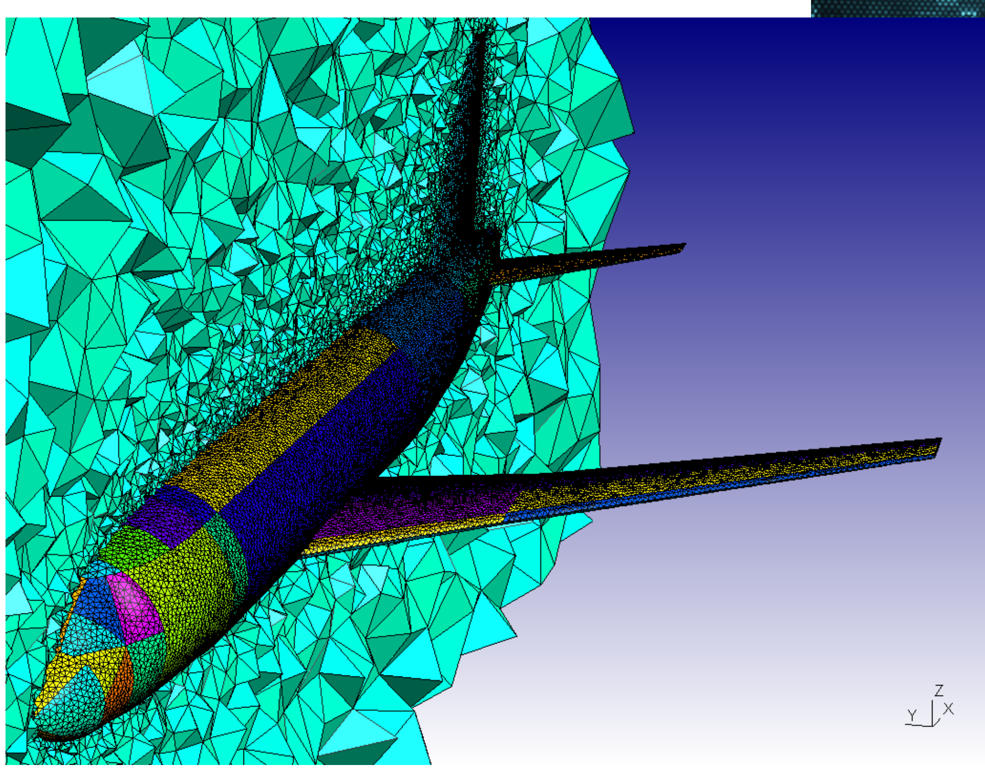
ETH zürich

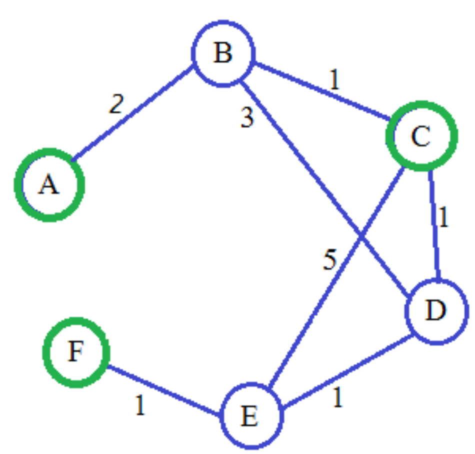
Introduction to Scientific Computation
Lecture 2
Fall 2018

