Abstract

"Improving the iterative methods in TNO DIANA using physical properties of the underlying model"

Alex Sangers - December 10, 2013 at 16:00 in EEMCS Room 1.2

DIANA is a multi-purpose finite element software package that is dedicated to e.g. structural and geotechnical engineering problems. The solution of one or more systems of linear equations is a computational intensive part of a finite element analysis. For this purpose a number of direct and iterative methods are available in DIANA.

As the demand for larger finite element analysis grows every year, so lead the corresponding models to large linear systems. Iterative methods have proved to be effective to solve these systems.

The purpose of this research is to find out how the iterative methods of DIANA can be improved. One technique considered is deflation, based on the physical properties of the underlying problem. Another technique is scaling the system based on the type of degree of freedom, such as translation, rotation or pressure.

Iterative methods at DIANA

Improving the iterative methods in TNO DIANA using physical properties of the underlying model.

Alex Sangers

Delft Institute of Applied Mathematics
TNO DIANA

December 10, 2013

Content

Finite element analysis at DIANA (1/2)

Some relevant quantities:

u - displacements

ε - strain

σ - stress

Finite element analysis at DIANA (1/2)

Some relevant quantities:

u - displacements

ε - strain

σ - stress

In case of linear elastic behavior:

$$\varepsilon = B_m u,$$

with B_m the differential matrix,

$$\sigma = D_m \varepsilon,$$

with D_m the elasticity matrix.

Finite element analysis at DIANA (2/2)

K and f are assembled by

$$K^{e_m} = \int_{e_m} B_m^T D_m B_m \; dV, \qquad f^{e_m} = \int_{e_m} \sum_i f_i^{e_m} \; dV$$

Finite element analysis at DIANA (2/2)

K and f are assembled by

$$K^{e_m} = \int_{e_m} B_m^T D_m B_m \; dV, \qquad f^{e_m} = \int_{e_m} \sum_i f_i^{e_m} \; dV$$

$$\Rightarrow K = \sum_{m=1}^{n_{el}} T_m^T K^{e_m} T_m, \qquad f = \sum_{m=1}^{n_{el}} T_m^T f^{e_m},$$

Finite element analysis at DIANA (2/2)

K and f are assembled by

$$K^{e_m} = \int_{e_m} B_m^T D_m B_m \; dV, \qquad f^{e_m} = \int_{e_m} \sum_i f_i^{e_m} \; dV$$

$$\Rightarrow K = \sum_{m=1}^{n_{el}} T_m^T K^{e_m} T_m, \qquad f = \sum_{m=1}^{n_{el}} T_m^T f^{e_m},$$

Result: $Ku = f, \quad K \in \mathbb{R}^{n \times n}, \; f \in \mathbb{R}^n$.

Finite element analysis at DIANA (2/2)

K and f are assembled by

$$K^{e_m} = \int_{e_m} B_m^T D_m B_m \, dV, \qquad f^{e_m} = \int_{e_m} \sum_i f_i^{e_m} \, dV$$

$$\Rightarrow K = \sum_{m=1}^{n_{el}} T_m^T K^{e_m} T_m, \qquad f = \sum_{m=1}^{n_{el}} T_m^T f^{e_m},$$

Result: $Ku = f, \quad K \in \mathbb{R}^{n \times n}, \ f \in \mathbb{R}^n$.

Computationally intensive part!

Solution methods for linear systems

Direct and iterative methods.

Solution methods for linear systems

Direct and iterative methods.

Direct methods:

- LU or Cholesky decomposition
- Pardiso (parallel LU)

Solution methods for linear systems

Direct and iterative methods.

Direct methods:

- LU or Cholesky decomposition
- Pardiso (parallel LU)

Iterative methods:

- Conjugate Gradient Method
- Generalized Minimal Residual Method

Solution methods for linear systems

Direct and iterative methods.

