Introduction to multigrid methods. Application of Schwarz domain decomposition methods as smoothers.

Carmen Rodrigo

IUMA, Department of Applied Mathematics
Universidad de Zaragoza (Spain)

Departamento de Matemática Aplicada
Universidad Zaragoza

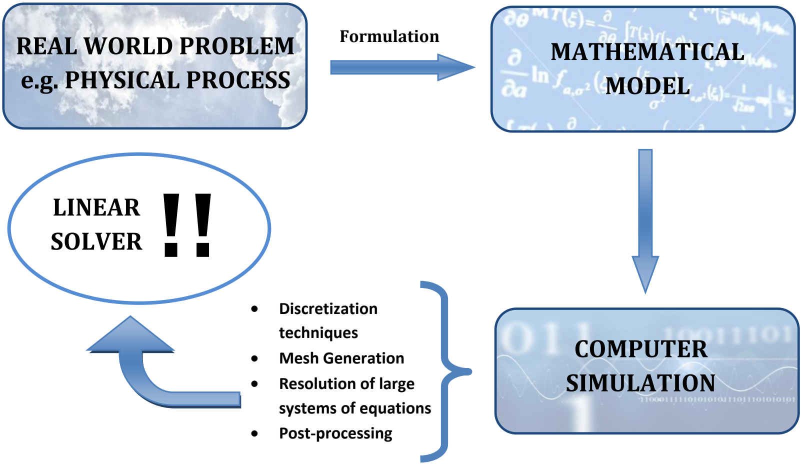
Instituto Universitario de Investigación de Matemáticas y Aplicaciones
Universidad Zaragoza

XXII Jacques-Louis Lions Hispano-French School on Numerical Simulation in Physics and Engineering, Institut Henri Poincaré, Paris (France), 12-15 January 2026

Contents

- Motivation
- Introduction to Multigrid (MG) methods
- Deterioration of the MG convergence
  - High-order discretizations
  - Saddle-point type problems
- Schwarz domain decomposition methods as smoothers
  - Multiplicative
  - Additive
  - Restricted Additive
- Numerical examples:
  - High-order discretization for Poisson
  - Saddle-point type problems: Biot's model

Motivation


REAL WORLD PROBLEM
e.g. PHYSICAL PROCESS
Formulation
MATHEMATICAL MODEL
COMPUTER SIMULATION
LINEAR SOLVER !!
• Discretization techniques
• Mesh Generation
• Resolution of large systems of equations
• Post-processing

Numerical Simulation: Solution of systems of equations

NUMERICAL SIMULATION OF PDE MODELS

DISCRETIZATION of PDEs ✓

EFFICIENT SOLUTION OF LARGE SYSTEMS OF EQUATIONS

Desirable properties

- Robust convergence with respect to the discretization and physical parameters.
- Efficiency.

Solution of large systems of equations

Solving linear system of equations:

Given $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ and $\boldsymbol{b} \in \mathbb{R}^{N}$, how to solve $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ efficiently?

Solution of large systems of equations

Solving linear system of equations:

Given $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ and $\boldsymbol{b} \in \mathbb{R}^{N}$, how to solve $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ efficiently?

DIRECT METHODS: Gaussian Elimination, LU

- Black-box, most user-friendly
- Robust, commonly used in practice
- Issue: computational cost is expensive, not efficient!

Solution of large systems of equations

Solving linear system of equations:

Given $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ and $\boldsymbol{b} \in \mathbb{R}^{N}$, how to solve $\boldsymbol{Ax} = \boldsymbol{b}$ efficiently?

DIRECT METHODS: Gaussian Elimination, LU

- Black-box, most user-friendly
- Robust, commonly used in practice
- Issue: computational cost is expensive, not efficient!

$$-\Delta u = f \xrightarrow{\text{FE/FV/FD discretization}} \boldsymbol{Ax} = \boldsymbol{b}$$

Computational cost of Gaussian elimination / LU: $\mathcal{O}(N^3)$

$\boldsymbol{A} \in \mathbb{R}^{N\times N}$	$N = 2^{20}$ $\approx 1.0 \times 10^6$	$N = 2^{24}$ $\approx 1.6 \times 10^7$	$N = 2^{28}$ $\approx 2.6 \times 10^8$	$N = 2^{32}$ $\approx 4.3 \times 10^9$
CPU time	3.8 sec	4.4 hrs	2.0 yrs	83.4 c

Solution of large systems of equations

Solving linear system of equations:

Given $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ and $\boldsymbol{b} \in \mathbb{R}^{N}$, how to solve $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ efficiently?

Efficiency means (nearly) optimal computational complexity
$\mathcal{O}(N \log^{\alpha} N)$, $\alpha = 0, 1$

CLASSICAL ITERATIVE METHODS:

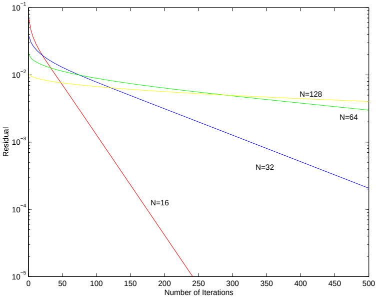
- Jacobi, Gauss-Seidel, etc...
- each iteration is cheap: computational cost is $\mathcal{O}(N)$ (sparse $\boldsymbol{A}$)
- converges if and only if $\rho < 1$
  ($\rho \equiv$ spectral radius of the iteration matrix)

Solution of large systems of equations: GS Convergence.

$$-\Delta u = f \text{ on } [0,1] \times [0,1] \xrightarrow{\text{FE/FV/FD discretization}} \boldsymbol{Ax} = \boldsymbol{b}$$

$$\rho = 1 - \mathcal{O}(h^2)$$

	$N = 16$	$N = 32$	$N = 64$	$N = 128$
it (ρ)	241 (0.9662)	833 (0.9910)	2938 (0.9977)	10405 (0.9994)



Basic linear iterative method

- may converge slowly and the convergence speed may depend on N

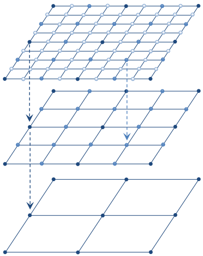
Solution of large systems of equations

Solving linear system of equations:

Given $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ and $\boldsymbol{b} \in \mathbb{R}^{N}$, how to solve $\boldsymbol{Ax} = \boldsymbol{b}$ efficiently?

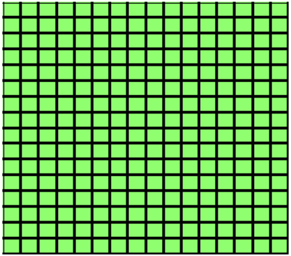
MULTIGRID METHODS

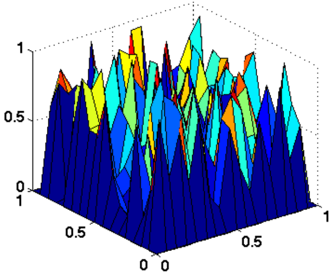
- Robust: Convergence independent of the discretization parameters
- Eficient: Optimal computational cost



Idea of Multigrid

$$-\Delta u = f \text{ on } [0,1]\times[0,1] \xrightarrow{\text{FE/FV/FD discretization}} \boldsymbol{Ax} = \boldsymbol{b}$$

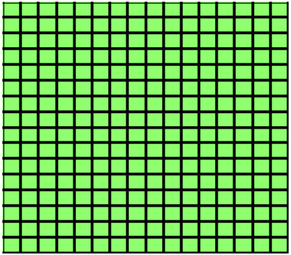


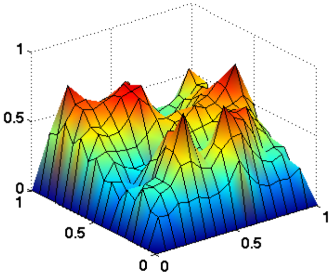


Initial error: $\boldsymbol{e}^0 = \boldsymbol{x} - \boldsymbol{x}^0$

Idea of Multigrid

$$-\Delta u = f \text{ on } [0,1] \times [0,1] \xrightarrow{\text{FE/FV/FD discretization}} \boldsymbol{Ax} = \boldsymbol{b}$$