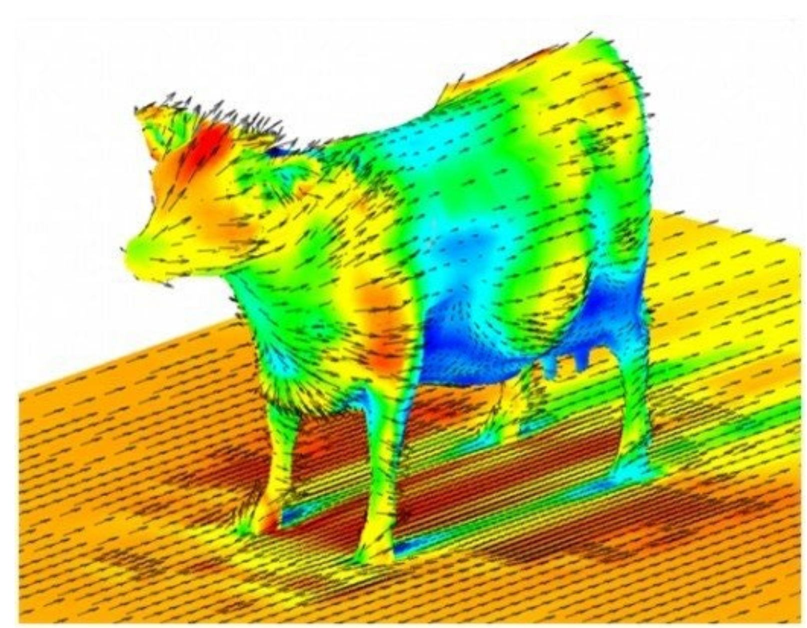
Liners Systems of Equations, SVD
FFT

Mikhail Usvyatsov October 3, 2018

Linear Systems of Equations







How to solve this ?

$$Ax = b$$

How to solve this ?

$$Ax = b$$

Direct methods

$$x = A^{-1}b$$

Exact solution,
except for rounding errors,
fixed number of computational steps

Iterative methods

$$x_t = f(x_{t-1}, A, b)$$

Often work better in practice,
due to care about numerical stability,
iteratively improve solution estimate

ETH zürich

Direct methods

If all linear equations are uncoupled:

If all linear equations are uncoupled:

$$A = \begin{bmatrix} a_{11} & & & \\ & a_{22} & & \\ & & \ddots & \\ & & & a_{nn} \end{bmatrix}$$

If all linear equations are uncoupled:

$$A = \begin{bmatrix} a_{11} & & & \\ & a_{22} & & \\ & & \ddots & \\ & & & a_{nn} \end{bmatrix}$$

Solution is obvious

If all linear equations are uncoupled:

$$A = \begin{bmatrix} a_{11} & & & \\ & a_{22} & & \\ & & \ddots & \\ & & & a_{nn} \end{bmatrix}$$

Solution is obvious

$$x_i = \frac{b_i}{a_{ii}}$$

Upper triangular matrix:

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ & a_{22} & \ddots & \vdots \\ & & \ddots & \vdots \\ & & & a_{nn} \end{bmatrix}$$

$$a_{ij} = 0, \forall i > j$$

$$a_{nn}x_n = b_n, \implies x_n = \frac{b_n}{a_{nn}}$$

$$a_{n-1,n-1}x_{n-1,n-1} = b_{n-1} - a_{n-1,n}x_n, \implies x_{n-1} = \frac{b_{n-1} - a_{n-1,n}x_n}{a_{n-1,n-1}}$$

Upper triangular matrix:

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ & a_{22} & \ddots & \vdots \\ & & \ddots & \vdots \\ & & & a_{nn} \end{bmatrix}$$

$a_{ij} = 0, \forall i > j$

$a_{nn}x_n = b_n, \implies x_n = \dfrac{b_n}{a_{nn}}$

$a_{n-1,n-1}x_{n-1,n-1} = b_{n-1} - a_{n-1,n}x_n, \implies x_{n-1} = \dfrac{b_{n-1} - a_{n-1,n}x_n}{a_{n-1,n-1}}$

Quiz: What is the final complexity ?

Upper triangular matrix:

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ & a_{22} & \ddots & \vdots \\ & & \ddots & \vdots \\ & & & a_{nn} \end{bmatrix}$$

$$a_{ij} = 0, \forall i > j$$

$$a_{nn}x_n = b_n, \implies x_n = \frac{b_n}{a_{nn}}$$

$$a_{n-1,n-1}x_{n-1,n-1} = b_{n-1} - a_{n-1,n}x_n, \implies x_{n-1} = \frac{b_{n-1} - a_{n-1,n}x_n}{a_{n-1,n-1}}$$

Quiz: What is the final complexity ?

