CISM COURSE COMPUTATIONAL ACOUSTICS

Solvers

Part 1: Direct Solvers

Ulrich Langer and Martin Neumüller
Institute of Computational Mathematics
Johannes Kepler University Linz
Udine, May 23-27, 2016

Outline

Outline

Algebraic Systems arising in CA

Given a regular (?) $n_h \times n_h$ system matrix $\mathbf{A} = [A_{ij}]_{i,j=1,\ldots,n_h}$ and a rhs $\underline{f} = [f_i]_{i=1,\ldots,n_h} \in \mathbb{R}^{n_h}$, find $\underline{u} = [u_j]_{j=1,\ldots,n_h} \in \mathbb{R}^{n_h}$:

$$\mathbf{A}\underline{u} = \underline{f} \qquad (1)$$

where $n = n_h = n_{eq} = O(h^{-d})$ - nr of dofs = nr of eqns,
h - discretization parameter, d - space dim. (PDE in $\Omega \subset \mathbb{R}^d$).

Possible system matrices in CA:

$\mathbf{A} = \mathbf{D}$ - diagonal matrix (mass lumping)
$\mathbf{A} = \mathbf{M}$ - mass matrix (MK3= Kaltenbacher 3)
$\mathbf{A} = \mathbf{K}$ - stiffness matrix (MK3)
$\mathbf{A} = \mathbf{M} + \gamma_H \Delta t\, \mathbf{C} + \beta_H (\Delta t)^2 \mathbf{K}$ - Newmark matrix (MK3)
$\mathbf{A} = \mathbf{K} - \omega^2 \mathbf{M}$ - time-harmonic case (SM=Marburg)
$\mathbf{A} = \mathbf{B}$ - fully populated BEM matrices (SM): $n_h = O(h^{-(d-1)})$

Model Problem from MK3

- Mixed BVP for Poisson equation ($\nu = 1$):

$$-\Delta u = f \text{ in } \Omega, \quad u = u_e := 0 \text{ on } \Gamma_e, \quad \frac{\partial u}{\partial \mathbf{n}} = q_n \text{ on } \Gamma_n \quad (2)$$

- Weak formulation: Find $u \in V_{u_e} : a(u,v) = \ell(v) \; \forall v \in V_0$
Find $u \in V_{u_e} := \{v \in H^1(\Omega) : v = u_e \text{ on } \Gamma_e\}$ such that (:)

$$\int_\Omega \nabla u \cdot \nabla v \, d\mathbf{x} = \int_\Omega f v d\mathbf{x} + \int_{\Gamma_n} q_n v \, ds \quad (3)$$

for all $v \in V_0 := \{v \in H^1(\Omega) : v = 0 \text{ on } \Gamma_e\}$, where

$$H^1(\Omega) = \{v \in L_2(\Omega) : \exists \text{ weak } \nabla v \in L_2(\Omega)\}$$

denotes the Sobolev space that is equipped with the norm

$$\|v\|_1^2 := \|v\|_0^2 + |v|_1^2 = \int_\Omega |v|^2 d\mathbf{x} + \int_\Omega |\nabla v|^2 d\mathbf{x}$$

Model Problem from MK3: $\exists\,!$

Lax-Milgram Lemma delivers existence and uniqueness provided that the following assumptions are fulfilled:

1. rhs $\ell(\cdot)$ is a continuous (bounded), linear functional:

$$|\ell(v)| \leq (\|f\|_0 + c\,\|q_n\|_{L_2(\Gamma_n)})\|v\|_1, \;\forall v \in V_0,$$

2. bilinear form $a(\cdot,\cdot)$ is continuous (bounded) on V_0:

$$|a(u,v)| \leq 1\,\|u\|_1\|v\|_1 = \mu_2\,\|u\|_1\|v\|_1, \;\forall u,v \in V_0,$$

3. bilinear form $a(\cdot,\cdot)$ is V_0 elliptic (coercive):

$$a(v,v) = |v|_1^2 \geq \frac{1}{2}(1 + c_F^{-2})\|v\|_1^2 = \mu_1\,\|v\|_1^2, \;\forall v \in V_0,$$

by Friedrichs' inequality: $\|v\|_0 \leq c_F(\Gamma_e)|v|_1, \;\forall v \in V_0.$