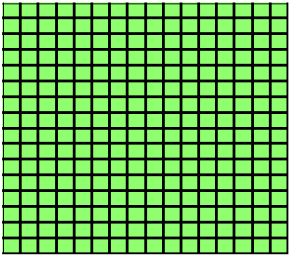


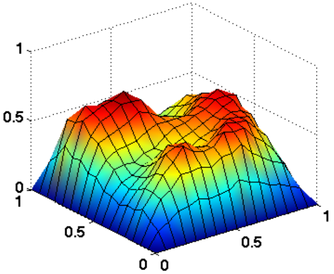


Error after 1 step of Gauss-Seidel

Idea of Multigrid

$$-\Delta u = f \text{ on } [0,1]\times[0,1] \xrightarrow{\text{FE/FV/FD discretization}} \boldsymbol{Ax} = \boldsymbol{b}$$

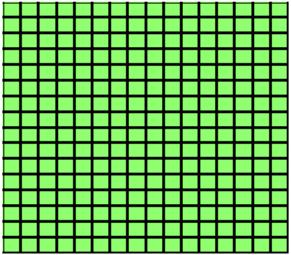


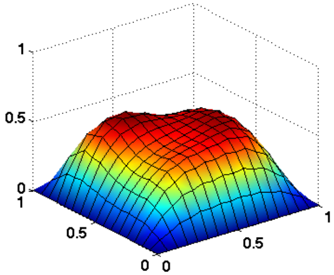


Error after 2 steps of Gauss-Seidel

Idea of Multigrid

$$-\Delta u = f \text{ on } [0,1] \times [0,1] \xrightarrow{\text{FE/FV/FD discretization}} \boldsymbol{Ax} = \boldsymbol{b}$$

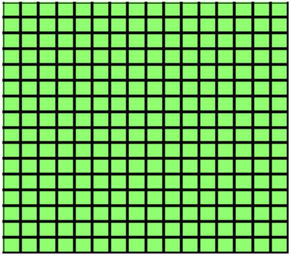


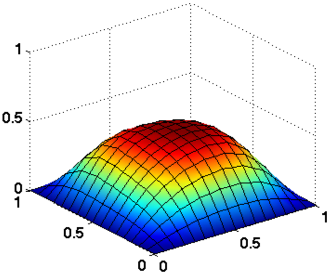


Error after 5 steps of Gauss-Seidel

Idea of Multigrid

$$-\Delta u = f \text{ on } [0,1] \times [0,1] \xrightarrow{\text{FE/FV/FD discretization}} \boldsymbol{Ax} = \boldsymbol{b}$$

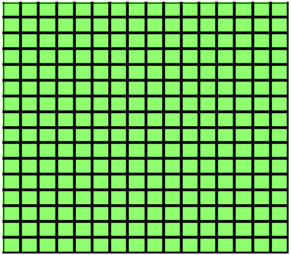


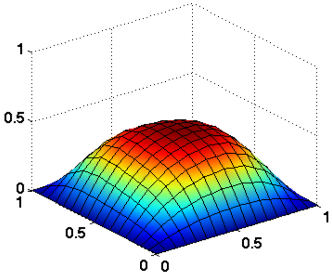


Error after 10 steps Gauss-Seidel

Idea of Multigrid

$$-\Delta u = f \text{ on } [0,1] \times [0,1] \xrightarrow{\text{FE/FV/FD discretization}} \boldsymbol{Ax} = \boldsymbol{b}$$

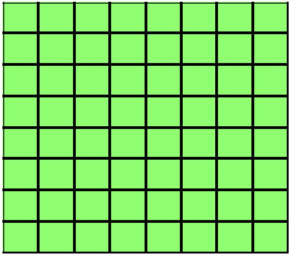


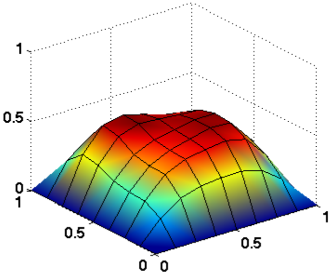


Error after 20 steps Gauss-Seidel

Idea of Multigrid

$$-\Delta u = f \text{ on } [0,1] \times [0,1] \xrightarrow{\text{FE/FV/FD discretization}} \boldsymbol{Ax} = \boldsymbol{b}$$

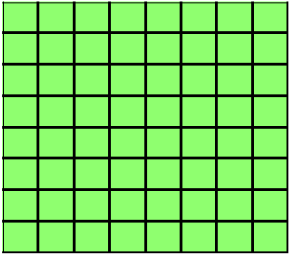


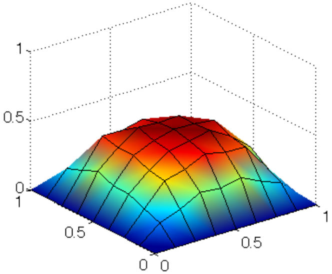


Go to coarse level (after 5 steps Gauss-Seidel)

Idea of Multigrid

$$-\Delta u = f \text{ on } [0,1]\times[0,1] \xrightarrow{\text{FE/FV/FD discretization}} \boldsymbol{Ax} = \boldsymbol{b}$$

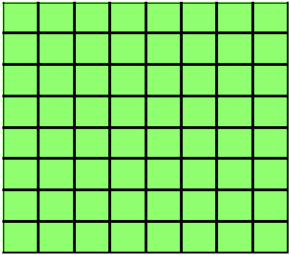


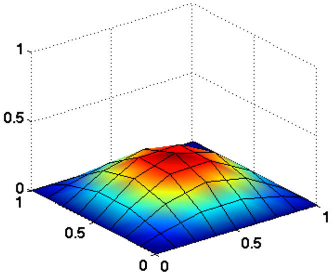


Error after 1 step of Gauss-Seidel on coarse grid

Idea of Multigrid

$$-\Delta u = f \text{ on } [0,1]\times[0,1] \xrightarrow{\text{FE/FV/FD discretization}} \boldsymbol{Ax} = \boldsymbol{b}$$

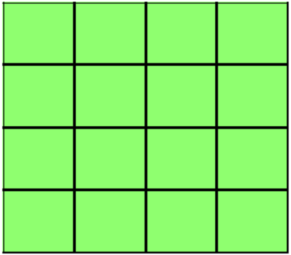


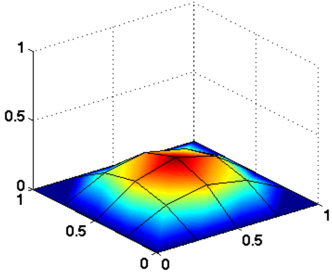


Error after 5 steps of Gauss-Seidel on coarse grid

Idea of Multigrid

$$-\Delta u = f \text{ on } [0,1] \times [0,1] \xrightarrow{\text{FE/FV/FD discretization}} \boldsymbol{Ax} = \boldsymbol{b}$$

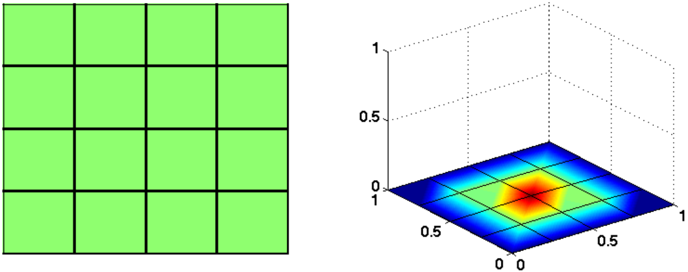




Go to coarser level

Idea of Multigrid

$$-\Delta u = f \text{ on } [0,1] \times [0,1] \xrightarrow{\text{FE/FV/FD discretization}} \boldsymbol{Ax} = \boldsymbol{b}$$

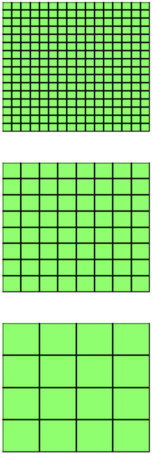


Error after coarse grid correction

Geometric Multigrid Method: $-\Delta u = f \Longrightarrow \boldsymbol{Ax} = \boldsymbol{b}$

Two main components:

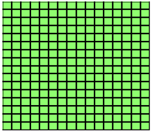
- smoothers (Jacobi, Gauss-Seidel, SOR, ...)
- transfer operators (prolongation, restriction)

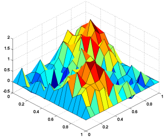


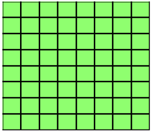
Geometric Multigrid Method: $-\Delta u = f \Longrightarrow \boldsymbol{Ax} = \boldsymbol{b}$

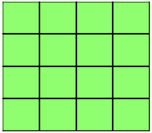
Two main components:

- smoothers (Jacobi, Gauss-Seidel, SOR, ...)
- transfer operators (prolongation, restriction)





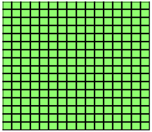


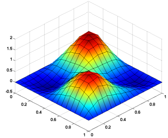


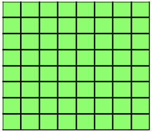
Geometric Multigrid Method: $-\Delta u = f \Longrightarrow \boldsymbol{Ax} = \boldsymbol{b}$

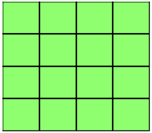
Two main components:

- smoothers (Jacobi, Gauss-Seidel, SOR, ...)
- transfer operators (prolongation, restriction)





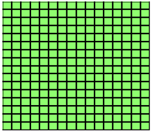


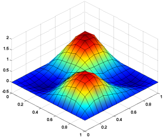


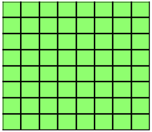
Geometric Multigrid Method: $-\Delta u = f \Longrightarrow \boldsymbol{Ax} = \boldsymbol{b}$

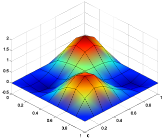
Two main components:

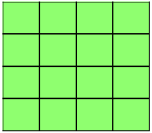
- smoothers (Jacobi, Gauss-Seidel, SOR, ...)
- transfer operators (prolongation, restriction)







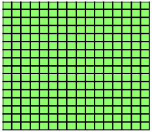


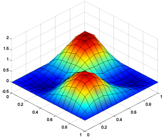


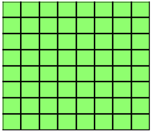
Geometric Multigrid Method: $-\Delta u = f \Longrightarrow \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$

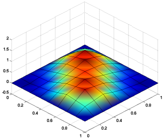
Two main components:

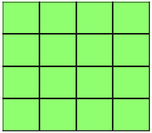
- smoothers (Jacobi, Gauss-Seidel, SOR, ...)
- transfer operators (prolongation, restriction)







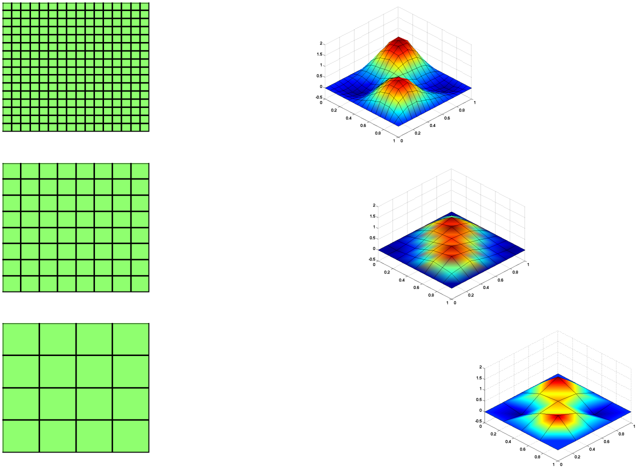




Geometric Multigrid Method: $-\Delta u = f \Longrightarrow \boldsymbol{Ax} = \boldsymbol{b}$

Two main components:

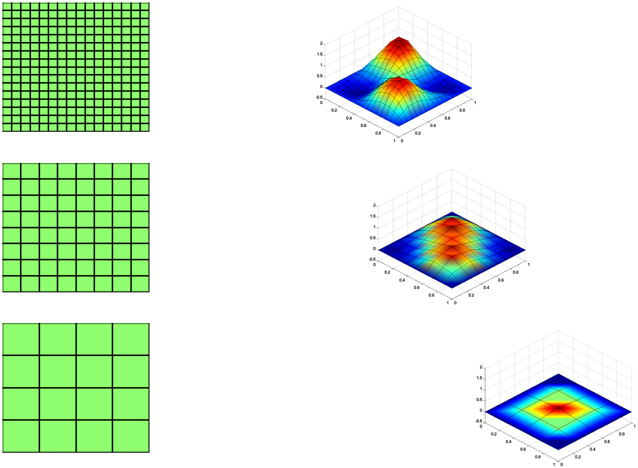
- smoothers (Jacobi, Gauss-Seidel, SOR, ...)
- transfer operators (prolongation, restriction)



Geometric Multigrid Method: $-\Delta u = f \Longrightarrow \boldsymbol{Ax} = \boldsymbol{b}$

Two main components:

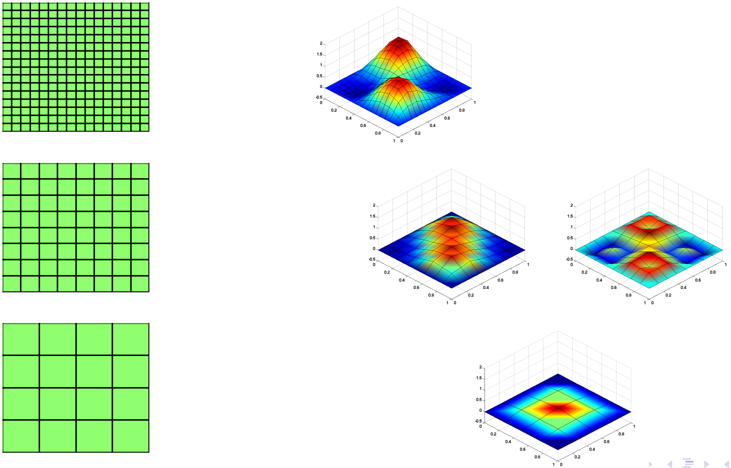
- smoothers (Jacobi, Gauss-Seidel, SOR, ...)
- transfer operators (prolongation, restriction)



Geometric Multigrid Method: $-\Delta u = f \Longrightarrow \boldsymbol{Ax} = \boldsymbol{b}$

Two main components:

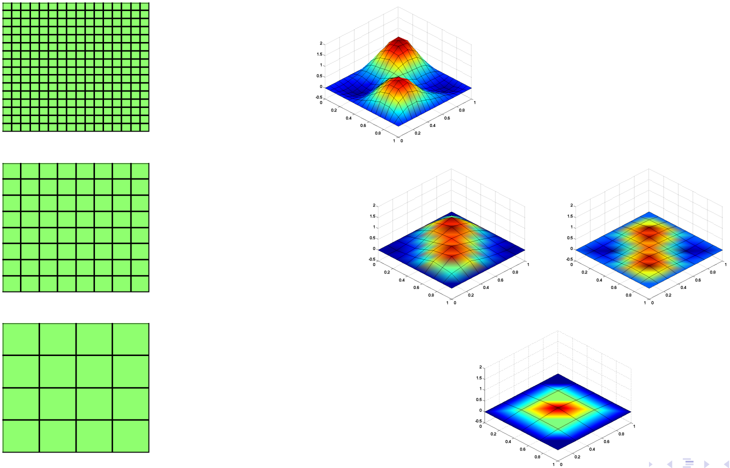
- smoothers (Jacobi, Gauss-Seidel, SOR, ...)
- transfer operators (prolongation, restriction)



Geometric Multigrid Method: $-\Delta u = f \Longrightarrow \boldsymbol{Ax} = \boldsymbol{b}$

Two main components:

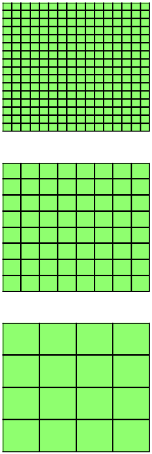
- smoothers (Jacobi, Gauss-Seidel, SOR, ...)
- transfer operators (prolongation, restriction)

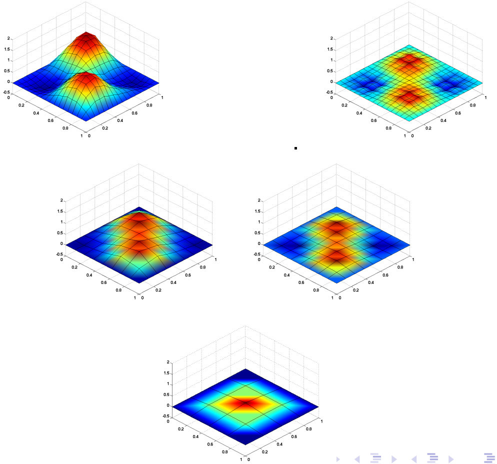


Geometric Multigrid Method: $-\Delta u = f \Longrightarrow \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$

Two main components:

- smoothers (Jacobi, Gauss-Seidel, SOR, ...)
- transfer operators (prolongation, restriction)

