Chaps 6+7: Iterative Methods

Part 2: Poisson Equation in higher dim. using Kronecker Products

Summary of all methods applied to Poisson Eq, sorted by speed

$$T_N = \begin{bmatrix} 2 & -1 & & 0 \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ 0 & & -1 & 2 \end{bmatrix}^{N \times N} \qquad T_{N \times N} = \begin{bmatrix} T_N + 2I_N & -I_N & & \\ -I_N & \ddots & \ddots & \\ & \ddots & \ddots & -I_N \\ & & -I_N & T_N + 2I_N \end{bmatrix}$$

Def: Let X be $m \times n$ matrix, then $vec(X) = m \cdot n \times 1$ vector consisting of cols of X stacked on top of one another from left to right

$$vec(X) = \begin{bmatrix} x_{11} \\ x_{21} \\ \vdots \\ x_{m1} \\ x_{21} \\ \vdots \\ x_{mn} \end{bmatrix}$$

Matlab: $vec(X) = $ reshape$(X, m \cdot n, 1)$

Def: Let A be $m \times n$, B is $p \times q$
then $A \otimes B$ is $m \cdot p \times n \cdot q$ matrix:

$$\begin{bmatrix} A_{11} \cdot B & A_{12} \cdot B & \cdots & A_{1n} \cdot B \\ A_{21} \cdot B & & & \vdots \\ \vdots & & \ddots & \\ A_{m1} \cdot B & & \cdots & A_{mn} B \end{bmatrix}$$

called Kronecker product of A and B
Matlab: $A \otimes B = \text{kron}(A, B)$

Lemma: A be $m \times m$, B $n \times n$, X, C are $m \times n$

1) $\text{vec}(A \cdot X) = (I_n \otimes A) \cdot \text{vec}(X)$
2) $\text{vec}(X B) = (B^T \otimes I_m) \cdot \text{vec}(X)$
3) 2D Poisson Equation $T_N V + V T_N = h^2 F$
can be written
$(I_N \otimes T_N + T_N \otimes I_N) \cdot \text{vec}(V) = h^2 \, \text{vec}(F)$

Proof of 1): $I_n \otimes A = \text{diag}(\overbrace{A, A, A, \ldots A}^{n})$
so $(I_n \otimes A) \cdot \text{vec}(X) =$

$\text{diag}(A, \ldots, A) \cdot [X(:,1); X(:,2); \ldots; X(:,n)]$

$= [A \cdot X(:,1); A \cdot X(:,2); \ldots; A \cdot X(:,n)]$

$= \text{vec}(A \cdot X)$

Proof of 2): similar (HW)

Proof of 3): apply 1) to $T_N \cdot V$

apply 2) to $U \cdot T_N$

$$I_N \otimes T_N + T_N \otimes I_N =$$

$$\begin{bmatrix} T_N & & 0 \\ & \ddots & \\ 0 & & T_N \end{bmatrix} + \begin{bmatrix} 2 \cdot I_N & -I_N & & \\ -I_N & \ddots & \ddots & \\ & \ddots & \ddots & -I_N \\ & & -I_N & 2 I_N \end{bmatrix}$$

Matlab: T_N = 2·eye(N) − diag(ones(N−1,1),−1)
− diag(ones(N−1,1),+1)

T_{NN} = kron(eye(N), T_N) + kron(T_N, eye(N))

Lemma: (homework)

1) Suppose $A \cdot C$ and $B \cdot D$ well defined

Then $(A \otimes B) \cdot (C \otimes D) = (A \cdot C) \otimes (B \cdot D)$

2) If A, B invertible then

$$(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}$$

3) $(A \otimes B)^T = A^T \otimes B^T$

Prop: $T^{N\times N} = Z \Lambda Z^T$ be eigendecomp of $T = T^T$

then eigendecomp of

$I \otimes T + T \otimes I$

$= (Z \otimes Z)(I \otimes \Lambda + \Lambda \otimes I)(Z^T \otimes Z^T)$

Here $I \otimes \Lambda + \Lambda \otimes I$ is diagonal with

entry $(i \cdot N + j)$th diagonal is $\lambda_i + \lambda_j$

and column $(i \cdot N + j)$ of Z is $z_i \otimes z_j$

Proof: $(Z \otimes Z)(I \otimes \Lambda + \Lambda \otimes I)(Z^T \otimes Z^T)$

$= (Z \cdot I \otimes Z \cdot \Lambda + Z \cdot \Lambda \otimes Z \cdot I)(Z^T \otimes Z^T)$

$= (Z \cdot I \cdot Z^T \otimes Z \cdot \Lambda \cdot Z^T + Z \cdot \Lambda \cdot Z^T \otimes Z \cdot I \cdot Z^T)$

$= (I \otimes T + T \otimes I)$

as desired

$(Z \otimes Z)(Z \otimes Z)^T = \cdots = I \otimes I = I$

Apply to Express 3D Poisson:

$$T_{N\times N\times N} = (T_N \otimes I_N \otimes I_N) + (I_N \otimes T_N \otimes I_N) + (I_N \otimes I_N \otimes T_N)$$

$$= (Z \otimes Z \otimes Z)(\Lambda \otimes I_N \otimes I_N + I_N \otimes \Lambda \otimes I_N + I_N \otimes I_N \otimes \Lambda)(Z^T \otimes Z^T \otimes Z^T)$$

Eigenvalues consist of all sums $\lambda_i + \lambda_j + \lambda_k$ for all triples (i,j,k)

