Solving lower/upper bound approaches of limit analysis by an interior-point method for convex programming

Franck Pastor, Malorie Trillat, Joseph Pastor, Etienne Loute

F. Pastor and E. Loute: Facultés Universitaires Saint-Louis (FUSL), Brussels, Belgium,
E. Loute: Center for Operations Research and Econometrics (CORE)
Université Catholique de Louvain (UCL)
Louvain-la-Neuve, Belgium
Malorie Trillat, Joseph Pastor: Laboratoire LOCIE, Université de Savoie, Chambéry, France

Optimization and Engineering, Louvain-La-Neuve, May 2006

Introduction

Purpose of this work:

- study the structure of a class of Limit Analysis (LA) problems

Introduction

Purpose of this work:

- study the structure of a class of Limit Analysis (LA) problems
- investigate how to design and implement (matlab) a convex interior point (IP) method suited to the problem

Introduction

Purpose of this work:

- study the structure of a class of Limit Analysis (LA) problems
- investigate how to design and implement (matlab) a convex interior point (IP) method suited to the problem
- test the algorithm on a series of both static and kinematic Limit Analysis problems of increasing size

Introduction

Purpose of this work:

- study the structure of a class of Limit Analysis (LA) problems
- investigate how to design and implement (matlab) a convex interior point (IP) method suited to the problem
- test the algorithm on a series of both static and kinematic Limit Analysis problems of increasing size
- demonstrate that large problems can be solved by a matlab-based implementation

Introduction

Purpose of this work:

- study the structure of a class of Limit Analysis (LA) problems
- investigate how to design and implement (matlab) a convex interior point (IP) method suited to the problem
- test the algorithm on a series of both static and kinematic Limit Analysis problems of increasing size
- demonstrate that large problems can be solved by a matlab-based implementation
- show the first result of a domain decomposition-like technique

Statement of the static Limit Analysis problem

An infinite bar is compressed under two rough rigid plates. Figure

A quarter of the section is meshed in triangular Lagrange p1 elements, with appropriate symmetry and boundary conditions. Solving the static problem leads to maximize a linear function of the stresses variables $(\sigma_x, \sigma_y, \sigma_{xy})$ of each triangle's apex, under linear equalities constraints (equilibrium, continuity, symmetry and boundary conditions), and one non-linear inequality per apex. The latter depends on the selected criterion. For example, with the Mises criterion:

$$(\sigma_x - \sigma_y)^2 + (2\sigma_{xy})^2 \leqslant (2c)^2.$$

cont. tests

The solution is a lower bound for the Limit Analysis problem

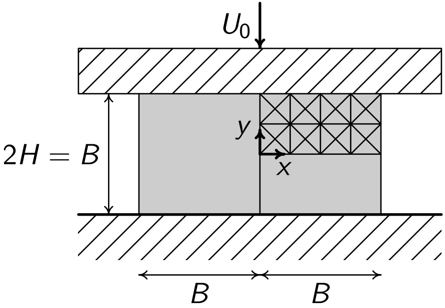


Figure: Compression of a bar between rough rigid plates

Typical optimization problems from static Limit Analysis

General form of the static Limit Analysis mechanical optimization problems :

$$\begin{array}{lrcl} \max & c^T x & & \\ \text{s.t.} & Ax & = & b, \\ & g(x) & \leqslant & 0, \end{array}$$

where

- $c, x \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $A \in \mathbb{R}^{m \times n}$
- $g = (g_1, \ldots, g_p)$ is a vector-valued function of p convex numeric functions g_i.