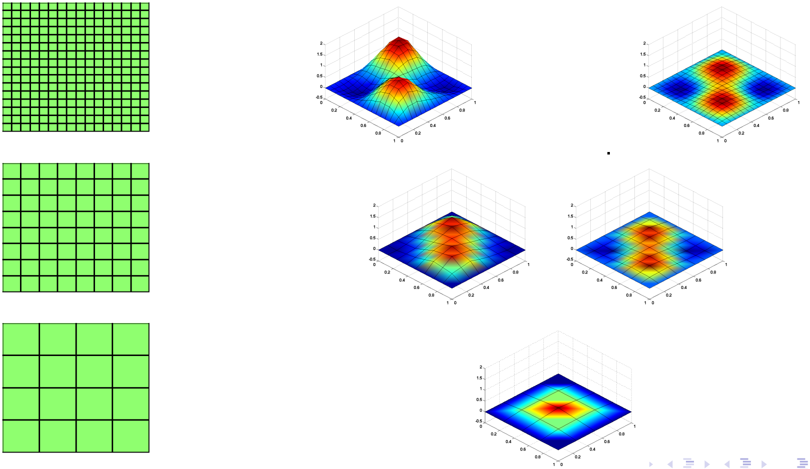




Geometric Multigrid Method: $-\Delta u = f \Longrightarrow \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$

Two main components:

- smoothers (Jacobi, Gauss-Seidel, SOR, ...)
- transfer operators (prolongation, restriction)



One-dimensional problem

1D boundary value problem: $\begin{cases} -u''(x) = f(x), \quad 0 < x < 1, \\ u(0) = u(1) = 0. \end{cases}$

Grid: $\Omega_h = \{x_i = ih,\ i = 0, \ldots, n\}, \quad nh = 1.$

Discrete problem: $\begin{cases} -u_{i-1} + 2u_i - u_{i+1} = h^2 f_i,\ 1 \le i \le n-1, \\ u_0 = u_n = 0. \end{cases}$

$$Au = f, \quad \begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & \ddots & \\ & & & \ddots & \ddots & -1 \\ & & & & -1 & 2 \end{pmatrix} \begin{pmatrix} u_1 \\ \vdots \\ \\ \vdots \\ u_{n-1} \end{pmatrix} = \begin{pmatrix} h^2 f_1 \\ \vdots \\ \\ \vdots \\ h^2 f_{n-1} \end{pmatrix}.$$

Weighted Jacobi iteration

- Eigenvalues of A: $\lambda_k(A) = 4\sin^2\left(\frac{k\pi}{2n}\right),\ 1 \le k \le n-1.$
- Eigenvectors of A: $w_j^k = \sin\left(\frac{jk\pi}{n}\right),\ 1 \le k \le n-1,\ 1 \le j \le n-1.$

Consider the weighted Jacobi iteration:

$$u^{j+1} = R_\omega u^j + \omega D^{-1} f, \quad R_\omega = (1-\omega)I + \omega D^{-1}(L+U), \quad A = D - L - U,$$

$$R_\omega = I - \omega D^{-1} A = I - \frac{\omega}{2} A \ \Rightarrow$$

- Eigenvalues of R_ω: $\lambda_k(R_\omega) = 1 - 2\omega\sin^2\left(\frac{k\pi}{2n}\right),\ 1 \le k \le n-1.$
- Eigenvectors of R_ω: $w_j^k = \sin\left(\frac{jk\pi}{n}\right),\ 1 \le k \le n-1,\ 1 \le j \le n-1.$

For $0 < \omega \le 1,\ |\lambda_k(R_\omega)| < 1,\ \lambda_1(R_\omega) = 1 - 2\omega\sin^2\left(\frac{\pi}{2n}\right) \approx 1 - \frac{\omega\pi^2 h^2}{2}.$

Frequency components. Smoothing factor

Let e^0 be the error in an initial guess:

$$e^0 = \sum_{k=1}^{n-1} c_k w^k \quad \Rightarrow \quad e^m = R_\omega^m e^0 = \sum_{k=1}^{n-1} c_k R_\omega^m w^k = \sum_{k=1}^{n-1} c_k \lambda_k^m(R_\omega) w^k.$$

Low-frequency and high-frequency

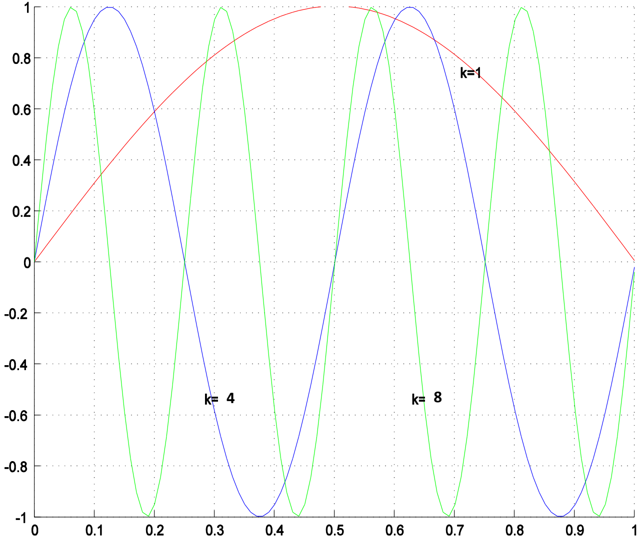
- **Low frequency or smooth mode :** w^k, $1 \leq k < n/2$.
- **High frequency or oscillatory mode:** w^k, $n/2 \leq k \leq n-1$.

Smoothing factor

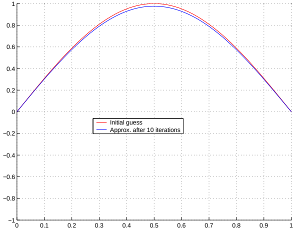
The smoothing factor is the worst factor by which high frequency error components are reduced per relaxation step:

$$\mu(\omega) = \sup\{|\lambda_k(R_\omega)|,\ n/2 \leq k \leq n-1\}.$$

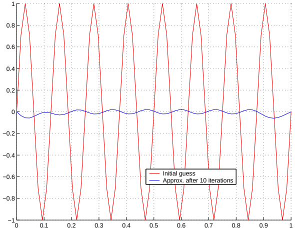
Low and high frequency modes



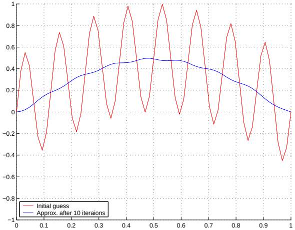
Smoothing property. (n=32).



w^1



w^{16}



$$\frac{w^1 + w^{16}}{2}$$

Frequency components. Smoothing factor

Let e^0 be the error in an initial guess:

$$e^0 = \sum_{k=1}^{n-1} c_k w^k \quad \Rightarrow \quad e^m = R_\omega^m e^0 = \sum_{k=1}^{n-1} c_k R_\omega^m w^k = \sum_{k=1}^{n-1} c_k \lambda_k^m(R_\omega) w^k.$$

Low-frequency and high-frequency

- **Low frequency or smooth mode :** w^k, $1 \le k < n/2$.
- **High frequency or oscillatory mode:** w^k, $n/2 \le k \le n-1$.

Smoothing factor

The smoothing factor is the worst factor by which high frequency error components are reduced per relaxation step:

$$\mu(\omega) = \sup\{|\lambda_k(R_\omega)|,\ n/2 \le k \le n-1\}.$$

Frequency components. Smoothing factor

Let e^0 be the error in an initial guess:

$$e^0 = \sum_{k=1}^{n-1} c_k w^k \quad \Rightarrow \quad e^m = R_\omega^m e^0 = \sum_{k=1}^{n-1} c_k R_\omega^m w^k = \sum_{k=1}^{n-1} c_k \lambda_k^m(R_\omega) w^k.$$

Low-frequency and high-frequency

- **Low frequency or smooth mode :** w^k, $1 \le k < n/2$.
- **High frequency or oscillatory mode:** w^k, $n/2 \le k \le n-1$.

