MA5233 Computational Mathematics

Lecture 13: Multigrid

Simon Etter

2019/2020

Multigrid

Multigrid algorithm

- Another algorithm for solving the Poisson equation.
- Key selling point: requires only $\mathcal{O}(N)$ FLOP.
  As before, N denotes the matrix size. $N = n^2$ for 2d Poisson on $n \times n$ grid.
  - LU factorisation: $\mathcal{O}\left(N^{3/2}\right)$ FLOP in 2d.
  - Fourier transform: $\mathcal{O}\left(N \log N\right)$ FLOP.
- Generality: somewhere between LU (solves any linear system) and fast Fourier transform (only solves very specific problem).

Outline

- Jacobi and Gauss-Seidel methods.
- Convergence analysis for Jacobi method.
- From Jacobi to multigrid.

Multigrid

Jacobi iteration for Poisson equation

Assume we have an initial guess $x^{(0)}$ for the linear system

$$(3+1)^2 \begin{pmatrix} 2 & -1 & \\ -1 & 2 & -1 \\ & -1 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}.$$

Solving each equation for the "diagonal" unknown yields

$$\begin{aligned} x_1^{(1)} &= \tfrac{1}{2}\left(\tfrac{b_1}{(3+1)^2} + x_2^{(0)}\right), \\ x_2^{(1)} &= \tfrac{1}{2}\left(\tfrac{b_2}{(3+1)^2} + x_1^{(0)} + x_3^{(0)}\right), \\ x_3^{(1)} &= \tfrac{1}{2}\left(\tfrac{b_3}{(3+1)^2} + x_2^{(0)}\right). \end{aligned}$$

The resulting $x^{(1)}$ is not the exact solution in general, but we may hope that it is a better approximation to x than $x^{(0)}$.

Idea: iterate the map $x^{(0)} \mapsto x^{(1)}$ until convergence.

See `jacobi_step()` in `13_multigrid.jl`.

Multigrid

Gauss-Seidel iteration

Minor modification on Jacobi:

$$
\begin{aligned}
x_1^{(1)} &= \tfrac{1}{2}\big(\tfrac{b_1}{(3+1)^2} + x_2^{(0)}\big),\\
x_2^{(1)} &= \tfrac{1}{2}\big(\tfrac{b_2}{(3+1)^2} + x_1^{(1)} + x_3^{(0)}\big),\\
x_3^{(1)} &= \tfrac{1}{2}\big(\tfrac{b_3}{(3+1)^2} + x_2^{(1)}\big).
\end{aligned}
$$

See `gauss_seidel_step()` in `13_multigrid.jl`

Comparison with Jacobi iteration:

- Good: iteration can be done *in-place*:
  - Jacobi: read from $x^{(0)}$, write to $x^{(1)}$.
  - Gauss-Seidel: read to and write from single vector.
- Good: convergence is faster.
- Bad: loss of parallelism:
  - Jacobi: every entry of $x^{(1)}$ can be computed independently.
  - Gauss-Seidel: $x_k^{(1)}$ must be computed after $x_\ell^{(1)}$ with $\ell < k$.

Multigrid

Discussion of Jacobi-type methods

- Good: performing a single iteration is very fast if matrix is sparse.
- Bad: many iterations are needed to reach a reasonable accuracy.

See `plot_convergence()` in `13_multigrid.jl`.

Next steps

Goal: convergence estimate $\|x^{(k)} - x\| = \mathcal{O}\big(f(k)\big)$.

Intermediate step: matrix formula for Jacobi iteration.

Remarks

- We will discuss only the Jacobi iteration in this lecture.
- Analysis of Gauss-Seidel is analogous but more complicated.

Multigrid

Jacobi iteration, general definition

Let A be an invertible matrix with nonzero diagonal D.

Jacobi iteration is defined as follows.

Algorithm 1 Jacobi iteration

1: **for** $k = 1, 2, \ldots$ **do**

2: $\quad x^{(k)} = D^{-1}\left(b - (A - D)\, x^{(k-1)}\right)$

3: **end for**

The next slide demonstrates that the general definition reduces to the concrete Jacobi iteration when applied to the Poisson matrix.

