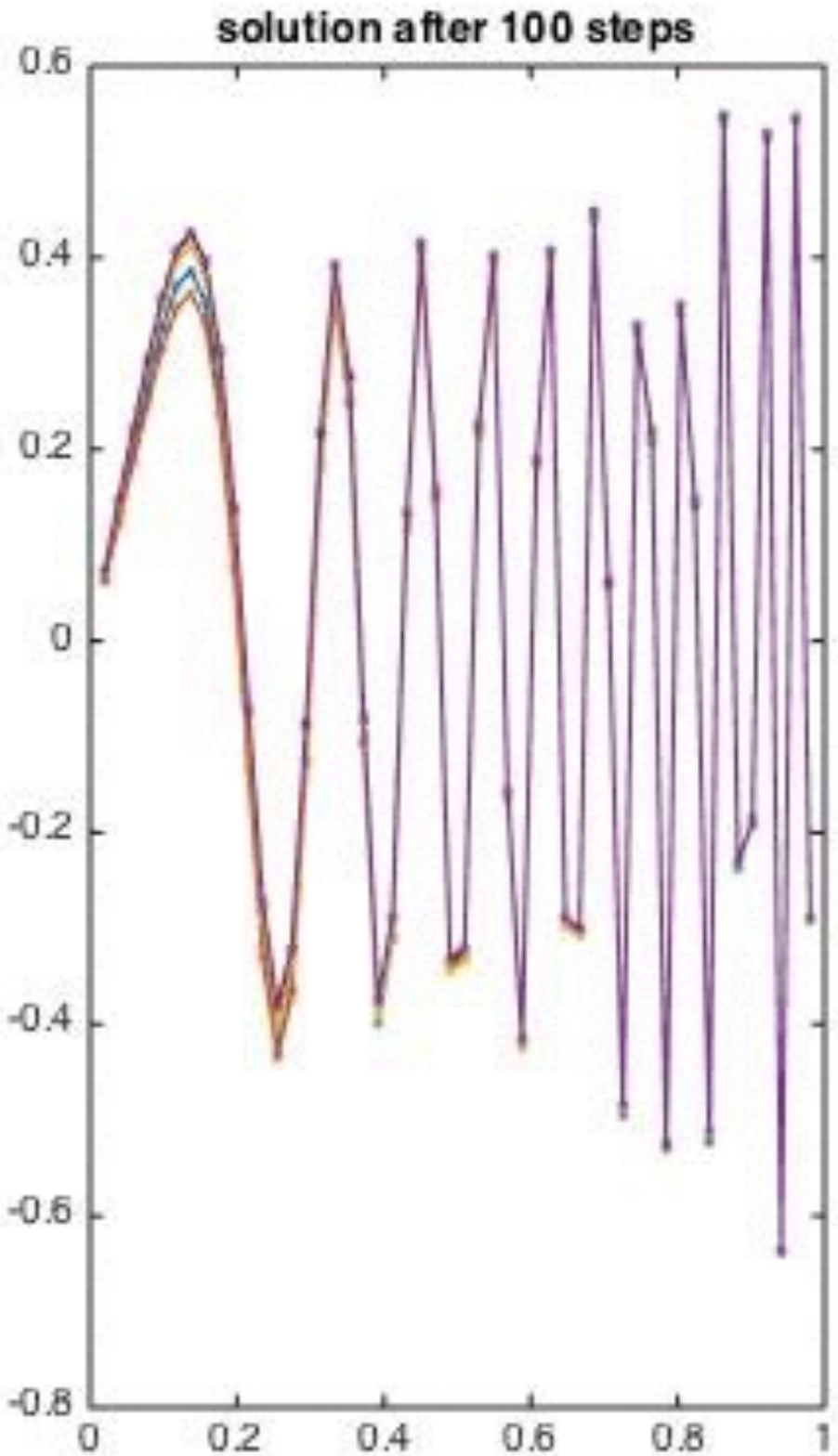
