Méthodes itératives

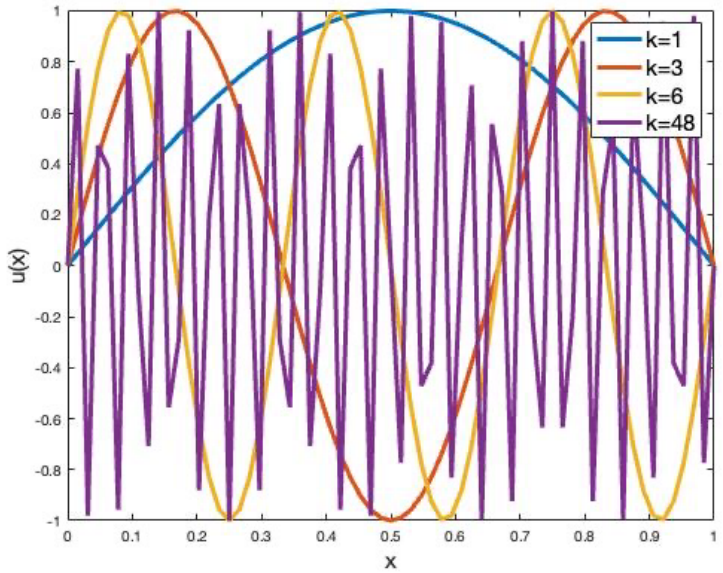
Carola Kruse

Cours 2, 25/03/2024
Multigrid methods

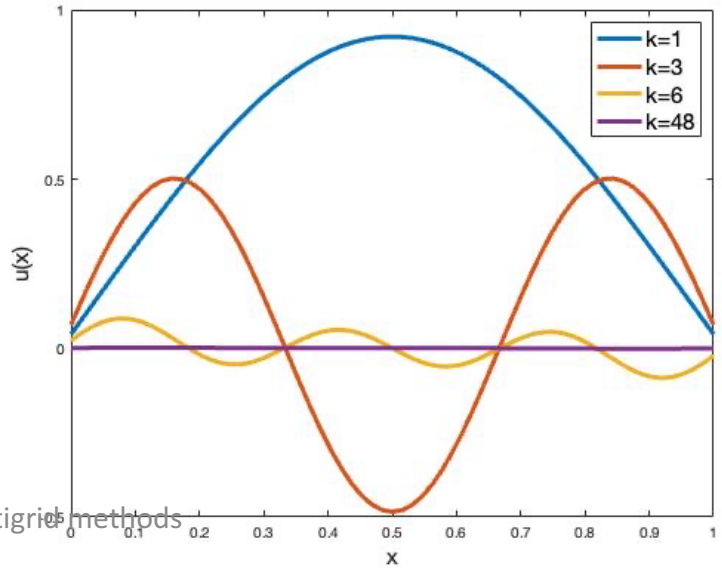
Last week

- Classical iterative schemes might damp highly oscillating discrete error modes very quickly.
- There is only a slow damping of the smooth discrete error modes.
- The smoothness of an error mode depends on the grid. A smooth error mode on a given grid is generally on a coarser grid less smooth.

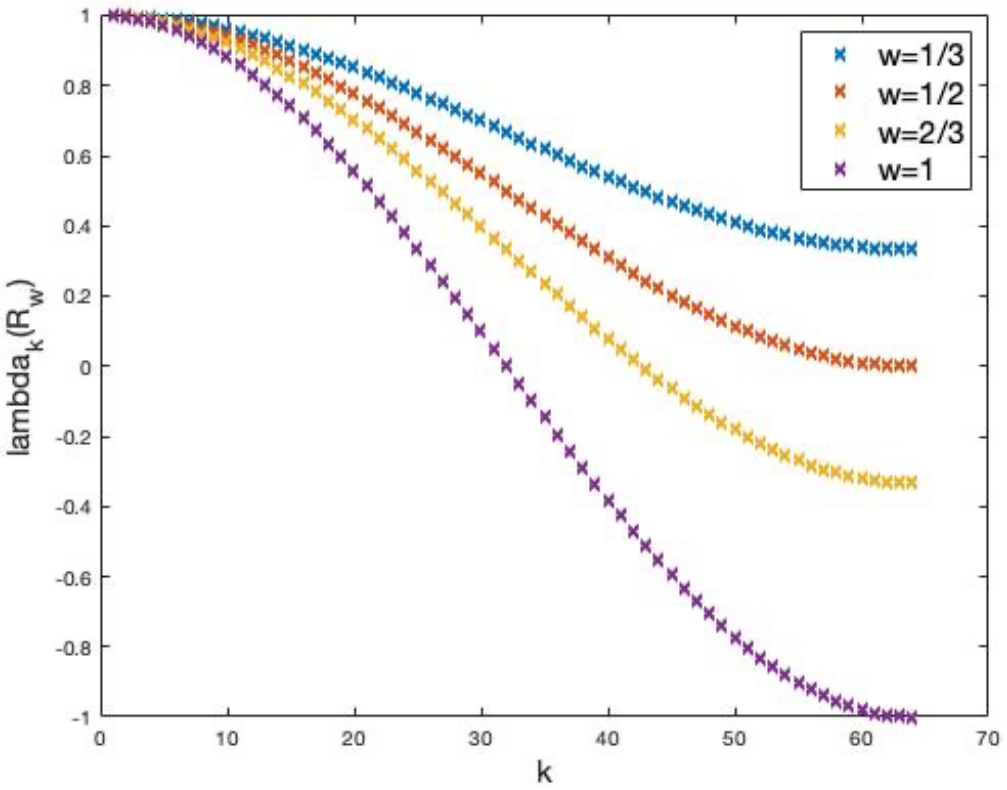
Initial error



Error after 100 iterations



Weighted Jacobi damping

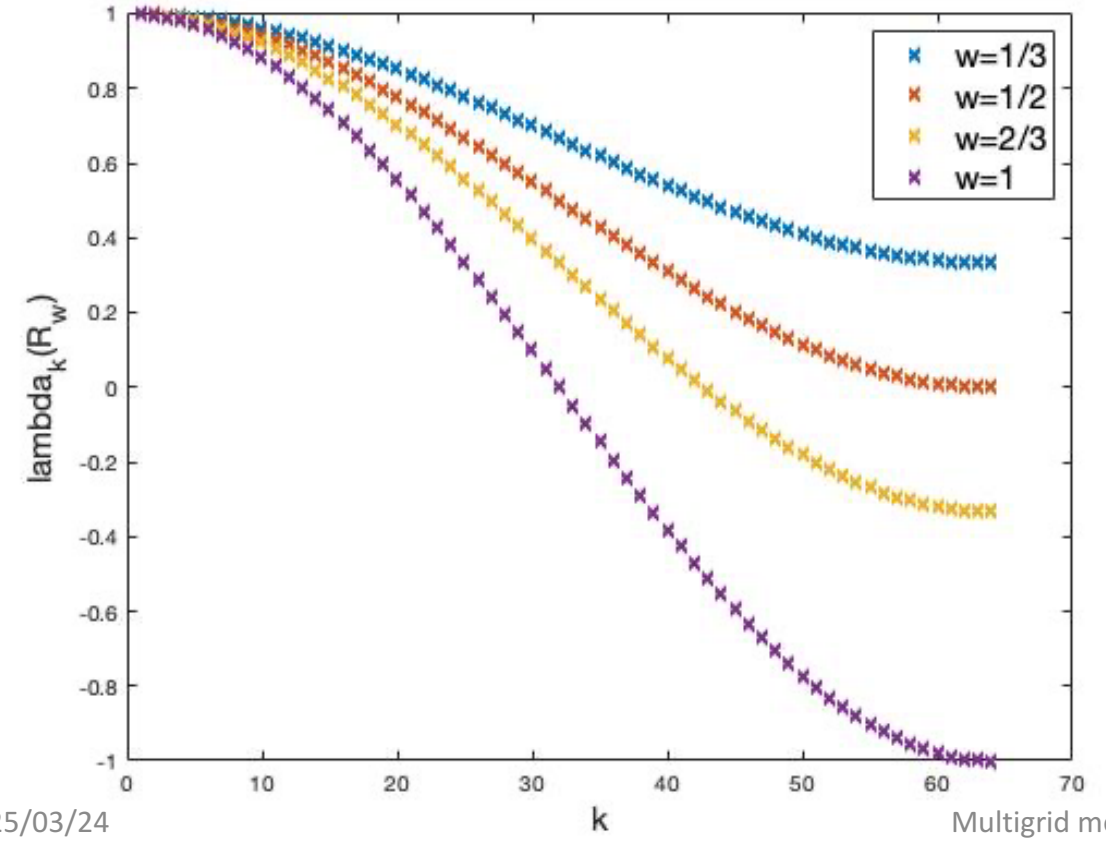


Case $\omega = 1$

- We obtain the non-weighted Jacobi method.
- Here, we see that for small *k*, but also for larger *k* close to *N-1* in the oscillatory spectrum, the eigenvalues are close to 1.
- Convergence is thus also slow in these cases.
- This illustrates why the weighted Jacobi method is advantageous.

Choice of ω

- Recall: for $0 < \omega \leq 1$, we have $|\lambda_k(R_\omega)| < 1$.
- Find the value ω that makes $|\lambda_k(R_\omega)|$ as small as possible for all k



For all values of ω, the eigenvalues associated to the 'smoothest' modes are close to 1.

We have

$$\lambda_1 = 1 - 2\omega \, sin^2\left(\frac{\pi h}{2}\right) \approx 1 - \frac{\omega \, \pi^2 h^2}{2}$$

Thus λ_1 will always be close to one, no matter which ω. It's getting even worse, the smaller h!

Optimal choice of ω

To find the optimal value of ω, we search for the smallest interval $\left[-\bar{\lambda}, \bar{\lambda}\right]$ with $\lambda_k(R_\omega) \in \left[-\bar{\lambda}, \bar{\lambda}\right]$ for $\frac{N}{2} \leq k \leq N-1$. This can be done for example by

$$-\lambda_{N/2}(R_\omega) = \lambda_{N-1}(R_\omega)$$

and we obtain

$$\omega = \frac{2}{3}$$

Definition: The multigrid smoothing factor

- The smoothing factor of a relaxation method R is the maximum magnitude of the upper half of the spectrum

$$\max_{k \in [\frac{N}{2}, N]} |\lambda_k(R_\omega)|$$

- In the example above, we have

$$\max_{k \geq N/2} |\lambda_k(R_{2/3})| = \max_{k \geq \frac{N}{2}} \left|1 - \frac{4}{3} sin^2\left(\frac{k\pi}{2N}\right)\right| \leq \frac{1}{3}$$

$\Longrightarrow$ The oscillatory components are reduced at least **by a factor of 3** at each relaxation.

- We thus see furthermore, that the bound is independent of the mesh size $h = \frac{1}{N}$.

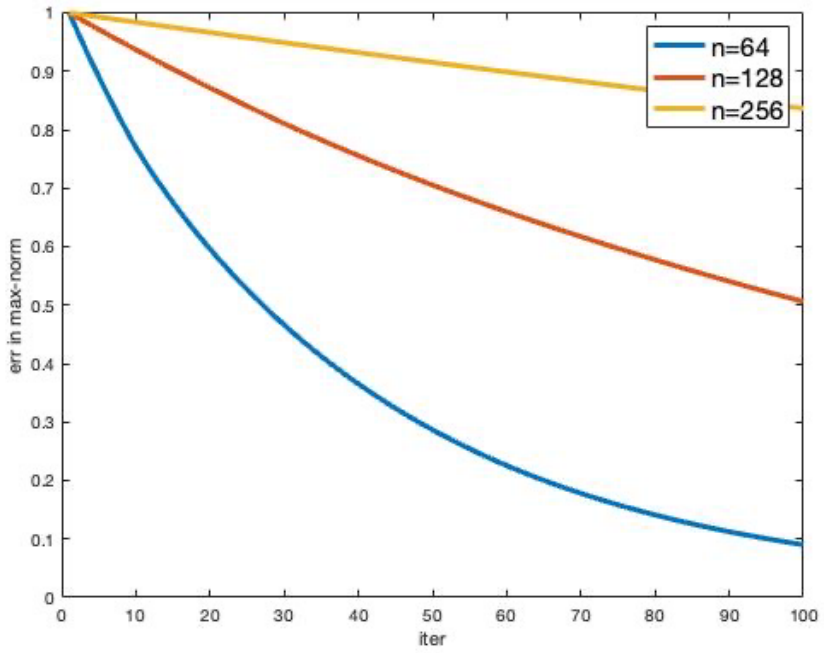
A common feature:

Oscillatory modes are quick to converge

Smooth modes are slow to converge

Fixed wave number on different grids

- Let us still see what happens for the fixed wave number *k=6* on different grids.
- We use weighted Jacobi relaxation with $\omega = \frac{2}{3}$ and show the error for the first 100 iterations



Observation: For a fixed wave number, the error is reduced better on a coarser than on a finer grid.