Multigrid

Example: Jacobi iteration for Poisson matrix

Consider the matrix

$$A = (3+1)^2 \begin{pmatrix} 2 & -1 & \\ -1 & 2 & -1 \\ & -1 & 2 \end{pmatrix}$$

The Jacobi iteration takes the form

$$\begin{aligned} x^{(1)} &= D^{-1}\left(b - (A-D)\,x^{(0)}\right) \\ &= \begin{pmatrix} 2 & & \\ & 2 & \\ & & 2 \end{pmatrix}^{-1} \left(\frac{1}{(3+1)^2} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} - \begin{pmatrix} & -1 & \\ -1 & & -1 \\ & -1 & \end{pmatrix} \begin{pmatrix} x_1^{(0)} \\ x_2^{(0)} \\ x_3^{(0)} \end{pmatrix} \right) \\ &= \begin{pmatrix} \frac{1}{2}\left(\frac{b_1}{(3+1)^2} + x_2^{(0)}\right) \\ \frac{1}{2}\left(\frac{b_2}{(3+1)^2} + x_1^{(0)} + x_3^{(0)}\right) \\ \frac{1}{2}\left(\frac{b_3}{(3+1)^2} + x_2^{(0)}\right) \end{pmatrix}. \end{aligned}$$

This is precisely the iteration we had before.

Multigrid

Towards a convergence estimate for Jacobi iteration

Using the Jacobi iteration formula and $b = Ax$, we obtain

$$\begin{aligned} x^{(k)} - x &= D^{-1}\left(b - (A-D)\,x^{(k-1)}\right) - x \\ &= D^{-1}\left(Ax - Dx - (A-D)\,x^{(k-1)}\right) \\ &= -D^{-1}\,(A-D)\left(x^{(k-1)} - x\right). \end{aligned}$$

Applying this formula repeatedly yields

$$x^{(k)} - x = R^k\left(x^{(0)} - x\right) \qquad \text{where} \qquad R := -D^{-1}\,(A-D).$$

Let us expand initial error in terms of eigenvectors u_ℓ of R,

$$x^{(0)} - x = \sum_{\ell=1}^{N} c_\ell\, u_\ell.$$

Denoting the eigenvalue associated with u_ℓ by λ_ℓ, we obtain

$$x^{(k)} - x = R^k\left(x^{(0)} - x\right) = \sum_{\ell=1}^{N} c_\ell\, R^k\, u_\ell = \sum_{\ell=1}^{N} c_\ell\, \lambda_\ell^k\, u_\ell.$$

Multigrid

Towards a convergence estimate for Jacobi iteration

From previous slide:

$$x^{(k)} - x = \sum_{\ell=1}^{N} c_\ell \, \lambda_\ell^k \, u_\ell.$$

Assume $\|u_\ell\| = 1$ and eigenvalues are sorted such that $|\lambda_1| \leq \ldots \leq |\lambda_N|$. Then,

$$\|x^{(k)} - x\| \leq \sum_{\ell=1}^{N} |c_\ell| \, |\lambda_\ell|^k \leq \left(\sum_{\ell=1}^{N} |c_\ell| \right) |\lambda_N|^k.$$

Conclusion

Jacobi iterates $x^{(k)}$ satisfy

$$\|x^{(k)} - x\| \leq C|\lambda_N|^k$$

where λ_N is the eigenvalue of largest absolute value of

$$R = -D^{-1}(A - D).$$

Multigrid

Convergence rate of Jacobi iteration for Poisson equation

Let us consider

$$A = (N+1)^2 \begin{pmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{pmatrix} \in \mathbb{R}^{N\times N} \quad \Longrightarrow \quad D = (N+1)^2 \begin{pmatrix} 2 & & \\ & \ddots & \\ & & 2 \end{pmatrix}.$$

Let us introduce the following notation:

- $\lambda_\ell = (N+1)^2 \left(2 - 2\cos\left(\pi \frac{\ell}{N+1}\right)\right)$: eigenvalues of A.
- $\hat{\lambda}_\ell$: eigenvalues of $R = -D^{-1}(A - D)$.