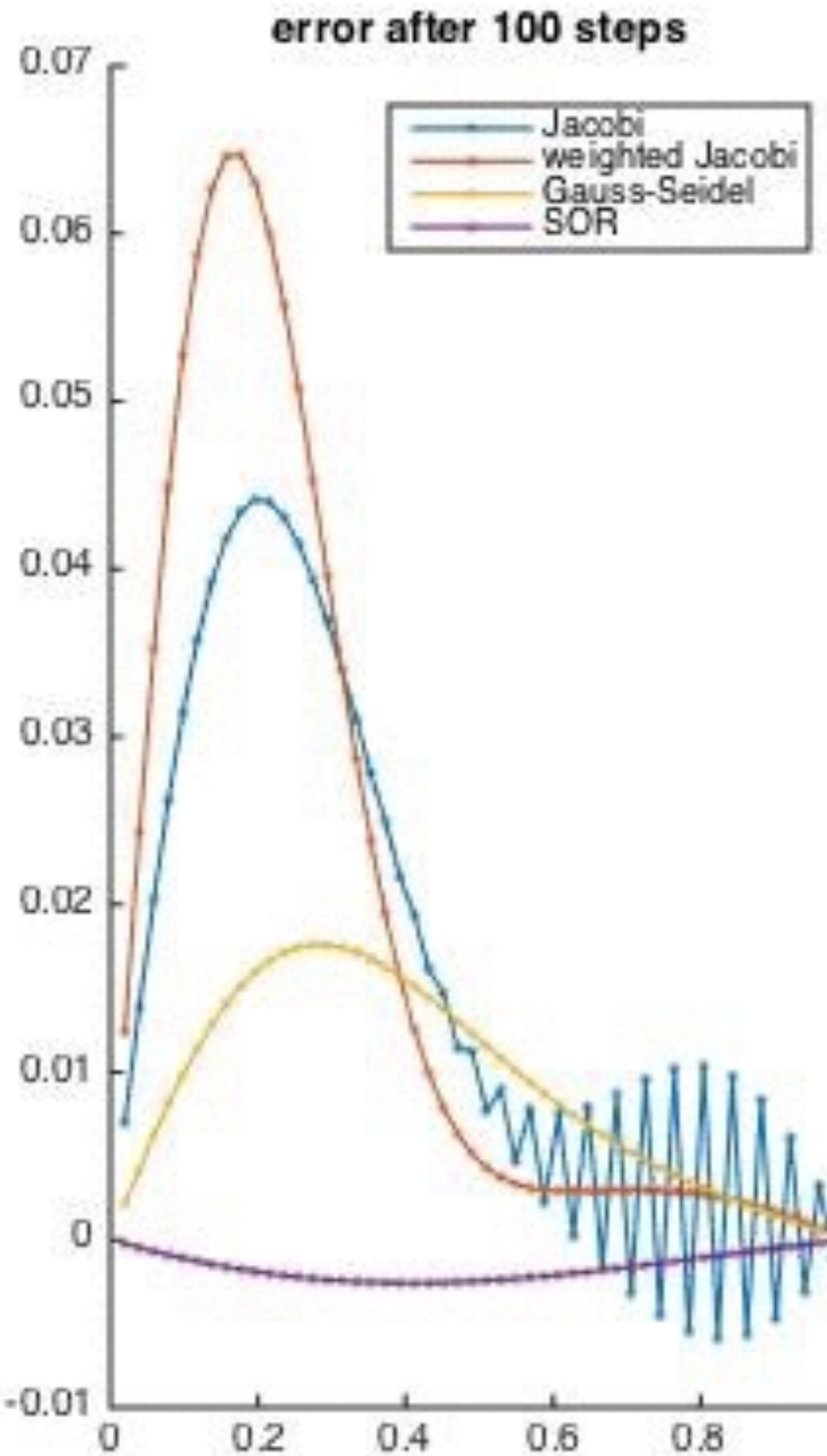
Lecture 16