Summary: Classification of modes

- Consider a fixed Fourier mode $\sin(k\pi x)$ and its discrete representation $\sin\left(\frac{jk\pi}{N}\right), j = 1, .., N-1.$

Then the classification of this mode depends on the fineness of the grid.

- If the grid is sufficiently fine, i.e. $k < \frac{N}{2}$, it is a smooth mode.
- If the grid is sufficiently coarse, i.e. $\frac{\mathrm{N}}{2} \leq k < N-1$, then it is an oscillatory mode.

The damping property depends on the smoothness of the mode.

- If it is a smooth mode, then the weigthed Jacobi method will damp it only slowly.
- If it is an oscillatory mode, then the weigthed Jacobi method may damp it quickly.

Relaxation schemes

Gauss-Seidel method

- Calculate an entry u_k , with $u = (u_1, .., u_n)$, of the new iteration and use it in the computation of $u_i, i = k + 1, .., n$.

$$\boldsymbol{u}^{(m+1)} = (D - L)^{-1} U \, \boldsymbol{u}^{(m)} + (D - L)^{-1} \boldsymbol{f}$$

Define
$S_{GS} := (D - L)^{-1} U$

SOR method

$$\boldsymbol{u}^{(m+1)} = -(\mathrm{D} - \omega L)^{-1}(-\omega U + (\omega - 1)D)\boldsymbol{u}^{(m)} + \omega(\mathrm{D} - \omega L)^{-1}\mathbf{f}$$

$S_{GS\omega} := (D - \omega L)^{-1}(-\omega U + (\omega - 1)D)$

- If $\omega = 1$, then the Gauss-Seidel method is recovered.

Gauss-Seidel method

- The eigenvalues of $S_{GS} = (D - L)^{-1}U$ are given by

$$\lambda_k(S_{GS}) = cos^2\left(\frac{k\,\pi}{N}\right)$$

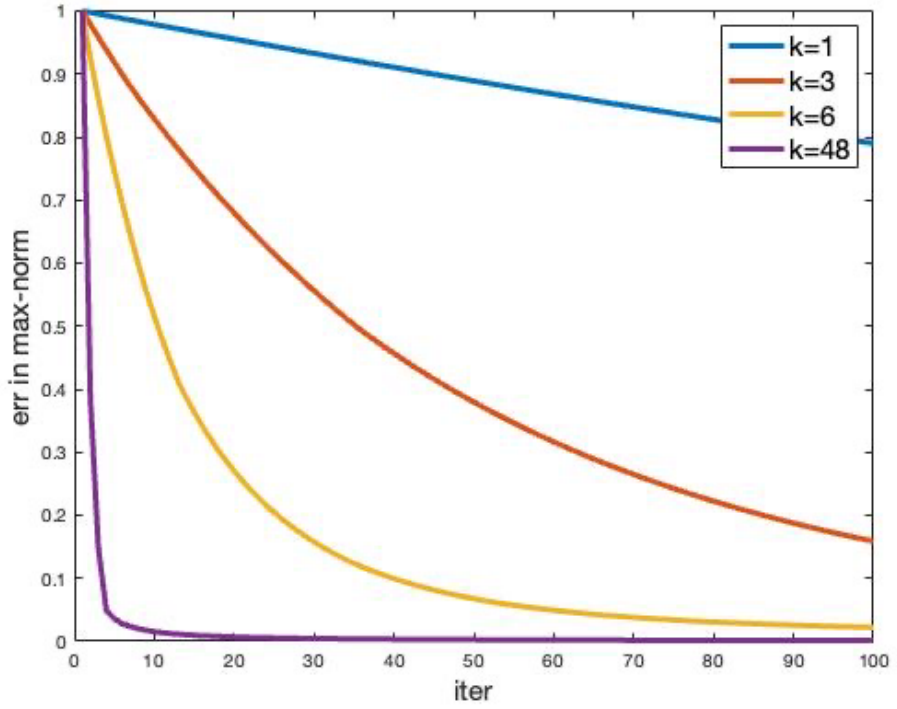
- The eigenvectors of S_{GS} are given by

$$w_{k,j}(S_{GS}) = \left[cos\,\left(\frac{k\,\pi}{N}\right)\right]^2 \sin\,\left(\frac{jk\,\pi}{N}\right)$$

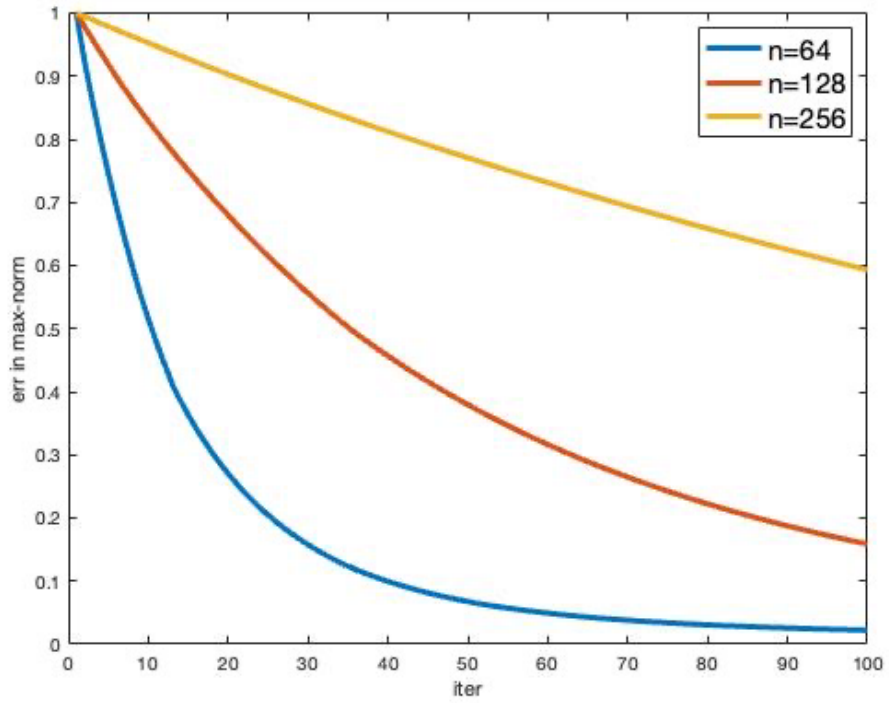
- These are not the same as for *A*. The convergence analysis has thus to be done carefully. We do not want to get into further details here.

Gauss-Seidel method

- With the eigenvectors of *A* as initial guess, the damping behaviour of the Gauss-Seidel method is qualitatively the same as for the Jacobi method.



Fixed number of nodes *n=64*



Fixed wave number *k*=6

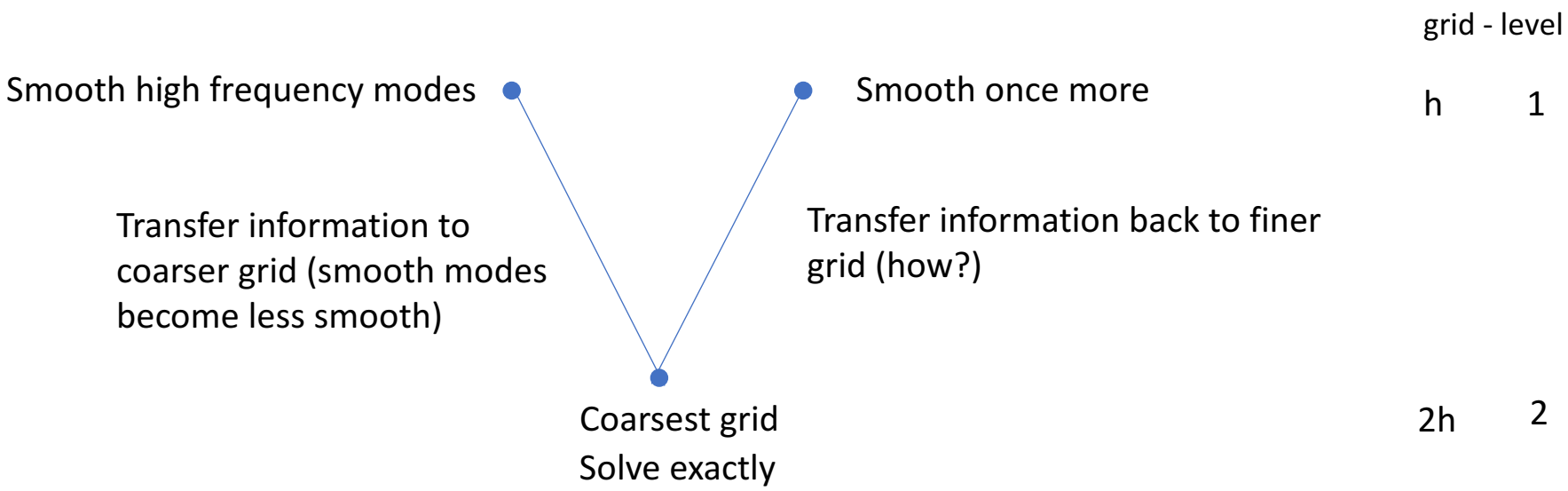
Summary: Relaxation schemes and damping

- Classical iterative schemes might damp highly oscillating discrete error modes very quickly.
- There is only a slow damping of the smooth discrete error modes.
- The smoothness of an error mode depends on the grid. A smooth error mode on a given grid is generally on a coarser grid less smooth.

Improvements possible for all error components?

- We have seen that with a good initial guess, the relaxation scheme converges faster (e.g. $k = 48$ against $k = 1$). Some iterations on a coarse grid could help.
- Could we use this coarse grid idea otherwise?

A two-grid scheme?