Since $D \propto I$, λ_ℓ and $\hat{\lambda}_\ell$ are related by $\hat{\lambda}_\ell = -\frac{1}{2}\left(-\frac{\lambda_\ell}{(N+1)^2} - 2\right)$.

Inserting the known values for λ_ℓ yields $\hat{\lambda}_\ell = \cos\left(\pi \frac{\ell}{N+1}\right)$.

Largest absolute value is achieved for $\ell = 1$ and $\ell = N$ for which we have

$$\lambda_1 = -\lambda_n = \cos\left(\frac{\pi}{N+1}\right) = 1 - \frac{\pi^2}{(N+1)^2} + \mathcal{O}\left(N^{-4}\right).$$

See `13_multigrid.jl`.

Multigrid

Summary of Jacobi convergence theory

Let λ_N be the eigenvalue of largest absolute value of $R = -D^{-1}(A - D)$. Then, there exists a constant $C \neq C(k)$ such that

$$\|x^{(k)} - x\| \leq C\,|\lambda_N|^k.$$

For Poisson matrix

$$A = \begin{pmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{pmatrix} \in \mathbb{R}^{N\times N},$$

we have

$$|\lambda_N| = 1 - \mathcal{O}\big(N^{-2}\big).$$

Multigrid

Why Jacobi iteration for Poisson equation must be slow

Consider

$$A = (N+1)^2 \begin{pmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{pmatrix} \in \mathbb{R}^{N \times N}, \qquad b_i := \begin{cases} (N+1)^2 & \text{if } i = 1, \\ 0 & \text{otherwise.} \end{cases}$$

Solution x is given by $x_i = 1 - \frac{i}{N+1} \neq 0$.

Jacobi iterates $x^{(k)}$ satisfy $x_i^{(k)} = 0$ if $i > k+1$.

Conclusion: $x^{(k)}$ cannot be a good approximation to x for $k < n$.

See `speed_of_propagation()` in `13_multigrid.jl`.

Why Krylov subspace methods for Poisson equation must be slow

- Above argument also applies to unpreconditioned Krylov methods since similarly $(A^k b)_i = 0$ if $i > k+1$.
- A good preconditioner must introduce non-locality, i.e. P^{-1} should be a dense matrix. (ILU satisfies this requirement.)

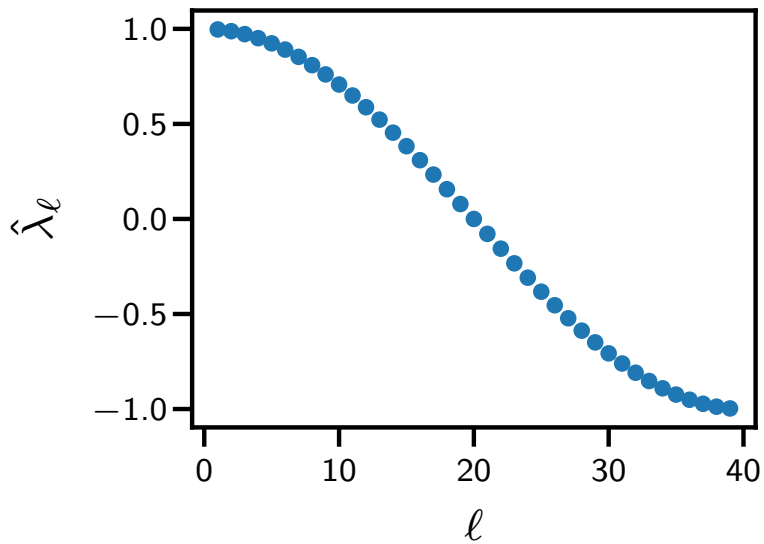
Multigrid

A closer look at Jacobi convergence theory

Recall: every Jacobi iteration multiplies error by R,

$$x^{(k)} - x = R^k \left(x^{(0)} - x\right).$$

Eigenvalues of R for Poisson matrix: $\hat{\lambda}_\ell = \cos\left(\pi \frac{\ell}{N+1}\right)$



Only error components associated with $\ell \approx 1$ and $\ell \approx N$ converge slowly.