18.086


solution after 100 steps
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
0
0.2
0.4
0.6
0.8
1

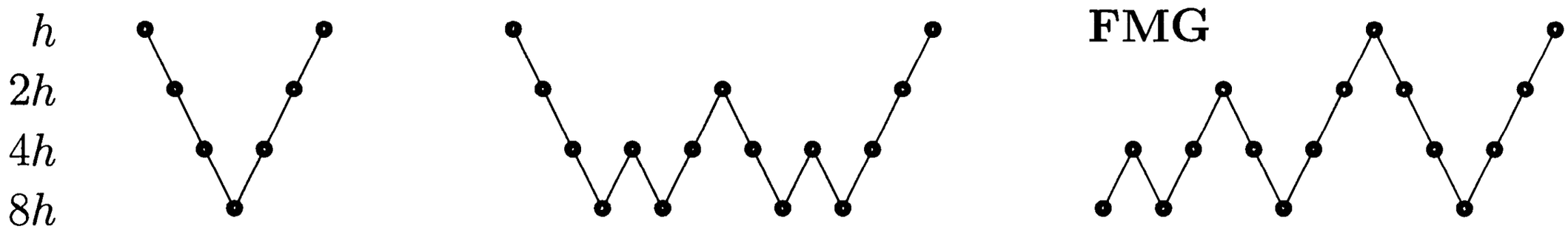

error after 100 steps
Jacobi
weighted Jacobi
Gauss-Seidel
SOR
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
-0.01
0
0.2
0.4
0.6
0.8
1

Multigrid methods (7.3)

- Low freq. on fine mesh => high frequency on coarse mesh

- More than two meshes possible!

- “Notation”:



- Simplest one is the v-cycle with just two meshes (h and 2h grid spacing)

v-cycle algorithm

1. **Iterate** on $A_h u = b_h$ to reach u_h (say 3 Jacobi or Gauss-Seidel steps).

2. **Restrict** the residual $r_h = b_h - A_h u_h$ to the coarse grid by $r_{2h} = R_h^{2h} r_h$.

3. **Solve** $A_{2h} E_{2h} = r_{2h}$ (or come close to E_{2h} by 3 iterations from $E = 0$).

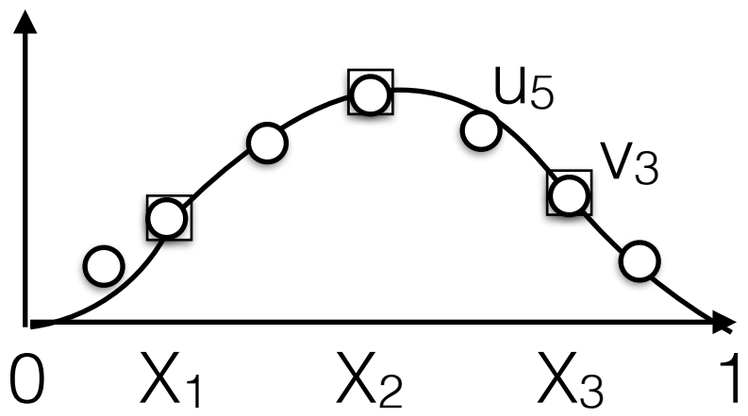
4. **Interpolate** E_{2h} back to $E_h = I_{2h}^h E_{2h}$. Add E_h to u_h.

5. **Iterate** 3 more times on $A_h u = b_h$ starting from the improved $u_h + E_h$.