This problem is convex, both equality and inequality constrained, potentially from medium to large scale, sparse.
IP methods are particularly well suited for this kind of problem

Transformation in a barrier problem

We have adapted an IP algorithm proposed originally by VIAL (1992) for convex programming problems

The algorithm is of the type “primal-dual interior point method”.
The development of the algorithm is as follows:
The original problem, is transformed in an unconstrained “barrier problem”, with a parameter $\mu > 0$, the “barrier parameter”:

$$\begin{array}{ll} \max & c^T x + \mu \sum_{i=1}^{p} \ln(s_i) \\ \text{s.t.} & Ax = b, \\ & g(x) + s = 0. \end{array}$$

The optimality condition for the barrier problem

The KKT conditions are:

$$\begin{aligned} -c + A^T w + \left(\frac{\partial g}{\partial x}\right)^T y = 0 &= F_d(x, y, w, s), \\ Ax - b = 0 &= F_{p_1}(x, w, y, s), \\ g(x) + s = 0 &= F_{p_2}(x, w, y, s), \\ YSe - \mu e = 0 &= F_c(x, w, y, s), \end{aligned}$$

where $w \in \mathbb{R}^m$, $y \in \mathbb{R}^p$ and Y, S are the diagonal matrices associated to y and s respectively. $e \in \mathbb{R}^p$ is a vector of ones.

The Newton system

The KKT conditions for the barrier problem is expressed as:
$F = (F_d, F_{p_1}, F_{p_2}, F_c) = 0$ (the Newton system).
The following linear system must be solved:

$$\begin{bmatrix} H_0 & A^{\mathrm{T}} & \left(\frac{\partial g}{\partial x}\right)^{\mathrm{T}} & 0 \\ A & 0 & 0 & 0 \\ \frac{\partial g}{\partial x} & 0 & 0 & I \\ 0 & 0 & S & Y \end{bmatrix} \begin{bmatrix} dx \\ dw \\ dy \\ ds \end{bmatrix} = \begin{bmatrix} -F_d \\ -F_{p_1} \\ -F_{p_2} \\ -F_c \end{bmatrix}.$$

Given that g is convex, $H_0 = \sum_{i=1}^{p} y_i \frac{\partial^2 g_i}{\partial x^2}$ is positive semi-definite, and in some cases positive definite.

Solving the Newton system: the equilibrium system

Some row and column reordering and a "block-elimination" reduction lead to:

$$\begin{bmatrix} Y & S & 0 & 0 \\ 0 & -Y^{-1}S & \frac{\partial g}{\partial x} & 0 \\ 0 & 0 & H_0 + \left(\frac{\partial g}{\partial x}\right)^{\mathrm{T}} YS^{-1}\frac{\partial g}{\partial x} & A^{\mathrm{T}} \\ 0 & 0 & A & 0 \end{bmatrix} \begin{bmatrix} ds \\ dy \\ dx \\ dw \end{bmatrix} = \begin{bmatrix} -F_c \\ -F_{p_2} + Y^{-1}F_c \\ -F_d - \left(\frac{\partial g}{\partial x}\right)^{\mathrm{T}} YS^{-1} r \\ -F_{p_1} \end{bmatrix},$$

with $r = -F_{p_2} + Y^{-1}F_c$.

Solving the Newton system: the equilibrium system (cont.)

Let us define $H = H_0 + \left(\frac{\partial g}{\partial x}\right)^{\mathrm{T}} YS^{-1}\frac{\partial g}{\partial x}$. Thus we first have to solve the following system:

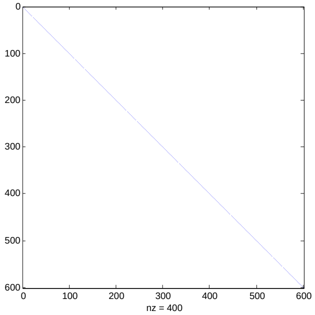
$$\begin{bmatrix} H & A^{\mathrm{T}} \\ A & 0 \end{bmatrix} \begin{bmatrix} dx \\ dw \end{bmatrix} = \begin{bmatrix} -F_d - \left(\frac{\partial g}{\partial x}\right)^{\mathrm{T}} YS^{-1} r \\ -F_{p_1} \end{bmatrix}.$$

- This kind of system is known as an “equilibrium system”.
- The equilibrium system is symmetric, never definite...