Direct methods:

- LU or Cholesky decomposition
- Pardiso (parallel LU)

Iterative methods:

- Conjugate Gradient Method
- Generalized Minimal Residual Method

Properties of iterative methods:

- Require less memory
- In general less robust
- Effective for important classes of problems

Iterative solution methods available at DIANA

Symmetric case:
Conjugate Gradient (CG)

- Lanczos-based algorithm
- Strategy: orthogonalizes the residuals
- Optimality
- Short-recurrence

Iterative solution methods available at DIANA

Symmetric case:
Conjugate Gradient (CG)

- Lanczos-based algorithm
- Strategy: orthogonalizes the residuals
- Optimality
- Short-recurrence

Nonsymmetric case:
(Restarted) Generalized Minimal Residual (GMRES(s))

- Arnoldi-based algorithm
- Strategy: minimizes the residuals
- Optimality (if no restart)
- Long-recurrence

Preconditioning

The convergence of iterative methods depend on eigenvalues and eigenvectors.

For SPD matrix K holds for the m-th CG iteration:

$$||u - u_m||_K \leq 2 \left[\frac{\sqrt{\lambda_{\max}/\lambda_{\min}} - 1}{\sqrt{\lambda_{\max}/\lambda_{\min}} + 1} \right]^m ||u - u_0||_K$$

Preconditioning

The convergence of iterative methods depend on eigenvalues and eigenvectors.

For SPD matrix K holds for the m-th CG iteration:

$$||u - u_m||_K \leq 2 \left[\frac{\sqrt{\lambda_{\max}/\lambda_{\min}} - 1}{\sqrt{\lambda_{\max}/\lambda_{\min}} + 1} \right]^m ||u - u_0||_K$$

- Clustered eigenvalues yield good convergence for CG
- Clustered eigenvalues and well-conditioned eigenvectors yield good convergence for GMRES

Preconditioning

The convergence of iterative methods depend on eigenvalues and eigenvectors.

For SPD matrix K holds for the m-th CG iteration:

$$||u - u_m||_K \leq 2 \left[\frac{\sqrt{\lambda_{\max}/\lambda_{\min}} - 1}{\sqrt{\lambda_{\max}/\lambda_{\min}} + 1} \right]^m ||u - u_0||_K$$

- Clustered eigenvalues yield good convergence for CG
- Clustered eigenvalues and well-conditioned eigenvectors yield good convergence for GMRES

Preconditioning $Ku = f \;\Rightarrow\; P^{-1}Ku = P^{-1}f$,

where:

- $P \approx K \Rightarrow P^{-1}K \approx I$.
- $Px = y$ is easy to solve.

Substructuring and Domain decomposition

Substructuring and Domain decomposition

Substructuring

$$K \sim \begin{pmatrix} A_1 & & & B_1 \\ & \ddots & & \vdots \\ & & A_{n_s} & B_{n_s} \\ B_1^T & \dots & B_{n_s}^T & C \end{pmatrix}$$

- No parallel implementation
- Partitioning the elements
- No overlap in partitions
- One preconditioner

Substructuring and Domain decomposition

Substructuring

$$K \sim \begin{pmatrix} A_1 & & & B_1 \\ & \ddots & & \vdots \\ & & A_{n_s} & B_{n_s} \\ B_1^T & \dots & B_{n_s}^T & C \end{pmatrix}$$

- No parallel implementation
- Partitioning the elements
- No overlap in partitions
- One preconditioner

Domain decomposition

$$K = \sum_{i=1}^{n_d} L_i^T K_i R_i$$

- Parallel implementation
- Partitioning the nodes
- Overlap allowed in partitions
- Dual preconditioner

Enhancements

Required improvements:

- Jumps in material properties
- Multiple types of degrees of freedom
- Interface elements: ‘coupling’ elements
- Mixture elements: pressure-translation elements
- Nonlinear loops

Enhancements

Required improvements:

- Jumps in material properties
- Multiple types of degrees of freedom
- Interface elements: ‘coupling’ elements
- Mixture elements: pressure-translation elements
- Nonlinear loops

Techniques:

- Deflation
- Scaling
- IDR(s)

Deflation

Define

$$\Pi^{\in} = I - Z(Y^T K Z)^{-1} Y^T K,$$
$$\Pi^{\perp} = I - KZ(Y^T K Z)^{-1} Y^T,$$

so that $\Pi^{\perp} K = K \Pi^{\in}$.

