Solid Topology Optimization Using Truss Lattices

*M. Victor PRICOP, Albert ZAHAROVITS
*Corresponding author
INCAS - National Institute for Aerospace Research “Elie Carafoli”
Iuliu Maniu 220, Bucharest 061136, Romania
pricop.victor@incas.ro
TO – today is a mature method
A-380, A-400, etc.

Ansys – TO for 2D and 3D
Altair – Optistruct
Nastran

-From single load to multiple load cases
-From static stress control to eigenfrequencies etc.

Intentions
-To develop an in-house code and experiment the algorithms
-Start with the most simple finite element – truss (*lattices* or not)
-Solid Isotropic Material with Penalization (SIMP)
-Power law
SIMP – Solid Isotropic Material with Penalization

- Power law \( p=3 \)
- Compliance minimization
- Mass constraint - inequality
- Density positivity - inequality
- Mechanical equilibrium

\[
\rho = x_e \rho_0 \quad E = x_e^p E_0 \quad E = E_{\text{min}} + x_e^p (E_0 + E_{\text{min}})
\]

\[
k_e = x_e^p k_0
\]

\[
\begin{cases}
\min : c(\rho) \\
a.i. : V = \sum_{e=1}^{N} \rho_e v_e \leq V_0
\end{cases}
\]
- Optimality Criteria OC
- Sequential Linear Programming (SLP)
- Method of Moving Asymptotes (MMA)

\[
\begin{align*}
\text{min : } c(\rho) &= U^T KU = \sum_{e=1}^{N} x_e^p u_e^T k_0 u_e \\
\text{a.i. : } &\frac{V(x)}{V_0} = f \\
KU &= F \\
0 < x_{\text{min}} &\leq x \leq 1
\end{align*}
\]

\[X_{\text{min}}=0.001\]
-M.P. Bendsoe. Evolutionary structural optimization

**Lagrange** Function, multipliers

\[
L(x) = c(x) + \lambda_0 \left( V(x) - fV_0 \right) + \lambda_1^T (KU - F) + \sum_{e=1}^{N} \lambda_{2e} (x_{\min} - x_e) + \sum_{e=1}^{N} \lambda_{3e} (x_e - x_{\max})
\]

\[
\frac{\partial L}{\partial x_e} = \frac{\partial c}{\partial x_e} + \lambda_0 \frac{\partial V}{\partial x_e} + \lambda_1^T \frac{\partial KU}{\partial x_e} - \lambda_{2e} + \lambda_{3e}
\]

Bounding constraints disabled
\[
\lambda_{2e} = 0 \quad \lambda_{3e} = 0
\]

If loading is independent wrt design space:
\[
\frac{\partial F}{\partial x} = 0
\]
Minimization of lagrangean gives:

\[
\frac{px_e^{p-1}q_c}{\lambda_0 V_e} = 1
\]

Equality can be seen as the limit of an iterative process

\[
\frac{p(x_e^{(n)})^{p-1}q_c}{\lambda_0 V_e} = \left(\frac{x_e^{(n+1)}}{x_e^{(n)}}\right)^{1/\zeta} = x_e^{(n+1)} = x_e^{(n)} \left(\frac{p(x_e^{(n)})^{p-1}q_c}{\lambda_0 V_e}\right)^{\zeta}
\]

\[\zeta = 0.5\]
TO for truss structures can be seen as a sizing method

- Fortran 95, OO
- serial, CSR – compressed row storage for stiffness matrix

INPUT
- physical parameters
- numerical parameters
- mesh – structured (lattices) or unstructured, 2D or 3D

OUTPUT
- tecplot and vtk files
- convergence history
- Ansys input file - validation

The prove that code exists
Lagrange multiplier equation – secant method

- two values are required for initialization, see input file
- not fully robust – depending on the overall process convergence
- requires a significant number of iterations

\[ g(\lambda_0) = V(x(\lambda_0)) - fV_0 = 0 \quad \Rightarrow \quad x_{e}^{(n+1)} = x_{e}^{(n)} \left( \frac{p(x_{e}^{(n)})^{p-1} q_c}{\lambda_0 V_e} \right)^{\varsigma} \]

Linear solvers:
- Initially SOR – slow, robust
- Conjugate gradient – efficient, robust – took an hour of programming, keeping an eye on the Wiki page

- Filtration – switch and radius provided – open issue
Results
-cantilever beam
-left side clamped
-lower right corner downward force

Results sensitivity with mesh topology – lattice diagonals
Results

Left: no filtration (null radius), right: filtration (radius = 2 x truss edge)

- link disappearance, islanding – filtration related issues
Results

Unstructured mesh lattices

- linkage disappearance, islanding – filtration related issues
Results

Matlab plotting
- variable thickness lines, proportional to element cross area
- folder for frame images
- movie output
Efforts to improve

- Going to beam elements
- Part of the work done
- Matlab symbolic calculations for the stiffness matrix

Parallelization/optimization of CG
- OpenMP – no impact

- Rewriting **CG** in CUDA for Tesla M2070
- No external library
- Modest improvement
- Lack of robustness
- Reordering of the matrix - no impact
- Rewriting the CG in C and using various compilation options
- Replace division with invert multiplication – IEEE non-conformal plus -ipo or -pgo => 10% speedup
Спасибо