Answer: $O(N^2)$

Gaussian Elimination

Exact solution,
except for rounding errors

$$A = LU$$

$$Ly = b$$

$$Ux = y$$

$$PA = LU$$

Not unique

Problems when A is singular

Solution is LUP - additional permutation matrix which aims to maximise absolute values of diagonal elements

Finding LU (LUP) has complexity $O(N^3)$

Special case: SPD matrices

<u>Symmetric positive definite</u>: if $A = A^T$ and $x^T A x > 0, \forall x \neq 0$

$A = LL*$

Two times less memory than LU

Conditioning

Relative error: $\dfrac{||x-\hat{x}||}{||x||}$

$$r = b - A\hat{x} = Ax - A\hat{x} = A(x-\hat{x}) \qquad A\hat{x} = b + \Delta b \implies ||\Delta b|| = ||r||$$

$$x - \hat{x} = A^{-1}r \implies ||x-\hat{x}|| = ||A^{-1}r|| \le ||A^{-1}||\,||r||$$

$$\frac{||x-\hat{x}||}{||x||} \le ||A^{-1}||\,||r||\frac{||A||}{||b||}$$

Condition number: $k(A) = ||A||\,||A^{-1}||$

$$\frac{x-\hat{x}}{||x||} \le k(A)\frac{||\Delta b||}{||b||}$$

the relative error in the solution is bounded by the condition number of the matrix A times the relative error in the residual.

Iterative methods

- Jacobi
- Gauss-Seidel
- SOR

Iterative methods

$$x_t = f(x_{t-1}, A, b)$$

Stationary

$$Mx_t = Nx_{t-1} + b, A = M - N$$

- Jacobi
- Gauss-Seidel
- SOR

Non stationary

- Gradient descent
- Conjugate gradients (for SPD)

Stationary methods

$Mx_t = Nx_{t-1} + b, A = M - N$

We need initial guess for the solution: x_0

Error is: $e_0 = x - x_0$

We do not know the exact solution (if we know, we do we care ?) and cannot compute

$r_0 = Ae_0 = b - Ax_0$

We can compute

$x = x_0 + e_0 = x_0 + A^{-1}r_0$

Set $M\widetilde{e} = r_k$

Preconditioner: M

Then $x_t = x_{t-1} + \widetilde{e}$

$A = D + E + F$

Jacobi: $M = D$

Gauss-Seidel: $M = D + E$

SOR: $M = D + \frac{1}{\omega} E$

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ a_{21} & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & 0 \end{bmatrix} \quad E$$

$$+ \begin{bmatrix} 0 & a_{12} & \dots & a_{1n} \\ 0 & 0 & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix} \quad F$$

$$+ \begin{bmatrix} a_{11} & 0 & \dots & 0 \\ 0 & a_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a_{nn} \end{bmatrix} \quad D$$