Deflation

Define

$$\Pi^{\in} = I - Z(Y^T K Z)^{-1} Y^T K,$$
$$\Pi^{\perp} = I - K Z(Y^T K Z)^{-1} Y^T,$$

so that $\Pi^{\perp} K = K \Pi^{\in}$.

Split u by

$$\begin{aligned} u &= u^{\in} + u^{\perp} \\ &= (I - \Pi^{\in})u + \Pi^{\in} u. \end{aligned}$$

Deflation

Define

$$\Pi^{\in} = I - Z(Y^T K Z)^{-1} Y^T K,$$
$$\Pi^{\perp} = I - K Z(Y^T K Z)^{-1} Y^T,$$

so that $\Pi^{\perp} K = K \Pi^{\in}$.

Split u by

$$\begin{aligned} u &= u^{\in} + u^{\perp} \\ &= (I - \Pi^{\in})u + \Pi^{\in} u. \end{aligned}$$

First part $u^{\in}$: $\quad u^{\in} = (I - \Pi^{\in})u = Z(Y^T K Z)^{-1} Y^T f.$

Deflation

Define

$$\Pi^{\in} = I - Z(Y^T K Z)^{-1} Y^T K,$$
$$\Pi^{\perp} = I - KZ(Y^T K Z)^{-1} Y^T,$$

so that $\Pi^{\perp} K = K \Pi^{\in}$.

Split u by

$$\begin{aligned} u &= u^{\in} + u^{\perp} \\ &= (I - \Pi^{\in})u + \Pi^{\in} u. \end{aligned}$$

First part $u^{\in}$: $\quad u^{\in} = (I - \Pi^{\in})u = Z(Y^T K Z)^{-1} Y^T f.$

Deflation

Define

$$\Pi^{\in} = I - Z(Y^T K Z)^{-1} Y^T K,$$
$$\Pi^{\perp} = I - K Z(Y^T K Z)^{-1} Y^T,$$

so that $\Pi^{\perp} K = K \Pi^{\in}$.

Split u by

$$\begin{aligned} u &= u^{\in} + u^{\perp} \\ &= (I - \Pi^{\in})u + \Pi^{\in} u. \end{aligned}$$

First part $u^{\in}$: $u^{\in} = (I - \Pi^{\in})u = Z(Y^T K Z)^{-1} Y^T f$. ✓

Second part $u^{\perp}$: $K u^{\perp} = K \Pi^{\in} u = \Pi^{\perp} K u$.

Deflation

Define

$$\Pi^{\in} = I - Z(Y^T K Z)^{-1} Y^T K,$$
$$\Pi^{\perp} = I - KZ(Y^T K Z)^{-1} Y^T,$$

so that $\Pi^{\perp} K = K \Pi^{\in}$.

Split u by

$$\begin{aligned} u &= u^{\in} + u^{\perp} \\ &= (I - \Pi^{\in})u + \Pi^{\in} u. \end{aligned}$$

First part $u^{\in}$: $\quad u^{\in} = (I - \Pi^{\in})u = Z(Y^T K Z)^{-1} Y^T f.$ ✓

Second part $u^{\perp}$: $\quad K u^{\perp} = K \Pi^{\in} u = \Pi^{\perp} K u.$

$$\Rightarrow \Pi^{\perp} K u = \Pi^{\perp} f.$$

Deflation

Define

$$\Pi^{\in} = I - Z(Y^T K Z)^{-1} Y^T K,$$
$$\Pi^{\perp} = I - KZ(Y^T K Z)^{-1} Y^T,$$

so that $\Pi^{\perp} K = K \Pi^{\in}$.

