Solving extremely ill-posed structures

Lars Beex  Marc Duflot^b  Laurent Adam^b

^b e-Xstream Engineering
Introduction
Introduction

480 µm

Engineering stress [MPa]

Engineering strain [-]
Problem: no out-of-plane stiffness
Creating out-of-plane stiffness: Approach 1

- $L_{\text{warp}}$
- $L_{\text{weft}}$
- $h \ll L_{\text{warp}}$
- $h \ll L_{\text{weft}}$
Creating out-of-plane stiffness: Approach 2

bending

torsion
Energy minimisation without Hessian
Energy minimisation: steepest descent

START: $u_0$  $E(u_0)$
Energy minimisation: steepest descent

START:

\[ u_0 \quad E(u_0) \quad -f(u_0) \]
Energy minimisation: steepest descent

UPDATE:

\[ u_{i+1} = u_i - \alpha f(u_i) \]

\[ E(u_{i+1}) \]
Energy minimisation: steepest descent

CONTINUE until $E(u_{i+1}) > E(u_i)$
Energy minimisation: steepest descent

CONDITION: small stepsize $\alpha$
$\rightarrow$ this makes the procedure inefficient
Steepest descent with backtracking linesearch (BTLS)

IDEA: once you know the search direction, optimise stepsize $\alpha$ such that $E(u_{i+1})$ is minimum
Steepest descent with backtracking linesearch (BTLS)

IDEA: once you know the direction, optimise stepsize $\alpha$ such that $E(u_{i+1})$ is minimum

Cross-section along search direction
Steepest descent with backtracking linesearch (BTLS)

BACKTRACKING: $\alpha_i = \alpha_0$

while $E(\alpha_{i+1}) < E(\alpha_i)$

$\alpha_{i+1} = 0.5 \alpha_i$

Cross-section along search direction
Steepest descent vs Steepest descent with BTLS

Steepest descent

Steepest descent with BTLS
Problem BACKTRACKING: New point is higher than starting point
Steepest descent with BTLS and Armijo Rule (AR)

Solution: apply Armijo Rule, before backtracking

$$E(u_i - \alpha_{i+1} f(u_i)) \leq E(u_i) - c_1 \alpha_0 f(u_i)^T f(u_i)$$
Steepest descent with BTLS vs Steepest descent with BTLS & AR
Consider the following case:

Cross-section along search direction
Include Curvature Rule to ensure that the slope changes enough.
Curvature Rule:

\[ \mathbf{f}(u_i)^T \mathbf{f}(u_{i+1}) \leq c_2 \mathbf{f}(u_i)^T \mathbf{f}(u_i) \]

Terminology:

Armijo Rule + Curvature Rule = Wolfe conditions
Steepest descent with BTLS & AR vs Steepest descent with BTLS & WC

Steepest descent AR

Steepest descent AR & CR
Conjugate gradient instead of Steepest descent

Enough about backtracking linesearch, consider this:

Combine current search direction with previous search direction
Conjugate gradient vs Steepest descent

Steepest descent update:

\[ u_{i+1} = u_i - \alpha f(u_i) \]

Conjugate gradient update for 2\textsuperscript{nd} step:

\[ u_2 = u_1 - \alpha \left( f(u_1) - \beta f(u_0) \right) \]

where \( \beta: \beta(f(u_1), f(u_0)) \)

(e.g. Polak-Ribiere)
Steepest descent with BTLS & AR vs Steepest descent with BTLS & WC

Steepest descent AR & CR

Conjugate gradient AR & CR
Truss network: Conjugate gradient with BTLS & WC
Truss network: Conjugate gradient with BTLS & WC

no scaling
Final remarks

Contact cannot be included via standard penalty methods or Lagrange multipliers

Energy landscape must be smooth
  \(\rightarrow\) no discontinuous potential
  \(\rightarrow\) no dissipation, BUT variational dissipation!!
  \(\rightarrow\) no friction in contact

Alternative to avoid ill-posedness for truss structures:
  \(\rightarrow\) beams