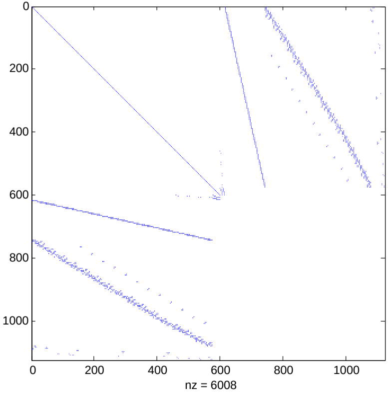
NB: To achieve the decrease of μ and computing the search direction dz along $\mathcal{C}$, we have implemented the Mehrotra predictor-corrector algorithm.

Solving the equilibrium system

The matrix H



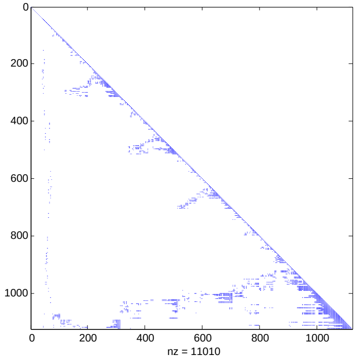
The matrix of the equilibrium system



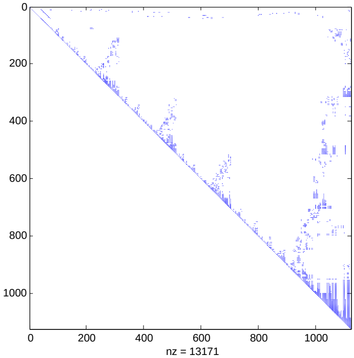
Solving the equilibrium system

Use of a specific method, or LU factorization

The factor L of LU decomposition



The factor U of LU decomposition



Application to the Mises and Gurson Criteria

We already gave an example of a plasticity criterion : the Mises one.

Another criterion: Gurson. The exact solution is not known in this case. Hereafter, $f = 0.16$.

$$(\sigma_x - \sigma_y)^2 + (2\sigma_{xy})^2 + 8c^2 f \cosh \frac{(\sigma_x + \sigma_y)}{2k} \leqslant 4c^2(1 + f^2).$$

- It gives rise to another convex programming, not a conic programming problem.
- A series of tests on the preceeding mechanical system, involving Mises and Gurson criteria, were performed.

Computational Statistics

Experiments with Matlab 6.5.1.
On Apple Macintosh dual G5 2.5Ghz, 4.5GB of ram, under MacOSX. Only one processor used, and 2GB of ram (32bits code).

		Constraints		Mises			Gurson		
N_{tr}	Vars.	Lin.	Conv.	Res.	Iter.	Time	Res.	Iter.	Time
800	7 440	6 340	2 480	2.41346	18	70s	1.64768	14	18s
7 200	65 520	56 220	21 840	2.42270	18	12m 21s	1.64950	19	44m
20 000	181 200	155 700	60 400	2.42465	20	1h 32 m	1.64989	27	7h 24m

Table: Mises and Gurson criteria : comparison.

NB: the exact solution of this problem is known for Mises: 2.42768.

The kinematic problem

- The kinematic problem aims at producing an upper bound for the Limit Analysis problem.
- Difficulty: following the usual way to solve this problem, one has to integrate the dissipated power π, which is sometimes very complicated to compute, if not possible
- Hence the interest of a kinematic method which does not use the dissipated power expression, only the plasticity criterion expression.

A new kinematic approach

The external power can be expressed as $Q \cdot q$, where :

- Q the load vector,
- $q = q(u)$ the generalized velocity vector, with u kinematically admissible (KA).

Virtual Power Principle

Q and $\sigma^\star$ are in equilibrium if, for all KA vectors u :

$$Q \cdot q = \int_V \sigma : v \, dV$$

where v is the strain rate tensor (which depends on u).