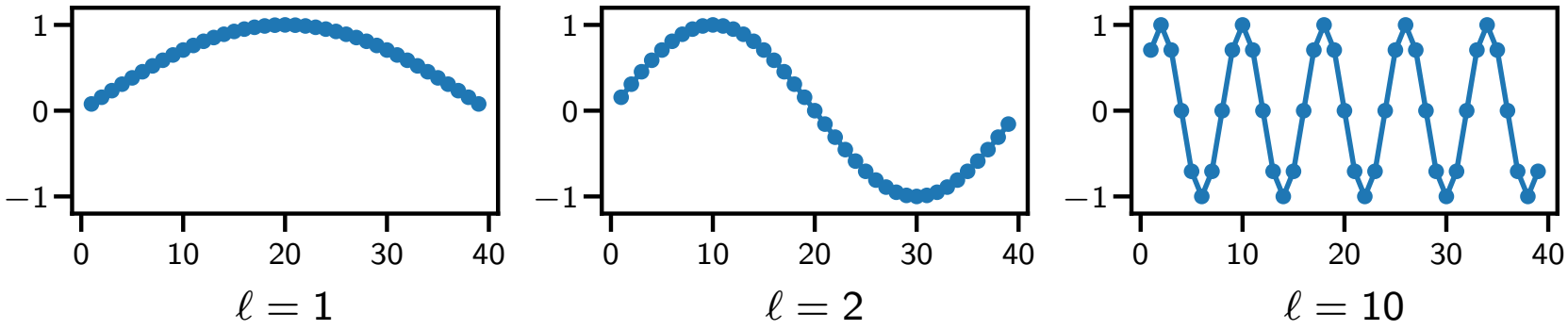
Error components associated with $\ell \approx \frac{N}{2}$ converge very fast.

Let us take a closer look at the eigenfunctions for $\ell \approx 1$.

Multigrid

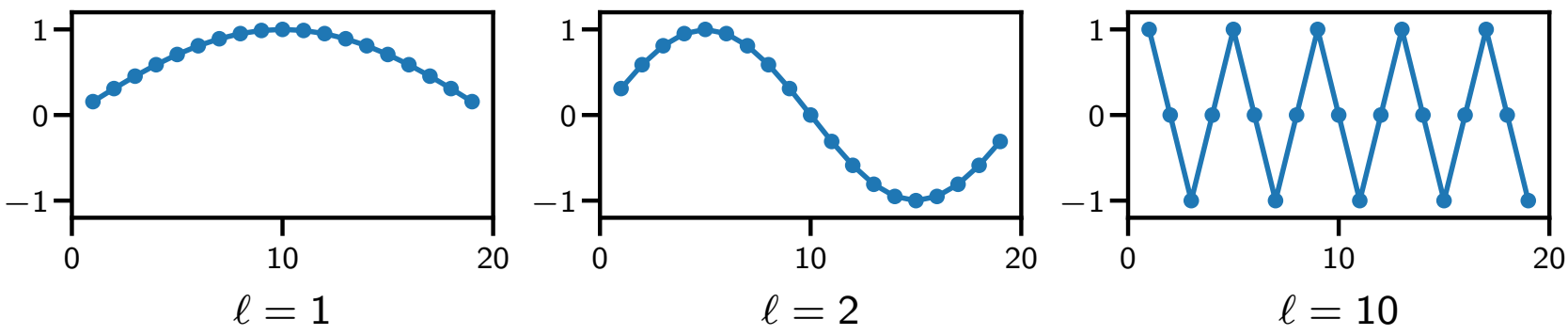
A closer look at Jacobi convergence theory

Eigenvectors of R are given by $(u_\ell)_i = \sin\left(\pi \frac{\ell i}{n+1}\right)$.



Observations:

- ℓ indicates the *frequency* (number of oscillations) of u_ℓ.
- u_ℓ with small ℓ are *smooth*: only few grid points are necessary to capture the rough shape of the curve.



Multigrid

The multigrid idea

1. Use Jacobi to reduce high-frequency errors.
   We have seen that Jacobi is good at reducing intermediate-frequency errors.
   We will see shortly how to use Jacobi to reduce high-frequency errors.
2. Solve for low-frequency errors on coarser grid ("coarse grid correction").
   This switching between grids is why the method is called multigrid.