Model Problem from MK3: FEM

- FE-Scheme: Find $u^h \in V^h_{u_e} : a(u^h, v^h) = \ell(v^h)\ \forall v^h \in V^h_0$
  Find $u^h(\mathbf{x}) = \sum_{j=1}^{n_{eq}} u_j N_j(\mathbf{x}) + \sum_{j=n_{eq}+1}^{n_n} u_e(\mathbf{x}_j) N_j(\mathbf{x}) \in V^h_{u_e}$:

$$\int_\Omega \nabla u^h \cdot \nabla v^h \, d\mathbf{x} = \int_\Omega f v^h d\mathbf{x} + \int_{\Gamma_n} q_n v^h \, ds \tag{4}$$

  for all $v^h \in V_0 := \text{span}\{N_1, N_2, \ldots, N_{n_{eq}}\}$.
- Since the FE basis is chosen, the FE scheme (4) is equivalent to the solution of a linear system of equations:
  Find $\underline{u} = [u_j]_{j=1,\ldots,n_h} \in \mathbb{R}^{n_h = n_{eq}}$:

$$\mathbf{K}\underline{u} = \underline{f}, \tag{5}$$

  where $\mathbf{K} = [K_{ij}]_{i,j=1,\ldots,n_h}$, $K_{ij} = \int_\Omega \nabla N_j \cdot \nabla N_i \, d\mathbf{x}$

$$\underline{f} = [f_i]_{i=1,\ldots,n_h},\ f_i = \int_\Omega f N_i d\mathbf{x} + \int_{\Gamma_n} q_n N_i \, ds - \sum_{j=n_{eq}+1}^{n_n} K_{ij} u_e(\mathbf{x}_j)$$

Structural Properties of K

- Large scale: $n_h = O(h^{-d})$ = 10^6,...,10^9 dofs in practice !
- Sparse: $K_{ij} = 0 \; \forall i,j :$ supp$N_i \cap$ supp$N_j = \varnothing$, i.e.
  NNE = Number of Non-zero Elements = $O(h^{-d}) = n_h$
- Band resp. profile strucure, i.e.
  $K_{ij} = 0$ if $|i-j| > b_w$ = bandwidth = $O(h^{-(d-1)})$, **BUT**
  band resp. profile depend on the numbering of the nodes !
  $\Longrightarrow$ Heuristic algorithms of band or profile optimization like
  - □ Cuthill-McKee algorithm
  - □ Reverse Cuthill-McKee algorithm
  - □ Minimal degree algorithm

Heredity Properties of $\mathbb{K}$

- Heredity relation:

$$(\mathbf{K}\underline{u}, \underline{v}) := (\mathbf{K}\underline{u}, \underline{v})_{R^n} = a(u^h, v^h)\ \forall \underline{u}, \underline{v} \leftrightarrow u^h, v^h \in V_0^h \quad (6)$$

- Consequences:
  1. $a(u^h, v^h) = a(v^h, u^h)\ \forall u^h, v^h \in V_0^h \Rightarrow \mathbf{K} = \mathbf{K}^T$
  2. $a(v^h, v^h) > 0\ \forall v^h \in V_0^h \setminus \{0\} \Rightarrow \mathbf{K}$ is positive definite !
  3. MK3 model problem (3): $\mathbf{K} = \mathbf{K}^T > 0$ is SPD
     since $a(.,.)$ is symmetric and even V_0-elliptic.

SPD Stiffness Matrix $\mathbb{K}$: Spectral Properties

Let us assume that $a(.,.)$ is symmetric, V_0-elliptic and V_0-bounded as in our MK3 model problem (3):

- Consequences:
  1. $\mathbf{K}$ is SPD
  2. $\mathbf{K}$ has $n = n_h$ positive real eigenvalues (EV) λ_k with the corresponding eigenvectors $\underline{\varphi}_k$: $\quad \mathbf{K}\underline{\varphi}_k = \lambda_k \underline{\varphi}_k$

     $$0 < \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$$

     $$\underline{\varphi}_1, \quad \underline{\varphi}_2, \quad \cdots \quad \underline{\varphi}_n,$$

     where the eigenvectors are orthogonal, i.e.

