Underdetermined system

$Ax=b$

Find least squares soln

$$\begin{array}{ccc}\begin{bmatrix}3 & 1 & 0\\ 1 & 0 & 1\end{bmatrix} & \begin{bmatrix}x\\ y\\ z\end{bmatrix} & = \begin{bmatrix}1\\ 2\end{bmatrix}\\ 2\times 3 & 3\times 1 & 2\times 1\end{array} \quad \text{expect} \quad \begin{bmatrix}5/11\\ -4/11\\ 17/11\end{bmatrix}$$

minimize $\|x\|$
subject $Ax=b$

Assume
$$x_{LN}=A^T(AA^T)^{-1}b$$

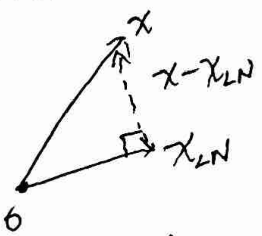
$$x_{LN}^T x_{LN} = \cancel{(A^T(AA^T)^{-1}b)^T(A^T(AA^T)^{-1}b)} = \cancel{b}$$

$$(x-x_{LN})^T x_{LN}$$
$$=(x-x_{LN})^T A^T (AA^T)^{-1} b$$
$$= \underbrace{(A(x-x_L))^T}_{\substack{\parallel \\ 0}} (AA^T)^{-1} b = 0$$

because $Ax=y$
$Ax_{LN}=y$

$\therefore (x-x_{LN}) \perp x_{LN}$

$$\|x\|^2 = \|x_{LN}+x-x_{LN}\|^2$$
$$= \|x_{LN}\|^2 + \|x-x_{LN}\|^2 \geq \|x_{LN}\|^2$$




$x_{LN}=A^T(AA^T)^{-1}b$ is a least-norm solution

~~Does A~~

QR decomposition

orthonormal
upper triangular

$A^T = QR$ $\qquad A=R^TQ^T$

$Ax=b$

$A=R^TQ^T$

$\cancel{A x = b}$ $R^TQ^Tx=b$
$Q^Tx=R^{-T}b$

$$x_{LN}=A^T(AA^T)^{-1}b$$
$$=QR(R^T\underbrace{Q^TQ}_{I}R)^{-1}b$$
$$=QR(R^TR)^{-1}b$$

? $\cancel{b \to QQ^TR^{-T}b}$ $=QR^{-T}b$

Just do ~~forward~~ ~~back~~ substitution
forward

$\|x_{LN}\|=\|R^{-T}b\|$

$\therefore x_{QR} = x_{LN}$

but column pivot QR doesn't return $x_{LN}$?

Test on Eigen

only SVD gives right solution
(i.e. equal to normal equation solution)

$P\begin{pmatrix}R\\0\end{pmatrix}^T \cancel{Q^TQ} \begin{pmatrix}R\\0\end{pmatrix}P^T$

$A=Q\begin{pmatrix}R\\0\end{pmatrix}P^T$

$P(R^T\ 0)\begin{pmatrix}R\\0\end{pmatrix}P^T$

$$\|Ax-b\|^2 = x^T(A^TA)x - 2(A^Tb)^Tx + b^Tb$$

$$(P\,R^TR\,P^T)x - P(R^T\ 0)Q^Tb = 0$$

$$\cancel{2}(A^TA)x - \cancel{2}(A^Tb) = 0$$

other options

- Complete orthogonal decomposition
- ABA / Featherstone — what's good about this?
- exploit structure $\boxed{\text{check speed for this}}$ — $\boxed{\text{get typical equation}}$

$$\begin{array}{c|c} & \begin{matrix} n_1 & n_2 \end{matrix} \\ \begin{matrix} m_1 \\ m_2 \end{matrix} & \begin{bmatrix} M & -J^T \\ J & 0 \end{bmatrix} \end{array} \begin{bmatrix} \dot{v} \\ f \end{bmatrix}\begin{matrix} n_1 \\ n_2 \end{matrix} = \begin{bmatrix} rhs_1 \\ rhs_2 \end{bmatrix}$$

Complexities

QR : $O(mn^2 - n^3/3)$

SVD : $O(mn^2 + n^3)$

$$[J \quad 0]\begin{bmatrix} \dot{v} \\ f \end{bmatrix} = rhs_2$$

solve 2 smaller problems vs. 1 big one

$J\dot{v} = rhs_2$

block backward sub

$\dot{v}_{sol}$

$M\dot{v} - J^T f = rhs_2$

$-J^T f = rhs_2 - M\dot{v}$

$f = J^{-T}(rhs_2 - M\dot{v})$

COD

orthogonal = unitary + real

$$AP = Q\begin{bmatrix} T & 0 \\ 0 & 0 \end{bmatrix} Z$$

permutation

upper triangular matrix rank x rank $R$

$[H][\dot{v}] = rhs$

or

$$\begin{bmatrix} H & -J^T \\ J & 0 \end{bmatrix}\begin{bmatrix} \dot{v} \\ f \end{bmatrix} = [rhs]$$

SVD : 1300000 ns

COD : 90000 ns

solve() 319211 Ir per call

5946001 Ir per call

Intro to HPC ch5.3
Eijkhout

## LU factorization

$$A = \begin{bmatrix} 6 & -2 & 2 \\ 12 & -8 & 6 \\ 3 & -13 & 3 \end{bmatrix} \begin{array}{c} ① \\ ② \\ ③ \end{array}$$

With Gaussian elimination:

