $\Delta\mathbf{u}'$ holds the velocity jumps in terms of the local co-ordinate system of Figure 4, and is related to the global velocity jumps $\Delta\mathbf{u}$ by the expression

$$\Delta\mathbf{u}' = \boldsymbol{\beta}\Delta\mathbf{u}$$

where $\boldsymbol{\beta}$ is the appropriate matrix of direction cosines. To remove the absolute value sign in the flow rule relation (11), and thus avoid non-differentiable constraints, $|\Delta u_{\mathrm{t}}|$ is replaced by

$$\sum_{i=1}^{D-1} \left(\overset{+}{v}{}_{i} + \overset{-}{v}{}_{i} \right)$$

so that the normal velocity jump is given by

$$\Delta u_{\mathrm{n}} = \Delta u'_D = \sum_{i=1}^{D-1} \left(\overset{+}{v}_i + \overset{-}{v}_i \right) \tan \phi \tag{13}$$

Sloan and Kleeman [3] have proved that this substitution preserves the upper bound nature of the solution, as it is equivalent to modelling a stronger discontinuity with properties (c^*, ϕ^*) where $c^* \geqslant c$ and $\phi^* \geqslant \phi$. Since the velocities vary linearly over each element, Equations (12) and (13) are satisfied for every point on the discontinuity if the flow rule constraints are imposed at each pair of its nodes (l, m) according to

$$\Delta u_i'^{lm} = \overset{+}{v}{}_i^{lm} - \overset{-}{v}{}_i^{lm}; \quad i = 1, \ldots, D-1$$

$$\Delta u_D'^{lm} = \sum_{i=1}^{D-1} \left(\overset{+}{v}{}_i^{lm} + \overset{-}{v}{}_i^{lm} \right) \tan \phi$$

$$\Delta \mathbf{u}'^{lm} = \boldsymbol{\beta}^{lm} \Delta \mathbf{u}^{lm}$$

$$\Delta \mathbf{u}^{lm} = \mathbf{u}^m - \mathbf{u}^l$$

$$\overset{+}{v}{}_i^{lm} \geqslant 0, \quad \overset{-}{v}{}_i^{lm} \geqslant 0; \quad i = 1, \ldots, D-1 \tag{14}$$

In matrix notation, conditions (14) can be written for discontinuity d as

$$\mathbf{A}^d_{u\,\mathrm{flow}} \mathbf{u}^d + \mathbf{A}^d_{v\,\mathrm{flow}} \mathbf{v}^d = \mathbf{b}^d_{\mathrm{flow}} \tag{15}$$

$$\mathbf{v}^d \geqslant \mathbf{0} \tag{16}$$

where

$$\mathbf{A}^d_{u\,\mathrm{flow}} = \begin{bmatrix} \boldsymbol{\beta}^{l_1 m_1} & & -\boldsymbol{\beta}^{l_1 m_1} & & & \mathbf{0} & \\ & \vdots & & \ldots & & \vdots & \\ & \mathbf{0} & & & \boldsymbol{\beta}^{l_D m_D} & & -\boldsymbol{\beta}^{l_D m_D} \end{bmatrix}$$

$$\mathbf{A}^d_{v\,\mathrm{flow}} = \begin{bmatrix} -1 \ldots & 1 \ldots & 0 & & & & \\ \vdots & \ldots & \vdots & & & \mathbf{0} & \\ \tan\phi & \ldots & \tan\phi & & & & \\ & \vdots & & \ldots & & \vdots & \\ & & & & -1 \ldots & 1 \ldots & 0 \\ & \mathbf{0} & & & \vdots & \ldots & \vdots \\ & & & & \tan\phi & \ldots & \tan\phi \end{bmatrix}$$

$$\mathbf{u}^d = \left\{ \{\mathbf{u}^{m_1}\}^{\mathrm{T}}, \{\mathbf{u}^{l_1}\}^{\mathrm{T}}, \ldots, \{\mathbf{u}^{m_D}\}^{\mathrm{T}}, \{\mathbf{u}^{l_D}\}^{\mathrm{T}} \right\}^{\mathrm{T}}$$

$$\mathbf{v}^d = \left\{ \left\{ \overset{+}{\mathbf{v}}{}^{l_1 m_1} \right\}^{\mathrm{T}}, \left\{ \overset{-}{\mathbf{v}}{}^{l_1 m_1} \right\}^{\mathrm{T}}, \ldots, \left\{ \overset{+}{\mathbf{v}}{}^{l_D m_D} \right\}^{\mathrm{T}}, \left\{ \overset{-}{\mathbf{v}}{}^{l_D m_D} \right\}^{\mathrm{T}} \right\}^{\mathrm{T}}$$

$$\mathbf{b}^d_{\mathrm{flow}} = \{\mathbf{0}\}$$

Thus each discontinuity gives rise to a total of D^2 equality constraints and $2D(D-1)$ inequality constraints on the nodal velocities and discontinuity variables.

2.4. Velocity boundary conditions

To be kinematically admissible, the computed velocity field must satisfy the prescribed boundary conditions. Consider a distribution of prescribed velocities w_p, $p \in P$, where P is a set of N_p prescribed components, over part of the boundary area A_w (Figure 6). If these velocities are specified in terms of global co-ordinates and distributed over the linearized boundary area A_w^{d}, we can cast the velocity boundary conditions for every node l on this area as

$$u_p^l = w_p^l,\ p \in P \tag{17}$$

If the prescribed velocities are defined in terms of local co-ordinates, as shown in Figure 6, then the velocity boundary conditions may be written as

$$u_p'^l = \beta_{pi}^l u_i^l = w_p^l,\ p \in P \tag{18}$$

where β_{pi}^l are the direction cosines of the local Cartesian co-ordinate system for node l.

Note that it is generally preferable to impose the constraints (18) on the nodal velocities with respect to the original, and not the linearized, boundary. If the latter is used with complex surface geometries, the surface normal is constant over each boundary segment and the velocity boundary conditions are represented less accurately. Considering (17) as a variant of (18) with $\boldsymbol{\beta} = \mathbf{I}$, we can cast both of the above conditions in the general matrix form

$$\mathbf{A}_{\mathrm{bound}}^b \mathbf{u}^b = \mathbf{b}_{\mathrm{bound}}^b \tag{19}$$

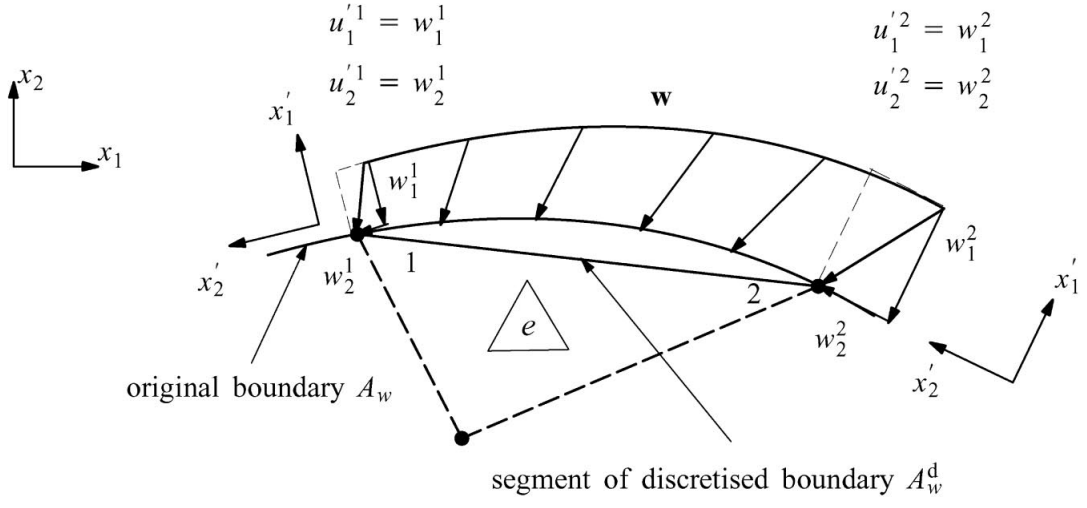


Figure 6. 2D velocity boundary conditions.

where

$$\mathbf{u}^b = \mathbf{u}^l$$

$$\mathbf{A}^b_{\text{bound}} = \left[\boldsymbol{\beta}^{l\mathrm{T}}_{p_1} \cdots \boldsymbol{\beta}^{l\mathrm{T}}_{p_{Np}}\right]^{\mathrm{T}}$$

$$\mathbf{b}^d_{\text{bound}} = \{w^l_p\}^{\mathrm{T}},\ p \in P$$

Every node that is subject to prescribed velocities generates a maximum of D equality constraints on the problem unknowns.

2.5. *Objective function*

Plastic flow may occur in both the continuum and the velocity discontinuities. Consequently, the total power dissipated in these regions constitutes the objective function. For each continuum element of volume V_e, the power dissipated by the plastic stresses is given by

$$W^e_c = \int_{V_e} \boldsymbol{\sigma}^{\mathrm{T}} \dot{\boldsymbol{\varepsilon}}\, \mathrm{d}V = \int_{V_e} \boldsymbol{\sigma}^{\mathrm{T}} \mathbf{B}\mathbf{u}\, \mathrm{d}V = V^e \boldsymbol{\sigma}^{e\mathrm{T}} \mathbf{B}^e \mathbf{u}^e$$

where $\mathbf{B}^e$ is defined in (6). For each velocity discontinuity of surface area S_d, the power dissipated by plastic shearing is given by an integral of the form

$$W^s_d = \int_{S_d} (|\tau \Delta u_{\mathrm{t}}| + \sigma_{\mathrm{n}} \Delta u_{\mathrm{n}}) \mathrm{d}S$$

which, using the flow rule (11) and the Mohr–Coulomb yield criterion $|\tau| = c - \sigma_{\mathrm{n}} \tan\phi$, becomes

$$W^s_d = \int_{S_d} c|\Delta u_{\mathrm{t}}| \mathrm{d}S = \int_{S_d} c \left[\sum_{i=1}^{D-1} \left(\overset{+}{v}_i - \overset{-}{v}_i\right)^2\right]^{1/2} \mathrm{d}S$$

Unfortunately, W^s_d is non-differentiable with respect to the problem unknowns when $\Delta u_{\mathrm{t}} = 0$. To avoid this singularity, the proposed formulation uses the result that

$$|\Delta u_{\mathrm{t}}| \leqslant \sum_{i=1}^{D-1} \left(\overset{+}{v}_i + \overset{-}{v}_i\right)$$

so that an upper bound on the power dissipated in each discontinuity is given by

$$W^s_d = \int_{S_d} c \sum_{i=1}^{D-1} \left(\overset{+}{v}_i + \overset{-}{v}_i\right) \mathrm{d}S \tag{20}$$

Assuming the discontinuity variables and cohesion c vary linearly over the discontinuity, integrating (20) gives

$$W^s_d = \mathbf{c}^{\mathrm{T}} \left[\int_{S_d} (\mathbf{N}^s_c \otimes \mathbf{e}) \mathbf{N}^s_v \, \mathrm{d}S\right] \mathbf{v}^d \tag{21}$$

where

$$\mathbf{c} = \{c_{l_1}, \ldots, c_{l_D}\}^{\mathrm{T}}$$

$$\mathbf{N}_c^s = \{N_{l_1}, \ldots, N_{l_D}\}^{\mathrm{T}}$$

$$\mathbf{N}_v^s = [\mathbf{I}N_{l_1}, \ldots, \mathbf{I}N_{l_D}]$$

and $\mathbf{e}$ is a $2(D-1)$-dimensional unit vector, $\mathbf{I}$ is a $2(D-1) \times 2(D-1)$-dimensional identity matrix, and the shape functions $\mathbf{N}_c^s$ and $\mathbf{N}_v^s$ are for $(D-1)$-dimensional space.