Benefits of switching to coarser grid

- Obvious: problem becomes smaller and hence more manageable.
- Less obvious: switching between grids allows us to reuse Jacobi smoothing on coarser levels.

Topics for the following slides

- How to use Jacobi to reduce high-frequency errors?
- How to perform coarse grid correction?

Multigrid

Jacobi smoothing for high-frequency errors

Observation in plot on slide 13:

High-frequency errors oscillate: $Ru_\ell \approx (-1)\, u_\ell$ for $\ell \approx N$.

Consequences:

- If we replace Jacobi iteration $x^{(k)} = D^{-1}\left(b - (A-D)\, x^{(k-1)}\right)$ with

$$x^{(k)} = (1-\theta)\, x^{(k-1)} + \theta\, D^{-1}\left(b - (A-D)\, x^{(k-1)}\right)$$

  for some $\theta \in (0, 1]$, then high-frequency errors approximately cancel.

- More precisely, the error recursion formula becomes

$$x^{(k)} - x = \underbrace{\Big((1-\theta)\, I - \theta\, D^{-1}(A-D)\Big)}_{R_\theta} (x^{(k-1)} - x),$$

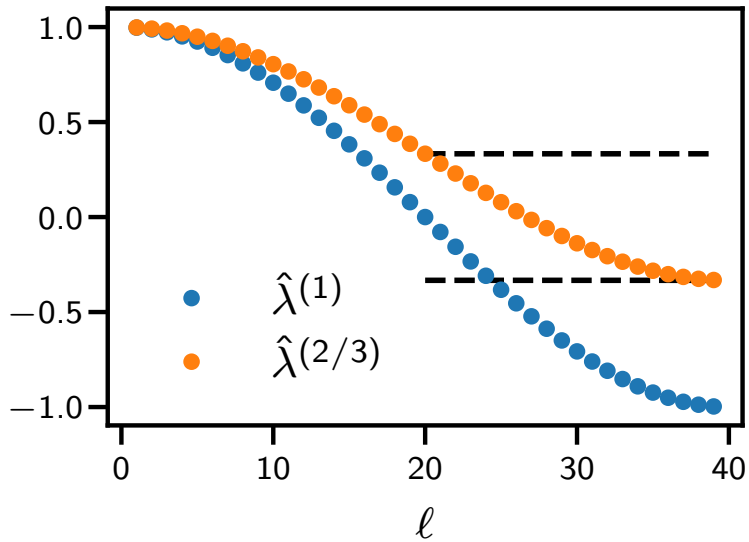
  and the eigenvalues of R_θ are

$$\hat{\lambda}_\ell^{(\theta)} = (1-\theta) + \theta\, \hat{\lambda}_\ell = (1-\theta) + \theta\, \cos\left(\pi \tfrac{\ell}{N+1}\right).$$

Multigrid

Jacobi smoothing for high-frequency errors (continued)

- For $\theta = \frac{2}{3}$, all error components for $\ell \geq \frac{N}{2}$ are multiplied by $\frac{1}{3}$ in every Jacobi iteration.



Multigrid

Jacobi smoothing for high-frequency errors (conclusion)

Relaxed Jacobi iteration

$$x^{(k)} = \frac{1}{3}\, x^{(k-1)} + \frac{2}{3}\, D^{-1}\left(b - (A - D)\, x^{(k-1)}\right)$$

efficiently reduces high-frequency error.

Method is called *relaxed* because it takes only $\frac{2}{3}$ of the step proposed by Jacobi.

After a few steps of relaxed Jacobi, we obtain $\tilde{x}^{(0)}$ such that

$$\tilde{x}^{(0)} - x = \underbrace{\sum_{\ell=1}^{N/2} c_\ell\, u_\ell}_{A} + \underbrace{\sum_{\ell=N/2+1}^{N} c_\ell\, u_\ell}_{B}$$

where B is small and A is smooth.

Next topic: coarse grid correction for eliminating the smooth error A.