② − ① × 2

③ − ① × $\frac{1}{2}$

$$\begin{bmatrix} 6 & -2 & 2 \\ 0 & -4 & 2 \\ 0 & -12 & 2 \end{bmatrix} \begin{array}{c} ①' \\ ②' \\ ③' \end{array} \xrightarrow{③ - ② \times 3} \begin{bmatrix} 6 & -2 & 2 \\ 0 & -4 & 2 \\ 0 & 0 & -4 \end{bmatrix}$$

$Ax = b$

This can be done with

$L_1 A x = L_1 b$, $L_1 A$

where $L_1 = \begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -\frac{1}{2} & 0 & 1 \end{bmatrix}$

$L_1^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ \frac{1}{2} & 0 & 1 \end{bmatrix}$

$L_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -3 & 1 \end{bmatrix} \rightarrow L_2^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 3 & 1 \end{bmatrix}$

$Ax = b \longrightarrow \underbrace{L_2 L_1 A}_{\text{upper triangular}} x = L_2 L_1 b$

$U = L_2 L_1 A \rightarrow A = L_1^{-1} L_2^{-1} U$

easy to compute

For triangular matrices, inverse is just taking negative of off-diagonals

Always has ones on diagonal because we eliminate by subtracting other row multiplied by $c$.

$L = L_1^{-1} L_2^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ \frac{1}{2} & 3 & 1 \end{bmatrix}$

$A = LU$

Inverse of operations taken to transform $A$ to $U$

result of Gaussian elimination

$diag(U) =$ "pivots"

~~Can be done in-place~~

L can be computed in-place, but we need to store U somewhere else

With cholesky, $A = LL^T$ so fully in place

Algorithm

for $k = [1 : n-1]$:
    EliminateValuesInCol($k$) — to make upper triangular

EliminateValuesInCol($k$)
  for $i = k+1 : n$
    $a_{ik} = a_{ik} / a_{kk}$ — compute multiplier for row $i$
    — what if $a_{kk} = 0$? Pivot row w/ one of the lower rows
    for $j = k+1 : n$ — update row $i$
      $a_{ij} = a_{ij} - a_{ik} \times a_{kj}$

This can be done for blocks too

Solutions are not unique, "up to diagonal scaling"
Makes sense because there's lots of ways to do Gaussian elimination.

Can make unique by requiring $diag(L) = 1$

Review 5.3.4

Pivots can be represented by matrix mult
$\rightarrow PA = LU$
$A = P^{-1} LU$

## Solving $Ax = b$

$A = LU$ $\quad O(n^2)$ given factorization

$LUx = b$ $\quad O(\frac{2}{3} n^3)$ for LU factorization

$Ly = b \rightarrow Ux = y$

Forward and backwards sub

Note that this doesn't work if any $diag(U) == 0$

Note that both steps (factorization, substitution) of solving with LU use recursion → not easy to parallelized.

# Cholesky factorization

symmetric matrix $A \longrightarrow A = LL^T$

$$A = \begin{bmatrix} A_{11} & A_{21}^T \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} \begin{bmatrix} L_{11} & L_{21}^T \\ 0 & L_{22}^T \end{bmatrix} = LL^T$$

$$L_{11}^2 = A_{11}$$

$$L_{11}L_{21}^T = A_{21}^T \longrightarrow L_{21}^T = L_{11}^{-1} A_{21}^T$$

$$L_{21}L_{21}^T + L_{22}L_{22}^T = A_{22} \longrightarrow \underbrace{L_{22}L_{22}^T}_{} = A_{22} - L_{21}L_{21}^T$$

$L_{22}$ is Cholesky factor of update $A_{22}$ block

Recursive definition

Base case: $L_{22} \in \mathbb{R}^{1\times 1}$

$$A = \begin{bmatrix} 9 & 2 \\ 2 & 5 \end{bmatrix} = \begin{bmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} \begin{bmatrix} L_{11} & L_{21}^T \\ 0 & L_{22}^T \end{bmatrix}$$

$L_{11}^2 = A_{11} = 9$ $\qquad L_{11}L_{21}^T = A_{21}^T$ $\qquad L_{22}L_{22}^T = A_{22} - L_{21}L_{21}^T$

$L_{11} = 3$ $\qquad 3L_{21} = 2$ $\qquad L_{22}^2 = 5 - \frac{2}{3}\cdot\frac{2}{3} = 5 - \frac{4}{9} = 4\frac{5}{9}$

$L_{21} = 2/3$ $\qquad = \frac{41}{9}$

$L_{22} = \sqrt{\frac{41}{9}}$

- Pivoting:
  Always do row exchanges to get the largest remaining element in current column into pivot position (row pivoting).

Big advantage of decompositions: re-use left side for multiple algorithms.

Gaussian elimination on LU factorization can be interpreted as removing all edges between a node (column being zeroed) and the rest of the graph. This might result in other edges being added as "fill-in". We want to find an ordering of elimination that reduces fill-in.

Iterative methods

Jacobi method

Instead of taking decomposition of $A$ and solving $x = A^{-1}b$, just do iterative: $\begin{cases} \text{Start @ } x_0 \\ x_{i+1} = Bx_i + c \\ \text{until stopping conditions} \end{cases}$

eg Iterative solve $[\diagdown][x] = [b]$ or $[\triangle]x = [b]$ — This in $O(n^2)$ or $O(n)$ vs. $O(n^3)$ for direct solve

Requires some assumption about the solution?

Init $x_0$
For $i \geq 0$
  Let $r_i = Ax_i - b$ (residual)
  compute $e_i$ from $Ke_i = r_i$
  update $x_{i+1} = x_i - e_i$

$$x_{i+1} = x_i - K^{-1}r_i$$

$\cancel{xA}$