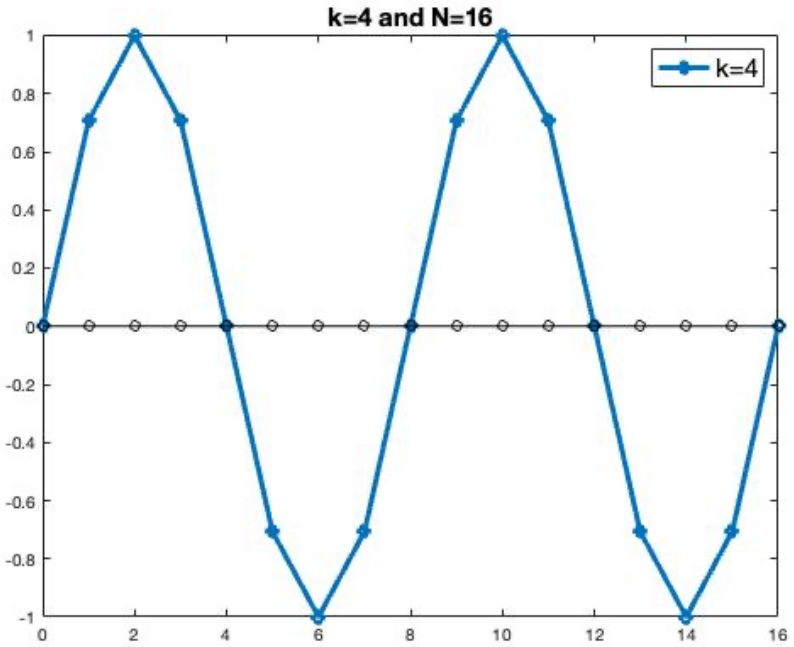
Elements of multigrid

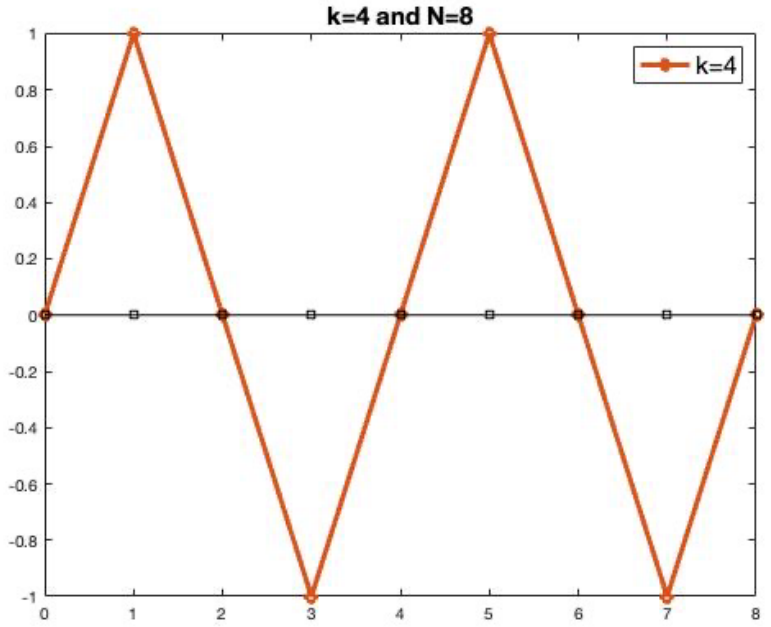
Grid transfer

Coarse grids

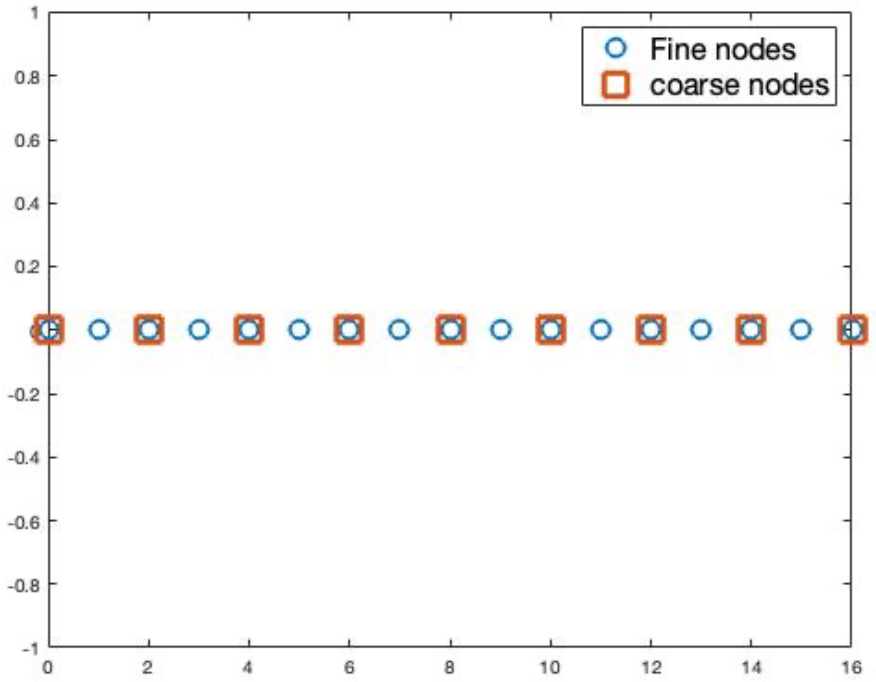
- Imagine all oscillatory modes are **damped out** and we are left only with **smooth modes**.
- Smooth modes look like **oscillatory modes** when sampled on a **coarse grid**.

4-mode of 15 versus 4-mode of 7





Coarse modes – Low frequency



We look at the *k*-th mode on the fine grid evaluated at the even-numbered grid points.

If $1 \leq k \leq \frac{N}{2}$, i.e. **lower part** of **spectrum**, then the components may be rewritten as

$$w^h_{k,2j} = \sin\left(\frac{2jk\pi}{N}\right) = \sin\left(\frac{jk\pi}{\frac{N}{2}}\right) = w^{2h}_{k,j}, \qquad 1 \leq j \leq \frac{N}{2}$$

- The *k*-th mode on Ω^h becomes the *k*-th mode on Ω^{2h}
- Passing from fine to coarse grid, a smooth mode becomes more oscillatory

Coarse modes – High frequency

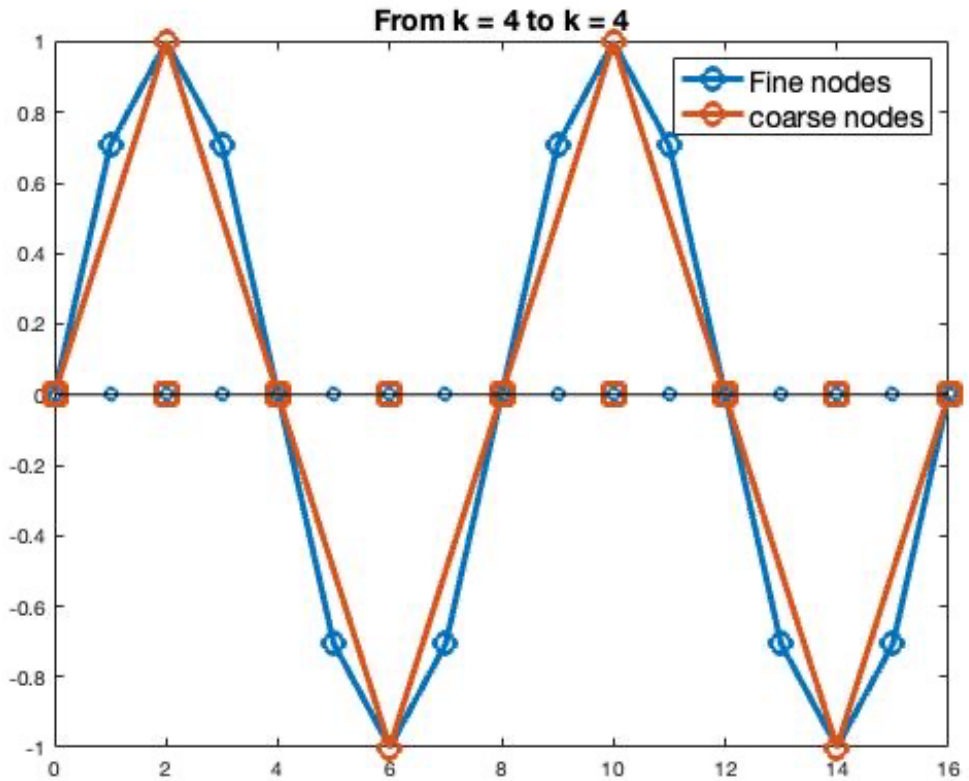
For the **upper part** of the **spectrum**, i.e. $\frac{N}{2} < k \leq N-1$, compute:

$$
\begin{aligned}
-w^{2h}_{N-k,j} &= -\sin\left(\frac{2j(N-k)\pi}{N}\right) \\
&= -\sin\left(\frac{2jN\pi}{N} - \frac{2jk\pi}{N}\right) \\
&= -\sin(2j\pi)\cos\left(\frac{2jk\pi}{N}\right) + \cos(2j\pi)\sin\left(\frac{2jk\pi}{N}\right) \\
&= \sin\left(\frac{2jk\pi}{N}\right) = w^{h}_{k,2j}
\end{aligned}
$$

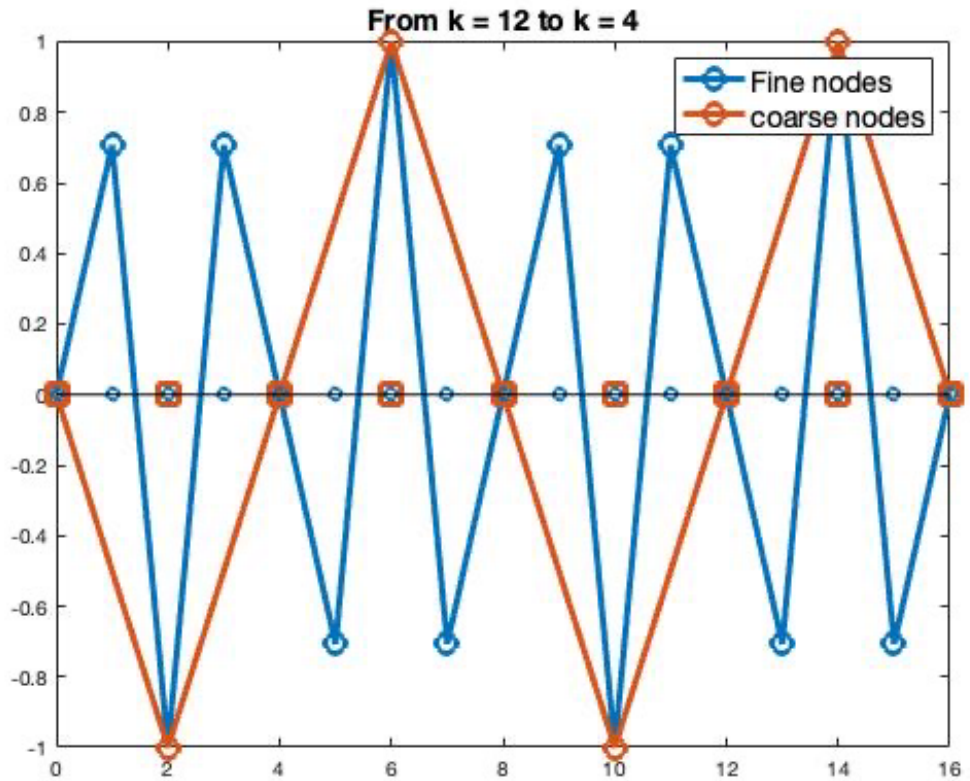
- The *k*-th mode on Ω^h becomes the negative of the *(N-k)*-th mode on Ω^{2h}.
- The oscillatory modes on the fine grid appear relatively smooth on the coarse grid.

- It is advisable to **damp out** the oscillatory modes **before** passing to the coarser grid. Otherwise we obtain additional smooth modes on the coarser grid.

Coarse modes



Low modes -> k-modes are *preserved*



High modes -> k-modes are *aliased*

Some key observations so far

1. Relaxation may be extremely efficient for smoothing the error relative to the grid.

2. A smooth error can be approximated well on a coarser grid.

3. A coarser grid implies less variables, hence less computation.

4. On the coarser grid, the error is no longer as smooth relative to the grid, so relaxation may once again be efficient.

First strategy: Nested iterations

- Use coarse grids to obtain an initial guess for the next finer grid.

- Relax on $A\mathbf{u}^{h_0} = \mathbf{f}^{h_0}$ on a very coarse grid to obtain an initial guess for the next finer grid

$\vdots$

- Relax on $A\mathbf{u}^{4h} = \mathbf{f}^{4h}$ on Ω^{4h} to obtain an initial guess for Ω^{2h}
- Relax on $A\mathbf{u}^{2h} = \mathbf{f}^{2h}$ on Ω^{2h} to obtain an initial guess for Ω^{h}
- Relax on $A\mathbf{u}^{h} = \mathbf{f}^{h}$ on Ω^{h} to obtain a final approximation to the solution.

Questions:

- What does it mean to relax on the coarser grid (i.e., how to define the equations to be solved thereon?)
- What if some smooth components remain? –> The algorithm will stall on the fine grid.

The residual equation

- An iterative method for $A\boldsymbol{u} = \boldsymbol{f}$ can either be applied to this equation directly, or to an equation formulated for the error. The next iterate will then be corrected by the defect

$$\mathbf{v}^{(m+1)} = \mathbf{v}^{(m)} + \mathbf{e}^{(m)}$$

- Let $\boldsymbol{u}^{(m)}$ be the approximation of $\boldsymbol{u}$ at the *m*-th iteration. The error is then given by $\mathbf{e}^{(m)} = \mathbf{u} - \mathbf{u}^{(m)}$.
- The error satisfies the equation

$$A\mathbf{e}^{(m)} = A\mathbf{u} - A\mathbf{u}^{(m)} = \mathbf{f} - A\mathbf{u}^{(m)} =: \mathbf{r}^{(m)}.$$

- This equation is called the residual equation.

Second strategy: Coarse grid correction (Two-level method)

- We now use the residual equation and relax on the actual error.
- The new iterate is then an update of the previous iterate corrected with the new residual

Smooth $A^h \boldsymbol{u}^h = \boldsymbol{f}^h$ on Ω^h, call solution $\boldsymbol{v}^h$. → We obtain an approximation of the solution, that is to be updated.

Compute the residual $\boldsymbol{r}^h = \boldsymbol{f}^h - A^h \boldsymbol{v}^h$.

Project (restrict) the residual to Ω^{2h}, called $R(\boldsymbol{r}^h)$. → Information is transferred to the coarse grid.