When we equate the rate of work of the external forces

$$\int_{A_t} \mathbf{t}^{\mathrm{T}}\mathbf{u}\,\mathrm{d}A + \int_{A_q} \mathbf{q}^{\mathrm{T}}\mathbf{u}\,\mathrm{d}A + \int_V \mathbf{g}^{\mathrm{T}}\mathbf{u}\,\mathrm{d}V + \int_V \mathbf{h}^{\mathrm{T}}\mathbf{u}\,\mathrm{d}V$$

to the rate of internal dissipation

$$\int_{V_e} \boldsymbol{\sigma}^{\mathrm{T}}\dot{\boldsymbol{\varepsilon}}\,\mathrm{d}V + \int_{S_d} (|\tau\Delta u_{\mathrm{t}}| + \sigma_{\mathrm{n}}\Delta u_{\mathrm{n}})\,\mathrm{d}S = \int_{V_e} \boldsymbol{\sigma}^{\mathrm{T}}\mathbf{B}\mathbf{u}\,\mathrm{d}V + \int_{S_d} c|\Delta u_{\mathrm{t}}|\,\mathrm{d}S$$

then, if some of the external forces are fixed, their work contributions need to be subtracted from the objective function. For side S_t of an element with nodes $l_1, \ldots, l_D$ that is subject to fixed surface tractions $\mathbf{t}$ we have

$$W_t^s = \int_{S_t} \mathbf{t}^{\mathrm{T}}\mathbf{u}\,\mathrm{d}S = \int_{S_t} (\mathbf{N}^s\mathbf{t}^s)^{\mathrm{T}}(\mathbf{N}^s\mathbf{u}^s)\mathrm{d}S = \mathbf{t}^{s\mathrm{T}}\left(\int_{S_t} \mathbf{N}^{s\mathrm{T}}\mathbf{N}^s\,\mathrm{d}S\right)\mathbf{u}^s \tag{22}$$

where

$$\mathbf{N}^s = [\mathbf{I}N_{l_1}, \ldots, \mathbf{I}N_{l_D}]$$

$$\mathbf{u}^s = \left\{\{\mathbf{u}^{l_1}\}^{\mathrm{T}}, \ldots, \{\mathbf{u}^{l_D}\}^{\mathrm{T}}\right\}^{\mathrm{T}}$$

$$\mathbf{t}^s = \left\{\{\mathbf{t}^{l_1}\}^{\mathrm{T}}, \ldots, \{\mathbf{t}^{l_D}\}^{\mathrm{T}}\right\}^{\mathrm{T}}$$

Similarly, for an element e subjected to fixed body forces $\mathbf{g}$ we can write

$$W_g^e = \int_{V_e} \mathbf{g}^{\mathrm{T}}\mathbf{u}\,\mathrm{d}V = \int_{V_e} \mathbf{g}^{e\mathrm{T}}\mathbf{N}^e\mathbf{u}^e\,\mathrm{d}V = \mathbf{g}^{e\mathrm{T}}\left(\int_{V_e} \mathbf{N}^e\,\mathrm{d}V\right)\mathbf{u}^e \tag{23}$$

where

$$\mathbf{N}^e = [\mathbf{I}N_{l_1}, \ldots, \mathbf{I}N_{l_{D+1}}]$$

$$\mathbf{g}^e = \{g_1, \ldots, g_D\}^{\mathrm{T}}$$

and $\mathbf{I}$ is now a $D \times D$ dimensional identity matrix and the shape functions $\mathbf{N}^e$ are for D-dimensional space. Thus, the final objective function Q is defined by

$$Q = \sum_{e=1}^{E} W_c^e + \sum_{s=1}^{Ds} W_d^s - \sum_{s=1}^{N_{\mathrm{t}}} W_t^s - \sum_{e=1}^{E} W_g^e \tag{24}$$

where E is the total number of elements, Ds is the total number of discontinuities, and N_{t} is the total number of element sides which are subject to prescribed surface tractions. Collecting

coefficients, (24) can be written as

$$Q = \boldsymbol{\sigma}^{\mathrm{T}}\mathbf{B}\mathbf{u} + \mathbf{c}_u^{\mathrm{T}}\mathbf{u} + \mathbf{c}_v^{\mathrm{T}}\mathbf{v} \tag{25}$$

where $\mathbf{B}$ is the global compatibility matrix introduced in Section 2.2, and $\mathbf{c}_u$ and $\mathbf{c}_v$ are vectors of linear coefficients for velocity and discontinuity variables, respectively, accumulated from expressions (21), (22) and (23).

2.6. *Loading constraints*

Additional constraints are often needed to specify the loading pattern and introduce relations between the components of the velocity field. For loading with a proportional displacement pattern, it is convenient to introduce the notion of a velocity profile. Figure 7 shows a boundary element segment, defined by the nodes 1 and 2, where the velocity field is optimized subject to the constraints of a known normal velocity profile $Prf(u_{\mathrm{n}})$. Under conditions of proportional displacement, we have

$$\frac{(u_{\mathrm{n}}^1)_{\mathrm{opt}}}{Prf(u_{\mathrm{n}})^1} = \frac{(u_{\mathrm{n}}^2)_{\mathrm{opt}}}{Prf(u_{\mathrm{n}})^2}$$

where the precise values of $Prf(u_{\mathrm{n}})^1$ and $Prf(u_{\mathrm{n}})^2$ may be chosen arbitrarily, provided they fit the required distribution. Each velocity component may have its own profile and each node l on the profile has a sign, denoted $sign(Prf(u_{\mathrm{n}})^l)$, which is equal to ± 1 and determines the direction of the optimized component. In cases where the final distribution of the optimized velocities is unknown, all nodes on the velocity profile are specified to have the same sign.

The most common types of loading constraints are load orientation and load profile constraints. The former occur when the orientation of the velocities on a loaded part of the body is known, so that the ratio of velocity components at a particular point is fixed. In Figure 8(a), for example, the ratio of the normal and tangential velocities at node 1 can be prescribed in terms of local co-ordinates by choosing appropriate values for $Prf(u_1')^1$ and $Prf(u_2')^1$. Load profile constraints are different to load orientation constraints in that they are used to specify the distribution of a single component of the unknown velocities over the loaded segment. In Figure 8(b), the required distribution of the normal and tangential velocities can be obtained by prescribing suitable values, respectively, for $Prf(u_1')^1$ and $Prf(u_1')^2$ and $Prf(u_2')^1$, $Prf(u_2')^2$. Restricting the normal velocities to be uniform over part of the boundary is one very common type of load profile constraint.

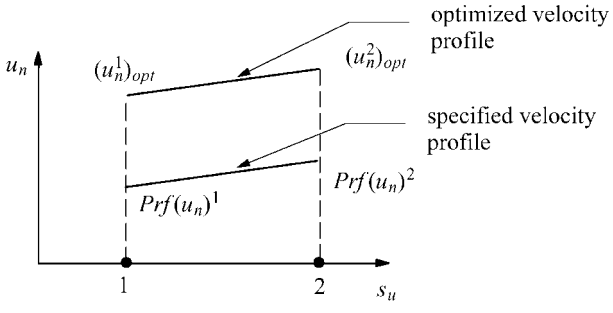


Figure 7. Velocity profile for applied boundary traction.

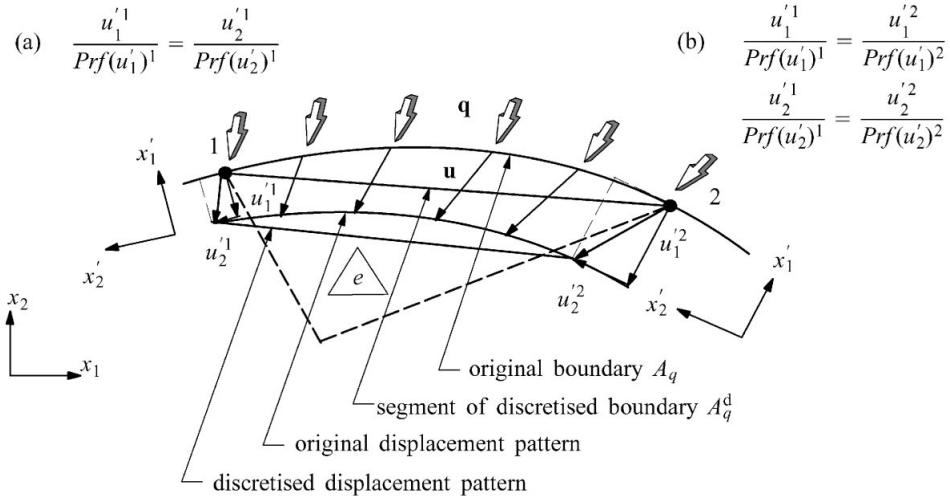


Figure 8. 2D example of loading constraints: (a) velocity orientation constraints for node 1 and (b) velocity profile constraints for nodes 1 and 2.

To specify the directions of the optimized deformations, we can introduce constraints which specify the ratio of the velocity components at all nodes along a loaded segment. Since the velocities are allowed to vary linearly within an element, this ratio need not be constant along the segment. When the deformation pattern is specified with respect to the global co-ordinate system, the load orientation constraints for the velocities at node l can be written as

$$\frac{u^l_{o_m}}{Prf(u_{o_m})^l} = \frac{u^l_{o_{m+1}}}{Prf(u_{o_{m+1}})^l},\ o_m,\ o_{m+1} \in O,\ m = 1, \ldots,\ N_O - 1$$

where O is the set and N_O is the number of optimized velocity components. When the optimization is conducted with respect to a local co-ordinate system, the above expressions take the form

$$\frac{\beta^l_{o_m i} u^l_i}{Prf(u'_{o_m})^l} = \frac{\beta^l_{o_{m+1} i} u^l_i}{Prf(u'_{o_{m+1}})^l}$$

or

$$\left[\frac{1}{Prf(u'_{o_m})^l}\boldsymbol{\beta}^l_{o_m} - \frac{1}{Prf(u'_{o_{m+1}})^l}\boldsymbol{\beta}^l_{o_{m+1}}\right]\mathbf{u}^l = 0$$

where

$$o_m,\ o_{m+1} \in O, \quad m = 1, \ldots,\ N_O - 1$$

As each load orientation constraint is a simple linear equality, it can be stated in matrix notation as

$$\mathbf{A}^{o}_{\text{orient}}\mathbf{u} = \mathbf{b}^{o}_{\text{orient}} \tag{26}$$

where the maximum number of constraints on the velocities is $(D-1)$ per node on A^{d}_{q}.

If any component of the unknown velocity field is to be optimized according to a particular distribution (or profile), then the loading constraints need to include node-to-node relations to reflect this. When the optimization is done with respect to global co-ordinates, the node-to-node load profile constraints for the oth component of the velocities can be written as

$$\frac{u_o^{l_m}}{Prf(u_o)^{l_m}} = \frac{u_o^{l_{m+1}}}{Prf(u_o)^{l_{m+1}}}, \quad o \in O, \quad l_m,\ l_{m+1} \in A^{\text{d}}_{q}$$

When the load optimization is done in terms of local co-ordinates, the above expressions become

$$\frac{\beta_{oi}^{l_m} u_i^{l_m}}{Prf(u_o)^{l_m}} = \frac{\beta_{oi}^{l_{m+1}} u_i^{l_{m+1}}}{Prf(u_o)^{l_{m+1}}}$$

or

$$\frac{1}{Prf(u_o)^{l_m}}\boldsymbol{\beta}_o^{l_m}\mathbf{u}^{l_m} - \frac{1}{Prf(u_o)^{l_{m+1}}}\boldsymbol{\beta}_o^{l_{m+1}}\mathbf{u}^{l_{m+1}} = 0$$

where

$$o \in O,\ l_m,\ l_{m+1} \in A^{\text{d}}_{q}$$

Each load profile constraint is a linear equality of the form

$$\mathbf{A}^{o}_{\text{profile}}\mathbf{u} = \mathbf{b}^{o}_{\text{profile}} \tag{27}$$

and adds a maximum of D equalities per pair of nodes on A^{d}_{q}.

To ensure the velocity field is unique we may, in some cases, need to impose an additional scaling constraint on the velocity variables. For proportional loading this is typically of the form

$$\int_{A_q} \mathbf{q}^{\text{T}}\mathbf{u}\,\text{d}A + \int_{V} \mathbf{h}^{\text{T}}\mathbf{u}\,\text{d}V = 1$$

Letting N_q denote the number of element sides which are subject to a surface traction $\mathbf{q}$, this equation can be written in matrix notation as

$$\sum_{s=1}^{N_q} W_q^s + \sum_{e=1}^{E} W_h^e = 1 \tag{28}$$

where

$$W_q^s = \int_{S_t} \mathbf{q}^{\text{T}}\mathbf{u}\,\text{d}S = \int_{S_t} (\mathbf{N}^s\mathbf{q}^s)^{\text{T}}(\mathbf{N}^s\mathbf{u}^s)\,\text{d}S = \mathbf{q}^{s\text{T}}\left(\int_{S_t} \mathbf{N}^{s\text{T}}\mathbf{N}^s\,\text{d}S\right)\mathbf{u}^s$$

$$W_h^e = \int_{V_e} \mathbf{h}^{\text{T}}\mathbf{u}\,\text{d}V = \int_{V_e} \mathbf{h}^{e\text{T}}\mathbf{N}^e\mathbf{u}^e\,\text{d}V = \mathbf{h}^{e\text{T}}\left(\int_{V_e} \mathbf{N}^e\,\text{d}V\right)\mathbf{u}^e$$

and

$$\mathbf{q}^s = \left\{ \{\mathbf{q}^{l_1}\}^{\mathrm{T}}, \ldots, \{\mathbf{q}^{l_D}\}^{\mathrm{T}} \right\}^{\mathrm{T}}$$

$$\mathbf{h}^e = \{h_1, \ldots, h_D\}^{\mathrm{T}}$$

$$\mathbf{u}^e = \left\{ \{\mathbf{u}^{l_1}\}^{\mathrm{T}}, \ldots, \{\mathbf{u}^{l_{D+1}}\}^{\mathrm{T}} \right\}^{\mathrm{T}}$$

As before, the shape functions $\mathbf{N}^e$ are for D-dimensional space while the shape functions $\mathbf{N}^s$ are for $(D-1)$-dimensional space.

