Fast Poisson Solvers

- Multigrid
- Fast Fourier Transforms
- Domain Decomposition
- Fast Multipoles

Multigrid

Here for 2D, Finite Differences, $N \times N$ square, $N = 2^m$.

- Direct inversion of $N^2 \times N^2$ matrix $\rightarrow \frac{1}{3}N^6$ operations
- Gauss-Seidel N^2 iterations $\rightarrow N^4$ operations
- Successive-Over-Relaxtion N iterations $\rightarrow N^3$ operations
- Multigrid $\rightarrow N^2$ operations.

Problem with Gauss-Seidel: slow diffusion across grid of longwave errors, shortwave errors diffuse rapidly
Hence tackle longwave errors on a faster coarse grid

Coarsest grid $\Delta x = \frac{1}{2}$, one interior point
Finest grid $\Delta x = \frac{1}{2^m}$, $(2^m - 1)^2$ interior points

Multigrid - sequence of problems

Sequence of Poisson problems

$$A_k x_k = b_k,$$

for grids $k = m$, the finest, to $k = 1$, the coarsest.

Make several V-cycles

Each cycle starts at the finest, descends one level at a time to the coarsest and then ascends back to the finest.

For the first cycle, start iteration with $x_m = 0$.
For subsequent cycles, start with x_m from previous V-cycle.

V-cycle, the descent

Starting with $k = m$

- Make a couple of Gauss-Seidel iterations of $A_k x_k = b_k$.
- Produces x_k^{approx}. Store for later use
- Calculate residue

$$\mathrm{res}_k = b_k - A_k x_k^{\mathrm{approx}}.$$

- Coarsen residue for forcing on the next coarser grid

$$b_{k-1} = C_k \mathrm{res}_k \quad \text{where} \quad C_k = \frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}.$$

- Store b_{k-1} for later use
- Zero x_{k-1} for starting iterations
- To courser grid: $k \to k-1$
- If $k > 1$ go to the top of this list

End descent on coursest grid ($k = 1$) with just one internal point, so $A_1 x_1 = b_1$ is one equation in one unknown, solved exactly.

V-cycle, the ascent

Starting with $k = 2$.

- Courser solution x_{k-1} interpolated to finer grid

$$x_k^{\text{correction}} = I_k x_{k-1} \quad \text{where} \quad I_k = \frac{1}{4}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}.$$

- Add this to stored x_k^{approx} from descent

$$x_k^{\text{better approx}} = x_k^{\text{approx}} + x_k^{\text{correction}}$$

- Make a couple of Gauss-Seidel iterations of $A_k x_k = b_k$ starting from $x_k^{\text{better approx}}$, using stored b_k
- To finer grid: $k \to k+1$
- If $k < m$ go to top of this list

End ascent with x_m

Multigrid not: first solve coursest Poisson, then interpolate for starting finer. Coarsening residue gives different forcing

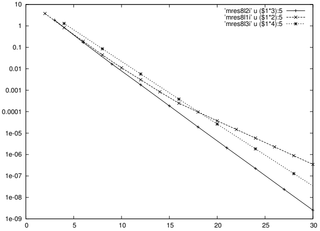
Multigrid – costs

Solve on 256×256 grid

$$\nabla^2 \psi = -2\pi^2 \sin(\pi x) \sin(\pi y)$$

Residue vs
#V-cycles $\times$ #GS iterations

From top, GS iterations = 1,3,2



Error reduces by 10 with 2 GS iterations at each level per V-cycle

$8N^2$ cost per V-cycle

Hence for 10^{-4} accuracy, cost is $32N^2$ cf $2N^3$ by SOR

Fast Fourier Transforms

See spectral methods for details of making fast transform

Poisson problem trivial in Fourier space. Cost in transforms.

For $N \times N$ problem in 2D, there are N^2 Fourier amplitudes.

- Simple calculation of amplitudes cost N^4.
- Orszag speedup gives N^3.
- Fast Fourier Transform reduces to $N^2 \ln N$

For 3D channel flow, FT in 2 periodic directions, FD in 3rd

Invert FD tridiagonal $\rightarrow$ cost $N^3 \ln N$

Domain decompostion

Good for complex geometry, very large problems – reduces memory requirements, FE and FD, parallelisable

- Divide domain into many sub-domains
- For each sub-domain, identify internal points which only involve internal variables x and boundary variables y.
- Solve internal variables x in terms of boundary variables y
- Solve reduced ‘Schur complement’ for boundary variables y.