In the following:
u_i: quantities on fine grid
v_i: quantities on coarse grid

Technicalities of multigrain algorithms

- Interpolation matrix I from coarse (v_i) => fine (u_i) grid: linear interpolation



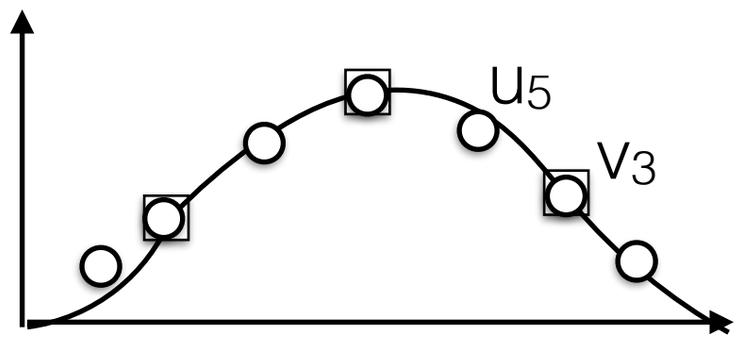
assuming Dirichlet BC

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \end{pmatrix} = \frac{1}{2} \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 2 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \\ 0 & 0 & 1 \end{pmatrix}}_{I} \cdot \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}$$

- 2D (see blackboard)

Technicalities of multigrain algorithms

- Restriction matrix R from fine (u_i) => coarse (v_i) grid:



assuming Dirichlet BC

A possible choice: $v_1 = u_2$ etc.

- Smarter choice: Weighted average, i.e. $v_1 = (u_1 + 2u_2 + u_3)/4$. Then:

$$R = \frac{1}{2} I^T$$

Technicalities of multigrain algorithms

- Restriction of system matrix A from fine (A_h) => coarse (A_{2h}) grid:

$$A_{2h} = RA_hI$$

- Example using $A=K_5/h^2$

$$A_h = \frac{1}{h^2}\begin{pmatrix} 2 & -1 & 0 & \dots \\ -1 & 2 & -1 & \dots \\ \dots & \dots & \dots & \dots \\ 0 & \dots & -1 & 2 \end{pmatrix} \Bigg\updownarrow 5$$

5 fine grid points =>
2 coarse grid points

$$A_{2h} = RA_hI = \dots = \frac{1}{(2h)^2}\begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}$$

This is just the K-matrix on the coarse mesh!

v-cycle algorithm

1. **Iterate** on $A_h u = b_h$ to reach u_h (say 3 Jacobi or Gauss-Seidel steps).

2. **Restrict** the residual $r_h = b_h - A_h u_h$ to the coarse grid by $r_{2h} = R_h^{2h} r_h$.

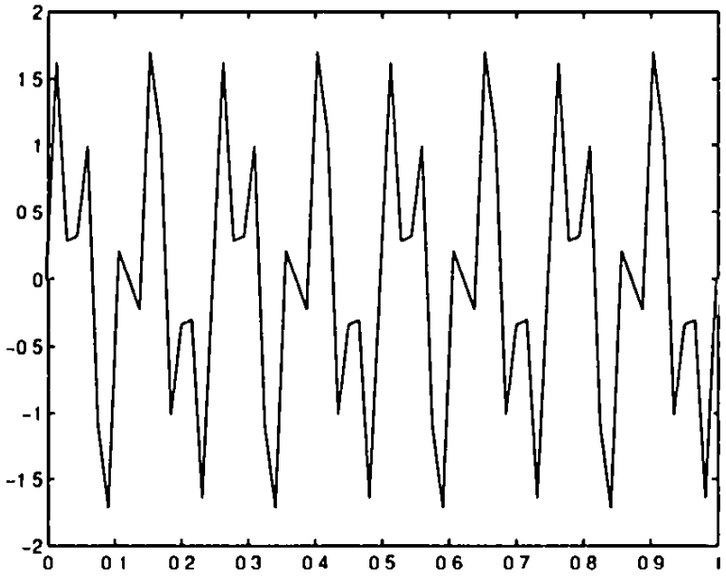
3. **Solve** $A_{2h} E_{2h} = r_{2h}$ (or come close to E_{2h} by 3 iterations from $E = 0$).