Finite element discretization

Assuming a 1-dimensional loading problem $Q^\star$, for sake of simplicity, and the velocities u linear and continuous:

$$\left.\begin{aligned} qQ^\star &= \{u\}^T\{\beta\}Q^\star \\ \int_V \sigma^\star : v\, dV &= \{u\}^T[\alpha]\{\sigma^\star\} \end{aligned}\right\} \Rightarrow \{u\}^T\left([\alpha]\{\sigma^\star\} - \{\beta\}Q^\star\right) = 0 \;\; \forall \{u\} \text{ KA}$$

It leads to the following problem, in the case of the compressed bar with $q = U_0$:

$$\begin{array}{ll} \max & qQ^\star \\ \text{s.t.} & [\alpha]\{\sigma^\star\} - \{\beta\}Q^\star = 0, \\ & f(\sigma^\star) \leqslant 0,\ \sigma^\star \text{ constant in each finite element} \\ & + \text{ limit, symmetric and loading linear conditions.} \end{array}$$

A KKT condition on the discretized problem:

$$-c + A^T w + \left(\frac{\partial f}{\partial x}\right)^T y = 0,$$

where :

$$A = \Big[[\alpha], -\{\beta\}\Big]^T, \; x = \begin{Bmatrix} \{\sigma^\star\} \\ Q^\star \end{Bmatrix}.$$

- w: dual variables associated to linear constraints;
- y: dual variables associated to non linear constraints;
- c: coefficients of the objective function.

By analyzing the structure of the following equations:

$$\{u\}^T \left[[\alpha] \{\sigma^\star\} - \{\beta\} Q^\star \right] = 0,$$
$$-\{c\}^T + \{w\}^T A + \{y\}^T \left\{ \frac{\partial f}{\partial \sigma} \right\} = 0,$$

it can be proved that the field $u = -w$ is admissible (ie KA and PA). Hence the method is rigorously kinematic, requiring *only* the plasticity criterion as information about the material.

Application: compressed bar, plane strain

- for this kind of problem (Gurson), it has been possible to use a Cholesky factorisation to solve the main system
- For the largest problems, the main linear system had to be slightly pertubed (diagonal perturbation of 10^{-8} on H)

Figure

Application: compressed bar, plane strain

- for this kind of problem (Gurson), it has been possible to use a Cholesky factorisation to solve the main system
- For the largest problems, the main linear system had to be slightly pertubed (diagonal perturbation of 10^{-8} on H)

Figure

		Constraints		Lin. cont. u	
N_{tr}	Variables	Lin.	Conv.	Opt. Value	Time.
400	2 403	790	801	1.6779	4s
3 200	9603	3 180	3 201	1.6655	44s
7 200	7 201	21 603	7 170	1.6611	2m 19s
20 000	60 003	19 950	20 001	1.6572	16m 3s

Table: The compressed bar and the *Gurson* material - *kinematic* results for $f = 0.16$ using *linear continuous* velocity fields

Static bound for a Gurson material ($f = 0.16$): 1.6499

$$1.6499 < \frac{F}{2Bk} < 1.6572$$

Only *0,7%* of gap between the two bounds.

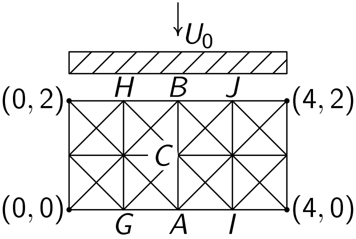
For bigger problems: a “divide and conquer” strategy

- When problems get too large, “out of memory” occurs within Matlab and matrices are increasingly bad-conditionned.
- Hence the idea: splitting the problem in two (or more).
- It happens to be possible in the static-dual algorithm, because of the mechanical meaning of all numerical variables in this problem.

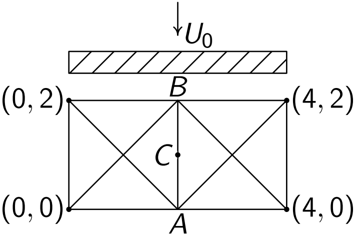
The general idea on a simple example

Purpose : solving a static-dual kinematic problem with linear continuous velocity fields.