Thus, for proportional loading, upper bound estimates on the applied forces can be expressed as

$$\mathbf{q}^{\text{upper}} = Q^* \mathbf{q}$$

$$\mathbf{h}^{\text{upper}} = Q^* \mathbf{h}$$

where Q^* is the value of the objective function (24) at the optimum point $\mathbf{x}^*$.

When the external traction or body force distribution is unknown, the upper bound technique can be used to estimate the collapse load in terms of mean tractions or mean body forces. In this case, the profile of the mean load ($\bar{\mathbf{q}}$, $\bar{\mathbf{h}}$) should be also specified so as to distinguish the force components. The scaling constraints now become

$$\int_{A_q} \bar{\mathbf{q}}^{\mathrm{T}} \mathbf{u} \, \mathrm{d}A + \int_V \bar{\mathbf{h}}^{\mathrm{T}} \mathbf{u} \, \mathrm{d}V = 1$$

or

$$\sum_{s=1}^{N_q} \overline{W}_q^s + \sum_{e=1}^{E} \overline{W}_h^e = 1 \tag{29}$$

where

$$\overline{W}_q^s = \bar{\mathbf{q}}^{\mathrm{T}} \int_{S_t} \mathbf{u} \, \mathrm{d}S = \bar{\mathbf{q}}^{\mathrm{T}} \int_{S_t} (\mathbf{N}^s \mathbf{u}^s) \, \mathrm{d}S = \bar{\mathbf{q}}^{\mathrm{T}} \left(\int_{S_t} \mathbf{N}^s \, \mathrm{d}S \right) \mathbf{u}^s$$

$$\overline{W}_h^e = \bar{\mathbf{h}}^{\mathrm{T}} \int_{V_e} \mathbf{u} \, \mathrm{d}V = \bar{\mathbf{h}}^{\mathrm{T}} \int_{V_e} \mathbf{N}^e \mathbf{u}^e \, \mathrm{dV} = \bar{\mathbf{h}}^{\mathrm{T}} \left(\int_{V_e} \mathbf{N}^e \, \mathrm{d}V \right) \mathbf{u}^e$$

and the mean surface tractions and body forces are

$$\bar{\mathbf{q}} = \{\bar{q}_1, \ldots, \bar{q}_D\}^{\mathrm{T}}$$

$$\bar{\mathbf{h}} = \{\bar{h}_1, \ldots, \bar{h}_D\}^{\mathrm{T}}$$

Thus, the mean upper bound estimates on the applied forces can be expressed as

$$\bar{\mathbf{q}}^{\text{upper}} = Q^* \bar{\mathbf{q}}$$

$$\bar{\mathbf{h}}^{\text{upper}} = Q^* \bar{\mathbf{h}}$$

Finally, all the various loading constraints may be summarized by grouping Equations (26), (27) and (28) or (29) to give a matrix equation of the form

$$\mathbf{A}_{\text{load}}\mathbf{u} = \mathbf{b}_{\text{load}} \tag{30}$$

2.7. Yield condition

The only requirement for the stresses in the upper bound formulation is that they satisfy the yield condition. For a perfectly plastic solid, we thus have

$$f(\sigma_{ij}) \leqslant 0 \tag{31}$$

where f is a convex function which is dependent on the stresses and certain material parameters. The solution procedure presented here does not depend on a particular type of yield function, but does require it to be both convex and smooth. Convexity is necessary to ensure the solution obtained from the optimization process is the global optimum, and is actually guaranteed by the assumptions of perfect plasticity. Smoothness is essential because the solution algorithm, described later, needs to be able to compute first and second derivatives of the yield function with respect to the unknown stresses. For yield functions which have singularities in their derivatives, such as the Tresca and Mohr–Coulomb criteria, it is necessary to adopt a smooth approximation of the original yield surface.

As the element stresses are assumed to be constant, there is only one yield condition per finite element. This implies that the stresses in the finite element model must satisfy the following inequality:

$$f(\sigma_{ij}^e) \leqslant 0, \quad e = 1, \ldots, E \tag{32}$$

or

$$f_j(\boldsymbol{\sigma}) \leqslant 0, \quad j \in J_\sigma \tag{33}$$

for all E elements. Thus, in total, the yield conditions give rise to E non-linear inequality constraints on the element stresses. Because each element is associated with a unique set of stress variables, it follows that each yield inequality is a function of an uncoupled set of stress variables σ_{ij}^e. This feature, discussed again later, can be exploited to give a very efficient solution algorithm.

2.8. Assembly of constraint equations

The steps necessary to formulate the upper bound theorem as an optimization problem have now been covered, and all that remains is to group the linear constraints for the whole mesh. Using Equations (15), (19) and (30) the various equality constraints may be assembled to give the linear equality constraint matrices according to

$$\mathbf{A}_u = \sum_{d=1}^{Ds} \mathbf{A}_{u\,\text{flow}}^d + \sum_{b=1}^{Bn} \mathbf{A}_{\text{bound}}^b + \sum_{l=1}^{Ld} \mathbf{A}_{\text{load}}^l$$

$$\mathbf{A}_v = \sum_{d=1}^{Ds} \mathbf{A}_{v\,\text{flow}}^d$$

where Ds is the total number of discontinuities, Bn is the total number of boundary nodes subject to prescribed surface tractions, Ld is the total number of loading constraints, and all

coefficients are inserted into the appropriate rows and columns using the usual element assembly rules.

Similarly, the corresponding right-hand side vector $\mathbf{b}$ is assembled according to

$$\mathbf{b} = \sum_{d=1}^{Ds} \mathbf{b}_{\text{flow}}^{d} + \sum_{b=1}^{Bn} \mathbf{b}_{\text{bound}}^{b} + \sum_{l=1}^{Ld} \mathbf{b}_{\text{load}}^{l}$$

Thus the problem of finding a kinematically admissible velocity field which minimizes the internal power dissipation may be stated as

$$\text{minimize} \quad Q = \boldsymbol{\sigma}^{\mathrm{T}}\mathbf{B}\mathbf{u} + \mathbf{c}_u^{\mathrm{T}}\mathbf{u} + \mathbf{c}_v^{\mathrm{T}}\mathbf{v} \quad \text{on } (\mathbf{u}, \mathbf{v}) \tag{34}$$

$$\begin{aligned} \text{subject to} \quad & \mathbf{A}_u\mathbf{u} + \mathbf{A}_v\mathbf{v} = \mathbf{b} \\ & \mathbf{B}\mathbf{u} = \sum_{j \in J_\sigma} \lambda_j \nabla f_j(\boldsymbol{\sigma}) \\ & \lambda_j \geqslant 0, \quad j \in J_\sigma \\ & \lambda_j f_j(\boldsymbol{\sigma}) = 0, \quad j \in J_\sigma \\ & f_j(\boldsymbol{\sigma}) \leqslant 0, \quad j \in J_\sigma \\ & \mathbf{v} \geqslant \mathbf{0} \\ & \mathbf{u} \in \mathrm{R}^{n_u},\ \mathbf{v} \in \mathrm{R}^{n_v},\ \boldsymbol{\sigma} \in \mathrm{R}^{n_\sigma},\ \boldsymbol{\lambda} \in \mathrm{R}^{E} \end{aligned} \tag{35}$$

where $\mathbf{B}$ is an $n_\sigma \times n_u$ global compatibility matrix, $\mathbf{c}_u$ is an n_u-dimensional vector of objective function coefficients for the velocities, $\mathbf{c}_v$ is an n_v-dimensional vector of objective function coefficients for the discontinuity variables, $n_\sigma = ED(D+1)$, $n_u = DN$, N is the number of nodes in the mesh, $n_v = 2D(D-1)Ds$, $\mathbf{A}_u$ is an $r \times n_u$ matrix of equality constraint coefficients for the velocities, $\mathbf{A}_v$ is an $r \times n_v$ matrix of equality constraint coefficients for the discontinuity variables, $r = D^2Ds+$ number of boundary condition and loading constraints, $f_j(\boldsymbol{\sigma})$ are yield functions, λ_j are non-negative multipliers, and $\mathbf{u}$, $\mathbf{v}$ and $\boldsymbol{\sigma}$ are problem unknowns.

3. SOLUTION OF UPPER BOUND OPTIMIZATION PROBLEM

The numerical formulation of the upper bound theorem presented in the previous sections results in a convex optimization problem. Standard optimization theory [6] indicates that, with an appropriate assumption of differentiability, the Kuhn–Tucker optimality conditions are both necessary and sufficient for the solution to a such a problem. This fact means that we can transform (34) into a set of Kuhn–Tucker optimality conditions and explore the possibility of using a rapidly convergent Newton method to solve the resulting system.

Recently, Zouain *et al.* [4] have proposed a feasible point algorithm for solving problems arising from their mixed limit analysis formulation. Although the optimization problem considered by these authors is different from the one considered here, this algorithm can be adapted to suit our requirements. In brief, the Zouain *et al.* scheme uses a quasi-Newton iteration formula for the set of all equalities in the optimality conditions. After each Newton iteration, the resulting search direction is deflected in order to preserve feasibility with respect to the inequality conditions. In our algorithm, we use a new deflection strategy, plus a number of other modifications, to account for the nature of the optimization problem that is generated by

the upper bound formulation. To proceed with the algorithm description, we first write down the Kuhn–Tucker optimality conditions.

3.1. Kuhn–Tucker optimality conditions

To begin, we note that Equations (2)–(5) of (35) are the Kuhn–Tucker optimality conditions for the following problem:

$$\begin{aligned} &\text{maximize} \quad Q = \boldsymbol{\sigma}^{\mathrm{T}}\mathbf{B}\mathbf{u} + \mathbf{c}_u^{\mathrm{T}}\mathbf{u} + \mathbf{c}_v^{\mathrm{T}}\mathbf{v} \quad \text{on } \boldsymbol{\sigma} \\ &\text{subject to} \quad f_j(\boldsymbol{\sigma}) \leqslant 0, \quad j \in J_\sigma \\ &\qquad\qquad \mathbf{u} \in \mathrm{R}^{n_u},\ \mathbf{v} \in \mathrm{R}^{n_v},\ \boldsymbol{\sigma} \in \mathrm{R}^{n_\sigma} \end{aligned}$$

Incorporating this optimization problem in the original problem (34), and dropping the corresponding optimality conditions, leads to

$$\begin{aligned} &\text{maximize} \quad Q = \boldsymbol{\sigma}^{\mathrm{T}}\mathbf{B}\mathbf{u} + \mathbf{c}_u^{\mathrm{T}}\mathbf{u} + \mathbf{c}_v^{\mathrm{T}}\mathbf{v} \quad \text{on } \boldsymbol{\sigma} \\ &\text{minimize} \quad Q = \boldsymbol{\sigma}^{\mathrm{T}}\mathbf{B}\mathbf{u} + \mathbf{c}_u^{\mathrm{T}}\mathbf{u} + \mathbf{c}_v^{\mathrm{T}}\mathbf{v} \quad \text{on } (\mathbf{u}, \mathbf{v}) \\ &\text{subject to} \quad \mathbf{A}_u\mathbf{u} + \mathbf{A}_v\mathbf{v} = \mathbf{b} \\ &\qquad\qquad f_j(\boldsymbol{\sigma}) \leqslant 0, \quad j \in J_\sigma \\ &\qquad\qquad f_j(\mathbf{v}) \leqslant 0, \quad j \in J_v \\ &\qquad\qquad \mathbf{u} \in \mathrm{R}^{n_u},\ \mathbf{v} \in \mathrm{R}^{n_v},\ \boldsymbol{\sigma} \in \mathrm{R}^{n_\sigma} \end{aligned} \tag{36}$$

where the original conditions $\mathbf{v} \geqslant 0$ are, for the sake of convenience, written as $f_j(\mathbf{v}) \leqslant 0$, $j \in J_v$. Since the objective function and the functions f_j are convex, and all equality constraints are linear, the solution to (34), $(\mathbf{u}^*, \mathbf{v}^*, \boldsymbol{\sigma}^*)$, must satisfy the Kuhn–Tucker optimality conditions

$$\begin{aligned} &\mathbf{B}\mathbf{u}^* - \sum_{j \in J_\sigma} \lambda_{\sigma j} \nabla f_j(\boldsymbol{\sigma}^*) = \mathbf{0} \\ &\mathbf{c}_v + \mathbf{A}_v^{\mathrm{T}}\boldsymbol{\mu} + \sum_{j \in J_\sigma} \lambda_{vj} \nabla f_j(\mathbf{v}^*) = \mathbf{0} \\ &\mathbf{B}^{\mathrm{T}}\boldsymbol{\sigma}^* + \mathbf{c}_u + \mathbf{A}_u^{\mathrm{T}}\boldsymbol{\mu} = \mathbf{0} \\ &\mathbf{A}_u\mathbf{u} + \mathbf{A}_v\mathbf{v} = \mathbf{b} \\ &\lambda_{\sigma j} f_j(\boldsymbol{\sigma}^*) = 0, \quad j \in J_\sigma \\ &\lambda_{vj} f_j(\mathbf{v}^*) = 0, \quad j \in J_v \\ &f_j(\boldsymbol{\sigma}^*) \leqslant 0, \quad j \in J_\sigma \\ &f_j(\mathbf{v}^*) \leqslant 0, \quad j \in J_v \\ &\lambda_{\sigma j} \geqslant 0, \quad j \in J_\sigma \\ &\lambda_{vj} \geqslant 0, \quad j \in J_v \\ &\mathbf{u}^* \in \mathrm{R}^{n_u},\ \mathbf{v}^* \in \mathrm{R}^{n_v},\ \boldsymbol{\sigma}^* \in \mathrm{R}^{n_\sigma} \\ &\boldsymbol{\lambda}_\sigma \in \mathrm{R}^{E},\ \boldsymbol{\lambda}_v \in \mathrm{R}^{n_v}, \boldsymbol{\mu} \in \mathrm{R}^{r} \end{aligned} \tag{37}$$

A detailed description of the two stage quasi-Newton strategy for solving the system (37) is given in the following sections.