     $$(\underline{\varphi}_i, \underline{\varphi}_j) := (\underline{\varphi}_i, \underline{\varphi}_j)_{R^n} = \delta_{i,j} \tag{7}$$

  3. Spectral condition number:

     $$\kappa_2(\mathbf{K}) := \|\mathbf{K}\|_2 \|\mathbf{K}^{-1}\|_2 = \frac{\lambda_n}{\lambda_1} = \frac{\lambda_{\text{max}}(\mathbf{K})}{\lambda_{\text{min}}(\mathbf{K})} \tag{8}$$

SPD $\mathbb{K}$: Eigenvalue Estimates

- Rayleigh quotion representation:
  1. Maximal eigenvalue $\lambda_n = \lambda_{\max}(\mathbf{K})$ of $\mathbf{K}$:

$$\lambda_{\max}(\mathbf{K}) = \max_{\underline{v} \in R^n} \frac{(\mathbf{K}\underline{v}, \underline{v})}{(\underline{v}, \underline{v})} \leq c_2 h^{d-2} \tag{9}$$

P: $(\mathbf{K}\underline{v}, \underline{v}) = a(v^h, v^h) = \sum (\mathbf{K}^e \underline{v}^e, \underline{v}^e) \leq \sum \lambda_{\max}(\mathbf{K}^e)(\underline{v}^e, \underline{v}^e)$

  2. Minimal eigenvalue $\lambda_1 = \lambda_{\min}(\mathbf{K})$ of $\mathbf{K}$:

$$\lambda_{\min}(\mathbf{K}) = \min_{\underline{v} \in R^n} \frac{(\mathbf{K}\underline{v}, \underline{v})}{(\underline{v}, \underline{v})} \geq c_1 h^{d} \tag{10}$$

P: $(\mathbf{K}\underline{v}, \underline{v}) = a(v^h, v^h) \geq \mu_1 \|v^h\|_1^2 \geq \mu_1 \|v^h\|_0^2 = \mu_1 (\mathbf{M}\underline{v}, \underline{v})$

- The spectral condition number estimate

$$\kappa_2(\mathbf{K}) = \frac{\lambda_{\max}(\mathbf{K})}{\lambda_{\min}(\mathbf{K})} \leq \frac{c_2}{c_1} h^{-2} \tag{11}$$

is sharp wrt h, i.e. $\kappa_2(\mathbf{K}) = O(h^{-2})$ for $h \to 0$ (example).

Example: $-u'' = f \ in \, (0,1), \ u(0) = u(1) = 0$

Let us consider the 1d example

$$-u''(x) = f(x), \ x \in (0,1), \quad u(0) = u(1) = 0 \tag{12}$$

yielding the FE stiffness matrix

$$\mathbf{K} = \frac{1}{h} \begin{pmatrix} 2 & -1 & 0 & \cdots & \cdots & 0 \\ -1 & 2 & -1 & \ddots & \ddots & \vdots \\ 0 & -1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & -1 & 0 \\ \vdots & \ddots & 0 & -1 & 2 & -1 \\ 0 & \cdots & \cdots & 0 & -1 & 2 \end{pmatrix} \tag{13}$$

for hat functions $N_1, \ldots, N_{n_h = n-1}$ on a uniform grid $0 = x_0 < x_1 < \ldots < x_{n-1} < x_n = 1$ with $x_{i+1} - x_i = h = 1/n$.

Example: $-u'' = f \; in \, (0,1), \; u(0) = u(1) = 0$

- Eigenvalues: $\lambda_k = \frac{4}{h} \sin^2 \frac{k\pi}{2n}, \;\; k = 1, 2, \ldots, n-1 = \overline{1, n-1}$
- Eigenvectors: $\underline{\varphi}_k = [\sqrt{2n} \sin(k\pi i h)]_{i=1,\ldots,n-1}, \, k = \overline{1, n-1}$
- Minimal eigenvalue:

$$\lambda_1 = \frac{4}{h} \sin^2 \frac{1\pi}{2n} = \frac{4}{h} \sin^2 \frac{\pi h}{2} = O(h)$$