Domain decomposition

For Poisson problem $Ax = b$, and K subdomains,
internal variables $x_1, x_2, \ldots, x_K$ boundary variables y

Internal problems

$$A_k x_k + B_k y = b_k.$$

Boundary problem

$$C_1 x_1 + C_2 x_2 + \ldots + C_K x_K + Dy = b_0.$$

i.e.

$$\begin{pmatrix} A_1 & & & & B_1 \\ & A_2 & & & B_2 \\ \vdots & \vdots & \ddots & & \vdots \\ & & & A_K & B_K \\ C_1 & C_2 & \cdots & C_K & D \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_K \\ y \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_K \\ b_0 \end{pmatrix}$$

Domain decomposition

Solution of internal problems, parallelisable, small memory each

$$x_k = A_k^{-1}(b_k - B_k y).$$

Hence problem for boundary variables

$$\left(D - C_1 A_1^{-1} B_1 - \cdots - C_K A_K^{-1} B_K\right) y = b_0 - C_1 A_1^{-1} b_1 - \cdots - C_K A_K^{-1} b_K.$$

If using direct LU inversion

- $N \times N$, full domain costs N^6
- K subdomains, cost N^6/K^3 per subdomain + $N^3 K^{3/2}$ boundary
- e.g. $N = 100$, $K = 25$: full 10^{12}, DD parallel 10^9 operations

- $N \times N \times N$, full domain costs N^9
- K subdomains, cost N^9/K^3 per subdomain + $N^6 K$ boundary
- e.g. $N = 100$, $K = 27$: full 10^{18}, DD parallel 10^{14} operations

Fast Multipole Method

For long range interations (potential flow or Stokes flow)

between N point-particles seems N^2 problem

Clustering effect of far particles (Barnes-Hut) gives $N \ln N$

Making clusters multipoles + polynomial local effects

(Greengard-Rokhlin) gives N

Here in 2D for

$$w(z_i) = \sum_{j \neq i}^{N} q_j \ln(z_i - z_j),$$

Trees, roots and leaves

Hierarchy of domains: divide initial square box into 4 equal squares; divide each sub-square into 4;
continue through $\ln_4 N$ levels, so on average only one in smallest.
Some smallest will be empty, some contain more than one.

Tree structure: at any level, smaller box within is a 'child', larger box which contains it is the 'parent'.
Top of tree is 'root'.
Once branch contains no particle stop subdivision,
Smallest non-empty box down a branch is a 'leaf'.

Barnes-Hut algorithm

Upward pass from leaves to root, one level at a time

- Sum charges q_c to charge of parent $q_p = \sum q_c$.
- Find center of mass of charges $z_p = \sum z_c q_c / \sum q_c$.

Downward pass for each particle, starting one below root

- If box is far, then contribution from cluster
- If box is not far and not end, go down a level
- If box is not far and end, sum contributions of individual particles

A box which is not adjacent is far.

Cost in 2D is $27N \ln_4 N$, beats N^2 if $N > 200$
Cost in 3D is $189N \ln_8 N$, beats N^2 if $N > 2000$

Fast Multipoles – upward pass

Far shifts of point charge at z_i to multipoles about center z_c

$$\ln(z - z_i) = \ln(z - z_c) + \sum_{r=1} \frac{(z_c - z_i)^r}{r(z - z_c)^r}.$$

Similary shift multipole at z_i

$$\frac{1}{(z - z_i)^m} = \sum_{r=0} b_r^m \frac{(z_c - z_i)^r}{(z - z_c)^{m+r}},$$

where b_r^m is a binomial coefficient.

Upward pass from leaves to root

- Use far shifts to move multipoles of children to centre of parent

Fast Multipoles – downward pass

Local shift of polynomial variation centred on parent z_p to centred on child z_c

$$(z - z_p)^m = \sum_{r=0}^{m} c_r^m (z - z_c)^r (z_c - z_p)^{m-r},$$

where c_r^m is a binomial coefficient.

Local expansion about centre of child at z_c of multipole at z_b

$$\frac{1}{(z - z_b)^m} = \sum_{r=o}^{\infty} b_r^m \frac{(z - z_c)^r}{(z_c - z_b)^{m+r}}.$$

Fast Multipoles - downward pass

Downward pass starting at root-2

- Box inherits from parent via local shift
- Plus local expansion input from 27 newly far boxes with parent-boxes adjacent to own parent

At lowest level

- Evaluate resulting field at each particle
- Add direct particle-particle from particle within own box and 8 adjacent boxes

Fast Multipoles

Errors from first multipole order not included $m_{\max}$, in 2D

$$\text{Error} \leq \left(\frac{1}{2\sqrt{2}}\right)^{m_{\max}+1}$$

Need $m_{\max} = 6$ for 10^{-3} accuracy ($m_{\max} = 8$ in 3D)

Costs in 2D

$$8N + \tfrac{4}{3}(m_{\max}+1)N + 36(m_{\max}+1)^2 N$$

So for 10^{-3} accuracy, need $N > 10^4$ before faster than N^2 direct particle-particle interactions

Costs in 3D

$$26N + m_{\max}^2 N + 189 m_{\max}^4 N$$

So for 10^{-3} accuracy, need $N > 10^6$ before faster than N^2 direct particle-particle interactions