Nonstationary methods

$$x_t = x_{t-1} + \alpha_t p_t$$

Gradient descent

$$x_t = x_{t-1} + \alpha_t p_t$$
$$p_k = r_k$$

or

$$M\widetilde{e} = r_t$$
$$M = \alpha_t I$$

Gradient descent

$$x_t = x_{t-1} + \alpha_t p_t$$

$$p_k = r_k$$

or

$$M\widetilde{e} = r_t$$

$$M = \alpha_t I$$

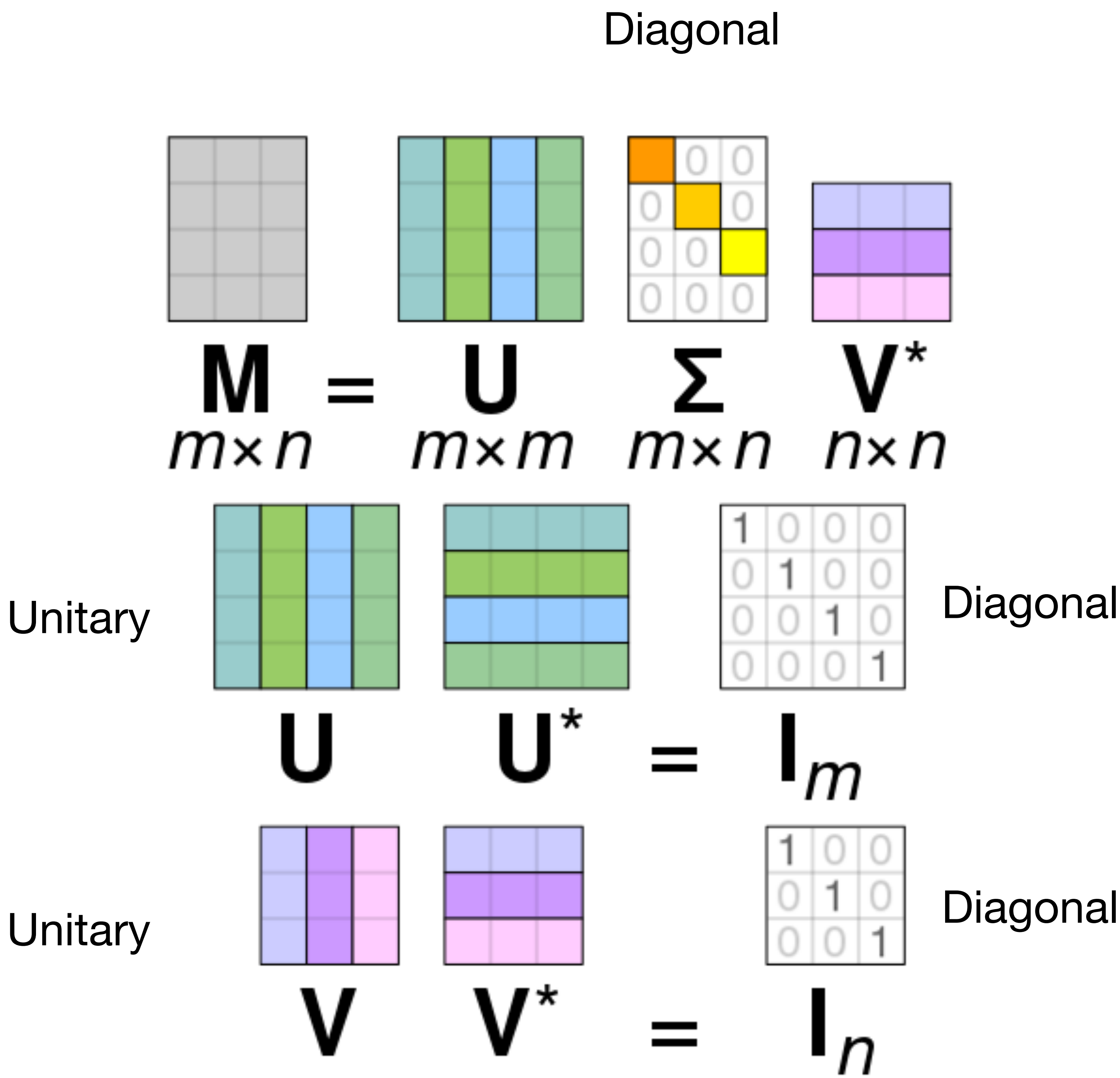
Conjugate gradient (SPD only)