- Maximal eigenvalue:

$$\lambda_{n-1} = \frac{4}{h} \sin^2 \frac{(n-1)\pi}{2n} = \frac{4}{h} \cos^2 \frac{\pi h}{2} = O(h^{-1})$$

- Spectral condition number:

$$\kappa_2(\mathbf{K}) = \frac{\lambda_{\mathsf{max}}(\mathbf{K})}{\lambda_{\mathsf{min}}(\mathbf{K})} = \frac{\cos^2 \frac{\pi h}{2}}{\sin^2 \frac{\pi h}{2}} = \cot^2 \frac{\pi h}{2} = O(h^{-2})$$

Outline

Carl Friedrich Gauss (1777- 1855)

Gaussian Elimination: Idea

Let us write our system (1) $\mathbf{A}\underline{u} = \underline{b}$ in detail as

$$\begin{array}{ccccccccc} A_{11}^{(0)} u_1 & + & A_{12}^{(0)} u_2 & + & \cdots & + & A_{1n}^{(0)} u_n & = & b_1^{(0)} \\ A_{21}^{(0)} u_1 & + & A_{22}^{(0)} u_2 & + & \cdots & + & A_{2N}^{(0)} u_n & = & b_2^{(0)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ A_{n1}^{(0)} u_1 & + & A_{n2}^{(0)} u_2 & + & \cdots & + & A_{nN}^{(0)} u_n & = & b_n^{(0)} \,. \end{array}$$

Use the first eqn to eliminate u_1 from the other eqns:

$$\begin{aligned} U_{1j} &= A_{1j}^{(0)} = A_{1j},\ j = 1, 2, \ldots, n, \\ L_{i1} &= A_{i1}^{(0)} / A_{11}^{(0)},\ i = 2, \ldots, n, \\ A_{ij}^{(1)} &= A_{ij}^{(0)} - L_{i1} U_{1j},\ i, j = 2, \ldots, n, \\ c_1 &= b_1^{(0)} = b_1 \\ b_i^{(1)} &= b_i^{(0)} - L_{i1} c_1,\ i, j = 2, \ldots, n. \end{aligned}$$

Gaussian Elimination: Idea

Let us write our system (1) $\mathbf{A}\underline{u} = \underline{b}$ in detail as

$$\begin{array}{ccccccccc} U_{11}u_1 & + & U_{12}u_2 & + & \cdots & + & U_{1n}u_n & = & c_1 \\ & & A^{(1)}_{22}u_2 & + & \cdots & + & A^{(1)}_{2N}u_n & = & b^{(1)}_2 \\ & & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ & & A^{(1)}_{n2}u_2 & + & \cdots & + & A^{(1)}_{nN}u_n & = & b^{(1)}_n\,. \end{array}$$

Use the first eqn to eliminate u_1 from the other eqns:

$$\begin{aligned} U_{1j} &= A^{(0)}_{1j} = A_{1j},\ j = 1, 2, \ldots, n, \\ L_{i1} &= A^{(0)}_{i1} / A^{(0)}_{11},\ i = 2, \ldots, n, \\ A^{(1)}_{ij} &= A^{(0)}_{ij} - L_{i1}U_{1j},\ i, j = 2, \ldots, n, \\ c_1 &= b^{(0)}_1 = b_1 \\ b^{(1)}_i &= b^{(0)}_i - L_{i1}c_1,\ i, j = 2, \ldots, n. \end{aligned}$$

Gaussian Elimination: Algorithm

If we simply replace superscript (0) by $(k-1)$ and (1) by (k), then we arrive at the Gaussian Elimination Algorithm

Algorithm (Gaussian Elimination Algorithm)