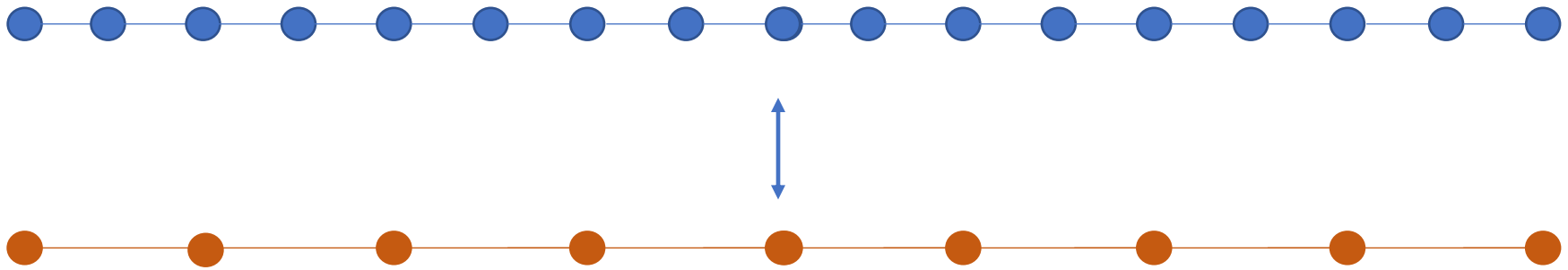
Solve $A^{2h} \boldsymbol{e}^{2h} = R(\boldsymbol{r}^h)$ on Ω^{2h} → We obtain an approximation $\boldsymbol{e}^{2h}$ of the error

Project (prolongate) $\boldsymbol{e}^{2h}$ to Ω^h, denoted $P(\boldsymbol{e}^{2h})$. → Information is transferred back to fine grid

Update the approximate solution on Ω^h by $\boldsymbol{v}^h = \boldsymbol{v}^h + P(\boldsymbol{e}^{2h})$

- How to define the system on the coarse grid?
- How to restrict and prolongate from one grid to the other?

Questions



- How to transfer between fine and coarse grid?
- Which equations do we want to solve on the coarse grid?
- We use superscripts *h* on the variable, indicating the grid they are defined on, e.g., $\boldsymbol{u}^h$ is defined on Ω^h and $\boldsymbol{u}^{2h}$ is defined on Ω^{2h}.

Grid transfer

As we have seen, we need to smooth the error on the fine grid first and only then solve the coarse-grid problem.

Hence, we need two types of intergrid transfer operators:

- A restriction operator (fine-to-coarse): I_h^{2h}
- A prolongation operator (coarse-to-fine): I_{2h}^h

Let us start with the prolongation operator first.

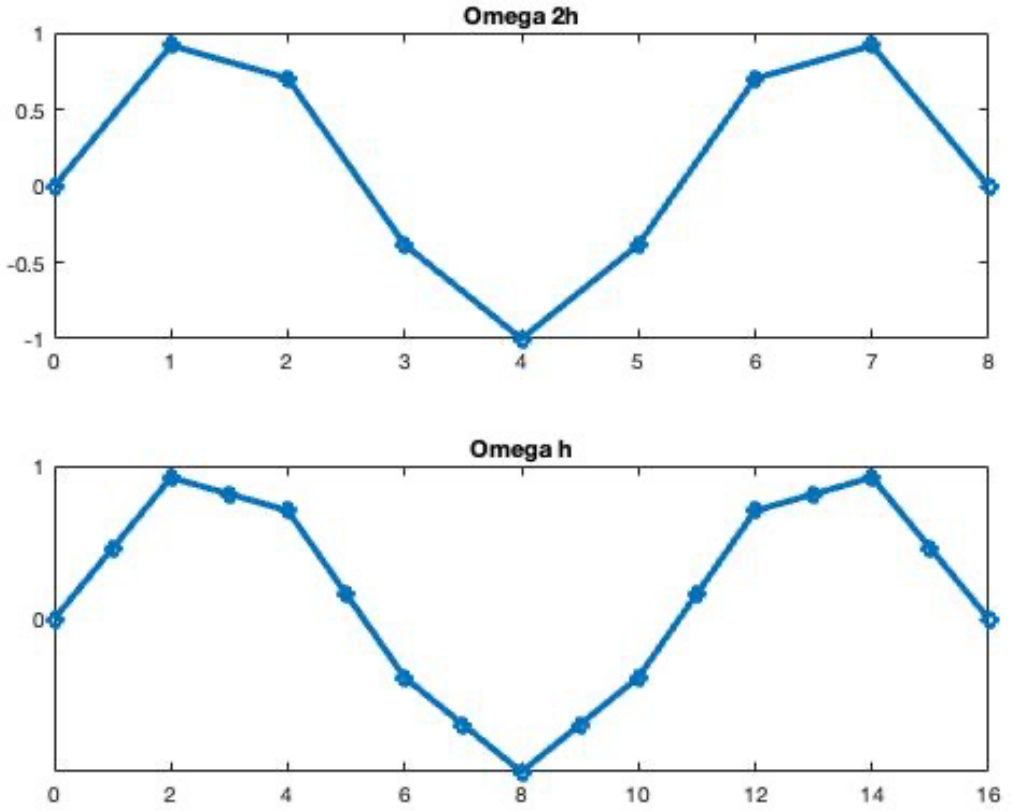
Prolongation: Linear interpolation

- Find a linear interpolation operator $I_{2h}^{h} : \mathbb{R}^{\frac{N}{2}-1} \rightarrow \mathbb{R}^{N-1}$.
- I_{2h}^{h}: coarse grid -> fine grid, with

$$v_{2j}^{h} = v_{j}^{2h},$$
$$v_{2j+1}^{h} = \frac{1}{2}(v_{j}^{2h} + v_{j+1}^{2h})$$

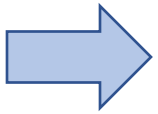
- It can be written in matrix form (here for the case *N=8*):

$$I_{2h}^{h} v^{2h} = \frac{1}{2} \begin{bmatrix} 1 & & \\ 2 & & \\ 1 & 1 & \\ & 2 & \\ & 1 & 1 \\ & & 2 \\ & & 1 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}_{2h} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \end{bmatrix}_{h} = v^h$$



How well does interpolation work?

- Assume that the real error is smooth on the fine grid.
- Assume also that a coarse-grid approximation is given on Ω^{2h} and that it is exact on the coarse nodes.
- When this coarse grid correction is interpolated to the fine grid, the interpolant is also smooth.

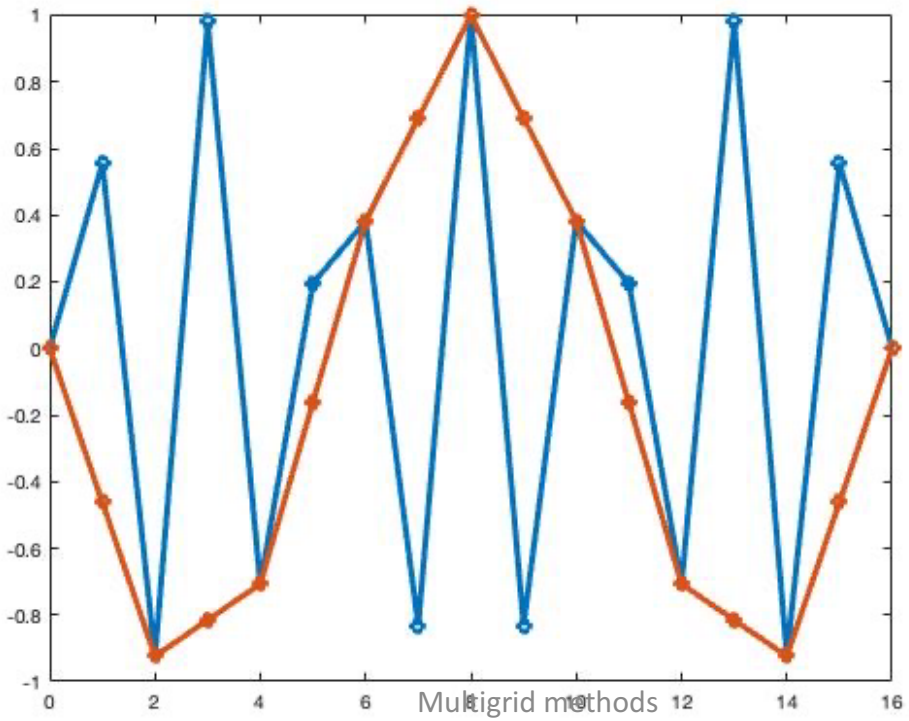


We expect a relatively good approximation to the fine grid error.

Oscillatory real error

- Assume that the real error is oscillatory

Even a very good coarse-grid approximation may produce an interpolant that is not very accurate.



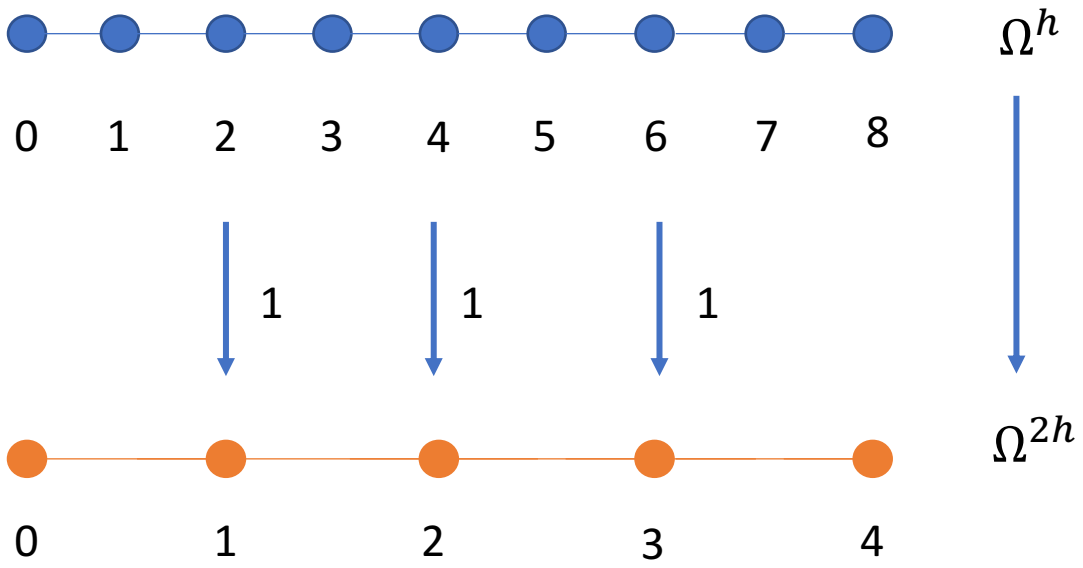
Summary prolongation

- All in all, interpolation gives the best result, if the error on the fine grid is smooth.
- The prolongation is an appropriate complement to the smoother, that works most efficiently if the error is oscillating.