Smoothing factor

The smoothing factor is the worst factor by which high frequency error components are reduced per relaxation step:

$$\mu(\omega) = \sup\{|\lambda_k(R_\omega)|,\ n/2 \le k \le n-1\}.$$

Frequency components. Smoothing factor

Let e^0 be the error in an initial guess:

$$e^0 = \sum_{k=1}^{n-1} c_k w^k \quad \Rightarrow \quad e^m = R_\omega^m e^0 = \sum_{k=1}^{n-1} c_k R_\omega^m w^k = \sum_{k=1}^{n-1} c_k \lambda_k^m(R_\omega) w^k.$$

Low-frequency and high-frequency

- **Low frequency or smooth mode :** $w^k, \ 1 \le k < n/2.$
- **High frequency or oscillatory mode:** $w^k, \ n/2 \le k \le n-1.$

Smoothing factor

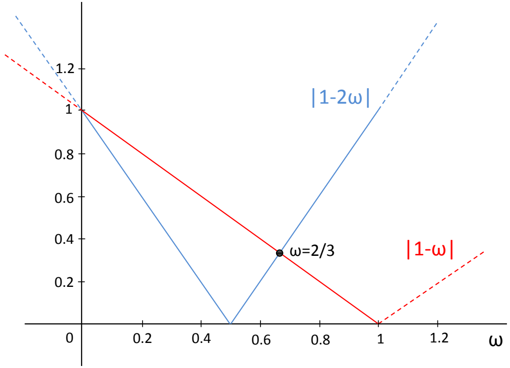
The smoothing factor is the worst factor by which high frequency error components are reduced per relaxation step:

$$\mu(\omega) = \sup\{|\lambda_k(R_\omega)|, \ n/2 \le k \le n-1\}.$$

Smoothing factor for weighted Jacobi

$$\mu(\omega) = \sup\left\{ \left| 1 - 2\omega \sin^2\left(\frac{k\pi}{2n}\right) \right|,\ n/2 \le k \le n-1 \right\}.$$

$$\mu(\omega) = \max\{|1-\omega|, |1-2\omega|\} = \begin{cases} 1, & \omega = 1, \\ 1/3, & \omega = 2/3. \end{cases}$$



Modes on the coarse grid

- Low frequency: kth mode on Ω_h is the kth mode on Ω_{2h}:

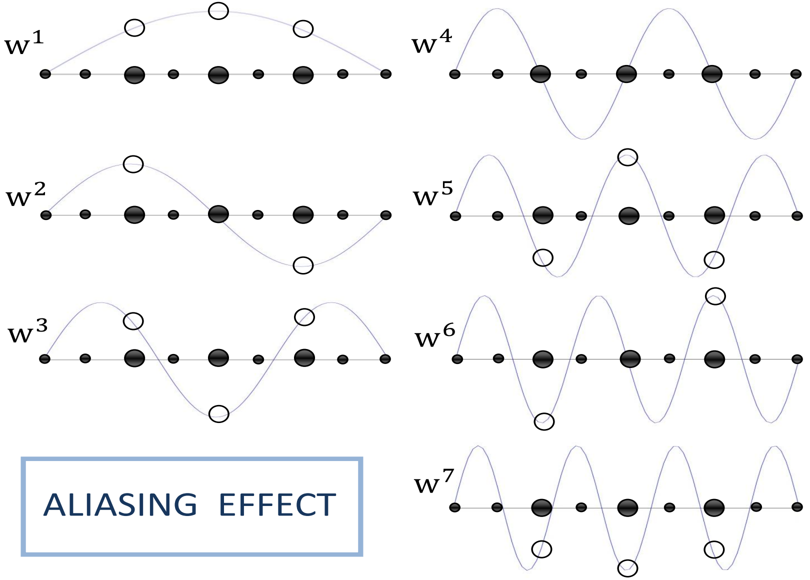
$$w^k_{h,2j} = \sin\left(\frac{2jk\pi}{n}\right) = \sin\left(\frac{jk\pi}{n/2}\right) = w^k_{2h,j}, \quad 1 \le k < n/2.$$

A low-frequency looks more oscillatory on Ω_{2h}.

- The $k = n/2$ mode on Ω_h becomes the zero vector on Ω_{2h}.

- When $k > n/2$: the kth mode on Ω_h becomes the $(n-k)th$ mode on Ω_{2h} $\Rightarrow$ aliasing of frequencies: high frequencies are not visible on the Ω_{2h} grid.

Aliasing of frequencies



ALIASING EFFECT

Coarse grid correction

Discrete linear elliptic boundary value problem: $L_h\, u_h = f_h, \text{ in } \Omega_h$

Let u_h^m be an approximation of the solution u_h, then:

- Error: $e_h^m = u_h - u_h^m$
- Defect or residual: $r_h^m = f_h - L_h\, u_h^m$

Defect equation: $L_h\, e_h^m = r_h^m$

Let Ω_H be a coarser grid $(H > h) \Rightarrow L_H\, \widehat{e_H^m} = r_H^m$

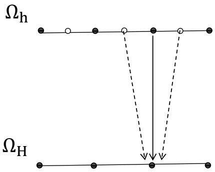
Transfer operators between Ω_h and Ω_H are considered:

- Restriction operator:

$$I_h^H : G(\Omega_h) \to G(\Omega_H)$$

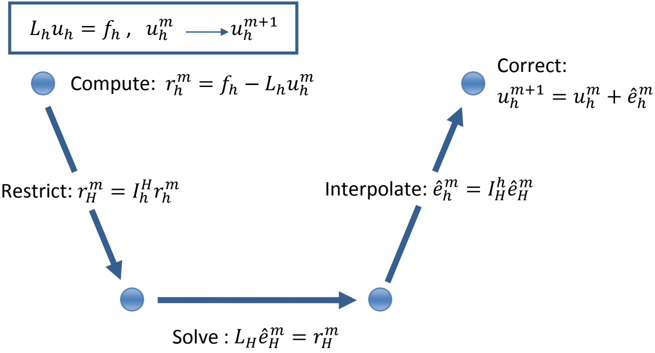
- Interpolation operator:

$$I_H^h : G(\Omega_H) \to G(\Omega_h)$$



Coarse grid correction, $u_h^m \to u_h^{m+1}$

- Compute the defect: $r_h^m = f_h - L_h\, u_h^m$
- Restrict the defect: $r_H^m = I_h^H\, r_h^m$
- Solve exactly the defect equation on Ω_H: $L_H\, \widehat{e}_H^m = r_H^m$
- Interpolate the correction: $\widehat{e}_h^m = I_H^h\, \widehat{e}_H^m$
- Compute a new approximation: $u_h^{m+1} = u_h^m + \widehat{e}_h^m$



Two-grid cycle, $u_h^m \to u_h^{m+1}$

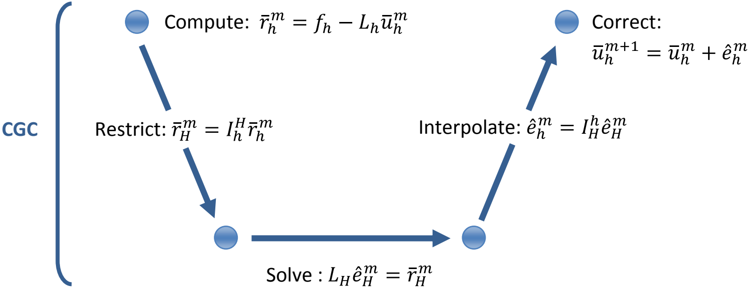
- **Pre-smoothing:**
  - $u_h^m \to \overline{u}_h^m$, by applying ν_1 steps of a given smoothing procedure.
    $$\overline{u}_h^m = S_h^{\nu_1}(u_h^m)$$
- **Coarse grid correction (CGC):**
  - Compute the defect: $\overline{r}_h^m = f_h - L_h \overline{u}_h^m$
  - Restrict the defect: $\overline{r}_H^m = I_h^H \overline{r}_h^m$
  - Solve exactly the defect equation on Ω_H: $L_H \widehat{e}_H^m = \overline{r}_H^m$
  - Interpolate the correction: $\widehat{e}_h^m = I_H^h \widehat{e}_H^m$
  - Compute a new approximation: $\overline{u}_h^{m+1} = \overline{u}_h^m + \widehat{e}_h^m$
- **Post-smoothing:**
  - $\overline{u}_h^{m+1} \to u_h^{m+1}$, by applying ν_2 steps of a given smoothing procedure.
    $$u_h^{m+1} = S_h^{\nu_2}(\overline{u}_h^{m+1})$$