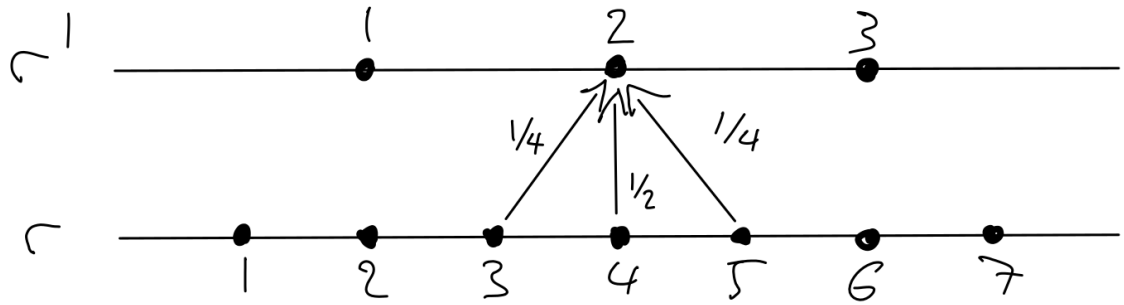
Multigrid

Coarse grid correction

Assume $N = 2N' + 1$.

1. Compute residual $r := b - A\tilde{x}^{(0)}$.
2. Approximate residual on coarser grid:

$$r' \in \mathbb{R}^{N'}, \qquad r'_i = \frac{r_{2i-1}}{4} + \frac{r_{2i}}{2} + \frac{r_{2i+1}}{4}.$$



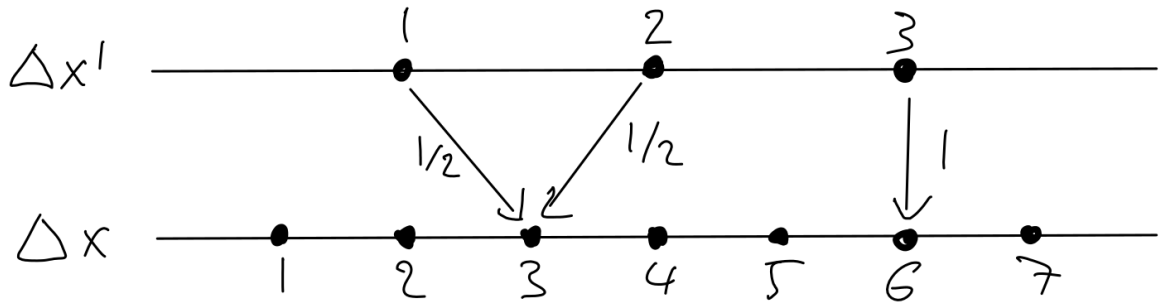
3. Solve on coarse grid: $\Delta x' := A'^{-1} r'$.

Multigrid

Coarse grid correction (continued)

4. Interpolate coarse grid correction $\Delta x'$ to fine grid:

$$\Delta x \in \mathbb{R}^N, \qquad \Delta x_{2i} := \Delta x'_i, \qquad \Delta x_{2i+1} := \frac{\Delta x'_i + \Delta x'_{i+1}}{2}.$$



5. Update $\tilde{x}^{(1)} := \tilde{x}^{(0)} + \Delta x$.

See `twogrid_step()` in `13_multigrid.jl`.

Idea of coarse grid correction

Smoothness of $\tilde{x}^{(0)} - x$ implies smoothness of $r = -A(\tilde{x}^{(0)} - x)$.
This smoothness ensures that coarse problem is "close" to fine problem.
Hence, $\Delta x \approx A^{-1} r = x - \tilde{x}$ and $\tilde{x} + \Delta x \approx x$.