Restriction operator - Injection

- Want to find a good restriction operator $I_h^{2h} : \mathbb{R}^{N-1} \to \mathbb{R}^{\frac{N}{2}-1}$.
- The most obvious restriction operator I_h^{2h} is injection, where

$$v_j^{2h} = v_{2j}^h$$

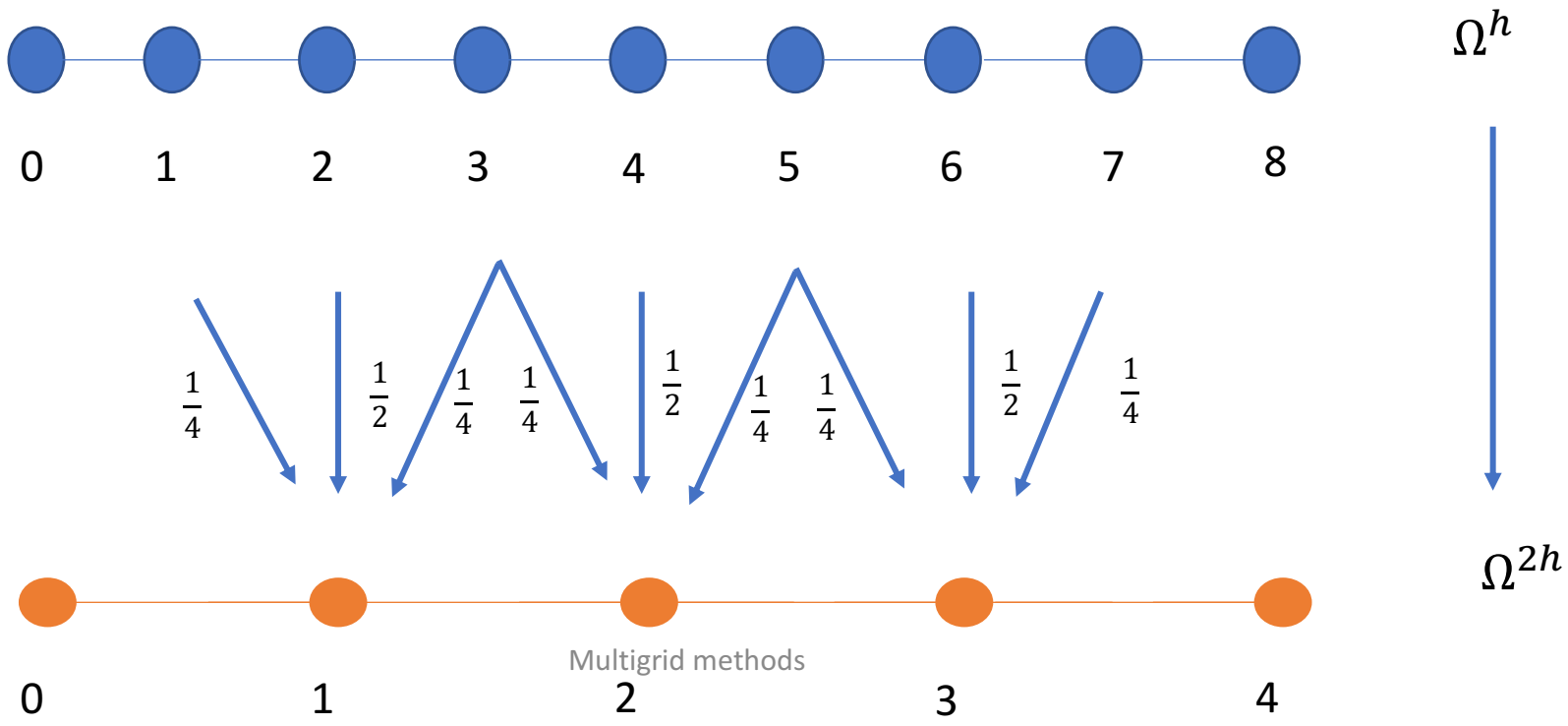


- However, the injection operator often does not lead to an efficient method.

Restriction operator – full weighting

- The full weighting restriction operator $I_h^{2h} : \mathbb{R}^{N-1} \to \mathbb{R}^{\frac{N}{2}-1}$. takes all grid points into account.

$$v_j^{2h} = \frac{1}{4}(v_{2j-1}^h + 2v_{2j}^h + v_{2j+1}^h)$$



Full weighting – matrix representation

For the case *N = 8,* we have

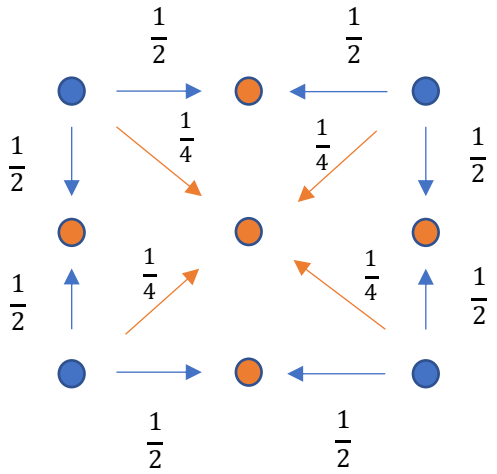
$$I_h^{2h} v^h = \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 & & & & \\ & & 1 & 2 & 1 & & \\ & & & & 1 & 2 & 1 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \end{bmatrix}_h = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}_{2h} = v^{2h}$$

Note that $I_{2h}^h = c(I_h^{2h})^T$ for $c \in \mathbb{R}$, i.e.

- the full weighting operator is the transpose of the linear interpolation operator
- It is a linear operator and has a rank of $\frac{N}{2} - 1$ and a null space of dimension $\frac{N}{2}$

Prolongation operator for 2D problem

- The interpolation operator may be defined in a similar way.
- Let $I_{2h}^{h}\boldsymbol{v}^{2h} = \boldsymbol{v}^{h}$. The components of $\boldsymbol{v}^{h}$ are given for $0 \leq i, j \leq \frac{N}{2} - 1$ by



$$v_{2i,2j}^{h} = v_{ij}^{2h}$$

$$v_{2i+1,2j}^{h} = \frac{1}{2}(v_{ij}^{2h} + v_{i+1,j}^{2h})$$

$$v_{2i,2j+1}^{h} = \frac{1}{2}(v_{ij}^{2h} + v_{i,j+1}^{2h})$$

$$v_{2i+1,2j+1}^{h} = \frac{1}{4}(v_{ij}^{2h} + v_{i+1,j}^{2h} + v_{i,j+1}^{2h} + v_{i+1,j+1}^{2h})$$

Full weighting operator in 2D

- Let $I_h^{2h}\mathbf{v}^h = \mathbf{v}^{2h}$. The components of $\boldsymbol{v^h}$ are given for $1 \leq i, j, \leq \frac{N}{2} - 1$ by

$$v_{i,j}^{2h} = \frac{1}{16}\left[v_{2i-1,2j-1}^h + v_{2i-1,2j+1}^h + v_{2i+1,2j-1}^h + v_{2i+1,2j+1}^h \right.$$
$$\left. + 2(v_{2i,2j-1}^h + v_{2i,2j+1}^h + v_{2i-1,2j}^h + v_{2i+1,2j}^h) + 4v_{i,j}^h\right]$$

The two level method

- ✅ Smooth $A^h \boldsymbol{u}^h = \boldsymbol{f}^h$ on Ω^h.
- ✅ Compute the residual $\boldsymbol{r}^h = \boldsymbol{f}^h - A^h \boldsymbol{u}^h$.
- ✅ Project (restrict) the residual to Ω^{2h}, called $R(\boldsymbol{r}^h)$.
- Solve $A^{2h} \boldsymbol{e}^{2h} = R(\boldsymbol{r}^h)$ on Ω^{2h}
- ✅ Project (prolongate) $\boldsymbol{e}^{2h}$ to Ω^h, called $P(\boldsymbol{e}^{2h})$
- ✅ Update the approximation of the solution on Ω^h by $\mathbf{v}^h = \boldsymbol{v}^h + P(\boldsymbol{e}^{2h})$

How to construct the coarse grid matrix A^{2h} ?

Coarse grid matrix

Straightforward approach:

- Define A^{2h} by applying a discretization method for the differential operator on Ω^{2h}.

Second approach:

- Use the intergrid transfer operators to define the coarse grid matrix A^{2h}.

Galerkin projection

- Here we use the prolongation and restriction operators to define the coarse grid matrix.
- Starting point is the residual equation

$$A^h \mathbf{e}^h = \mathbf{r}^h$$

- Let $\boldsymbol{e}^h$ be in the range of the prolongation operator I_{2h}^h, i.e.

$$\mathbf{e}^h = I_{2h}^h(\mathbf{e}^{2h})$$

- We substitute the latter equation into the first equation and obtain

$$A^h I_{2h}^h(\mathbf{e}^{2h}) = \mathbf{r}^h$$

Galerkin projection

Applying on both sides the restriction operator, we obtain

$$I_h^{2h} A^h I_{2h}^h(\mathbf{e}^{2h}) = I_h^{2h}(\mathbf{r}^h)$$

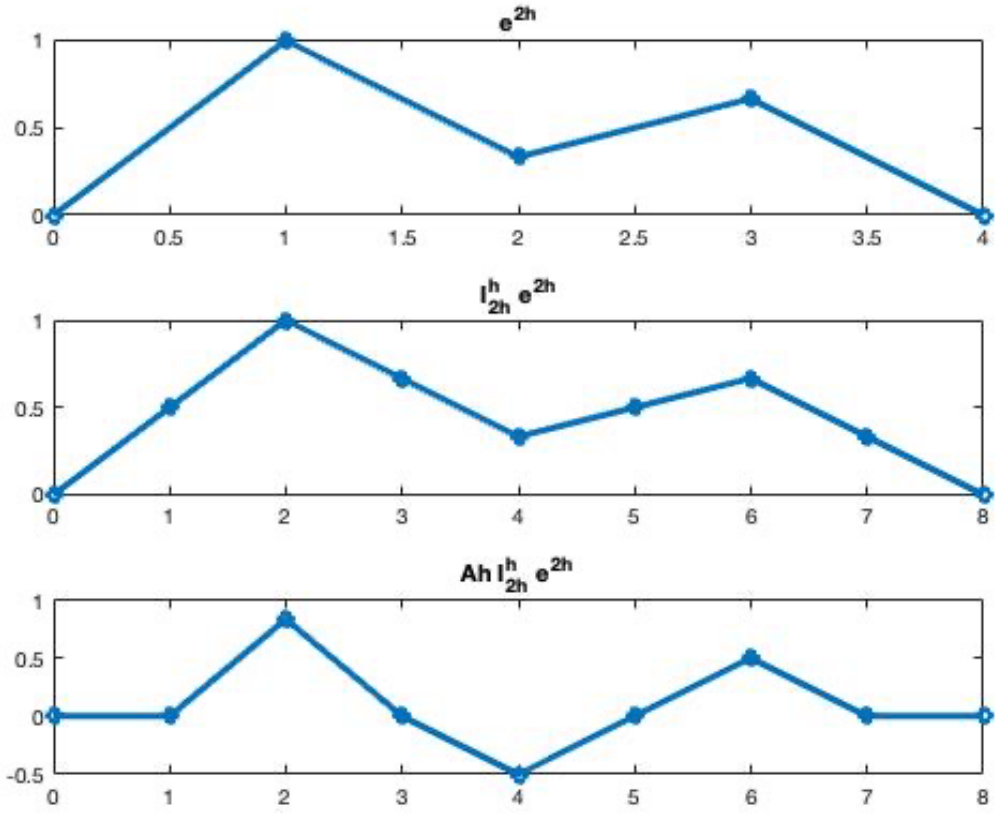
Remember that the coarse grid problem is given as

$$A^{2h}\mathbf{e}^{2h} = I_h^{2h}(\mathbf{r}^h)$$

We can thus conclude that

$$A^{2h} := I_h^{2h} A^h I_{2h}^h$$

Galerkin operator graphically – Poisson 1D



$$A = \frac{1}{h^2}\begin{pmatrix} 2 & -1 & 0 & \dots & 0 \\ -1 & 2 & -1 & \ddots & \vdots \\ 0 & -1 & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & 2 & -1 \\ 0 & \dots & 0 & -1 & 2 \end{pmatrix}$$

- Assume that $\boldsymbol{e}^h = I_{2h}^h \boldsymbol{e}^{2h}$

Apply A^h to obtain $A^h I_{2h}^h \boldsymbol{e}^{2h}$, then:

- The odd rows are 0
- The even rows correspond to the coarse grid nodes

Coarse grid matrix – 1D Poisson FDM

$$A = \frac{1}{h^2}\begin{pmatrix} 2 & -1 & 0 & \dots & 0 \\ -1 & 2 & -1 & \ddots & \vdots \\ 0 & -1 & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & 2 & -1 \\ 0 & \dots & 0 & -1 & 2 \end{pmatrix}$$

- To obtain the coarse grid matrix, we apply the Galerkin operator to the *j*-th unit vector $\hat{\boldsymbol{e}}_j^{2h}$

	J-1		j		J+1
$\hat{\boldsymbol{e}}_j^{2h}$	0		1		0
$I_{2h}^h \hat{\boldsymbol{e}}_j^{2h}$	0	$\frac{1}{2}$	1	$\frac{1}{2}$	0
$A^h I_{2h}^h \hat{\boldsymbol{e}}_j^{2h}$	$-\frac{1}{2h^2}$	0	$\frac{1}{h^2}$	0	$-\frac{1}{2h^2}$
$I_h^{2h} A^h I_{2h}^h \hat{\boldsymbol{e}}_j^{2h}$	$-\frac{1}{4h^2}$		$\frac{1}{2h^2}$		$-\frac{1}{4h^2}$

- This is the indeed the finite difference stiffness matrix A^{2h} on Ω^{2h}.

Galerkin projection

- The preceding argument is based on the assumption that $\boldsymbol{e}^h$ lies in the range of interpolation.
- In general, this is not the case.
- However, it gives a sensible definition for A^{2h}.

- Assume that e^h does lie in the range of iterpolation
- Then doing the two-grid correction would give the exact (direct) solution.