Two-grid cycle

$$L_h u_h = f_h\,, \quad u_h^m \longrightarrow u_h^{m+1}$$

Pre-smoothing (ν_1): $\bar{u}_h^m = S_h^{\nu_1} u_h^m$

Post-smoothing (ν_2): $u_h^{m+1} = S_h^{\nu_2} \bar{u}_h^{m+1}$

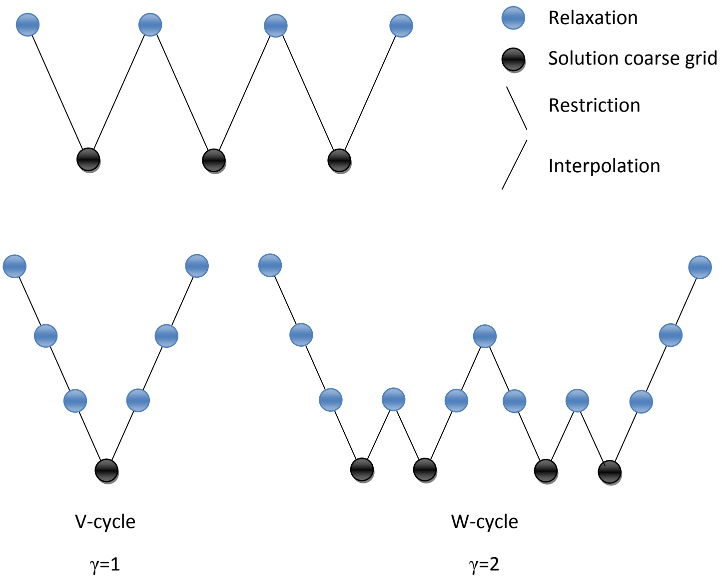


Two-grid iteration operator: $M_h^H = S_h^{\nu_2}[I_h - I_H^h (L_H)^{-1} I_h^H L_h] S_h^{\nu_1}$

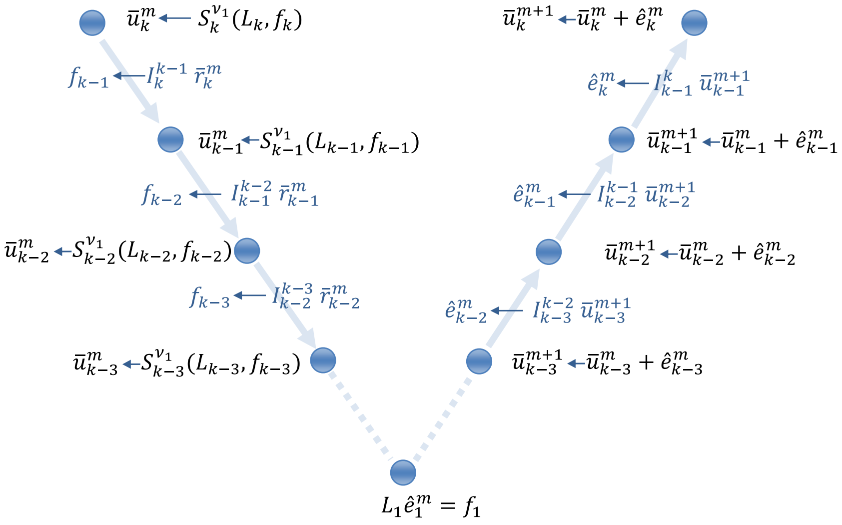
Linear multigrid: $u_k^{m+1} = MG(k, \gamma, L_k, f_k, u_k^m, \nu_1, \nu_2)$

- **Pre-smoothing.** Compute $\bar{u}_k^m$ by applying $\nu_1 (\geq 0)$ smoothing steps to u_k^m.
- **Coarse grid correction:**
  - Compute the defect. $\bar{r}_k^m = f_k - L_k \bar{u}_k^m$.
  - Restrict the defect. $\bar{r}_{k-1}^m = I_k^{k-1} \bar{r}_k^m$.
  - Compute an approximate solution $\widehat{e}_{k-1}^m$ of the defect equation on Ω_{k-1}, $L_{k-1} \widehat{e}_{k-1}^m = \bar{r}_{k-1}^m$ by:
    - If $k = 1$, use a direct or fast iterative solver.
    - If $k > 1$, perform $\gamma \geq 1$ iterations using the zero grid function as a first approximation. $\widehat{e}_{k-1}^m = MG(k-1, \gamma, L_{k-1}, \bar{r}_{k-1}^m, 0_{k-1}, \nu_1, \nu_2)$
  - Interpolate the correction. $\widehat{e}_k^m = I_{k-1}^k \widehat{e}_{k-1}^m$.
  - Correction: $\overline{u}_k^{m+1} = \overline{u}_k^m + \widehat{e}_k^m$.
- **Post-smoothing.** Compute u_k^{m+1} by applying $\nu_2 (\geq 0)$ smoothing steps to $\overline{u}_k^{m+1}$.

Type of cycles in multigrid



V-cycle

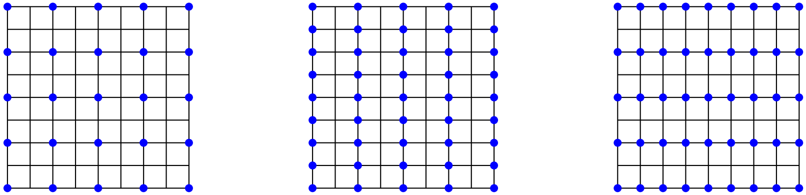

$\bar{u}_k^m \longleftarrow S_k^{\nu_1}(L_k, f_k)$
$f_{k-1} \longleftarrow I_k^{k-1} \bar{r}_k^m$
$\bar{u}_{k-1}^m \leftarrow S_{k-1}^{\nu_1}(L_{k-1}, f_{k-1})$
$f_{k-2} \longleftarrow I_{k-1}^{k-2} \bar{r}_{k-1}^m$
$\bar{u}_{k-2}^m \leftarrow S_{k-2}^{\nu_1}(L_{k-2}, f_{k-2})$
$f_{k-3} \longleftarrow I_{k-2}^{k-3} \bar{r}_{k-2}^m$
$\bar{u}_{k-3}^m \leftarrow S_{k-3}^{\nu_1}(L_{k-3}, f_{k-3})$
$L_1 \hat{e}_1^m = f_1$
$\bar{u}_{k-3}^{m+1} \leftarrow \bar{u}_{k-3}^m + \hat{e}_{k-3}^m$
$\hat{e}_{k-2}^m \longleftarrow I_{k-3}^{k-2} \bar{u}_{k-3}^{m+1}$
$\bar{u}_{k-2}^{m+1} \leftarrow \bar{u}_{k-2}^m + \hat{e}_{k-2}^m$
$\hat{e}_{k-1}^m \longleftarrow I_{k-2}^{k-1} \bar{u}_{k-2}^{m+1}$
$\bar{u}_{k-1}^{m+1} \leftarrow \bar{u}_{k-1}^m + \hat{e}_{k-1}^m$
$\hat{e}_k^m \longleftarrow I_{k-1}^{k} \bar{u}_{k-1}^{m+1}$
$\bar{u}_k^{m+1} \leftarrow \bar{u}_k^m + \hat{e}_k^m$

Components of multigrid

The following individual components of the multigrid method have to be specified:

- The smoothing procedure.
- The numbers ν_1, ν_2 of smoothing steps.
- The coarse grid Ω_H.
- The fine-to-coarse restriction operator I_h^H.
- The coarse grid operator L_H.
- The coarse-to-fine interpolation operator I_H^h.

The coarse grid Ω_H and the coarse grid operator L_H



Standard coarsening | x-semi coarsening | y-semi coarsening

- DCA (Direct coarse aproximation): the direct analog of L_h on the coarse grid Ω_H.
- GCA (Galerkin coarse approximation, $L_H = I_h^H L_h I_H^h$): discontinuous coefficients. Algebraic multigrid methods.

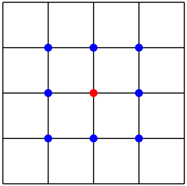
The transfer operators I_h^H and I_H^h

Restriction operator:

Full weighting (FW) operator is frequently used:

We denote the operator by using a stencil [ ] :

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$



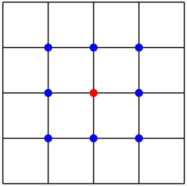
Centered over a coarse point (•) it shows what fractions of the neighboring (•) fine points value is contributed to the value at the coarse point.

Prolongation operator:

Bilinear interpolation is frequently used:

We denote the operator by using a stencil] [:

$$\frac{1}{4}\left]\begin{matrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{matrix}\right[$$

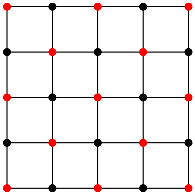


Centered over a coarse-point (•) it shows what fraction of the coarse points value is contributed to neighboring fine points (•).