Multigrid

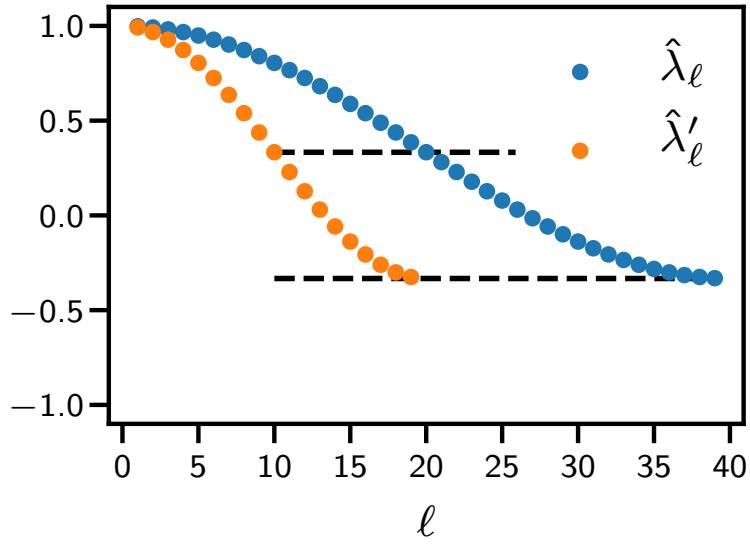
Solving the coarse grid problem

Step 3 of coarse grid correction: $\Delta x' = A'^{-1}\, r'$.

How to solve this linear system?

Key observation

Intermediate frequencies on fine grid $\rightarrow$ high frequencies on coarse grid.



Idea: apply smoothing $\rightarrow$ coarsening $\rightarrow$ smoothing $\rightarrow$ ... recursively.

See `multigrid_step()` in `13_multigrid.jl`.

Multigrid

Multigrid convergence theory

The algorithm implemented in `twogrid_step()` is as follows:

1. Compute $\tilde{x}^{(0)}$ from $x^{(0)}$ using one relaxed Jacobi step.
2. Compute and apply coarse grid correction, $\tilde{x}^{(1)} = \tilde{x}^{(0)} + \Delta x$
3. Compute $x^{(1)}$ from $\tilde{x}^{(1)}$ using one relaxed Jacobi step.

The error recursion formula for relaxed Jacobi from slide 16 yields

$$\tilde{x}^{(0)} - x = R_{2/3}\left(x^{(0)} - x\right) \qquad \text{and} \qquad x^{(1)} - x = R_{2/3}\left(\tilde{x}^{(1)} - x\right).$$

The following slides derive a similar formula for the coarse grid correction.

Multigrid

Multigrid convergence theory (continued)

Observations:

- Approximation step can be written as $r' = \frac{1}{2} P^T r$ where

$$\frac{1}{2} P^T = \begin{pmatrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & & & & & \\ & & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & & & \\ & & & & \frac{1}{4} & \frac{1}{2} & \frac{1}{4} & \\ & & & & & \ddots & \ddots & \ddots \end{pmatrix}.$$

- Interpolation step can be written as $\Delta x = P \, \Delta x'$ where

$$P = \begin{pmatrix} \frac{1}{2} & & & & \\ 1 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ & 1 & & & \\ & \frac{1}{2} & \frac{1}{2} & & \\ & & 1 & \ddots & \\ & & \frac{1}{2} & \ddots & \\ & & & \ddots & \end{pmatrix}.$$

Multigrid

Multigrid convergence theory (continued)

Coarse grid correction takes the form $\tilde{x}^{(1)} = \tilde{x}^{(0)} + \Delta x$ where

$$\Delta x = \tfrac{1}{2}\, P\, A'^{-1}\, P^T\, r = -\tfrac{1}{2}\, P\, A'^{-1}\, P^T\, A\left(\tilde{x}^{(0)} - x\right).$$

Error recursion becomes

$$\begin{aligned}\tilde{x}^{(1)} - x &= \tilde{x}^{(0)} - x + \Delta x \\ &= \left(I - \tfrac{1}{2}\, P\, A'^{-1}\, P^T\, A\right)\left(\tilde{x}^{(0)} - x\right).\end{aligned}$$

Inserting the relaxed Jacobi error recursion yields

$$x^{(1)} - x = R_{2/3}\left(I - \frac{1}{2}\, P\, A'^{-1}\, P^T\, A\right) R_{2/3}\left(x^{(0)} - x\right).$$

Multigrid

Multigrid convergence theory (continued)