About the convergence of the two-level method

Two-level method

- We use the two-level method only with pre-smoothing (no post-smoothing).

- Remember that a stationary linear iteration scheme may be expressed in the form

$$\boldsymbol{v}^{(m)} = (1 - BA)^m \boldsymbol{v}^{(0)} + Bf = S^m \ \boldsymbol{v}^{(0)} + C(\boldsymbol{f})$$

- We want to find the iteration matrix S_{2lev} of the two-level method

Let

- S_{sm} be the iteration matrix of the smoother
- $\boldsymbol{v}^{(m)}$ is the approximation before the pre-smoothing and $\boldsymbol{v}^{(m+1)}$ the result after the update
- $\boldsymbol{v}_{\nu}^{(m)}$ is the approximation at step *m* after ν pre-smoothing steps

Iteration matrix

We know that for the error in the pre-smoothing iterations holds

$$\mathbf{e}^{(\nu)} = S_{sm}^{\nu}\mathbf{e}^{(0)}$$

with

$$\mathbf{e}^{(0)} = \mathbf{u} - \mathbf{v}^{(m)}, \quad \mathbf{e}^{(\nu)} = \mathbf{u} - \mathbf{v}_{\nu}^{(m)}$$

It follows that

$$\mathbf{v}_{\nu}^{(m)} = \mathbf{u} - S_{sm}^{\nu}(\mathbf{u} - \mathbf{v}^{(m)})$$

By substitution, we obtain

$$\mathbf{r} = \mathbf{f} - A^h\mathbf{v}_{\nu}^{(m)} = \mathbf{f} - A^h\mathbf{u} + A^h S_{sm}^{\nu}(\mathbf{u} - \mathbf{v}^{(m)}) = A^h S_{sm}^{\nu}(\mathbf{u} - \mathbf{v}^{(m)}).$$

Iteration matrix

- We develop the matrix starting from the update step

- $\boldsymbol{v}^{(m+1)} = \boldsymbol{v}_{\nu}^{(m)} + I_{2h}^{h}(e^{2h})$

$= \boldsymbol{u} - S_{sm}^{\nu}\left(\boldsymbol{u} - \boldsymbol{v}^{(m)}\right) + I_{2h}^{h}\left(A^{2h}\right)^{-1} I_{h}^{2h}\boldsymbol{r}$

$= S_{sm}^{\nu}\boldsymbol{v}^{(m)} + (I - S_{sm}^{\nu})\left(A^{h}\right)^{-1}\boldsymbol{f}$

$+ I_{2h}^{h}\left(A^{2h}\right)^{-1} I_{h}^{2h} A^{h} S_{sm}^{\nu}\left(\left(A^{h}\right)^{-1}\boldsymbol{f} - \boldsymbol{v}^{(m)}\right)$

$= (I - I_{2h}^{h}\left(A^{2h}\right)^{-1} I_{h}^{2h} A^{h}) S_{sm}^{\nu}\boldsymbol{v}^{(m)}$

$+((I - S_{sm}^{\nu}) + I_{2h}^{h}\left(A^{2h}\right)^{-1} I_{h}^{2h} A^{h} S_{sm}^{\nu})\left(A^{h}\right)^{-1}\boldsymbol{f}$

Def. $v_{\nu}^{(m)}$ on previous slide and coarse grid eqn $A^{2h}\boldsymbol{e}^{2h} = I_{h}^{2h}\boldsymbol{r}$

$A^{h}\mathbf{u} = \boldsymbol{f}$

Def. $\boldsymbol{r}$ on previous slide

Reordering.

Iteration matrix

- The two-level iteration matrix is thus given by

$$S_{2lev} = (I - I_{2h}^{h}(A^{2h})^{-1} I_{h}^{2h} A^{h}) S_{s}^{\nu}$$

- To prove convergence, one could do a rigorous Fourier analysis (see Trottenberg, Osterley et al., Multigrid).
- We will follow an approach of Hackbusch (Multi-grid methods and applications) and show mesh-independent convergence.

Goal of the convergence analysis

- From last week, we know that convergence of the two level method is given, if and only if

$$\rho(S_{2lev}) < 1$$

- This is in general hard to prove.

- However, for some induced matrix norm

$$\rho(S_{2lev}) \leq \|S_{2lev}\|$$

- We thus aim to show

$$\|S_{2lev}\| \leq \rho < 1$$

Convergence analysis

- The analysis is based on a splitting of S_{2lev} as

$$S_{2lev} = \left(\left(A^h\right)^{-1} - I_{2h}^h \left(A^{2h}\right)^{-1} I_h^{2h} \right) A^h S_{sm}^{\nu}$$

- Hence,

$$\|S_{2lev}\| \leq \|\left(A^h\right)^{-1} - I_{2h}^h \left(A^{2h}\right)^{-1} I_h^{2h}\| \cdot \|A^h S_{sm}^{\nu}\|$$

Approximation property

Smoothing property

- Approximation property: This is the effect of the coarse grid approximation.
- Smoothing property: The efficiency of the smoothing step.

Smoothing and approximation properties

Definition 1: The matrix S_{sm} is said to possess the smoothing property, if there exist functions $\eta(\nu)$ and $\bar{\nu}(t)$ independent of h, such that

$$\left\| A^h S_{sm}^{\nu} \right\| \leq \eta(\nu) h^{-\alpha}$$

for some number $\alpha > 0$ and for all $1 \leq \nu \leq \bar{\nu}(h)$, with $\eta(\nu) \to 0$ as $\nu \to \infty$

Definition 2: The approximation property holds if there is a constant C_α, which is independent of h, such that

$$\left\| \left(A^h\right)^{-1} - I_{2h}^{h} \left(A^{2h}\right)^{-1} I_{h}^{2h} \right\| \leq C_\alpha h^{\alpha}$$

with the same α as in the smoothing property.

Convergence of the two-level method

Theorem

Suppose that the smoothing and the approximation property hold. Let $\rho > 0$ be a fixed number. Then there exists a number ν^*, such that

$$\|S_{2lev}\| \leq C_\alpha \eta(\nu) \leq \rho,$$

whenever $\nu \geq \nu^*$.

Proof

Since we required that α is the same for the approximation as well as the smoothing property, we get

$$\|S_{2lev}\| \leq C_\alpha h^\alpha \eta(\nu) h^{-\alpha} = C_\alpha \eta(\nu)$$

Since $\eta(\nu) \to 0$ as $\nu \to \infty$, the right hand side can be smaller than any given ρ, namely as ν is sufficiently large.

Convergence of the two-level method

- Note that the constant $C_\alpha \eta(\nu)$ is independant of the mesh size h
- It converges, if sufficiently many smoothing steps are applied
- In practice only a few pre-smoothing steps (1 to 3) are often sufficient

- The estimate above is however often not very tight.

V(2,1)-cycle for 2D Poisson

	$n = 16$				$n = 32$				$n = 64$				$n = 128$			
V-cycle	$\|r^h\|_h$	ratio	$\|e\|_h$	ratio	$\|r^h\|_h$	ratio	$\|e\|_h$	ratio	$\|r^h\|_h$	ratio	$\|e\|_h$	ratio	$\|r^h\|_h$	ratio	$\|e\|_h$	ratio
0	6.75e+02		5.45e−01		2.60e+03		5.61e−01		1.06e+04		5.72e−01		4.16e+04		5.74e−01	
1	4.01e+00	0.01	1.05e−02	0.02	1.97e+01	0.01	1.38e−02	0.02	7.56e+01	0.01	1.39e−02	0.02	2.97e+02	0.01	1.39e−02	0.02
2	1.11e−01	0.03	4.10e−04	0.04	5.32e−01	0.03	6.32e−04	0.05	2.07e+00	0.03	6.87e−04	0.05	8.25e+00	0.03	6.92e−04	0.05
3	3.96e−03	0.04	1.05e−04	0.26	2.06e−02	0.04	4.41e−05	0.07	8.30e−02	0.04	4.21e−05	0.06	3.37e−01	0.04	4.22e−05	0.06
4	1.63e−04	0.04	1.03e−04	0.98*	9.79e−04	0.05	2.59e−05	0.59	4.10e−03	0.05	7.05e−06	0.17	1.65e−02	0.05	3.28e−06	0.08
5	7.45e−06	0.05	1.03e−04	1.00*	5.20e−05	0.05	2.58e−05	1.00*	2.29e−04	0.06	6.45e−06	0.91*	8.99e−04	0.05	1.63e−06	0.50
6	3.75e−07	0.05	1.03e−04	1.00*	2.96e−06	0.06	2.58e−05	1.00*	1.39e−05	0.06	6.44e−06	1.00*	5.29e−05	0.06	1.61e−06	0.99*
7	2.08e−08	0.06	1.03e−04	1.00*	1.77e−07	0.06	2.58e−05	1.00*	8.92e−07	0.06	6.44e−06	1.00*	3.29e−06	0.06	1.61e−06	1.00*
8	1.24e−09	0.06	1.03e−04	1.00*	1.10e−08	0.06	2.58e−05	1.00*	5.97e−08	0.07	6.44e−06	1.00*	2.14e−07	0.06	1.61e−06	1.00*
9	7.74e−11	0.06	1.03e−04	1.00*	7.16e−10	0.06	2.58e−05	1.00*	4.10e−09	0.07	6.44e−06	1.00*	1.43e−08	0.07	1.61e−06	1.00*
10	4.99e−12	0.06	1.03e−04	1.00*	4.79e−11	0.07	2.58e−05	1.00*	2.87e−10	0.07	6.44e−06	1.00*	9.82e−10	0.07	1.61e−06	1.00*
11	3.27e−13	0.07	1.03e−04	1.00*	3.29e−12	0.07	2.58e−05	1.00*	2.04e−11	0.07	6.44e−06	1.00*	6.84e−11	0.07	1.61e−06	1.00*
12	2.18e−14	0.07	1.03e−04	1.00*	2.31e−13	0.07	2.58e−05	1.00*	1.46e−12	0.07	6.44e−06	1.00*	4.83e−12	0.07	1.61e−06	1.00*
13	2.33e−15	0.11	1.03e−04	1.00*	1.80e−14	0.08	2.58e−05	1.00*	1.08e−13	0.07	6.44e−06	1.00*	3.64e−13	0.08	1.61e−06	1.00*
14	1.04e−15	0.45	1.03e−04	1.00*	6.47e−15	0.36	2.58e−05	1.00*	2.60e−14	0.24	6.44e−06	1.00*	1.03e−13	0.28	1.61e−06	1.00*
15	6.61e−16	0.63	1.03e−04	1.00*	5.11e−15	0.79	2.58e−05	1.00*	2.30e−14	0.88	6.44e−06	1.00*	9.19e−14	0.89	1.61e−06	1.00*

- 2D Poisson on unit square and homogeneous Dirichlet conditions
- Results taken from Briggs et al, A multigrid tutorial, Table 4.1

The Multigrid Method

Multigrid method

What is the best method to solve the coarse grid equation $A^{2h}e^{2h} = r^{2h}$?

- We could use a direct method. But what if the coarse grid problem is still very large?
- Let us think recursively: The coarse-grid problem is not much different from the original problem.
- We can apply the two-level method to the coarse grid problem, thus going to a grid on Ω^{4h}.
- We can repeat this process until a direct solution of the residual equation is possible.

In the following

- let *L* be the number of levels,
- for simplicity of the notation, we will call the restriction of the residual

$$f^{2^k h} := I_{kh}^{2^k h} r^{kh}.$$

- It is in fact only a new right-hand side.

V-cycle scheme

$v^h \leftarrow V^h(v^h, f^h)$

- Relax ν_1 times on $A^h u^h = f^h$ with initial guess v^h. The result is denoted by v^h.
- Compute $f^{2h} = I_h^{2h} r^h = I_h^{2h}(f^h - A^h v^h)$.
  - Relax ν_1 times on $A^{2h} u^{2h} = f^{2h}$ with initial guess $v^{2h} = 0$. The results is denoted by v^{2h}.
  - Compute $f^{4h} = I_{2h}^{4h} r^{2h} = I_{2h}^{4h}(f^{2h} - A^{2h} v^{2h})$.