Initialization: $\mathbf{A}^{(0)} = A$, $\underline{b}^{(0)} = \underline{b}$

Forward Elimination:

for $k = 1$ *step* 1 *until* $n-1$ *do*

for $i = k+1$ *step* 1 *until* n *do*

$L_{ik} = A_{ik}^{(k-1)} / A_{kk}^{(k-1)}$

$b_i^{(k)} = b_i^{(k-1)} - L_{ik} b_k^{(k-1)}$

for $j = k+1$ *step* 1 *until* n *do*

$A_{ij}^{(k)} = A_{ij}^{(k-1)} - L_{ik} A_{kj}^{(k-1)}$

endfor

endfor

endfor

Gaussian Elimination: Storage scheme

The intermediate results after $k-1$ can be stored as follows:

$$\begin{pmatrix} U_{11} & U_{12} & \cdots & U_{1k} & \cdots & U_{1n} \\ L_{21} & U_{22} & \cdots & U_{2k} & \cdots & U_{2n} \\ \vdots & \ddots & \ddots & \vdots & \vdots & \vdots \\ L_{k1} & \cdots & L_{k,k-1} & A_{kk}^{(k-1)} & \cdots & A_{kn}^{(k-1)} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ L_{n1} & \cdots & L_{n,k-1} & A_{nk}^{(k-1)} & \cdots & A_{nn}^{(k-1)} \end{pmatrix}$$

Backward Substitution

After n-1 steps, we obtain the upper triangular system

$$\mathbf{U}\underline{u} = \underline{c}$$

with the upper triangular matrix

$$\mathbf{U} = \begin{pmatrix} U_{11} & U_{12} & \cdots & U_{1,n-1} & U_{1n} \\ 0 & U_{22} & \cdots & U_{2,n-1} & U_{2n} \\ \vdots & 0 & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & U_{n-1,n-1} & U_{n-1,n} \\ 0 & 0 & \cdots & 0 & U_{nn} \end{pmatrix} \quad \text{und} \quad \underline{c} = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_{n-1} \\ c_n \end{pmatrix}$$

which can easily be solved by backward substitution:

$$u_n = c_n / U_{nn}; \quad u_i = (c_i - \sum_{j=i+1}^{n} U_{ij} u_j) U_{ii}/, \; i = n-1, n-2, \ldots, 1.$$

Feasibility and operation count

- **Feasibility** via pivoting strategies:
  To avoid $U_{kk} = A_{kk}^{k-1} = 0$, i.e. division by zero, we propose a pivot search in the remainder matrix $\mathbf{A}^{(k-1)}$:
  1. Total pivoting: column and row exchange defined by
     $i^*, j^* \in \{k, \ldots, n\} : |A_{i^*j^*}^{k-1}| \geq |A_{ij}^{k-1}| \;\; \forall i, j = k, \ldots, n.$
  2. column pivoting: column exchange
  3. row pivoting: row exchange

- **Operation count**: SAXPY $(ax + y)$ operations:
  1. Forward elimination $\mathbf{A} = \mathbf{LU}$: $\approx O(n^3) = (n-1)^2 + \ldots + 1^2$
  2. Forward substitution $\underline{c} = \mathbf{L}^{-1}\underline{b}$: $\approx O(n^2) = (n-1) + \ldots + 1$
  3. Bachward substitution $\underline{x} = \mathbf{U}^{-1}\underline{c}$: $\approx O(n^2)$

Gaussian Elimination as LU factorization

- **Exercise:** Show that the n-1 Gaussian elimination steps are equivalent to the LU factorization of **A**, i.e. ($n = 3$)

$$\mathbf{A} = \mathbf{LU} = \begin{pmatrix} 1 & 0 & 0 \\ L_{21} & 1 & 0 \\ L_{31} & L_{32} & 1 \end{pmatrix} \begin{pmatrix} U_{11} & U_{21} & U_{31} \\ 0 & U_{22} & U_{32} \\ 0 & 0 & U_{33} \end{pmatrix},$$

  with the entries L_{ij} and U_{ij} generated by the Gaussian elimination algorithm.
- Therefore, the solution of $\mathbf{A}\underline{u} = \underline{b}$ is equivalent to
  1. factorization: $\mathbf{A} = \mathbf{LU}$ by means of $O(n^3)$ ops
  2. forward substitution: $\mathbf{L}\underline{c} = \underline{b}$ by means of $O(n^2)$ ops
  3. backward substitution:$\mathbf{U}\underline{u} = \underline{c}$ by means of $O(n^2)$ ops