The smoothing procedure

- Classical iterative methods: Jacobi, Gauss-Seidel, ...
- Take into account:
  - Ordering in which we visit the grid points: Lexicographic order, pattern color, ...
  - Pointwise or blockwise (with or without overlapping) smoother.
  - Damping or relaxation parameter.

Example: Gauss-Seidel Red-Black. (A grid point (x_i, y_j) is red/black if $(i + j)$ is odd/even).



Numerical experiment: Model problem

Model problem: Discrete Poisson equation in 2D with Dirichlet boundary conditions.

Let Ω be the unit square, $\Omega = (0,1)^2 \subset \mathbb{R}^2$.

Model Problem

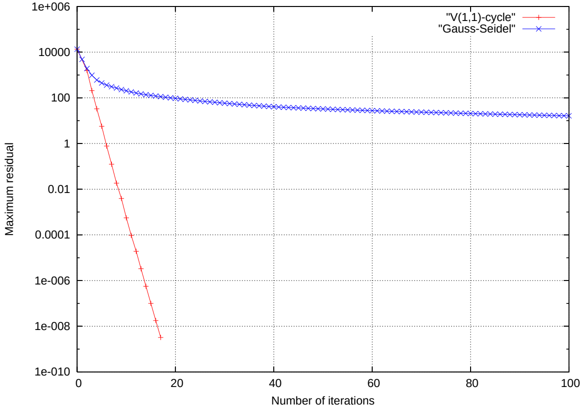
$$\begin{aligned} L_h\, u_h(x,y) &= f_h(x,y), \quad (x,y) \in \Omega_h, \\ u_h(x,y) &= 0, \quad (x,y) \in \partial\Omega_h, \end{aligned}$$

where L_h is the standard 5-point approximation of the operator $-\Delta$, given by

$$L_h\, u_h(x,y) = \frac{1}{h^2} \begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix}_h u_h(x,y).$$

Numerical experiment: Iterative Methods vs Multigrid

Convergence of a lexicographic Gauss-Seidel against a Multigrid method with the same iterative method as smoother.



Grid: 64×64

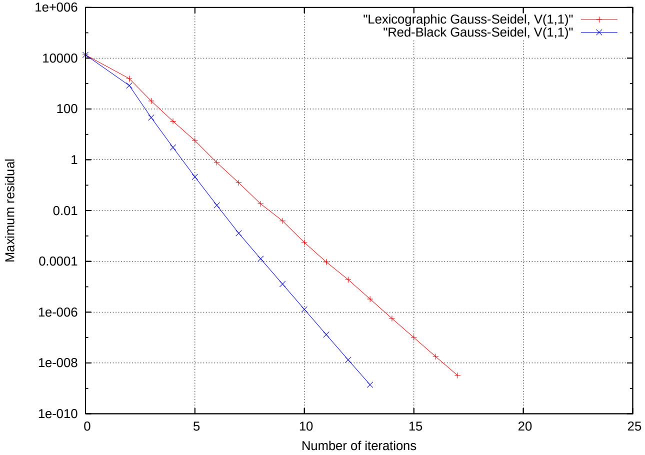
Multigrid features:

- 6 grid levels.
- FW and linear interpolation.
- Smoother: Lexicographic Gauss-Seidel.

Numerical experiment: Influence of the smoother

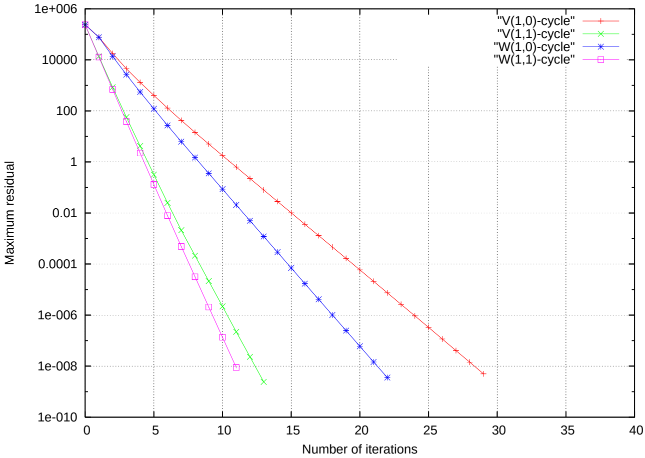
Lexicographic Gauss-Seidel vs Red-Black Gauss-Seidel.

V(1,1)-cycle, Grid: 64×64



Numerical experiment: Influence of the type of cycle

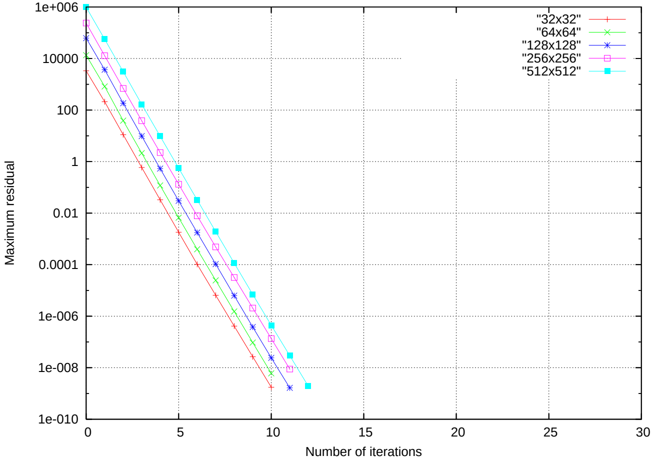
Multigrid convergence for different types of cycles: V-cycle, W-cycle, with different number of pre- and post- smoothing steps.



- Grid: 64×64, (6 grid levels)
- FW and linear interpolation.
- Smoother: Red-Black Gauss-Seidel.

Numerical experiment: h-independent convergence

V(1,1)-multigrid convergence for Red-Black Gauss-Seidel with different numbers of refinement levels.



Numerical experiment: Numerical efficiency of Multigrid

CPU-time and number of iterations necessary to reach a stopping criterion of $\| r^m \| \leq 10^{-10} \| r^0 \|$ with Red-Black smoother, for different cycles and different grids.

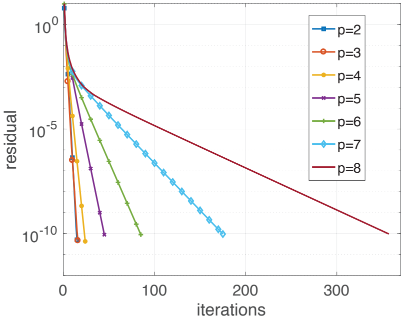
Cycle \ grid	128×128	256×256	512×512	1024×1024
V(1,0)	21 ($< 1''$)	21 ($1''$)	21 ($5''$)	21 ($20''$)
V(1,1)	9 ($< 1''$)	9 ($1''$)	9 ($3''$)	9 ($12''$)
V(2,1)	8 ($< 1''$)	8 ($1''$)	8 ($4''$)	8 ($15''$)
V(2,2)	7 ($< 1''$)	7 ($1''$)	7 ($3''$)	7 ($15''$)
F(1,0)	16 ($< 1''$)	16 ($2''$)	16 ($4''$)	16 ($19''$)
F(1,1)	9 ($< 1''$)	9 ($1''$)	9 ($4''$)	9 ($14''$)
F(2,1)	7 ($< 1''$)	7 ($1''$)	7 ($3''$)	7 ($15''$)
F(2,2)	7 ($< 1''$)	7 ($1''$)	7 ($4''$)	7 ($17''$)

Deterioration of the performance of “standard” multigrid

HIGH-ORDER DISCRETIZATIONS

- Condition number of the system matrix grows exponentially with the polynomial degree
- Naive application of multigrid methods results in an important deterioration of the convergence when the polynomial degree is increased.

Multigrid based
on Gauss-Seidel smoother



- Need of smoothers robust with respect to the polynomial degree

Deterioration of the performance of "standard" multigrid

SADDLE-POINT TYPE PROBLEMS

$$\begin{pmatrix} A & B^T \\ B & -C \end{pmatrix}, \text{ where } C = 0 \text{ or } C \text{ is very small.}$$

Some examples:

- Stokes' problem in fluid dynamics
- Porous media problems: Biot's model for example

Point-wise smoothers does not work properly

Block-wise smoothers - Schwarz DD methods

- Splitting of the grid into blocks of unknowns
- One smoothing step consists of solving local problems on each block one-by-one either in a Jacobi-type or Gauss-Seidel-type manner.