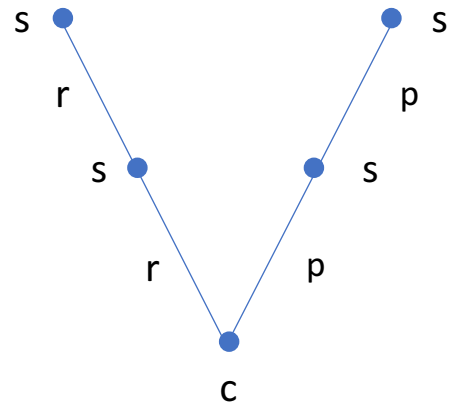
⋮

- Solve $A^{Lh} u^{Lh} = f^{Lh}$

Coarsest grid solve

⋮

  - Correct $v^{2h} := v^{2h} + I_{4h}^{2h} v^{4h}$
  - Apply smoother ν_2 times to $A^{2h} u^{2h} = f^{2h}$ with the initial guess v^{2h}
- Correct $v^h := v^h + I_{2h}^h v^{2h}$
- Apply smoother ν_2 times to $A^h u^h = f^h$ with the initial guess v^h



Remarks

In terms of the error:

- Remember what we said about the frequency components on different grids. Here, we do the following:
  - By applying smoothing iterations on Ω^h, we cause the *h* - high-frequency components to decrease rapidly. The smooth ones stay visible.
  - We then restrict the *h* – low-frequency components to Ω^{2h}, where they become more high-frequent.
  - We then apply again a smoother and damp out the *2h* - high-frequency components.
  - By continuing to the coarsest grid, we obtain a very fast overall reduction of the error.

Furthermore:

- The best number of pre-relaxation and post-relaxation iterations is normally 1 to 3.
- The boundary conditions for the coarse grid problem are 0 (because the coarse-grid variable is the error).
- The initial guess for the coarse-grid solution must be 0.

Multigrid with γ-cycle

- The previous V-cycle scheme is just one possibility to perform a multigrid method.
- It belongs to a family of multigrid methods, the so-called multigrid methods with γ-cycle.
- In these cases, the cycle does not necessarily have a shape of a 'V'.
- It has a compact recursive definition (see next slide).

We use some new notation:

- Number of levels L, thus giving $(L+1)$ grids.
- Let $k = 0, \dots, L$ be the numbering of the grids. Then $k = 0$ corresponds to the coarsest one, and $k = L$ to the finest one.

Multigrid with γ-cycle

Multigrid Cycle $u_k^{(m+1)} = MGC(k, \gamma, u_k^{(m)}, L_k, f_k, \nu_1, \nu_2)$

1. **Pre smoothing:** Compute $\bar{u}_k^{(m)}$ by applying ν_1 smoothing steps to $u_k^{(m)}$.
2. **Coarse grid correction:**

   Compute the residual and restrict it to the next coarser grid $r_{k-1}^{(m)} = I_k^{k-1}\left(f_k - A_k r_l^{(m)}\right)$

   If $\boldsymbol{k = 1}$:

   We are on the coarsest grid, then solve the problem directly and go to step 3.

   If $\boldsymbol{k > 1}$:

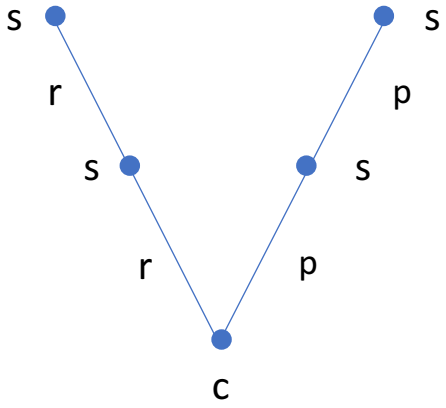
   Apply $\gamma \geq 1$ k – grid cycles using the zero initial guess $v = 0$:

$$v_{k-1}^{(m)} = MGC^{\gamma}(k-1, \gamma, 0, A_{k-1}, r_{k-1}^{(m)}, \nu_1, \nu_2)$$

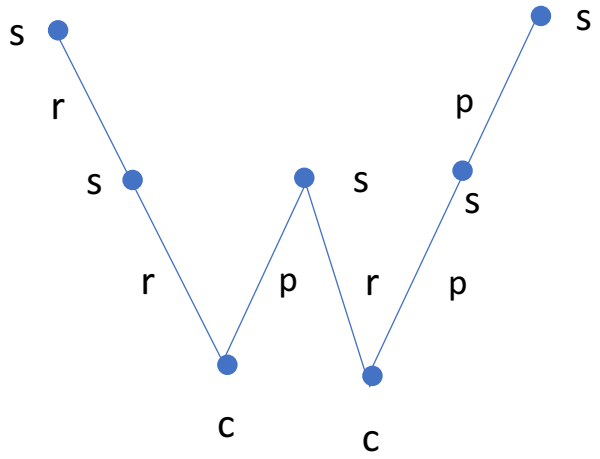
3. **Prolongation and update:** Compute $u_k^{(m)} = \bar{u}_k^{(m)} + I_{k-1}^k v_{k-1}^{(m)}$
4. **Post smoothing:** Apply the smoother ν_2 times on $A_k u_k^m = f^h$ and obtain $u_k^{(m+1)}$.

Three-grid methods

- In practice, mostly $\gamma = 1$ and $\gamma = 2$ are used.
- The following figures show one iteration step of a multigrid method with 3 grid - levels



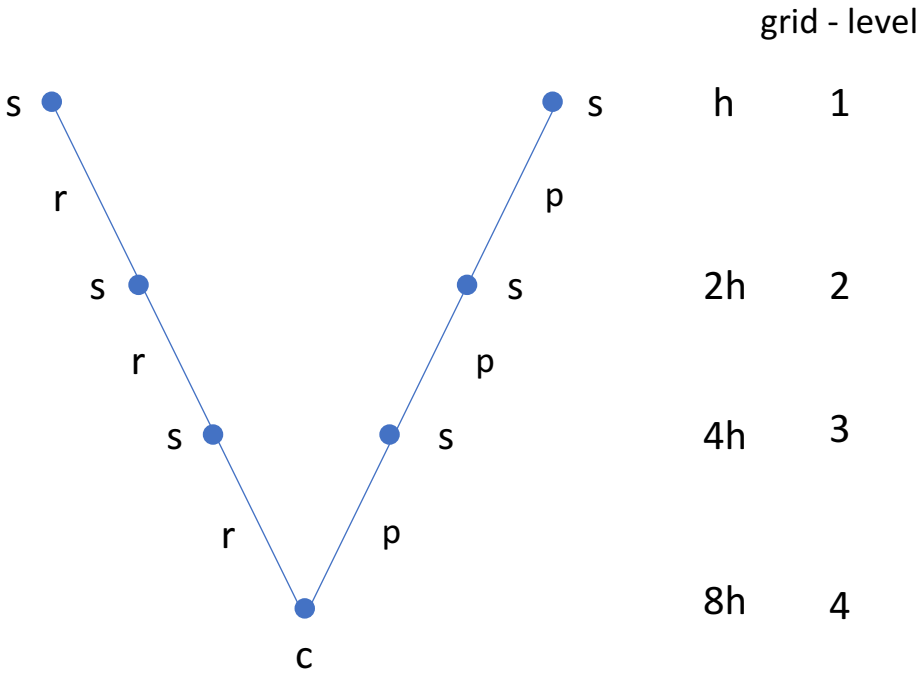
$\gamma = 1$: V-cycle scheme



$\gamma = 2$: W-cycle scheme

Four-grid methods

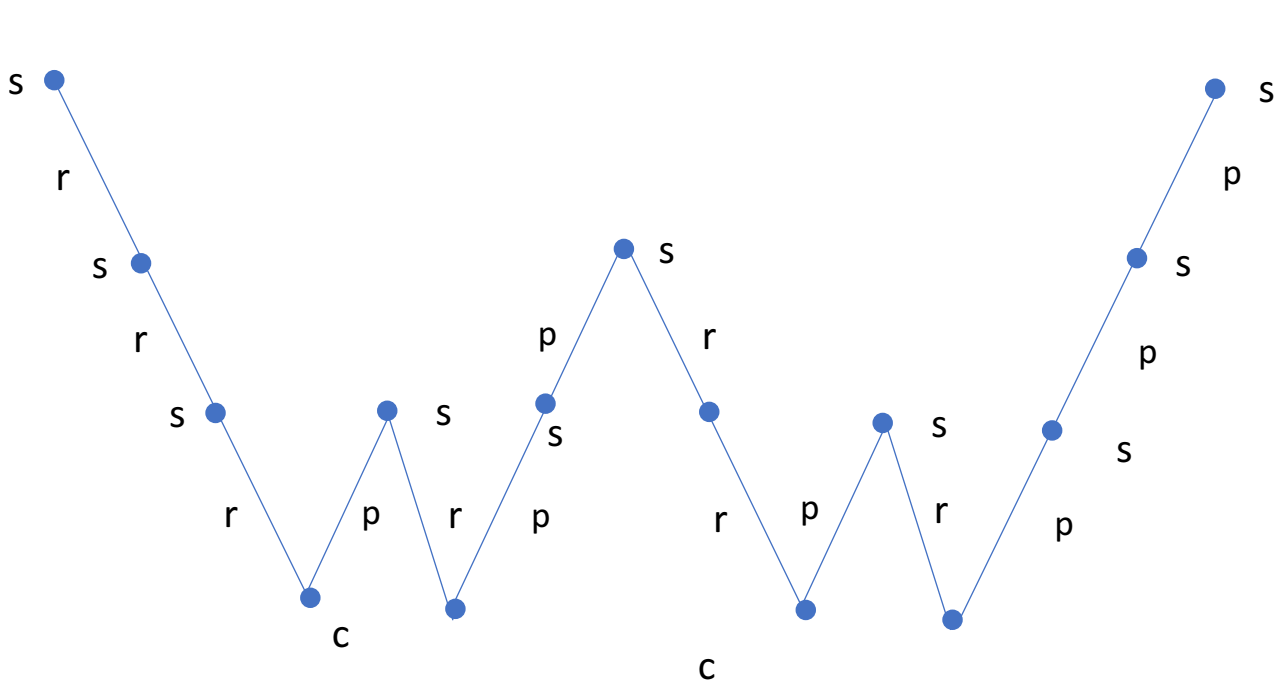
- Here we see a four-grid V-cycle



$\gamma = 1$: V-cycle scheme

Multigrid W-cycle

- Four-grid W-cycle scheme



$\gamma = 2$: W-cycle scheme

Iteration operator of γ-cycle

Remeber that the two-grid iteration operator from grid *h* to *2h* is given by

$$(S_{2lev})_h^{2h} = (I - I_{2h}^h \left(A^{2h}\right)^{-1} I_h^{2h} A^h) S_{sm}^{\nu}$$

What changes in the γ-cycle multigrid method?

$\left(A^{2h}\right)^{-1}$ is now an recursive loop of two-level operators.

Let L be the number of levels and $k = L, \dots, 0$ the grid level. The multigrid iteration operator M_k is given for $k = 1, .., L$ by

$$M_0 = 0$$
$$M_k = S_k^{\nu_2}\left(I - I_{k-1}^k (I_{k-1} - (M_{k-1})^{\gamma})(A_{k-1})^{-1} I_k^{k-1} A_k\right) S_k^{\nu_1}$$

The γ –cycle multigrid method also shows mesh independent convergence as the two-level method.

Computational work

Computational work of multigrid cycle

- Earlier, we have seen that the convergence speed of the two-level method does not depend on the mesh size of the finest grid (mesh-independent convergence).
- The same is true for the multigrid algorithm.
- But just knowing that its convergence is independent of the mesh size says nothing about its efficiency (e.g. a direct method always needs only ’one’ iteration).
- We thus want to estimate the computational work of a multigrid method.
- We follow the description in the book ‘Multigrid’ of Trottenberg et al.

Computational costs of V-Cycle

- From the recursive definition of a multigrid cycle, it follows that the computational work W_l per multigrid cycle on Ω_l is given recursively by

$$W_1 = W_1^0 + W_0, \qquad W_{k+1} = W_{k+1}^k + \gamma W_k, \qquad k = 1, \dots, L-1$$

- Here, W_{k+1}^k denotes the computational work of one (h_{k+1}, h_k) two-grid cycle, excluding the work needed to solve the residual equation.
- W_0 denotes the work needed to compute the exact solution on the coarsest grid Ω_0.
- By computational work, we mean some reasonable measure, typically the number of arithmetic operations.

We obtain for level $\mathrm{L} \geq 1$

$$W_L = \sum_{k=1}^{L} \gamma^{L-k} W_k^{k-1} + \gamma^{L-1} W_0$$

Computational costs

- We use standard coarsening in 2D, i.e. $h \rightarrow 2h \rightarrow 4h \rightarrow$ …, which gives in terms of grid points

$$N_k = 4N_{k-1}, \qquad (k = 1,2, \dots, L)$$

  with equality up to lower order terms (boundary effects).