$$x_t = x_{t-1} + \alpha_t p_t$$

$$p_t = r_t - \sum_{i<t} \frac{p_i^T A r_t}{p_i^T A p_i} p_i$$

$$\alpha_t = \frac{p_t^T r_t}{p_t^T A p_t}$$

SVD

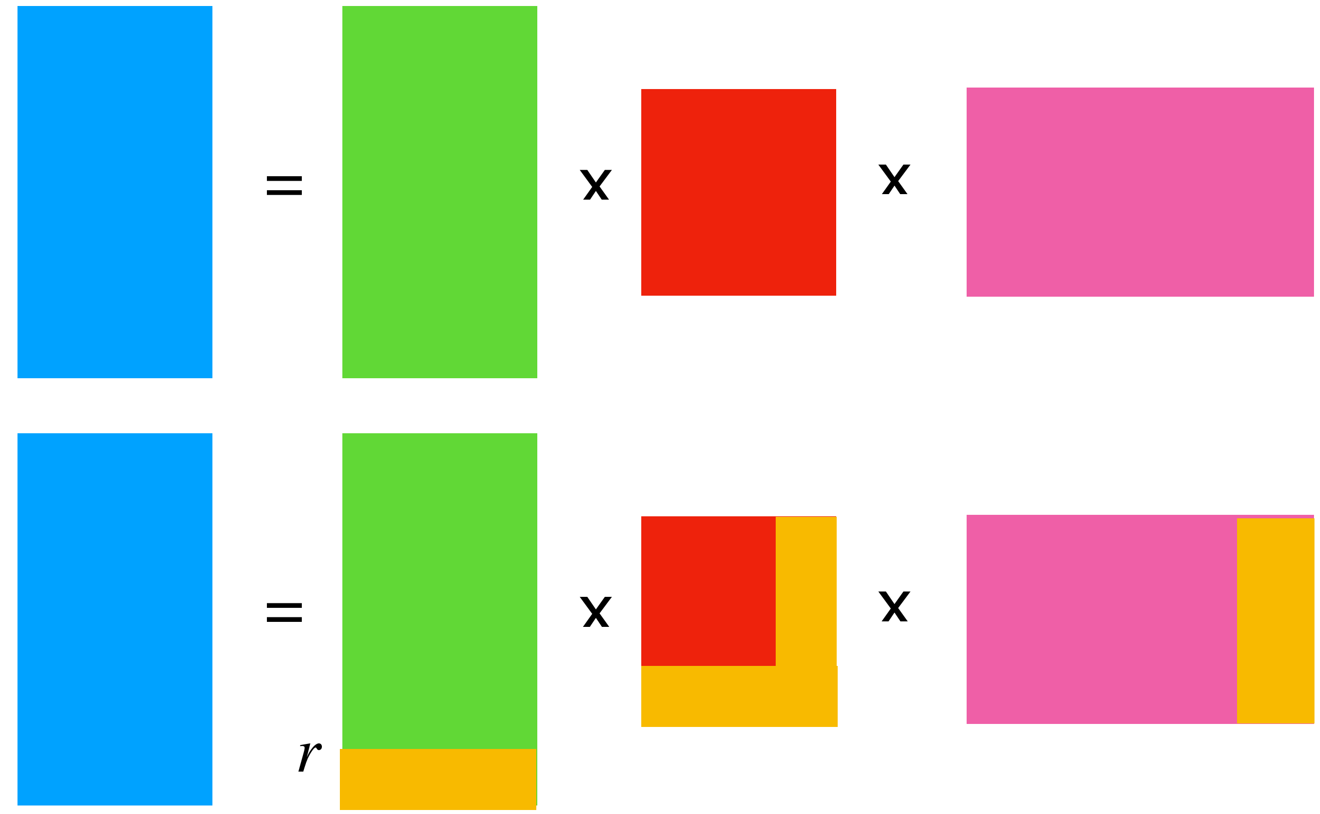


Diagonal
$\mathbf{M} = \mathbf{U}\,\boldsymbol{\Sigma}\,\mathbf{V}^*$
$m \times n$
$m \times m$
$m \times n$
$n \times n$
Unitary
$\mathbf{U}\,\mathbf{U}^* = \mathbf{I}_m$
Diagonal
Unitary
$\mathbf{V}\,\mathbf{V}^* = \mathbf{I}_n$
Diagonal

$M = U \Sigma V^*$

If $rank(A) = r \implies \forall i > r, \sigma_i = 0$

$\sigma_1 \geq \ldots \geq \sigma_n$



Eckart-Young theorem: best low-rank approximation can be computed with SVD

$$\min_{rank(B)=r} ||A - B||_2 = ||A - A_r||_2 = \sigma_{r+1}$$

r - rank approximation is setting $\forall i > r, \sigma_i = 0$

FFT

See code