3.2. Two-stage quasi-Newton algorithm

In Newton's method, the non-linear equations at the current point, k, are linearized using Taylor's series and the resulting system of linear equations is solved to obtain a new point, $k+1$. The process is repeated until the governing system of non-linear equations is satisfied. Applying this procedure to the equalities of system (37) and collecting terms leads to

$$\sum_{j\in J_\sigma} \lambda_{\sigma j}^k \nabla^2 f_j(\boldsymbol{\sigma}^k)(\boldsymbol{\sigma}^{k+1} - \boldsymbol{\sigma}^k) - \mathbf{B}(\mathbf{u}^{k+1} - \mathbf{u}^k) + \sum_{j\in J_\sigma} \lambda_{\sigma j}^{k+1} \nabla f_j(\boldsymbol{\sigma}^k) = \mathbf{B}\mathbf{u}^k \tag{38}$$

$$\sum_{j\in J_v} \lambda_{vj}^k \nabla^2 f_j(\mathbf{v}^k)(\mathbf{v}^{k+1} - \mathbf{v}^k) + \mathbf{A}_v^{\mathrm{T}} \boldsymbol{\mu}^{k+1} + \sum_{j\in J_v} \lambda_{vj}^{k+1} \nabla f_j(\mathbf{v}^k) = -\mathbf{c}_v \tag{39}$$

$$\mathbf{B}^{\mathrm{T}}(\boldsymbol{\sigma}^{k+1} - \boldsymbol{\sigma}^k) + \mathbf{A}_u^{\mathrm{T}} \boldsymbol{\mu}^{k+1} = -\mathbf{B}^{\mathrm{T}} \boldsymbol{\sigma}^k - \mathbf{c}_u \tag{40}$$

$$\mathbf{A}_u(\mathbf{u}^{k+1} - \mathbf{u}^k) + \mathbf{A}_v(\mathbf{v}^{k+1} - \mathbf{v}^k) = \mathbf{0} \tag{41}$$

$$\lambda_{\sigma j}^k \nabla f_j(\boldsymbol{\sigma}^k)(\boldsymbol{\sigma}^{k+1} - \boldsymbol{\sigma}^k) + \lambda_{\sigma j}^{k+1} f_j(\boldsymbol{\sigma}^k) = 0, \quad j \in J_\sigma \tag{42}$$

$$\lambda_{vj}^k \nabla f_j(\mathbf{v}^k)(\mathbf{v}^{k+1} - \mathbf{v}^k) + \lambda_{vj}^{k+1} f_j(\mathbf{v}^k) = 0, \quad j \in J_v \tag{43}$$

where the superscript k denotes the iteration number. Assuming that at $\mathbf{x}^k = (\mathbf{u}^k, \mathbf{v}^k, \boldsymbol{\sigma}^k)$ the multipliers $\boldsymbol{\lambda}^k = (\boldsymbol{\lambda}_\sigma^k, \boldsymbol{\lambda}_v^k)$ are positive (as will be forced by the updating rule), Equations (38)–(43) can be expressed in the more compact form

$$\mathbf{H}_\sigma \mathbf{d}_\sigma - \mathbf{B}\mathbf{d}_u + \mathbf{G}_\sigma^{\mathrm{T}} \boldsymbol{\lambda}_\sigma = \mathbf{r} \tag{44}$$

$$\mathbf{H}_v \mathbf{d}_v + \mathbf{A}_v^{\mathrm{T}} \boldsymbol{\mu} + \mathbf{G}_v^{\mathrm{T}} \boldsymbol{\lambda}_v = -\mathbf{c}_v \tag{45}$$

$$\mathbf{B}^{\mathrm{T}} \mathbf{d}_\sigma + \mathbf{A}_u^{\mathrm{T}} \boldsymbol{\mu} = \mathbf{a} \tag{46}$$

$$\mathbf{A}_v \mathbf{d}_v + \mathbf{A}_u \mathbf{d}_u = \mathbf{0} \tag{47}$$

$$\mathbf{G}_\sigma \mathbf{d}_\sigma + \mathbf{F}_\sigma \boldsymbol{\lambda}_\sigma = \mathbf{0} \tag{48}$$

$$\mathbf{G}_v \mathbf{d}_v + \mathbf{F}_v \boldsymbol{\lambda}_v = \mathbf{0} \tag{49}$$

where

$$\mathbf{H}_\sigma = \sum_{j \in J_\sigma} \lambda_{\sigma j}^k \nabla^2 f_j(\boldsymbol{\sigma}^k)$$

$$\mathbf{H}_v = \sum_{j \in J_v} \lambda_{vj}^k \nabla^2 f_j(\mathbf{v}^k)$$

$$\mathbf{G}_\sigma^{\mathrm{T}} = [\nabla f_1(\boldsymbol{\sigma}^k) \ldots \nabla f_E(\boldsymbol{\sigma}^k)]$$

$$\mathbf{G}_v^{\mathrm{T}} = [\nabla f_1(\mathbf{v}^k) \ldots \nabla f_{n_v}(\mathbf{v}^k)]$$

$$\mathbf{F}_\sigma = \mathrm{diag}(f_j(\boldsymbol{\sigma}^k)/\lambda_{\sigma j}^k) \tag{50}$$

$$\mathbf{F}_v = \mathrm{diag}(f_j(\mathbf{v}^k)/\lambda_{vj}^k) \tag{51}$$

$$\mathbf{r} = \mathbf{B}\mathbf{u}^k$$

$$\mathbf{a} = -\mathbf{B}^{\mathrm{T}}\boldsymbol{\sigma}^k - \mathbf{c}_u$$

and

$$\mathbf{d}_\sigma = \boldsymbol{\sigma}^{k+1} - \boldsymbol{\sigma}^k, \quad \mathbf{d}_v = \mathbf{v}^{k+1} - \mathbf{v}^k, \quad \mathbf{d}_u = \mathbf{u}^{k+1} - \mathbf{u}^k$$

$$\boldsymbol{\mu} = \boldsymbol{\mu}^{k+1}, \quad \boldsymbol{\lambda}_\sigma = \boldsymbol{\lambda}_\sigma^{k+1}, \quad \boldsymbol{\lambda}_v = \boldsymbol{\lambda}_v^{k+1}$$

If we denote

$$\mathbf{T} = \begin{bmatrix} -\mathbf{H}_\sigma & \mathbf{0} & \mathbf{B} & \mathbf{0} & -\mathbf{G}_\sigma^{\mathrm{T}} & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_v & \mathbf{0} & \mathbf{A}_v^{\mathrm{T}} & \mathbf{0} & \mathbf{G}_v^{\mathrm{T}} \\ \mathbf{B}^{\mathrm{T}} & \mathbf{0} & \mathbf{0} & \mathbf{A}_u^{\mathrm{T}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_v & \mathbf{A}_u & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ -\mathbf{G}_\sigma & \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{F}_\sigma & \mathbf{0} \\ \mathbf{0} & \mathbf{G}_v & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{F}_v \end{bmatrix}, \quad \mathbf{y} = \begin{Bmatrix} \mathbf{d}_\sigma \\ \mathbf{d}_v \\ \mathbf{d}_u \\ \boldsymbol{\mu} \\ \boldsymbol{\lambda}_\sigma \\ \boldsymbol{\lambda}_v \end{Bmatrix}, \quad \mathbf{w} = \begin{Bmatrix} -\mathbf{r} \\ -\mathbf{c}_v \\ \mathbf{a} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{Bmatrix}$$

then the system defined by Equations (44)–(49) may be rewritten as

$$\mathbf{T}\mathbf{y} = \mathbf{w} \tag{52}$$

Now we have a highly sparse, symmetric system of $n_\sigma + n_v + n_u + r + E + n_v$ linear equations that can be solved for the unknowns $\mathbf{y}$.

Since our formulation employs an uncoupled set of stress variables for each element, $\mathbf{H}_\sigma$ is block diagonal and $\mathbf{G}_\sigma$ also consists of disjoint blocks. The size of each block in $\mathbf{H}_\sigma$ is equal to the number of stress variables in the corresponding yield constraint (with a maximum of six in the 3D case) and $\mathbf{H}_\sigma^{-1}$ can be computed very quickly element-by-element. The matrices $\mathbf{H}_v$ and $\mathbf{G}_v$ are diagonal. All entries in $\mathbf{H}_v$ are artificially set to a small value to allow $\mathbf{H}_v^{-1}$ and the partial factorization of the matrix $\mathbf{T}$ to be computed.

For some yield functions, the Hessian $\mathbf{H}_\sigma^{-1}$ is not a positive definite matrix. In these cases, we apply a small perturbation $\varepsilon\mathbf{I}$ to restore positive definiteness according to

$$\mathbf{H}_\sigma = \sum_{j \in J_\sigma} \lambda_{\sigma j}^k [\nabla^2 f_j(\boldsymbol{\sigma}^k) + \varepsilon_j \mathbf{I}_j]$$

where $\varepsilon \in [10^{-6}, 10^{-4}]$.

3.2.1. Stage 1: estimating the increment. In the first stage of the algorithm, estimates for $\mathbf{d}_\sigma$, $\mathbf{d}_v$, $\mathbf{d}_u$, $\boldsymbol{\mu}$, $\boldsymbol{\lambda}_\sigma$ and $\boldsymbol{\lambda}_v$ are found by exploiting the above-mentioned features of the matrices $\mathbf{H}_\sigma$, $\mathbf{G}_\sigma$, $\mathbf{H}_v$ and $\mathbf{G}_v$. From (44) we have

$$\mathbf{d}_{\sigma 0} = \mathbf{H}_\sigma^{-1}(\mathbf{B}(\bar{\mathbf{u}} + \mathbf{d}_{u0}) - \mathbf{G}_\sigma^{\mathrm{T}}\boldsymbol{\lambda}_{\sigma 0}) \tag{53}$$

where the subscript 0 signifies that the solution is from stage one of the algorithm and $\mathbf{u}^k$ is replaced by $\bar{\mathbf{u}}$. Substituting (53) into (48) gives

$$\boldsymbol{\lambda}_{\sigma 0} = \mathbf{W}_\sigma^{-1}\mathbf{Q}_\sigma^{\mathrm{T}}\mathbf{B}(\bar{\mathbf{u}} + \mathbf{d}_{u0}) \tag{54}$$

where the diagonal matrix $\mathbf{W}_\sigma$ and the matrix $\mathbf{Q}_\sigma$ are computed from

$$\mathbf{W}_\sigma = \mathbf{G}_\sigma\mathbf{H}_\sigma^{-1}\mathbf{G}_\sigma^{\mathrm{T}} - \mathbf{F}_\sigma$$

$$\mathbf{Q}_\sigma = \mathbf{H}_\sigma^{-1}\mathbf{G}_\sigma^{\mathrm{T}}$$

Next we can eliminate $\boldsymbol{\lambda}_{\sigma 0}$ from (53) using (54) to give

$$\mathbf{d}_{\sigma 0} = \mathbf{D}_\sigma\mathbf{B}(\bar{\mathbf{u}} + \mathbf{d}_{u0}) \tag{55}$$

with

$$\mathbf{D}_\sigma = \mathbf{H}_\sigma^{-1} - \mathbf{Q}_\sigma\mathbf{W}_\sigma^{-1}\mathbf{Q}_\sigma^{\mathrm{T}}$$

Multiplying both sides of (55) by $\mathbf{B}^{\mathrm{T}}$ and taking into account (46) we obtain

$$\mathbf{K}_\sigma\mathbf{d}_{u0} + \mathbf{A}_u^{\mathrm{T}}\boldsymbol{\mu}_0 = \mathbf{a} - \mathbf{K}_\sigma\bar{\mathbf{u}} \tag{56}$$

where

$$\mathbf{K}_\sigma = \mathbf{B}^{\mathrm{T}}\mathbf{D}_\sigma\mathbf{B}$$

is an $n_u \times n_u$ symmetric positive semi-definite matrix [4] whose density is roughly double that of $\mathbf{B}$.