4. **Interpolate** E_{2h} back to $E_h = I_{2h}^h E_{2h}$. Add E_h to u_h.

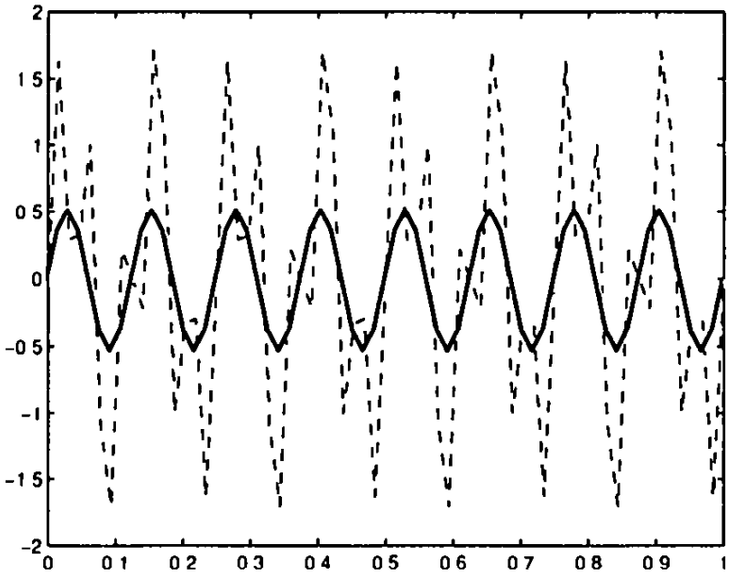
5. **Iterate** 3 more times on $A_h u = b_h$ starting from the improved $u_h + E_h$.

In the following:
u_i: quantities on fine grid
v_i: quantities on coarse grid

Error behavior

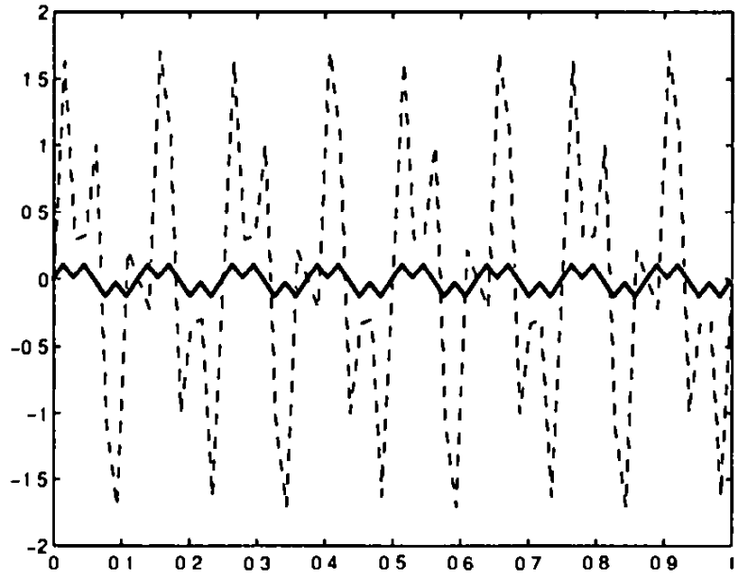


error of initial guess



error e_h

after 3 fine grid iter.