ILU Factorization as Preconditioner

- If we compute the coefficients L_{ij} and U_{ij} in the Gaussian Elimination Algorithm only for the indicies

$$(i,j) \in \mathcal{M} \supseteq \mathcal{M}_{NZE} := \{(i,j) : A_{ij} \neq 0\}$$

and set them to zero otherwise, then we obtain an Incomplete LU factorization of the form

$$\mathbf{A} = \tilde{\mathbf{L}}\tilde{\mathbf{U}} + \mathbf{R}, \text{ i.e., in general, } \mathbf{C} = \tilde{\mathbf{L}}\tilde{\mathbf{U}} \neq \mathbf{A}.$$

In particular, $\mathbf{R} = \mathbf{0}$ if $\mathcal{M} = \{(i,j) : i,j = 1,2,\ldots,n\}$, and the LU and ILU factorizations coincide.

ILU Factorization as Preconditioner

- If we compute the coefficients L_{ij} and U_{ij} in the Gaussian Elimination Algorithm only for the indicies

$$(i,j) \in \mathcal{M} \supseteq \mathcal{M}_{NZE} := \{(i,j) : A_{ij} \neq 0\}$$

  and set them to zero otherwise, then we obtain an Incomplete LU factorization of the form

$$\mathbf{A} = \tilde{\mathbf{L}}\tilde{\mathbf{U}} + \mathbf{R}, \text{ i.e., in general, } \mathbf{C} = \tilde{\mathbf{L}}\tilde{\mathbf{U}} \neq \mathbf{A}.$$

  In particular, $\mathbf{R} = \mathbf{0}$ if $\mathcal{M} = \{(i,j) : i,j = 1,2,\ldots,n\}$, and the LU and ILU factorizations coincide.

- **But who knows what it's good for ?**

ILU Factorization as Preconditioner

- If we compute the coefficients L_{ij} and U_{ij} in the Gaussian Elimination Algorithm only for the indicies

$$(i,j) \in \mathcal{M} \supseteq \mathcal{M}_{NZE} := \{(i,j) : A_{ij} \neq 0\}$$

  and set them to zero otherwise, then we obtain an Incomplete LU factorization of the form

$$\mathbf{A} = \tilde{\mathbf{L}}\tilde{\mathbf{U}} + \mathbf{R}, \text{ i.e., in general, } \mathbf{C} = \tilde{\mathbf{L}}\tilde{\mathbf{U}} \neq \mathbf{A}.$$

  In particular, $\mathbf{R} = \mathbf{0}$ if $\mathcal{M} = \{(i,j) : i,j = 1,2,\ldots,n\}$, and the LU and ILU factorizations coincide.

- **But who knows what it's good for ?** We can hope that $\mathbf{C} = \tilde{\mathbf{L}}\tilde{\mathbf{U}}$ can be used as a good **preconditioner** for $\mathbf{A}$ in iterative methods $\Longrightarrow$ see NL2 and NL3

Special Matrices: Band and Profile Matrices

- **Exercise:** Show that

$$L_{ij} = 0 \quad \text{and} \quad U_{ij} = 0 \quad \forall\, |i-j| > b_w$$

  if $A_{ij} = 0$ for all $|i-j| > b_w$ = bandwidth !

- **Results:**
  1. The bandwidth of $\mathbf{A}$ remains in the LU factors $\mathbf{L}$ and $\mathbf{U}$ of $\mathbf{A}$, but zero coefficients within the band of $\mathbf{A}$ can turn to non-zero coefficients of $\mathbf{L}$ and $\mathbf{U}$. This is call *“fill-in”* !
  2. Factorization needs $O(b_w^2 n)$ ops, whereas
     For- and backward substitutions need $O(b_w n)$ ops only !
  3. Storage requirement is of the order $O(b_w n)$.

- Similar results hold for profiles (sky lines): The row / column resp. column / row profils of $\mathbf{A}$ remains in the $\mathbf{LU}$ resp. $\mathbf{UL}$ factorization of $\mathbf{A}$.