Repeating the above steps for Equations (45), (49) and (47) we finally have

$$\boldsymbol{\lambda}_{v0} = -\mathbf{W}_v^{-1}\mathbf{Q}_v^{\mathrm{T}}(\mathbf{A}_v^{\mathrm{T}}\boldsymbol{\mu}_0 + \mathbf{c}_v) \tag{57}$$

$$\mathbf{d}_{v0} = -\mathbf{D}_v(\mathbf{A}_v^{\mathrm{T}}\boldsymbol{\mu}_0 + \mathbf{c}_v) \tag{58}$$

$$\mathbf{A}_u^{\mathrm{T}}\mathbf{d}_{u0} - \mathbf{K}_v\boldsymbol{\mu}_0 = \mathbf{A}_v\mathbf{D}_v\mathbf{c}_v \tag{59}$$

where

$$\mathbf{W}_v = \mathbf{G}_v\mathbf{H}_v^{-1}\mathbf{G}_v^{\mathrm{T}} - \mathbf{F}_v$$
$$\mathbf{Q}_v = \mathbf{H}_v^{-1}\mathbf{G}_v^{\mathrm{T}}$$
$$\mathbf{D}_v = \mathbf{H}_v^{-1} - \mathbf{Q}_v\mathbf{W}_v^{-1}\mathbf{Q}_v^{\mathrm{T}}$$

and

$$\mathbf{K}_v = \mathbf{A}_v\mathbf{D}_v\mathbf{A}_v^{\mathrm{T}}$$

is an $r \times r$ symmetric positive semi-definite matrix [4]. To start the computational sequence, $\mathbf{d}_{u0}$ and $\boldsymbol{\mu}_0$ are first found using (56) and (59)

$$\begin{Bmatrix} \mathbf{d}_{u0} \\ \boldsymbol{\mu}_0 \end{Bmatrix} = \mathbf{E}^{-1} \begin{Bmatrix} \mathbf{a} - \mathbf{K}_\sigma \bar{\mathbf{u}} \\ \mathbf{A}_v \mathbf{D}_v \mathbf{c}_v \end{Bmatrix} \tag{60}$$

where

$$\mathbf{E} = \begin{bmatrix} \mathbf{K}_\sigma & \mathbf{A}_u^{\mathrm{T}} \\ \mathbf{A}_u & -\mathbf{K}_v \end{bmatrix}$$

The quantities $\mathbf{d}_{\sigma 0}$, $\mathbf{d}_{v0}$, $\boldsymbol{\lambda}_{\sigma 0}$ and $\boldsymbol{\lambda}_{v0}$ are then computed, respectively, using (55), (58), (54) and (57).

For large 3D problems, this computational strategy is up to ten times faster than solving (52) by direct factorization of the matrix $\mathbf{T}$. It should be noted, however, that the condition number of the matrix $\mathbf{E}$ is equal to the square of the condition number of the matrix $\mathbf{T}$ as the optimum solution is approached. Fortunately, experience suggests that double precision arithmetic is sufficient to get an accurate solution using (60) before the matrix $\mathbf{E}$ becomes badly ill-conditioned or numerically singular.

3.2.2. Stage 2: computing a deflected feasible direction. From (48) we see that the increment vector $\mathbf{d}_{\sigma 0}$ found in the first stage of the algorithm is tangential to the active yield constraints $f_j(\boldsymbol{\sigma}^k) = 0$. To preserve feasibility, this search direction must be adjusted by setting the right side of every row in (48) to some negative value which is known as the deflection factor θ. Lyamin [7] and Lyamin and Sloan [8] have described such a deflection technique for lower bound limit analysis. This strategy, which is applicable to the upper bound formulation as well, has a simple geometrical interpretation and works for all common yield criteria.

Consider the component-wise form of Equation (48). Letting $\mathbf{d}_{\sigma j}$ denote the part of the vector $\mathbf{d}_\sigma$ which corresponds to the set of variables of the function f_j, and using the notation $\bar{\lambda}_j$ instead of λ_j^k, we can rewrite (48) in the form

$$\nabla f_{\sigma j}^{\mathrm{T}} \mathbf{d}_{\sigma j} + (f_{\sigma j} / \bar{\lambda}_{\sigma j}) \lambda_{\sigma j} = 0 \tag{61}$$

Using the geometrical definition of the scalar product, this may also be expressed as

$$\|\nabla f_{\sigma j}\| \|\mathbf{d}_{\sigma j}\| \cos \psi_j + (f_{\sigma j} / \bar{\lambda}_{\sigma j}) \lambda_{\sigma j} = 0$$

where ψ_j is the angle between the normal to the jth constraint and the vector $\mathbf{d}_{\sigma j}$. If we now introduce the deflection factor as the product $\|\nabla f_{\sigma j}\| \|\mathbf{d}_{\sigma 0 j}\| \theta_{\sigma j}$, then $\theta_{\sigma j}$ is equal to $\cos \psi_j$ for all active yield constraints. When an equal value of $\theta_{\sigma j}$ is used for these cases, this implies that the angle between the vector $\mathbf{d}_{\sigma j}$ and the tangential plane to the yield surface at the point $\boldsymbol{\sigma}_j^k$, which we term the deflection angle θ_σ, is identical for all active constraints. For many yield surfaces used in limit analysis, the curvature at a given point $\boldsymbol{\sigma}_j^k$ is strongly dependent on the direction $\mathbf{d}_{\sigma j}$. Therefore, it is better to make θ proportional not only to the length of the vector $\mathbf{d}_{\sigma j}$, but also to the curvature $\chi_{\sigma j}$ of the function $f_{\sigma j}$ in the direction $\mathbf{d}_{\sigma j}$. The latter may be estimated as the norm of the derivative of $\nabla f_{\sigma j}$ in the direction $\mathbf{d}_{\sigma 0 j}$ multiplied by $R_{\sigma j}$, where $R_{\sigma j}$ is the distance from the point $\boldsymbol{\sigma}_j^k$ to the axis of the yield surface in the octahedral plane. Combining all the above factors suggests that (61) should be replaced in stage 2 of the algorithm by

$$\nabla f_{\sigma j}^{\mathrm{T}} \mathbf{d}_{\sigma j} + (f_{\sigma j} / \bar{\lambda}_{\sigma j}) \lambda_{\sigma j} = -\theta_\sigma \|\nabla f_{\sigma j}\| \|\mathbf{d}_{\sigma 0 j}\| \chi_{\sigma j} \tag{62}$$

where

$$\theta_\sigma \in (0, 1]$$

$$\chi_{\sigma j} = \max(\chi^0_{\sigma j}/\chi_{\sigma\,\max},\ \chi_{\min})$$

$$\chi^0_{\sigma j} = \frac{\|\nabla^2 f_{\sigma j}(\boldsymbol{\sigma}^k)\mathbf{d}_{\sigma 0 j}\| R_{\sigma j}}{\|\nabla f_{\sigma j}\|\,\|\mathbf{d}_{\sigma 0 j}\|}$$

$$\chi_{\sigma\,\max} = \max_{j \in J_\sigma} \chi^0_{\sigma j}$$

$$\chi_{\min} \in (0, 1]$$

Note that θ_σ is equal to the sine of the maximum deflection angle $\theta_{\sigma\max}$ and can be related analytically to the increment in the objective function $\mathbf{d}^{\mathrm{T}}_{\sigma 0}\mathbf{B}\mathbf{u}_0$ (as will be shown later).

Applying the standard deflection strategy to the linear inequality constraints for the discontinuity variables leads to (49) being replaced for each component by

$$\nabla f^{\mathrm{T}}_{vj}\mathbf{d}_{vj} + (f_{vj}/\bar{\lambda}_{vj})\lambda_{vj} = -\theta_v \|\nabla f_{vj}\|\,\|\mathbf{d}_{v0j}\| \tag{63}$$

where

$$\theta_v = \chi_{\min}$$

Replacing (48) and (49) by, respectively, the matrix form of (62) and (63) furnishes the system of equations for computing the new feasible direction as

$$\mathbf{H}_\sigma \mathbf{d}_\sigma - \mathbf{B}\mathbf{d}_u + \mathbf{G}^{\mathrm{T}}_\sigma \boldsymbol{\lambda}_\sigma = \mathbf{r} \tag{64}$$

$$\mathbf{H}_v \mathbf{d}_v + \mathbf{A}^{\mathrm{T}}_v \boldsymbol{\mu} + \mathbf{G}^{\mathrm{T}}_v \boldsymbol{\lambda}_v = -\mathbf{c}_v \tag{65}$$

$$\mathbf{B}^{\mathrm{T}}\mathbf{d}_\sigma + \mathbf{A}^{\mathrm{T}}_u \boldsymbol{\mu} = \mathbf{a} \tag{66}$$

$$\mathbf{A}_v \mathbf{d}_v + \mathbf{A}_u \mathbf{d}_u = 0 \tag{67}$$

$$\mathbf{G}_\sigma \mathbf{d}_\sigma + \mathbf{F}_\sigma \boldsymbol{\lambda}_\sigma = -\theta_\sigma \boldsymbol{\omega}_\sigma \tag{68}$$

$$\mathbf{G}_v \mathbf{d}_v + \mathbf{F}_v \boldsymbol{\lambda}_v = -\theta_v \boldsymbol{\omega}_v \tag{69}$$

where $\boldsymbol{\omega}_\sigma$ and $\boldsymbol{\omega}_v$ are, respectively, E- and n_v-dimensional vectors with components

$$\omega_{\sigma j} = \|\nabla f_{\sigma j}\|\,\|\mathbf{d}_{\sigma 0 j}\|\chi_{\sigma j}$$

$$\omega_{vj} = \|\nabla f_{vj}\|\,\|\mathbf{d}_{v\,0 j}\|$$

Applying the same computational sequence that was used to solve the system (44)–(49), the solution of Equations (64)–(69) can be obtained using the steps

$$\mathbf{d}_\sigma = \mathbf{D}_\sigma \mathbf{B}(\bar{\mathbf{u}} + \mathbf{d}_u) - \theta_\sigma \mathbf{Q}_\sigma \mathbf{W}^{-1}_\sigma \boldsymbol{\omega}_\sigma \tag{70}$$

$$\boldsymbol{\lambda}_\sigma = \mathbf{W}^{-1}_\sigma \mathbf{Q}^{\mathrm{T}}_\sigma \mathbf{B}(\bar{\mathbf{u}} + \mathbf{d}_u) + \theta_\sigma \mathbf{W}^{-1}_\sigma \boldsymbol{\omega}_\sigma$$

$$\mathbf{d}_v = -\mathbf{D}_v(\mathbf{A}^{\mathrm{T}}_v \boldsymbol{\mu} + \mathbf{c}_v) - \theta_v \mathbf{Q}_v \mathbf{W}^{-1}_v \boldsymbol{\omega}_v$$

$$\boldsymbol{\lambda}_v = -\mathbf{W}^{-1}_v \mathbf{Q}^{\mathrm{T}}_v (\mathbf{A}^{\mathrm{T}}_v \boldsymbol{\mu} + \mathbf{c}_v) + \theta_v \mathbf{W}^{-1}_v \boldsymbol{\omega}_v$$

where $\mathbf{d}_u$ and $\boldsymbol{\mu}$ are first found from

$$\begin{Bmatrix} \mathbf{d}_u \\ \boldsymbol{\mu} \end{Bmatrix} = \mathbf{E}^{-1} \begin{Bmatrix} \mathbf{a} - \mathbf{K}_\sigma \bar{\mathbf{u}} + \theta_\sigma \mathbf{B}^{\mathrm{T}} \mathbf{Q}_\sigma \mathbf{W}_\sigma^{-1} \boldsymbol{\omega}_\sigma \\ \mathbf{A}_v \mathbf{D}_v \mathbf{c}_v + \theta_v \mathbf{A}_v \mathbf{Q}_v \mathbf{W}_v^{-1} \boldsymbol{\omega}_v \end{Bmatrix}$$

To derive an expression for θ_σ we multiply both sides of (70) by $(\bar{\mathbf{u}} + \mathbf{d}_{u0})\mathbf{B}^{\mathrm{T}}$, then use (54) and (55) to obtain

$$\theta_\sigma = (\mathbf{d}_{\sigma 0}^{\mathrm{T}} \mathbf{B}\mathbf{u} - \mathbf{d}_\sigma^{\mathrm{T}} \mathbf{B}\mathbf{u}_0)/\boldsymbol{\lambda}_0^{\mathrm{T}} \boldsymbol{\omega}$$

where

$$\mathbf{u}_0 = \bar{\mathbf{u}} + \mathbf{d}_{u0}, \quad \mathbf{u} = \bar{\mathbf{u}} + \mathbf{d}_u$$

Assuming $\mathbf{u} \approx \mathbf{u}_0$ and imposing the condition that the Stage 1 and Stage 2 changes in the element dissipated power are related by a fixed parameter $\beta \in (0, 1]$, such that $\mathbf{d}_\sigma^{\mathrm{T}} \mathbf{B}\mathbf{u} = \beta \mathbf{d}_{\sigma 0}^{\mathrm{T}} \mathbf{B}\mathbf{u}_0$, θ_σ can be computed as

$$\theta_\sigma = (1 - \beta)\mathbf{d}_{\sigma 0}^{\mathrm{T}} \mathbf{B}\mathbf{u}_0 / \boldsymbol{\lambda}_{\sigma 0}^{\mathrm{T}} \boldsymbol{\omega}$$

This gives the required deflection angle for each iteration. A typical value for β, which is used for all the runs in this paper, is 0.7.