error e'_h

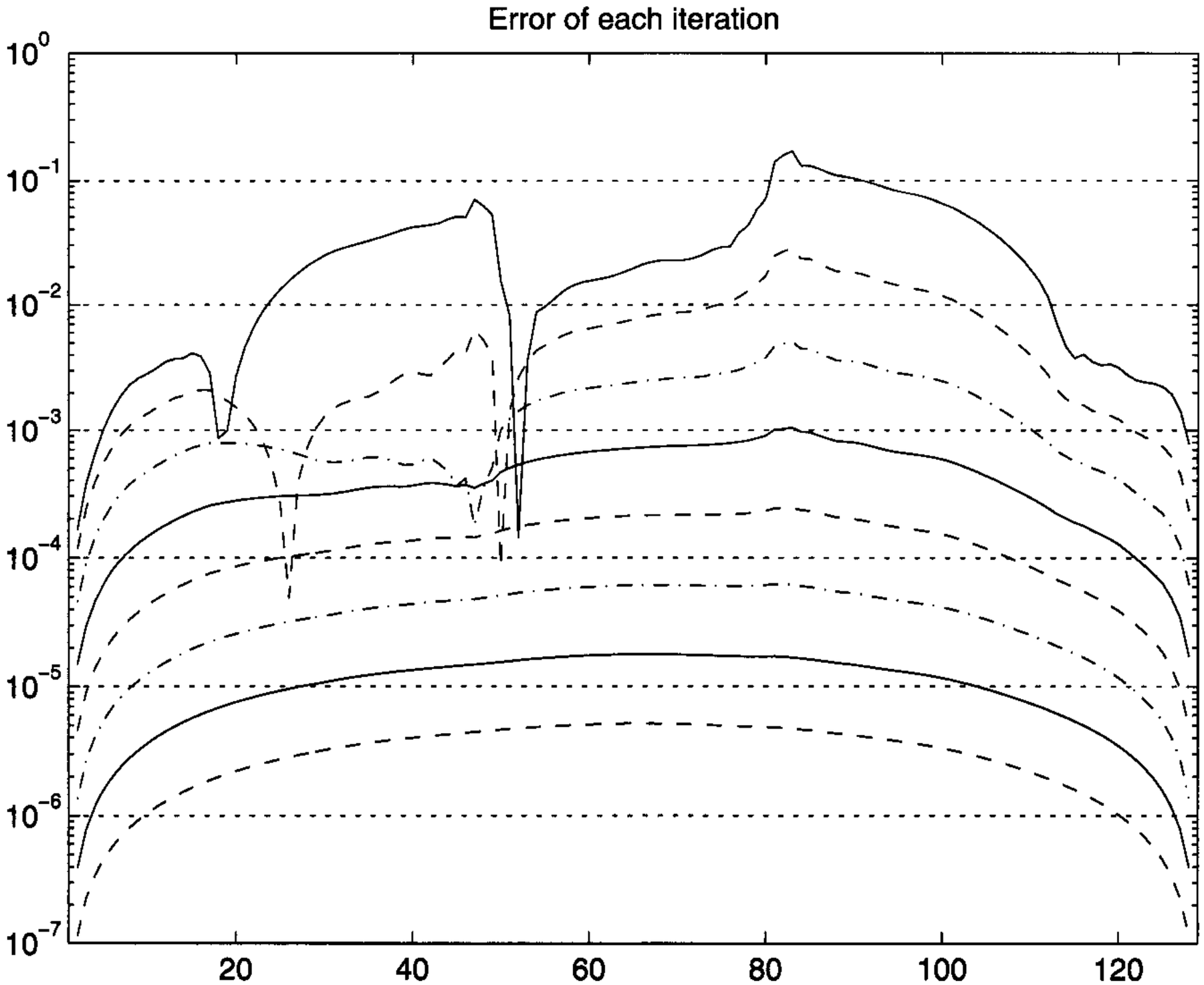
after 3 coarse grid iter.

Some notes on performance

- Can show: $\|\mathbf{error\ after\ step\ 5}\| \leq \rho \|\mathbf{error\ before\ step\ 1}\|$ with typically $\rho \approx 0.1$ **independent** on h (i.e. N)!

- Typically for Jacobi alone: $\rho \approx 0.99$

Some notes on performance



Some notes on performance

- Total cost per complete v-cycle: O(N) (N: number of grid points)
- Since error is reduced in each step by a constant independent on h (and thus N), the total cost to get arbitrary low errors still is only O(N):
- The finest level grid operations determine the cost!