- The loading vector: $Q = F$
- The generalized velocity vector: U_0



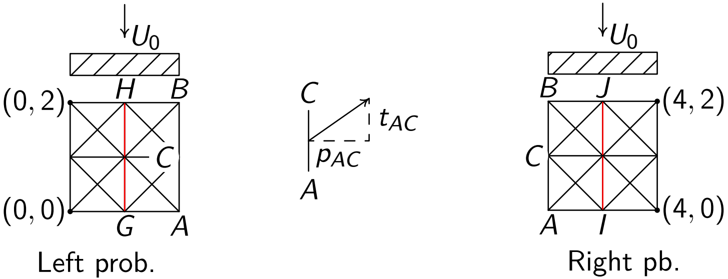
4×2 problem (to solve)



Starting 2×1 problem

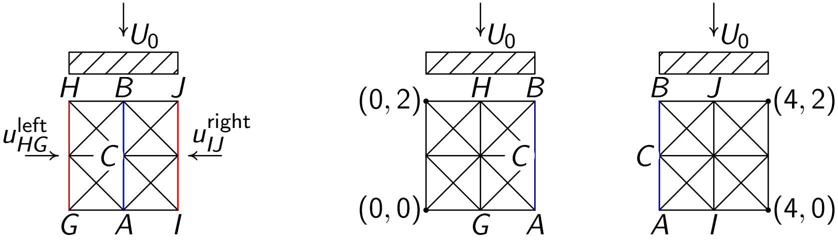
$u_C = (u_A + u_B)/2$

First iteration



- The loading vector: $Q = (F, p_{AC}, t_{AC}, p_{CB}, t_{CB})$
- The velocity vector: $q = \left(U_0, (u_A + u_B)^T/2, (u_B + u_C)^T/2\right)$, where vectors u_A, u_B, u_C are collected on the starting problem
- The sum of the objective values of these problems is a kinematic bound

Subsequent iterations



Middle pb. Left prob. Right pb.

- The sum of the objective values of the latter two problems is another kinematic bound, noticeably lower than the previous one. This bound improves steadily if we iterate this process.
- At each iteration, only the coefficients of the objective function change.

First experiments

- On the compressed bar
- Performed on a PowerbookG4, 1.33Ghz, 2Go of RAM.

	Original problem		Splitted problem		
Size	Result	Time	Global Iter.	Result	Time
16×8	2.53284	n.s.	5	2.53303	n.s.
32×16	2.48753	59s	2	2.48597	68s
64×32	2.45833	1030s	2	2.45957	720s

- The results are more and more accurate as iterations go on.
- However, only a few iteration are necessary, in a reasonable amount of time.

Conclusion and future work

- The IP algorithm for the original convex nonlinear programming problem is efficient, in terms of the number of iterations and even in terms of CPU time per iteration.

Conclusion and future work

- The IP algorithm for the original convex nonlinear programming problem is efficient, in terms of the number of iterations and even in terms of CPU time per iteration.
- We have demonstrated that a code developed in the Matlab environment, with reasonable resources, is almost as efficient as industrial-strength codes.

Conclusion and future work

- The IP algorithm for the original convex nonlinear programming problem is efficient, in terms of the number of iterations and even in terms of CPU time per iteration.
- We have demonstrated that a code developed in the Matlab environment, with reasonable resources, is almost as efficient as industrial-strength codes.
- LA problems are also interesting as test-bed problems for people working on large sparse symmetric structured systems of linear equations, positive definite systems and indefinite ones.

Conclusion and future work

- The IP algorithm for the original convex nonlinear programming problem is efficient, in terms of the number of iterations and even in terms of CPU time per iteration.
- We have demonstrated that a code developed in the Matlab environment, with reasonable resources, is almost as efficient as industrial-strength codes.
- LA problems are also interesting as test-bed problems for people working on large sparse symmetric structured systems of linear equations, positive definite systems and indefinite ones.
- The use of domain decomposition-like techniques makes parallel processing attractive.