3.3. Initialization

So far we have assumed that the current solution ($\mathbf{u}^k$, $\mathbf{v}^k$, $\boldsymbol{\sigma}^k$, $\boldsymbol{\lambda}_\sigma^k$, $\boldsymbol{\lambda}_v^k$) is a feasible point. To start the iterations, we therefore need a procedure which will furnish an initial feasible point ($\mathbf{u}^0$, $\mathbf{v}^0$, $\boldsymbol{\sigma}^0$, $\boldsymbol{\lambda}_\sigma^0$, $\boldsymbol{\lambda}_v^0$). For $\boldsymbol{\sigma}^0$, $\boldsymbol{\lambda}_\sigma^0$, $\boldsymbol{\lambda}_v^0$ we can start with

$$\boldsymbol{\sigma}^0 = \mathbf{0}, \quad \boldsymbol{\lambda}_\sigma^0 = \mathbf{1}, \quad \boldsymbol{\lambda}_v^0 = \mathbf{1}$$

but the pair ($\mathbf{u}^0$, $\mathbf{v}^0$) must satisfy the equality constraints

$$\mathbf{A}_u \mathbf{u}^0 + \mathbf{A}_v \mathbf{v}^0 = \mathbf{b} \tag{71}$$

In order to convert (71) to a linear system with a square symmetric coefficient matrix, it can be augmented by a set of dummy variables $\boldsymbol{\eta}$ to become

$$\begin{bmatrix} \mathbf{I} & \mathbf{A}^{\mathrm{T}} \\ \mathbf{A} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \mathbf{z}^0 \\ \boldsymbol{\eta} \end{Bmatrix} = \begin{Bmatrix} \mathbf{0} \\ \mathbf{b} \end{Bmatrix}$$

where

$$\mathbf{A} = [\mathbf{A}_u \mathbf{A}_v], \ \mathbf{z}^0 = \left\{ \mathbf{u}^{0\mathrm{T}}, \mathbf{v}^{0\mathrm{T}} \right\}^{\mathrm{T}}$$

Writing the equations in this form allows us to use the same solver as for the main system of equations (44)–(49). They can be readily solved using the relations

$$\mathbf{z}_0 = -\mathbf{A}^{\mathrm{T}} \boldsymbol{\eta}$$

$$\mathbf{A}\mathbf{A}^{\mathrm{T}} \boldsymbol{\eta} = -\mathbf{b}$$

When factorizing the symmetric matrix $\mathbf{A}\mathbf{A}^{\mathrm{T}}$, however, we need to be aware of possible redundancies in the matrix of equalities $\mathbf{A}$. These are detected when a zero (or near zero) entry occurs on the diagonal of the factored form of $\mathbf{A}\mathbf{A}^{\mathrm{T}}$, and are marked in a single factorization pass. If detected, the redundant equalities are deleted from further consideration in the algorithm (except when checking the feasibility of the current solution) and $\mathbf{A}\mathbf{A}^{\mathrm{T}}$ is factorized

afresh to obtain values of $\boldsymbol{\eta}$ and the initial feasible point $\mathbf{z}^0$. Once $\mathbf{z}^0$ has been obtained, we then substitute it into the inequalities and compute the scalar

$$\alpha = \max_{j \in J_v} f_{vj}(\mathbf{z}^0)$$

If $\alpha \leqslant 0$, $\mathbf{z}^0$ satisfies all the inequalities and the initial search direction $\mathbf{d}_0 = \{\mathbf{d}_u^{0\mathrm{T}}, \mathbf{d}_v^{0\mathrm{T}}\}^{\mathrm{T}}$ can be found by solving

$$\begin{bmatrix} \mathbf{I} & \mathbf{A}^{\mathrm{T}} \\ \mathbf{A} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \mathbf{d}^0 \\ \boldsymbol{\eta} \end{Bmatrix} = \begin{Bmatrix} \mathbf{c} \\ \mathbf{0} \end{Bmatrix}$$

where $\mathbf{c} = \{\mathbf{c}_u^{\mathrm{T}}, \mathbf{c}_v^{\mathrm{T}}\}^{\mathrm{T}}$ and $\boldsymbol{\eta}$ is again a set of dummy variables that are introduced for computational convenience. We can then start the iteration process. If, on the other hand, $\alpha > 0$, this signifies that $\mathbf{z}^0$ does not satisfy all the inequalities and it is necessary to solve the so-called 'phase one' problem

$$\begin{aligned} &\text{minimize } \alpha_{r+1} \\ &\text{subject to } \mathbf{A}_\alpha \mathbf{z}_\alpha = \mathbf{b} \\ &\qquad \alpha_{j+1} - \alpha_j = 0, \quad j \in J_v \\ &f_{vj}(\mathbf{z}_\alpha) - \alpha_j \leqslant 0, \quad j \in J_v \\ &\qquad -\alpha_{r+1} \leqslant 0 \\ &\mathbf{z}_\alpha \in \mathrm{R}^{n_u + n_v + r + 1} \end{aligned} \tag{72}$$

where $\mathbf{z}_\alpha^{\mathrm{T}} = \{\mathbf{z}^{\mathrm{T}}, \alpha_1, \ldots, \alpha_{r+1}\}$ and $\mathbf{A}_\alpha$ is the matrix $\mathbf{A}$ augmented by $r+1$ columns. Because of the similarity of problem (72) with problem (34), it can be solved using the same two-stage algorithm but dropping the yield inequalities. If the optimal solution to the phase one problem (72) is such that $\alpha = \alpha_1 = \cdots = \alpha_{r+1} = 0$, then $\mathbf{z}^0$ is found by discarding the last $r+1$ entries in $\mathbf{z}_a$. Otherwise, the original upper bound optimization problem has no feasible solution and the process is halted.

3.4. Convergence criteria

At the end of each Stage 1 iteration, various convergence tests are performed on the set of optimality conditions (37). Noting that a11 constraints of the optimization problem (36) are satisfied at every stage in the solution process, our dimensionless convergence checks are of the form

$$\left\| \mathbf{B}\mathbf{u} - \sum_{j \in J_\sigma} \lambda_{\sigma 0 j} \nabla f_j(\boldsymbol{\sigma}) \right\| \leqslant \varepsilon_u \|\mathbf{B}\mathbf{u}\| \tag{73}$$

$$\lambda_{\sigma 0 j} \geqslant -\varepsilon_\lambda \lambda_{\sigma\,\max}, \quad j \in J_\sigma \tag{74}$$

$$\lambda_{\sigma 0 j} \leqslant \varepsilon_\lambda \lambda_{\sigma\,\max} \quad \text{if } f_j(\boldsymbol{\sigma}) < -\varepsilon_f, \quad j \in J_\sigma \tag{75}$$

$$\left\| \mathbf{c}_v + \mathbf{A}_v^{\mathrm{T}} \boldsymbol{\mu} + \sum_{j \in J_v} \lambda_{v 0 j} \nabla f_j(\mathbf{v}) \right\| \leqslant \varepsilon_u \|\mathbf{c}_v\| \tag{76}$$

$$\lambda_{v 0 j} \geqslant -\varepsilon_\lambda \lambda_{v\,\max}, \quad j \in J_v \tag{77}$$

$$\lambda_{v 0 j} \leqslant \varepsilon_\lambda \lambda_{v\,\max} \quad \text{if } f_j(\mathbf{v}) < -\varepsilon_f, \quad j \in J_v \tag{78}$$

where

$$\lambda_{\sigma\,\max} = \max_{j\in J_\sigma} \lambda_{\sigma 0j} | \lambda_{\sigma 0j} > 0$$

$$\lambda_{v\,\max} = \max_{j\in J_v} \lambda_{v 0j} | \lambda_{v 0j} > 0$$

and the three different tolerances are typically defined so that ε_u, ε_λ, $\varepsilon_f \in [10^{-3}, 10^{-2}]$. All conditions (73)–(78) must be satisfied before convergence is deemed to have occurred.

3.5. Line search

As the deflected search direction $\mathbf{d}_v$ automatically obeys all linear equality constraints, the only requirement for the line search is that the resulting solution, $\mathbf{v}$, satisfies all inequality constraints. In our implementation, we prevent any inequality becoming active in a single iteration by setting the maximum step size s_v according to

$$s_v = \min\left[\min_{j\in J_v} s_{vj} \,|\, f_j(\mathbf{v}_j + s_{vj}\mathbf{d}_{vj}) = \delta_{vf} f_j(\mathbf{v}_j),\ 1\right]$$

where the parameter δ_{vf} is forced to converge to zero by means of the rule

$$\delta_{vf} = \min\left[\delta_f^0,\ \|\mathbf{c}_v^{\mathrm{T}}\mathbf{d}_{v0}\| / \|\mathbf{c}_v^{\mathrm{T}}\mathbf{v}\|\right]$$

and $\delta_f^0 \in (0,\ 1)$ is a given control parameter. As there are no equality constraints on the stresses in the upper bound formulation, the sole requirement for the line search is that the resulting stress field, $\boldsymbol{\sigma}$, satisfies all inequality yield constraints

$$s_{\sigma j} = \min[s_{\sigma j} \,|\, f_j(\boldsymbol{\sigma}_j + s_{\sigma j}\mathbf{d}_{\sigma j}) = \delta_{\sigma f} f_j(\boldsymbol{\sigma}_j),\ s_v],\ j \in J_\sigma$$

where the parameter $\delta_{\sigma f}$ is forced to converge to zero by means of the rule

$$\delta_{\sigma f} = \min[\delta_f^0,\ \|\mathbf{d}_{\sigma 0}^{\mathrm{T}}\mathbf{B}\mathbf{u}_0\| / \|\boldsymbol{\sigma}^{\mathrm{T}}\mathbf{B}\mathbf{u}_0\|]$$

Preventing inequality constraints from becoming active in a single iteration gives smoother convergence, since it avoids abrupt changes in the number of active constraints.

3.6. Updating

At the conclusion of each iteration, the solution vector ($\mathbf{u}$, $\mathbf{v}$, $\boldsymbol{\sigma}$) is updated using the computed search direction ($\mathbf{d}_u$, $\mathbf{d}_v$, $\mathbf{d}_\sigma$) and the corresponding step size (s_v, $\mathbf{s}_\sigma$) according to

$$\mathbf{u} = \mathbf{u} + s_v\mathbf{d}_u$$

$$\mathbf{v} = \mathbf{v} + s_v\mathbf{d}_v$$

$$\boldsymbol{\sigma}_j = \boldsymbol{\sigma}_j + s_{\sigma j}\mathbf{d}_{\sigma j}, \quad j \in J_\sigma$$

Next, the vector of Lagrange multipliers ($\boldsymbol{\lambda}_\sigma$, $\boldsymbol{\lambda}_v$) is updated so that each component is kept strictly positive. This restriction is necessary because each component is used as a denominator

in expressions (50), (51) and we wish to maintain negative entries on the diagonals of $\mathbf{F}_\sigma$ and $\mathbf{F}_v$ so that the deflection strategy of (62), (63) is able to maintain feasibility. The rule for updating the Lagrange multipliers is

$$\lambda_{\sigma j} = \max[\lambda_{\sigma 0 j},\ \delta_{\sigma\lambda}\lambda_{\sigma\,\max}], \quad j \in J_\sigma$$

$$\lambda_{vj} = \max[\lambda_{v0j},\ \delta_{v\lambda}\lambda_{v\,\max}], \quad j \in J_v$$

with

$$\delta_{\sigma\lambda} = \min[\delta_\lambda^0,\ \|\mathbf{d}_\sigma\|/\|\boldsymbol{\sigma}\|]$$

$$\delta_{v\lambda} = \min[\delta_\lambda^0,\ \|\mathbf{d}_v\|/\|\mathbf{v}\|]$$

where $\delta_\lambda^0 \in [10^{-4},\ 10^{-2}]$ is a prescribed tolerance. This rule allows the λ_j to converge to a positive value if λ_{0j} is positive, but sets λ_j to a small positive value if λ_{0j} is tending to zero.