Split u by

$$\begin{aligned} u &= u^{\in} + u^{\perp} \\ &= (I - \Pi^{\in})u + \Pi^{\in} u. \end{aligned}$$

First part $u^{\in}$: $\quad u^{\in} = (I - \Pi^{\in})u = Z(Y^T K Z)^{-1} Y^T f.$ ✓

Second part $u^{\perp}$: $\quad K u^{\perp} = K \Pi^{\in} u = \Pi^{\perp} K u.$

$$\Rightarrow \Pi^{\perp} K u = \Pi^{\perp} f.$$ ✓

Deflation: choice of Z

Firstly, $Y = Z$.
Secondly,

Deflation: choice of Z

Firstly, $Y = Z$.
Secondly,

- Eigenvector deflation:

$$Z = \begin{pmatrix} v_1 & \dots & v_k \end{pmatrix}.$$

Deflation: choice of Z

Firstly, $Y = Z$.
Secondly,

- Eigenvector deflation:

$$Z = \begin{pmatrix} v_1 & \dots & v_k \end{pmatrix}.$$

- Subdomain deflation:

$$Z_{ij} = \begin{cases} 1 & \text{if } i \in G_j, \\ 0 & \text{otherwise.} \end{cases}$$

Deflation: choice of Z

Firstly, $Y = Z$.
Secondly,

- Eigenvector deflation:

$$Z = \begin{pmatrix} v_1 & \dots & v_k \end{pmatrix}.$$

- Subdomain deflation:

$$Z_{ij} = \begin{cases} 1 & \text{if } i \in G_j, \\ 0 & \text{otherwise.} \end{cases}$$

- Rigid body mode deflation:
  Z is approximate null space of element matrices corresponding to 'near-rigid bodies'.

Rigid body mode deflation: Example

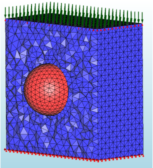


Figure: Stiff sphere in block.

Young's modulus $E(\underline{x}) = \begin{cases} 1 & \text{if } \underline{x} \in \text{block}, \\ 10^6 & \text{if } \underline{x} \in \text{sphere}. \end{cases}$

Rigid body mode deflation: Example

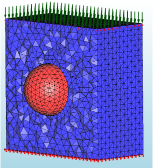


Figure: Stiff sphere in block.

Young's modulus $E(\underline{x}) = \begin{cases} 1 & \text{if } \underline{x} \in \text{block}, \\ 10^6 & \text{if } \underline{x} \in \text{sphere}. \end{cases}$

'Near-rigid body'!

Rigid body mode deflation: Z

Mathematically,

$K z_i \approx 0$, z_i rigid body mode with only value in sphere.

Rigid body mode deflation: Z

Mathematically,
$Kz_i \approx 0$, z_i rigid body mode with only value in sphere.

Suppose only one element in sphere with nodes
$\underline{x}_1 = (x_1, y_1, z_1)$ and $\underline{x}_2 = (x_2, y_2, z_2)$.

Rigid body mode deflation: Z

Mathematically,
$K z_i \approx 0$, z_i rigid body mode with only value in sphere.

Suppose only one element in sphere with nodes
$\underline{x}_1 = (x_1, y_1, z_1)$ and $\underline{x}_2 = (x_2, y_2, z_2)$.

$$\begin{array}{c} \\ \\ x_1 \\ x_2 \\ y_1 \\ y_2 \\ z_1 \\ z_2 \\ \\ \\ \end{array} \qquad Z = \begin{pmatrix} & & & \emptyset & & \\ & & & & & \\ 1 & 0 & 0 & 0 & -z_1 & y_1 \\ 1 & 0 & 0 & 0 & -z_2 & y_2 \\ 0 & 1 & 0 & z_1 & 0 & -x_1 \\ 0 & 1 & 0 & z_2 & 0 & -x_2 \\ 0 & 0 & 1 & -y_1 & x_1 & 0 \\ 0 & 0 & 1 & -y_2 & x_2 & 0 \\ & & & & & \\ & & & \emptyset & & \end{pmatrix}$$

Scaling

Example:

Translation degrees of freedom:	$\mathcal{O}(t)$
Rotation degrees of freedom:	$\mathcal{O}(r)$
Pressure degrees of freedom:	$\mathcal{O}(p)$

Scaling

Example:

Translation degrees of freedom:	$\mathcal{O}(t)$
Rotation degrees of freedom:	$\mathcal{O}(r)$
Pressure degrees of freedom:	$\mathcal{O}(p)$

Right preconditioning:

$$KP^{-1}x = f, \quad u = P^{-1}x,$$

with $P = \text{diag}(\{t, r, p\})$.