$\Rightarrow$ Now know enough to solve Poisson using FFT (direct, not iterative, but present for comparison with other methods)

$$T_N \cdot V + V \cdot T_N = h^2 F$$

$$Z^T (Z \Lambda Z^T \cdot V + V \cdot Z \Lambda Z^T = h^2 F) Z$$

$$(\Lambda Z^T V + Z^T V Z \Lambda Z^T = h^2 Z^T F Z$$

$$\Lambda (Z^T V Z) + (Z^T V Z) \Lambda = h^2 (Z^T F Z)$$

$$(\Lambda \cdot V' + V' \cdot \Lambda = h^2 F')$$

diagonal system of equations

$$\Lambda_{ii} V'_{ij} + V'_{ij} \Lambda_{jj} = h^2 F'_{ij}$$

$$\lambda_i V'_{ij} + V'_{ij} \lambda_j = h^2 F'_{ij}$$

$$V'_{ij} = h^2 F'_{ij} / (\lambda_i + \lambda_j)$$

1) Form $F' = Z^T F Z$
2) Solve for $V'_{ij} = h^2 F'_{ij} / (\lambda_i + \lambda_j)$
3) Form $V = Z V' Z^T$

Cost: 4 multiplications by Z or Z^T
$+ N^2$ multiplies/divides

straightforward matmul by Z costs $O(N^3)$
but Z closely related to FFT
so only costs $O(N^2 \log N)$

$FFT_{ij} = e^{\sqrt{-1}\, \pi i j / N}$, very similar to Z

$= \cos ij\pi/N + \sqrt{-1}\, \underline{\sin ij\pi/N}$

Same idea applies to 3D Poisson
and higher dimensions:

$$T^{-1} = \left[(Z \otimes Z \otimes Z)(\Lambda \otimes I \otimes I + I \otimes \Lambda \otimes I + I \otimes I \otimes \Lambda) \cdot (Z^T \otimes Z^T \otimes Z^T) \right]^{-1}$$

$= (Z \otimes Z \otimes Z) \cdot (\text{diagonal})^{-1} (Z^T \otimes Z^T \otimes Z^T)$

Next: Summarize performance of all iterative methods on 2D+3D Poisson

Metrics: #flops, memory, #parallel steps, #procs

Notation: Explicit Inverse: suppose we have precomputed and stored it, don't count cost computing it

SOR: Successive Over relaxation

SSOR/Chebyshev: Symmetric SOR with Chebyshev acceleration

FFT: Fast Fourier Transform

BCR: Block Cyclic Reduction

Lower Bound: assumes one flop per component of output

SPMV = Sparse matrix times dense vector multiply

2D Poisson on $n \times n$ mesh $\Rightarrow N = n^2$ unknowns

3D Poisson on $n \times n \times n$ mesh $\Rightarrow N = n^3$ unknowns

In table, entries are for 2D Poisson, if 3D different, shown in parenthesis

All table entries are $O(\ )$

Method	Direct or Iterative	#flops	Mem	#Par steps	#Procs
Dense Cholesky works on any s.p.d. A	D	N^3	N^2	N	N^2
Band Cholesky	D	N^2 $(N^{7/3})$	$N^{3/2}$ $(N^{5/3})$	N	N $(N^{(4/3)})$

works on any spd band matrix of bandwidth b with cost $O(N \cdot b^2)$

2D $b = n = N^{1/2}$, 3D $b = n^2 = N^{2/3}$

Method	Direct or Iterative	#flops	Mem	#Par steps	#Procs
Explicit Inverse	D	N^2	N^2	$\log N$	N^2
Jacobi	I	N^2 $(N^{5/3})$	N	N $(N^{2/3})$	N

$\#$ flops $= O(\#\text{flops}(\text{SPMV}) \cdot \#\text{iterations})$

$\#\text{flops}(\text{SPMV}) = O(\#\text{nonzeros in } A)$

$\#\text{iterations} \sim \text{cond}(A) = n^2$ in 2D and 3D in general: works on any "diagonally dominant" matrix

Gauss-Seidel	I	N^2 $(N^{5/3})$	N	N $(N^{2/3})$	N

analogous to Jacobi (constants differ)

works on any diagonally dominant or spd matrix

Sparse Cholesky	D	$N^{3/2}$ (N^2)	$N \log N$ $(N^{4/3})$	$N^{1/2}$ $(N^{2/3})$	N $(N^{4/3})$

Assumes we use best ordering of rows and columns: nested dissection

works on any spd matrix

Conjugate Gradients CG	I	$N^{3/2}$ $(N^{4/3})$	N	$N^{1/2} \log N$ $(N^{1/3} \log N)$	N

$\#\text{flops} = O(\#\text{flops}(\text{SPMV}) \cdot \#\text{iterations})$

$\#\text{iterations} = O\left(\sqrt{\text{cond}(A)}\right)$

works on any spd matrix

SOR	I	$N^{3/2}$ $(N^{4/3})$	N	$N^{1/2}$ $(N^{4/3})$	N

works on any spd matrix

SSOR/ Chebyshev	I	$N^{5/4}$ $(N^{7/6})$	N	$N^{1/4}$ $(N^{1/6})$	N

$\#\text{flops} = O(\#\text{flops}(\text{SAMU}) \cdot \#\text{iterations})$

$\#\text{iterations} = O\left((\text{cond}(A))^{\frac{1}{4}}\right)$

works on any spd matrix, but need to know range of eigenvalues

---

FFT	D	$N \cdot \log N$	N	$\log N$	N

Works on Poisson

BCR	D	$N \cdot \log N$	N	?	?

slightly more general than Poisson

Multigrid	I	N	N	$\log^2 N$	N

many variants, apply to many problems arising from elliptic PDE and FEM

Lower Bound	N	N	logN