4. NUMERICAL RESULTS

We now study some typical two- and three-dimensional soil stability problems to demonstrate the efficiency of the new upper bound algorithm. All of the examples are for Tresca or Mohr–Coulomb criteria. The latter, which is widely used to model the failure of cohesive–frictional soils, can be troublesome for non-linear optimization techniques because of its apex and corners which generate gradient singularities. In our implementation, these are smoothed using the hyperbolic approximation of Abbo and Sloan [9]. The yield surface curvatures which result from this model vary strongly, but are well controlled by the special deflection technique described in Section 3.4. The influence of the smoothing approximation on the resulting bounds has been evaluated by adjusting the values of the smoothing parameters. Although problem dependent, it was found that this influence is of the same order of magnitude as the smoothing parameters themselves. Therefore, a value of 10^{-6} was used to round the apex and a transition angle of 29.5° was used to round the corners (see Reference [9]).

For the two-dimensional cases, results are presented for three different meshes which are classified as coarse, medium and fine. Because of the improved upper bounds that they generate, velocity discontinuities are used at all interelement boundaries.

To solve the sparse symmetric system of equations (60) efficiently, we use the MA27 multifrontal code developed at Harwell. This is a direct solution method and employs threshold pivoting to maintain stability and sparsity in the factorization.

All CPU times given in the tables are for a Dell Precision 220 workstation with a Pentium III 800EB MHz processor operating under Windows 98 second edition. The compiler used was Visual Fortran Standard Edition 6.1.0 and full optimization was employed.

4.1. Rigid strip footing on cohesive–frictional soil

The exact collapse pressure for a rigid strip footing on a weightless cohesive–frictional soil with no surcharge is given by the Prandtl [10] solution

$$q/c' = (\exp(\pi \tan \phi')\tan^2(45 + \phi'/2) - 1)\cot \phi'$$

where c' and ϕ' are, respectively, the effective cohesion and the effective friction angle. For a soil with a friction angle of $\phi' = 35°$ it gives $q/c' = 46.14$. The velocity boundary conditions and

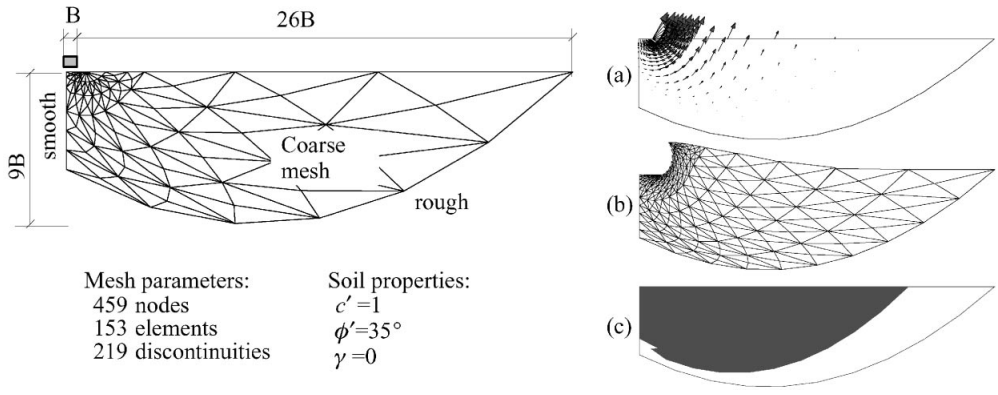


Figure 9. Upper bound mesh and collapse pattern for strip footing problem, (a) velocities, (b) deformations, (c) plastic zones.

Table I. Upper bounds for smooth strip footing on weightless cohesive–frictional soil ($c' = 1$, $\phi' = 35°$, $\gamma = 0$, $NSID$ = number of sides in linearized yield surface).

Mesh	Quantity	LP ($NSID = 24$)	NLP ($\varepsilon = 0.01$)	LP/NLP
Coarse	q/c'	50.48	49.82	1.01
459 nodes	No. of iterations	1396	28	49.9
153 elements	CPU (s)	6.76	1.82	3.7
219 disc	Error (%)	+9.4	+8.0	1.18
Medium	q/c'	48.77	48.00	1.02
1365 nodes	No. of iterations	6875	29	237
455 elements	CPU (s)	133.7	5.5	24.3
665 disc	Error (%)	+5.7	+4.0	1.43
Fine	q/c'	48.03	47.30	1.02
2751 nodes	No. of iterations	16771	30	559
917 elements	CPU (s)	2105	13.6	155
1351 disc	Error (%)	+4.1	+2.5	1.65

material properties used in the upper bound analysis, together with one of the meshes, are shown in Figure 9.

The results presented in Table I compare the performance of the new non-linear formulation with that of the linear programming formulation described by Sloan [2,11] and Sloan and Kleeman [3]. The new algorithm demonstrates fast convergence to the optimum solution and, most importantly, the number of iterations required is essentially independent of the problem size.

For the coarsest mesh, the non-linear formulation is nearly four times faster than the linear programming formulation and gives an upper bound limit pressure which is 1 per cent better. For the finest mesh the speed advantage of the non-linear procedure is more dramatic, with a 155-fold reduction in the CPU time. In this case, the upper bound limit pressure is 2.5 per cent

Table II. Results for smooth strip footing problem with different convergence tolerances $\varepsilon = \varepsilon_c = \varepsilon_\lambda = \varepsilon_f$.

LP				NLP			
CPU	Iterations	*NSID*	q/c'	q/c'	ε	Iterations	CPU
6.8	1396	24	50.48	50.49	0.05	20	1.04
30.4	3537	100	49.84	49.82	0.01	28	1.82
260	5990	200	49.82	49.80	0.005	30	1.84
405	8722	300	49.81	49.80	0.001	48	2.47

above the exact limit pressure and the analysis uses just 13.6 s of CPU time. Because the number of iterations with the new algorithm is essentially constant for all cases, its CPU time grows almost linearly with the problem size. This is in contrast to the linear programming formulation, where the iterations and CPU time grow at a much faster rate. The CPU time savings from the non-linear method become larger as the problem size increases.

The tests in Table I were all run with the convergence tolerances (defined in Section 3.6) set to $\varepsilon_c = \varepsilon_\lambda = \varepsilon_f = \varepsilon = 0.01$. The influence of changing these values, for the coarse mesh illustrated in Figure 9, is shown in Table II. Clearly, a uniform tolerance value of 0.01 is sufficiently accurate for practical calculations with the non-linear algorithm. Indeed, tightening the tolerances from 0.05 to 0.001 affects the collapse pressure error only marginally, reducing it from +9.4 to +8.0 per cent. For the linear programming formulation, at least 24 sides (*NSID*) need to be used in the yield surface linearization, but this value may increase with increasing friction angles. Increasing *NSID*, however, dramatically increases the solution cost. To obtain a solution with the same accuracy as the non-linear programming method using $\varepsilon = 0.01$, the linear programming formulation requires 100 sides in the yield surface linearization and over a 16-fold increase in CPU time. As expected, the linear and non-linear programming methods give the same collapse pressure when the former is used with an accurate linearization and the latter is used with stringent convergence tolerances.

4.2. *Thick cylinder expansion in cohesive–frictional soil*

We now consider the collapse of a thick cylinder of cohesive-frictional soil subject to a uniform internal pressure. The exact solution to this problem has been obtained by Yu [12] and is given by the formula

$$p/c' = \frac{Y + (\alpha - 1)p_0}{c'(\alpha - 1)}\Big((b/a)^{(\alpha-1)/\alpha} - 1\Big) + p_0/c'$$

where p is the collapse pressure, p_0 is the initial hydrostatic pressure acting throughout the soil, a and b are the inner and outer radii of the cylinder, and $Y = 2c'\cos\phi'/(1 - \sin\phi')$ and $\alpha = \tan^2(45 + \phi'/2)$ are material constants. For $p_0 = 0$, $b/a = 1.5$, $c' = 1$ and $\phi' = 30^\circ$, the exact collapse pressure is $p = 0.5376$. One of the meshes used for the upper bound analyses is shown in Figure 10, together with the assumed loading and boundary conditions. Because of symmetry, only a 90° sector of the cylinder is discretized. As in all numerical examples considered in the paper, velocity discontinuities are included between all adjacent elements to minimize the upper bound.

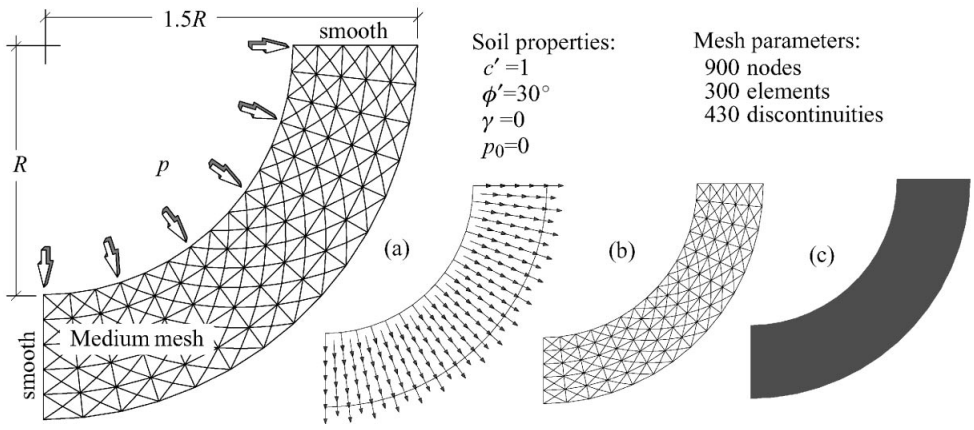


Figure 10. Upper bound mesh and collapse pattern for expansion of thick cylinder, (a) velocities, (b) deformations, (c) plastic zones.

Table III. Upper bounds for expansion of thick cylinder of cohesive–frictional soil ($p_0 = 0$, $b/a = 1.5$, $c' = 1$, $\phi' = 30^\circ$).

Mesh	Quantity	LP($NSID = 36$)	NLP($\varepsilon = 0.01$)	LP/NLP
Coarse	q/c'	0.5535	0.5527	1.001
36 nodes	No. of iterations	79	16	3.44
12 elements	CPU (s)	0.06	0.28	0.21
14 disc	Error (%)	+2.96	+2.80	1.06
Medium	q/c'	0.5394	0.5384	1.002
900 nodes	No. of iterations	4221	23	183.5
300 elements	CPU (s)	60.0	2.59	23.2
430 disc	Error (%)	+0.33	+0.15	2.2
Fine	q/c'	0.5386	0.5378	1.001
3600 nodes	No. of iterations	22325	20	1116
1200 elements	CPU (s)	12553	11.2	1121
1760 disc	Error (%)	+0.18	+0.04	4.5

The results for the three different meshes, shown in Table III, demonstrate similar trends to the footing problem in regard to the performance of the non-linear and linear programming techniques. The former needs, respectively, just 16, 23 and 20 iterations to obtain the solution for the coarse, medium and fine meshes, which suggests that its iteration counts are again largely independent of the problem size. This characteristic makes the non-linear programming procedure very efficient and, for the fine mesh, it takes only 11.2 s of CPU time to obtain an upper bound with 0.04 per cent error. In contrast, the iterations required by the linear programming formulation grow rapidly as the mesh is refined. Indeed, for the fine mesh, this procedure requires 22 325 iterations and a CPU time which is roughly 1120 times greater than that of the non-linear programming method.

4.3. Stability of an unsupported vertical cut under plane strain

All the problems considered so far have been concerned with the optimization of surface tractions applied along a specified part of the domain boundary. We now consider the case of a long unsupported vertical cut in a purely cohesive Tresca soil, where the unit weight γ is optimized for a given cohesion c_u and height of cut H. Since the failure of this problem is governed by the stability number $N_s = H\gamma/c_u$, the analysis may also be interpreted as finding the maximum height of a cut for a given unit weight and undrained cohesion. One of the three meshes used to analyse the cut is shown in Figure 11, together with the imposed velocity boundary conditions. Although this is a classical stability problem for construction in undrained clays, its exact solution remains unknown. To date, the best upper and lower bounds on the stability number for a plane strain vertical cut are $N_s^+ = 3.785864$ and $N_s^- = 3.77200$, and were obtained, respectively, by Pastor *et al.* [13] using a linear programming formulation with the IBM Optimization Subroutine Library and by Lyamin [7] using a quasi-Newton two-stage non-linear programming algorithm. Pastor *et al.* [13] report their CPU times for a processor which is about 3 times slower than the one used for our analyses, and it is thus interesting to compare the performance of their linear programming upper bound formulation against the performance of the new non-linear programming formulation presented here.

According to Pastor *et al.* [13], their linear programming formulation required about 1000 h ($\sim$60 000 min) of CPU time to solve for a mesh with $\sim$3700 triangular elements. In comparison, our non-linear programming scheme required only 2 min 23 s of CPU time for a mesh with 4230 elements (not shown here) to give a slightly better upper bound of $N_s^+ = 3.78445$. This solution would now appear to be the best known upper bound for the vertical cut problem, and highlights the outstanding speed and accuracy advantages to be gained from the non-linear programming formulation.