IDR(s)

Nonsymmetric systems: No 'best' algorithm.

IDR(s) forces residuals r_n in subspace $\mathcal{G}_j$ of decreasing dimension. Some freedom in algorithm remains.

GMRES(s)	IDR(s)
Long recurrences	Short recurrences
Optimal if no restart	Not optimal

IDR(s)

Nonsymmetric systems: No 'best' algorithm.

IDR(s) forces residuals r_n in subspace $\mathcal{G}_j$ of decreasing dimension. Some freedom in algorithm remains.

GMRES(s)	IDR(s)
Long recurrences	Short recurrences
Optimal if no restart	Not optimal

$\Rightarrow$ IDR(s) is likely to outperform GMRES(s) in case of restart

IDR(s)

Nonsymmetric systems: No 'best' algorithm.

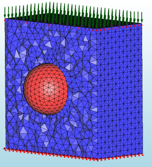
IDR(s) forces residuals r_n in subspace $\mathcal{G}_j$ of decreasing dimension. Some freedom in algorithm remains.

GMRES(s)	IDR(s)
Long recurrences	Short recurrences
Optimal if no restart	Not optimal

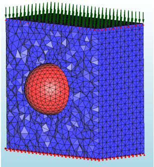
$\Rightarrow$ IDR(s) is likely to outperform GMRES(s) in case of restart

- Large systems of equations
- Ill-conditioned problems

Rigid body mode deflation so far...



Rigid body mode deflation so far...



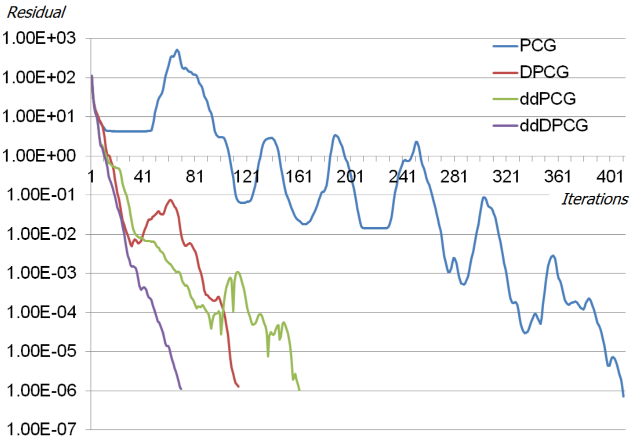


Figure: Convergence for *Block5*.

Analogue behavior for three stiff blocks in a block.

Challenges

Challenges

- Identification of rigid bodies
  - Young's modulus of materials
  - Interface elements
  - Spring elements
  - (Shell elements)
  - (Contact elements)

Challenges

- Identification of rigid bodies
  - Young's modulus of materials
  - Interface elements
  - Spring elements
  - (Shell elements)
  - (Contact elements)
- Scaling
  - Identifying orders of magnitude (units, expected solution)
  - Combination with other preconditioners

Challenges

- Identification of rigid bodies
  - Young's modulus of materials
  - Interface elements
  - Spring elements
  - (Shell elements)
  - (Contact elements)
- Scaling
  - Identifying orders of magnitude (units, expected solution)
  - Combination with other preconditioners
- Nonlinear iteration loop
  - When to reuse information?
  - How to reuse information (rigid bodies, Ritz vectors, etc.)?

Iterative methods at DIANA

Improving the iterative methods in TNO DIANA using physical properties of the underlying model.

Alex Sangers

Delft Institute of Applied Mathematics
TNO DIANA

December 10, 2013