- We assume that the multigrid components (relaxation, computation of residuals, fine-to-coarse and coarse-to-fine grid transfer) require a number of arithmetic operations per point of the respective grids, which is bounded by a small constant *C*, independent of *k:*

$$W_k^{k-1} \leq C\ N_k, \qquad (k = 1, 2, \dots, L),$$

  with, again, $' \leq '$ up to lower order terms.

The constant C

W_k^{k-1} or, in other words the constant C, is determined by the computational work of the multigrid components of the (h_k, h_{k-1}) two grid method, namely

$$W_k^{k-1} = \big((\nu_1 + \nu_2)w_0 + w_1 + w_2\big)N_k$$

Here, w_0, w_1, w_2 are measures for the computational work per grid point of Ω_k needed for the single components

- w_0: one smoothing step on Ω_k
- w_1: computation of the residual and its transfer to Ω_{k-1}
- w_2: interpolation of the correction to Ω_k and its addition to the previous approximation.
- ν_1, ν_2: number of pre- and postsmoothing steps

In practice, interpolation and restriction are often negligible and only the cost of one smoothing step in used.

Computational costs

- We then obtain

$$W_l \leq \begin{cases} \frac{4}{3} C N_L, & \text{for } \gamma = 1 \\ 2\, C N_L & \text{for } \gamma = 2 \\ 4\, C N_L & \text{for } \gamma = 3 \\ \mathcal{O}(N_L \log N_L) & \text{for } \gamma = 4 \end{cases}$$

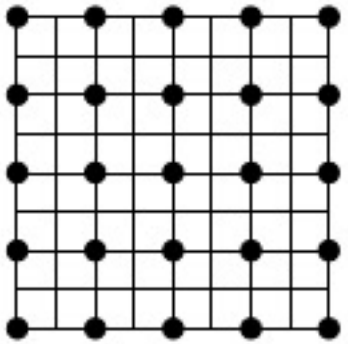
- This estimate shows that the computational work needed for one 2D multigrid cycle is proportional to the number of grid points on the finest grid for $\gamma \leq 3$ and standard coarsening.
- Together with the h- independent convergence, this means that multigrid methods achieve a fixed reduction of the error in $\mathcal{O}(N)$ operations.
- The constant C depends on γ, the type of coarsening and the other multigrid components.

Practical aspects

Coarsening

Coarse grids – standard coarsening

- So far, we have only talked about standard coarsening, i.e. we assume a number of 2^n elements and divide these for each coarser grid by 2.



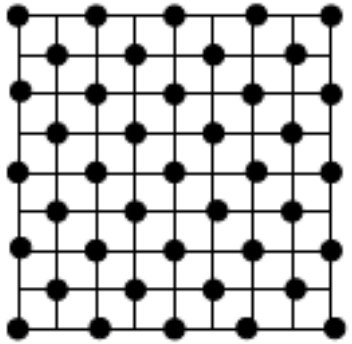
- In *d* dimensions, the relation between the number of grid points (neglecting boundary effects) is thus given by

$$\# \, \Omega_{2h} \approx \frac{1}{2^d} \# \, \Omega_h$$

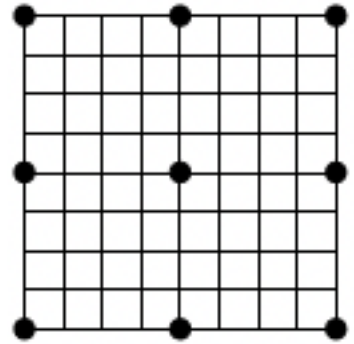
- There are however other choices…

Coarse grids – red-black and 4h

- Also of interest:



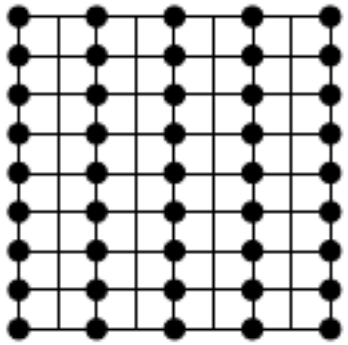
Red-black (or checkerboard) coarsening



4h-coarsening

Coarse grids – semi-coarsening

- For some aninsotropic problems, it might be more appropriate to coarsen only in the *x*- or *y*-direction (we will see an example next). This is called semi-coarsening.



- The mesh size is doubled in one direction only, and is left constant in the other, i.e. $H = (2h_x, h_y)$ or $H = (h_x, 2h_y)$
- The number of degrees of freedom do not decrease as fast as for standard coarsening. Some more levels might be needed to have a small coarse-grid problem.
- In this case, we have

$$\# \, \Omega_{2h} \approx \frac{1}{2} \# \, \Omega_h$$

Anisotropic Poisson equation

- Let us assume that we have a model problem with $0 < \epsilon \ll 1$

$$-u_{xx} - \epsilon u_{yy} = f \qquad \text{+ Bdy conditions}$$

- This leads to the same five-point as the Poisson equation ($\epsilon = 1$)

$$A^h = \frac{1}{h^2}\begin{pmatrix} & -\epsilon & \\ -1 & 2+2\epsilon & -1 \\ & -\epsilon & \end{pmatrix}$$

- We have only a weak connection in y-direction. This is best visible in the limiting case $\epsilon = 0$.

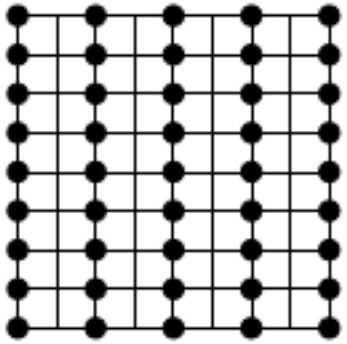
$$A^h = \frac{1}{h^2}\begin{pmatrix} & 0 & \\ -1 & 2 & -1 \\ & 0 & \end{pmatrix} \quad \leftarrow \quad \text{This is the 1D-Poisson 3-Point stencil in x-direction}$$

- Thus, Gauss-Seidel or weighted Jacobi would smooth well in x-direction (they do so in 1D), but each line is not well connected to the x-line above.

How can we overcome this problem?

Semi-coarsening and point relaxation

- For this problem, it makes sense to use semi-coarsening.
- Point relaxation (standard Gauss-Seidel or weighted Jacobi that acts on each point), will not smoothen well in y-direction. So it makes sense to keep every point also on the coarse grid to have a good resolution of the error.
- We thus only coarsen in x-direction, where the smoother works well as in 1D.
- Interpolation and restriction are done in a one-dimensional way.



Full coarsening and line relaxation

- If we order the unknowns in the direction of constant *x*, i.e. the direction of strong coupling, we obtain

$$A^h = \begin{pmatrix} \widetilde{D} & -cI & & & \\ -cI & \widetilde{D} & -cI & & \\ & -cI & \widetilde{D} & \ddots & \\ & & \ddots & \ddots & -cI \\ & & & -cI & \widetilde{D} \end{pmatrix} \quad \text{with} \quad \widetilde{D} = \begin{pmatrix} 2+2\epsilon & -1 & & & \\ -1 & 2+2\epsilon & -1 & & \\ & -1 & \ddots & \ddots & \\ & & \ddots & \ddots & -1 \\ & & & -1 & 2+2\epsilon \end{pmatrix} \quad \text{and} \quad c = \frac{\epsilon}{h^2}$$

- Each block is associated with an individual horizontal grid line.
- We can apply weighted Jacobi in the sense of a block-matrix, i.e.

$$D^{-1} = \begin{pmatrix} \widetilde{D}^{-1} & & & \\ & \widetilde{D}^{-1} & & \\ & & \ddots & \\ & & & \widetilde{D}^{-1} \end{pmatrix}$$

- The matrices $\widetilde{D}$ are tridiagonal and can thus be efficiently solved using some kind of Gaussian elimination.
- This method is called line-relaxation, as it treats one line together.

Practical aspects

Parallelization

Parallel properties of smoothers - Jacobi

- With growing problem sizes, we are interested in parallel implementations of the multigrid method. One important ingredient are the smoothers.

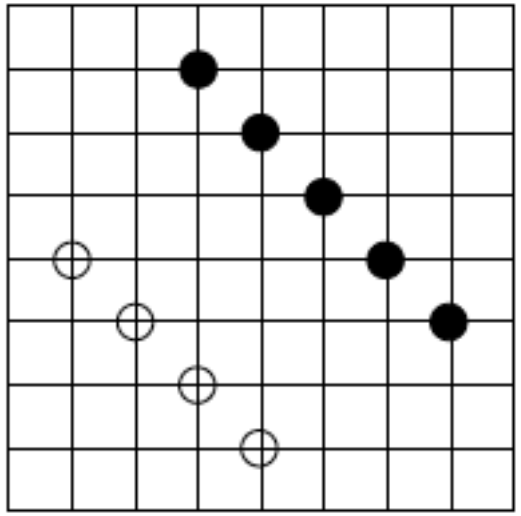
Weighted Jacobi:

- This smoother is fully parallel, as we need for the next iterate only the values of the previous iterate.
- The new values do not depend on each other, thus it can be applied to all grid points simultaneously.
- The degree of parallelism is the number of grid points, thus

$$\text{Par-deg}(\omega\text{-JAC}) = \sharp\, \Omega_h$$

Parallel properties of smoothers – Gauss-Seidel

- The classical Gauss-Seidel smoother is not parallel, as for each new value in a line, we access the already computed ones.
- However, there are special cases.

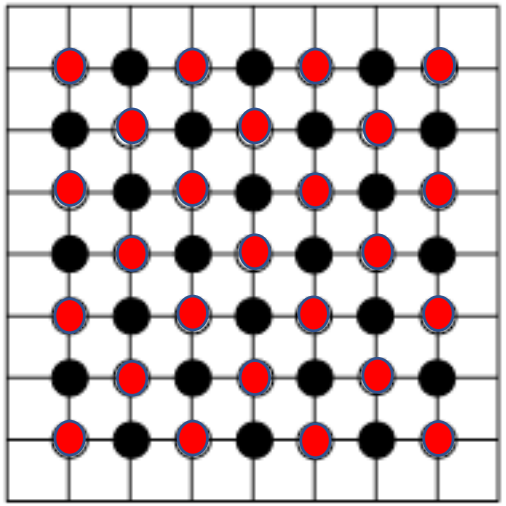


(These are the grid-points, not the matrix entries.)

- For the 2D Poisson problem with a 5-point stencil, the entries on a diagonal are independent from each other. These can be relaxed in parallel.
- From one diagonal to the other is sequential.
- The number of points in one diagonal varies.
- The degree of parallelism is restricted by

$$\text{Par-deg}(\text{GS}) = (\#\ \Omega_h)^{\frac{1}{2}}$$

Red-black Gauss-Seidel smoother



- The red-black Gauss-Seidel smoother consiste of two half-steps:
  1. All red grid points are treated simultaneously
  2. All black grid points are treated, using the updated red grid point values.

- The degree of parallelism is thus

$$\text{Par-deg(RB-GS)} = \frac{1}{2}(\#\ \Omega_h)$$

Smoothing vs Parallelism

Relaxation	Smoothing factor	Smoothing	Parallel degree
ω-Jacobi, $\omega = 1$	1	No	N
ω-Jacobi, $\omega = 0.5$	0.75	Unsatisfactory	N
ω-Jacobi, $\omega = 2/3$	0.33	Very good	N
Gauss-Seidel	0.5	Good	$\leq \sqrt{N}$
GS-RB	0.25	Very good	$\frac{1}{2}N$

Measure multigrid convergence factor

- In order to analyze a multigrid iteration, one often wants to determine its convergence factor ρ empirically.
- The only quantities that are available for the determination of ρ are the residuals on each level.
- For example, we can compute

$$q^{(m)} := \frac{\left\| r_h^{(m)} \right\|_2}{\left\| r_h^{(m-1)} \right\|_2}$$

Or as a product of all previous residuals

$$\hat{q}^{(m)} := \sqrt[m]{q^{(m)} q^{(m-1)} \dots q^{(1)}} = \sqrt[m]{\frac{\left\| r_h^m \right\|_2}{\left\| r_h^0 \right\|_2}}$$

- The quantity $\hat{q}^{(m)}$ represents an average defect reduction factor over m iterations.
- In general, for $r_h^0 \neq 0$, we have $\hat{q}^{(m)} \to \rho$ for m sufficiently large.
- The convergence history for $q^{(m)}$ might also be of interest.