The results for the various meshes, shown in Table IV, again compare the performance of the new non-linear programming method with that of the active set linear programming method described in [2,3,11]. As in all previous examples, the iteration counts for the linear programming formulation grow rapidly as the mesh is refined, while the iteration counts for

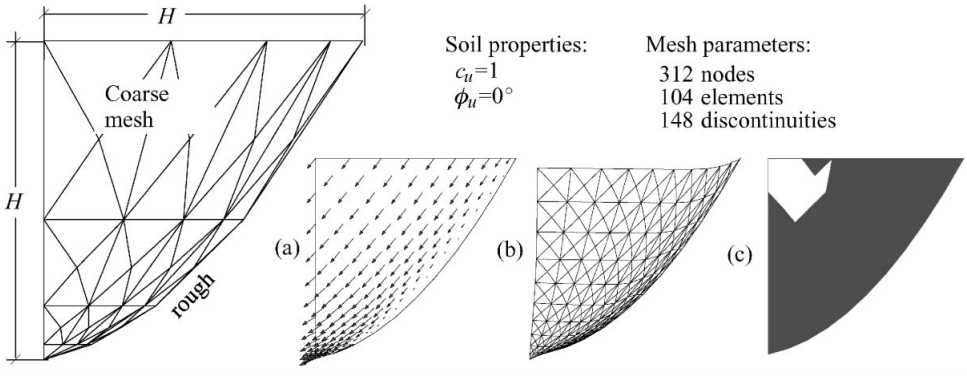


Figure 11. Upper bound mesh and collapse pattern for vertical cut problem (a) velocities, (b) deformations, (c) plastic zones.

Table IV. Upper bounds for plane strain vertical cut in purely cohesive soil ($c_u = 1$, $\phi_u = 0^\circ$).

Mesh	Quantity	LP($NSID = 24$)	NLP($\varepsilon = 0.01$)	LP/NLP
Coarse	N_s	3.867	3.855	1.003
156 nodes	No. of iterations	402	26	15.5
52 elements	CPU (s)	0.93	0.55	1.69
72 disc	Error (%)*	+2.5	+2.2	1.14
Medium	N_s	3.818	3.804	1.004
1110 nodes	No. of iterations	8272	30	278
370 elements	CPU (s)	132.3	4.34	30.5
540 disc	Error (%)*	+1.2	+0.85	1.41
Fine	N_s	3.808	3.794	1.004
2928 nodes	No. of iterations	31910	37	862
976 elements	CPU (s)	2303	13.5	171
1440 disc	Error (%)*	+0.96	+0.59	1.63

*Measured with respect to Lyamin [7] lower bound of $N_s^- = 3.77200$.

the non-linear programming method are essentially constant. For the finest mesh, these two solution schemes give upper bounds of $N_s^+ = 3.808$ and $N_s^+ = 3.794$, respectively, but the non-linear programming method is over 170 times faster. The best upper bound of $N_s^+ = 3.794$, differs by just 0.59 per cent from Lyamin's [7] lower bound of $N_s^- = 3.772$, and uses only 13.5 s of CPU time. In this example, the slight discrepancy between the linear programming and non-linear programming solutions is caused by the yield surface linearization used in the former.

4.4. *Stability of an unsupported circular excavation*

The stability of an unsupported circular excavation is another important practical problem whose exact solution remains unknown. For the case of undrained clays with a Tresca yield criterion, solutions have been obtained by Bjerrum and Eide [14], Prater [15], Pastor and Turgeman [16], Pastor [17], Britto and Kusakabe [18], and Sloan [19]. No rigorous results, however, appear to be available for the case of cohesive–frictional soils.

We now consider the stability of a circular cut, of height H and radius R, in a cohesive-frictional soil whose behaviour is governed by the Mohr–Coulomb yield criterion. Solutions from the new upper bound formulation are combined with solutions from a recently developed lower bound technique [7,8] to bracket the exact stability number from above and below. To exploit its inherent symmetry we consider only a 15° sector of the problem, as shown in Figure 12, and restrict our attention to the case of wall failure. As discussed by Britto and Kusakabe [18], this mode of failure is critical for circular cuts in clay provided H/R does not exceed 7.

The computed stability numbers are presented in Table V and Figure 13 for a range of soil friction angles and height to radius ratios H/R. For the special case of $\phi = 0^\circ$, which corresponds to a cut in a (Tresca) undrained clay, the new upper bounds improve substantially on the upper bounds of Britto and Kusakabe [18]. For all cases, including those with non-zero

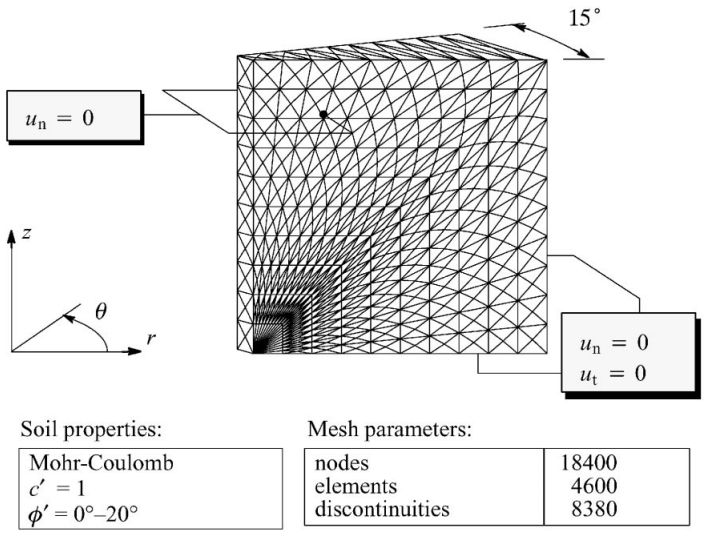


Figure 12. Typical upper bound mesh for circular excavation.

Table V. Upper and lower bounds for circular excavation in cohesive–frictional soil.

ϕ°	Stability number $\gamma H/c'$					
	$H/R = 1$		$H/R = 2$		$H/R = 3$	
	LB	UB	LB	UB	LB	UB
0	5.09	5.26	5.95	6.14	6.59	6.83
10	6.48	6.73	7.82	8.20	9.00	9.42
20	8.30	8.85	10.49	11.26	12.54	13.43

friction angles, the new upper and lower bounds bracket the exact stability number to 7 per cent or better. For $H/R = 1$ and $\phi = 0^\circ$, Pastor and Turgeman's [16] lower bound of $N_s^- = 3.464$ has been raised to $N_s^- = 5.09$. For the same H/R ratio Pastor's [17] upper bound has been reduced from $N_s^+ = 5.298$ to $N_s^+ = 5.26$. No rigorous results could be found for frictional soils, so the solutions presented in Table V are new. The gap between the upper and lower bound estimates of the computed stability numbers tends to widen as the friction angle increases, with an average discrepancy of 6.8 per cent for $\phi = 20^\circ$ compared to 3.3 per cent for $\phi = 0^\circ$.

The finest upper bound mesh for the circular excavation comprised 26 688 nodes, 6672 elements and 12 168 discontinuities. Bearing in mind that each node has three unknown velocities, each element has six unknown stresses, and each discontinuity has 12 unknown variables, this grid generates a very large optimization problem and it comes as no surprise that the method is computationally demanding. On average, the CPU time for the analyses shown in Figure 12 was 5000 s for a Dell Precision 220 workstation with a Pentium III 800EB MHz

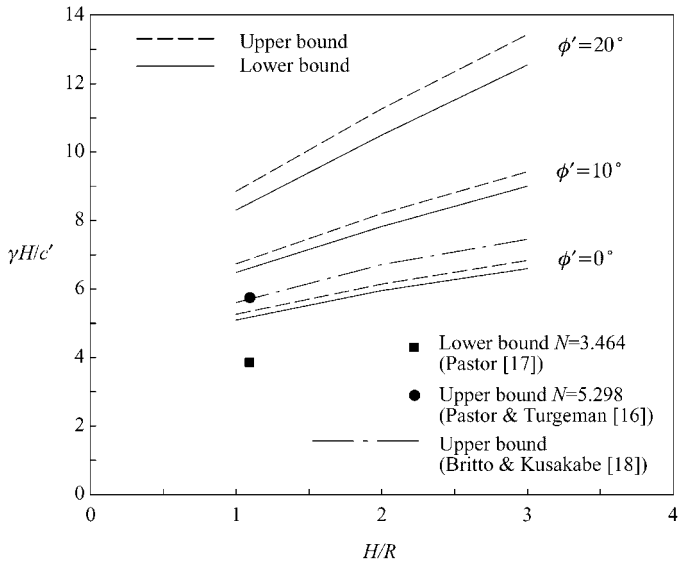


Figure 13. Upper and lower bounds for circular excavation.

processor operating under Windows 98 second edition. Although large, these timings are certainly competitive with those that would be needed for a three-dimensional incremental analysis with the displacement finite element method. Moreover, the difference between the upper and lower bounds provides a direct estimate of the error in the computed collapse load.

5. CONCLUSIONS

A new formulation for performing upper bound limit analysis has been presented. Detailed numerical results show that the approach is fast and accurate for a broad spectrum of two- and three-dimensional stability problems. In terms of CPU time, it is vastly superior to a widely used linear programming formulation, especially for large scale applications where the speedup can be several orders of magnitude. Indeed, the new upper bound scheme is so efficient that large three-dimensional problems can now be analysed on a desktop PC.

REFERENCES

1. Bottero A, Negre R, Pastor J, Turgeman S. Finite element method and limit analysis theory for soil mechanics problems. *Computer Methods in Applied Mechanics and Engineering* 1980; **22:**131–149.
2. Sloan SW. Upper bound limit analysis using finite elements and linear programming. *International Journal for Numerical and Analytical Methods in Geomechanics* 1989; **13:**263–282.
3. Sloan SW, Kleeman PW. Upper bound limit analysis using discontinuous velocity fields. *Computer Methods in Applied Mechanics and Engineering* 1995; **127:**293–314.
4. Zouain N, Herskovits J, Borges LA, Feijóo RA. An iterative algorithm for limit analysis with nonlinear yield functions. *International Journal of Solids and Structures* 1993; **30**(10):1397–1417.
5. Herskovits J. A two-stage feasible directions algorithm for nonlinearly constrained optimization. *Mathematical Programming* 1986; **36:**19–38.

6. Avriel L. *Nonlinear Programming, Analysis and Methods*. Prentice-Hall, Inc.: Englewood Cliffs, NJ, 1976.
7. Lyamin AV. Three-dimensional lower bound limit analysis using nonlinear programming. *Ph.D. Thesis*, Department of Civil, Surveying and Environmental Engineering, University of Newcastle, Australia, 1999.
8. Lyamin AV, Sloan SW. A comparison of linear and nonlinear programming formulations for lower bound limit analysis. *Proceedings of the 6th International Symposium on Numerical Models in Geomechanics*, Pietruszczak, Pande (eds), Balkema: Rotterdam, 1997; 367–373.
9. Abbo AJ, Sloan SW. A smooth hyperbolic approximation to the Mohr–Coulomb yield criterion. *Computers and Structures* 1995; **54:**427–441.
10. Prandtl L. Über die Härte plastischer Körper. *Nachrichten von der Gesellschaft der Wissenschaften zu Göttingen, Mathematisch -Physikalische Klasse* 1920; **12:**74–85.
11. Sloan SW. A steepest edge active set algorithm for solving sparse linear programming problems. *International Journal for Numerical Methods in Engineering* 1988; **26:**2671–2685.
12. Yu HS. Expansion of a thick cylinder of soils. *Computers and Geotechnics* 1992; **14:**21–41.
13. Pastor J, Thai T-H, Francescato P. New bounds for the height limit of a vertical slope. *International Journal for Numerical and Analytical Methods in Geomechanics* 2000; **24:**165–182.
14. Bjerrum L, Eide O. Stability of strutted excavations in clay. *Geotechnique*, 1956; **6**(1):32–47.
15. Prater EG. An examination of some theories of earth pressure on shaft linings. *Canadian Geotechnical Journal* 1977; **14**(1):91–106.
16. Pastor J, Turgeman S. Formulation linéaire des méthodes de l'analys limite en symétrie axiale. *Fourth Congrés Francais de Méchanique*, Nancy, France, 1979.
17. Pastor J. Analyse limite et stabilite des fouilles. *Proceedings of the 10th International Conference on Soil Mechanics and Foundation Engineering*, vol. 3, Stockholm, Sweden 1981; 505–508.
18. Britto AM, Kusakabe O. Stability of unsupported axisymmetric excavations in soft clay. *Geotechnique* 1982; **32**(3):261–270.
19. Sloan SW. Numerical analysis of incompressible and plastic solids using finite elements. *Ph.D. Thesis, University of Cambridge, 1981*.