Last formula on previous slide implies that convergence rate of `twogrid_step()` is given by eigenvalue of largest absolute value of

$$R_{2g} = R_{2/3}\left(I - \tfrac{1}{2}\, P\, A'^{-1}\, P^T\, A\right) R_{2/3}.$$

Observations:

- We already know the eigenvalues and eigenvectors of $R_{2/3}$, A and A'.
- It can be shown that with $(S_N)_{\ell k} := \sin\left(\pi \frac{\ell k}{N+1}\right)$ we have

$$\left(S_{2N'+1}\, P\, S_{N'}\right)_{\ell,\ell'} \neq 0 \quad \Longleftrightarrow \quad \ell = \ell' \text{ or } \ell = 2N' + 1 - \ell'.$$

- The nonzero entries of $S_{2N'+1}\, P\, S_{N'}$ can be computed explicitly.

Using these results, lengthy calculations will reveal that R_{2g} has only two distinct eigenvalues, namely $\frac{1}{9}$ and 0.

Disclaimer: I did not actually do these calculations, and it is possible that the above result is not 100% accurate. However, the $\frac{1}{9}$ convergence rate is confirmed numerically in `13_multigrid.jl`.

Multigrid

Multigrid convergence theory (final remarks)

Key result from previous slide:

> Convergence rate of multigrid is independent of grid size N!

This is what sets multigrid apart from Jacobi and Krylov type methods.

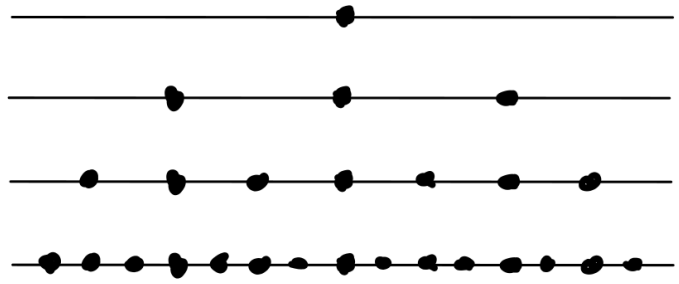
Further remarks:

- Convergence depends on dimension. Convergence is slower in 2d.
- `twogrid_step()` performs two relaxed Jacobi steps and hence reduces the high-frequency error by $\frac{1}{9}$. Heuristically, the result on the previous slide says that the coarse grid correction leads to the same reduction in the low-frequency errors.
- `twogrid_step()` is not a practical algorithm for more complicated problems because we solve the coarse grid problem explicitly. However, it is reasonable to assume and confirmed numerically that `multigrid_step()` performs only slightly worse.

Multigrid

Computational cost of multigrid

- Multigrid employs a stack of grids of sizes $N_\ell := 2^\ell - 1$.



- For each grid point in any grid in the multigrid stack of grids, we perform $\mathcal{O}(1)$ operations.
- Total number of grids points (on any level):

$$\sum_{\ell=1}^{L}(2^\ell - 1) = \frac{2^{L+1} - 1}{2 - 1} - L = \mathcal{O}(2^L) = \mathcal{O}(N_L).$$

Conclusion:

> FLOP count of multigrid is proportional to number of grid points on finest level!

Multigrid

Summary of multigrid method

Each iteration requires $\mathcal{O}(N)$ FLOP and reduces error by factor ρ independent of N.

Why multigrid for Poisson equation can be fast

Recall finite speed of propagation problem for Jacobi-type and Krylov subspace methods.
The coarse grid correction step eliminates this problem for multigrid algorithms: Jacobi iteration on coarse grid propagates information much faster than Jacobi step on fine grid.

Geometric and algebraic multigrid

The algorithm presented here is known as *geometric* multigrid since it exploits the geometric interpretation of the linear system that we try to solve.

There is a generalisation called *algebraic* multigrid which tries to form the stack of grids by only looking at the sparsity pattern of A. Generally, algebraic multigrid is easier to use (you only have to provide A and no further information) but performance is worse.

Multigrid

References and further reading

Most of this lecture is based on Chapter 6 of the following book:

J. W. Demmel. *Applied Numerical Linear Algebra.* Society for Industrial and Applied Mathematics (1997), doi:10.1137/1.9781611971446