Special Matrices: Symmetric Matrices

The LDL^T factorization of a symmetric and regular matrix $\mathbf{A}$ can be found by comparing the coefficients $(n = 3)$:

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & 0 \\ L_{21} & 1 & 0 \\ L_{31} & L_{32} & 1 \end{pmatrix} \begin{pmatrix} D_{11} & 0 & 0 \\ 0 & D_{22} & 0 \\ 0 & 0 & D_{33} \end{pmatrix} \begin{pmatrix} 1 & L_{21} & L_{31} \\ 0 & 1 & L_{32} \\ 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} D_{11} & D_{11}L_{21} & D_{11}L_{31} \\ L_{21}D_{11} & L_{21}^2 D_{11} + D_{22} & L_{21}L_{31}D_{11} + L_{32}D_{22} \\ L_{31}D_{11} & L_{31}L_{21}D_{11} + L_{32}D_{22} & L_{31}^2 D_{11} + L_{32}^2 D_{22} + D_{33} \end{pmatrix}$$

Algorithm (LDL^T factorization: Algorithm)

$j = 1, \ldots, n$: $D_{jj} = A_{jj} - \sum_{k=1}^{j-1} L_{jk}^2 D_{kk}$

$i = j+1, \ldots, n$: $L_{ij} = D_{jj}^{-1}(A_{jj} - \sum_{k=1}^{j-1} L_{ik}L_{jk}D_{kk})$

Special Matrices: SPD Matrices

The Cholesky factorizations LL^T or UU^T of a SPD matrix $\mathbf{A}$ can also be found by comparing the coefficients $(n = 3)$:

$$\begin{aligned}
\mathbf{A} &= \begin{pmatrix} L_{11} & 0 & 0 \\ L_{12} & L_{22} & 0 \\ L_{13} & L_{23} & L_{33} \end{pmatrix} \begin{pmatrix} L_{11} & L_{12} & L_{13} \\ 0 & L_{22} & L_{23} \\ 0 & 0 & L_{33} \end{pmatrix} \\
&= \begin{pmatrix} L_{11}^2 & L_{11}L_{12} & L_{11}L_{13} \\ L_{12}L_{11} & L_{12}^2 + L_{22}^2 & L_{12}L_{13} + L_{22}L_{23} \\ L_{13}L_{11} & L_{13}L_{12} + L_{23}L_{22} & L_{13}^2 + L_{23}^2 + L_{33}^2 \end{pmatrix}
\end{aligned}$$

Algorithm (Cholesky factorizations LL^T: Algorithm)

$L_{11} = \sqrt{A_{11}}$; **for** $j = 2$ *step* 1 *until* n *do* $L_{1j} = A_{1j}/L_{11}$;
if $j > 2$ *then* $i = 2, \ldots, j-1$: $L_{ij} = L_{jj}^{-1}(A_{ij} - \sum_{k=1}^{i-1} L_{ki}L_{kj})$;
$L_{jj} = \sqrt{A_{jj} - \sum_{k=1}^{j-1} L_{kj}^2}$ **endfor**

Outline

Sparse Direct Methods

- Sparse direct methods like
  - nested disection methods
  - multifrontal methods

  use special elimination strategies:
  1. ordering step: reorder the rows and columns
  2. symbolic facorization: nonzero structure of the facors
  3. numerical factorization: $\mathbf{L}$ and $\mathbf{U}$
  4. solution step: forward and backward substitution
- Software:
  - SuperLU (left-looking)
  - UMFPACK (multifrontal)
  - PARDISO (left-right looking)
  - MUMPS (multifrontal)
- References:
  1. I. Duff: Direct Methods for Sparse Matrices, 1987.
  2. T. Davis: Direct Methods for Sparse Linear Systems, 2006.

Outline

- Linear systems of algebraic equation arising in CA
- Properties of the system matrices
- Gaussian elimination as basic idea of direct methods
- Gaussian elimination and LU factorization
- ILU factorization as preconditioner
- Band and profile matrices
- LDL^T factorization for symmetric matrices
- Cholesky factorization for SPD matrices
- Sparse direct methods

References

[1] M. Jung and U. Langer.
Methode der finiten Elemente für Ingenieuer.
Springer Vieweg, Wiesbaden, 2013.

[2] G. Meurant.
Computer Solution of Large Linear Systems, volume 28 of *Studies in Mathematics and its Applications*.
North Holland, 2013.