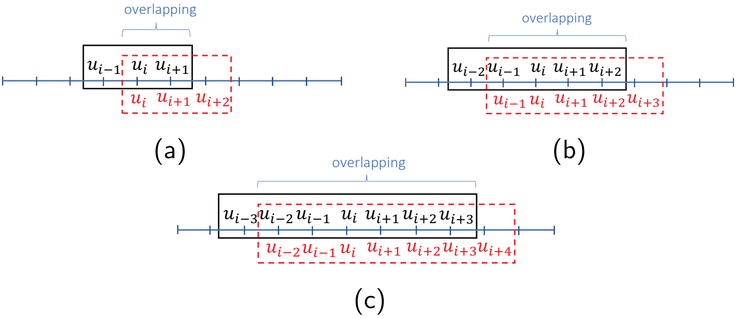


Figure: Splittings of a one-dimensional grid into: (a) blocks of three points, (b) blocks of five points and (c) blocks of seven points, all of them with maximum overlapping among the blocks.

Schwarz domain decomposition methods

- For every block of unknowns B_i, the corresponding local problem is set as follows: Let u_{B_i} be the vector with unknowns involved in block B_i, δu_{B_i} the local corrections and r_{B_i} the local defect. Then, every local system is given by

$$A^{B_i}\delta u_{B_i} = r_{B_i}, \tag{1}$$

  where A^{B_i} is the local matrix corresponding to block B_i.

- The construction of local matrices A^{B_i} is carried out by means of a projection operator V_{B_i} from the vector of unknowns u to the vector of unknowns involved in the block B_i:

$$A^{B_i} = V_{B_i} A V_{B_i}^T.$$

- One iteration of these smoothers consists of a loop over all the blocks such that the corresponding local systems (1) are solved.

Multiplicative Schwarz methods

- Multiplicative Schwarz methods make use of the most recent updates of u.
- For every local system the defect is obtained after a global update of u with the previous local corrections.

Algorithm 1: $u^{m+1} = \text{Multiplicative}(A, u^m, b, M_B, N_B, s_B)$

---

for $i = 1 : N_B$ **do**

$B_i \leftarrow M_B(i, :)$	*Indices of the unknowns involved in B_i.*
$V_i = 0_{s_B \times n}$	
$V_i(:, B_i) = I_{s_B}$	*Construction of projection operator V_i.*
$A^{B_i} \leftarrow V_i A V_i^T$	*Construction of local matrix.*
$r_{B_i} \leftarrow V_i(b - Au^m)$	*Construction of local defect.*
$A^{B_i} \delta u_{B_i} = r_{B_i}$	*Solve the local system.*
$u^{m+1} = u^m + V_i^T \delta u_{B_i}$	*Local correction of u.*

Additive Schwarz methods

- Additive Schwarz methods solve all the local systems by using the same global defect.
- The approximation is corrected at once by addition of all the local corrections.

Algorithm 2: $u^{m+1} = \text{Additive}(A, u^m, b, M_B, N_B, s_B, \omega)$

$r \leftarrow (b - Au^m)$ *Initial residual.*

for $i = 1 : N_B$ **do**

  $B_i \leftarrow M_B(i, :)$ *Indices of the unknowns involved in B_i.*

  $V_i = 0_{s_B \times n}$

  $V_i(:, B_i) = I_{s_B}$ *Construction of projection operator V_i.*

  $A^{B_i} \leftarrow V_i A V_i^T$ *Construction of local matrix.*

  $r_{B_i} \leftarrow V_i r$ *Construction of local defect.*

  $A^{B_i} \delta u_{B_i} = r_{B_i}$ *Solve the local system.*

$u^{m+1} = u^m + \sum_{i=1}^{N_B} V_i^T \omega \, \delta u_{B_i}$ *Global correction of u.*

Restrictive additive Schwarz methods

- Restrictive additive Schwarz methods omit multiple local corrections of the same grid point.

Algorithm 3: $u^{m+1} = \text{Restrictive}(A, u^m, b, M_B, N_B, s_B, \omega, ov)$

$r \leftarrow (b - Au^m)$ *Initial residual.*
for $i = 1 : N_B$ **do**
  $B_i \leftarrow M_B(i, :)$ *Indices of the unknowns involved in B_i.*
  $V_i = 0_{s_B \times n}$
  $\tilde{V}_i = 0_{s_B \times n}$ *Auxiliary projection operator $\tilde{V}_i$.*
  $V_i(:, B_i) = I_{s_B}$ *Construction of projection operator V_i.*
  $\tilde{V}_i(:, B_i(1 : s_B - ov)) = I_{s_B - ov}$ *ov is overlapping among blocks.*
  $A^{B_i} \leftarrow V_i A V_i^T$ *Construction of local matrix.*
  $r_{B_i} \leftarrow V_i r$ *Construction of local defect.*
  $A^{B_i} \delta u_{B_i} = r_{B_i}$ *Solve the local system.*

$u^{m+1} = u^m + \sum_{i=1}^{N_B} \tilde{V}_i^T \omega \, \delta u_{B_i}$ *Global correction of u.*

Poisson equation: Multiplicative Schwarz smoothers

First, let us consider the 2D Poisson problem defined in $\Omega = (0,1)^2$ with dirichlet boundary conditions in $\partial\Omega$:

$$\begin{aligned} -\Delta u &= f, \quad \text{in} \quad \Omega, \\ u &= 0, \quad \text{on} \quad \partial\Omega, \end{aligned}$$

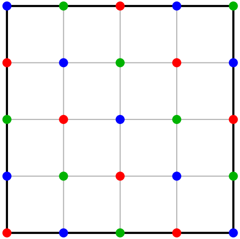
- High-oder finite elements with polynomial degree k.

For this problem, $9(3 \times 3)-$ $25(5 \times 5)-$ and $49(7 \times 7)-$point multiplicative Schwarz smoothers are considered:

	9p Schwarz	25p Schwarz	49p Schwarz
$k = 2$	0.099	0.066	0.050
$k = 3$	0.214	0.067	0.051
$k = 4$	0.455	0.145	0.053
$k = 5$	0.703	0.262	0.119
$k = 6$	0.872	0.440	0.165
$k = 7$	0.955	0.657	0.325
$k = 8$	0.982	0.807	0.440

Poisson equation: Multiplicative Schwarz smoothers

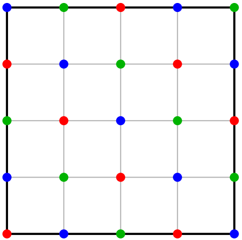
Coloured multiplicative Schwarz relaxations - Improved convergence



Poisson equation: Multiplicative Schwarz smoothers

Coloured multiplicative Schwarz relaxations - Improved convergence

Behavior of the multigrid method based on coloured multiplicative Schwarz smoothers: for a mesh of size 512×512, we display the number of V(1,0)-multigrid iterations together with the CPU times necessary to reduce the initial residual in a factor of 10^{-8}



	9p-Sch		c-9p-Sch		25p-Sch		c-25p-Sch		49p-Sch		c-49p-Sch	
k	it	cpu	it	cpu	it	cpu	it	cpu	it	cpu	it	cpu
2	8	1.29	**4**	**0.68**	7	4.89	4	2.88	6	14.21	3	7.46
3	8	2.13	**4**	**0.99**	6	5.60	4	4.02	6	17.80	3	9.30
4	14	4.66	**7**	**2.40**	7	8.38	4	5.08	6	22.40	4	15.56
5	19	17.26	17	10.40	9	17.83	**4**	**6.09**	6	39.96	3	14.69
6	31	21.53	40	28.40	11	20.01	**5**	**9.33**	7	39.90	3	16.06
7	60	48.04	112	98.70	15	34.32	9	21.80	9	60.44	**3**	**17.15**
8	88	105.62	254	310.55	22	71.19	15	48.87	11	88.17	**4**	**26.01**

Choice of block size depending on the polynomial degree

Poisson equation: Multiplicative Schwarz smoothers

Robust behavior with respect to mesh size h:

Proposed strategy for different grids with mesh-sizes from 128×128 to 1024×1024 and various spline degrees ranging from $k = 2$ to $k = 8$ in order to show the good scalability of the solver. Again, we use $V-$cycles with one pre-smoothing step and no post-smoothing steps

	Color 9p Schwarz						Color 25p Schwarz				Color 49p Schwarz			
	$k = 2$		$k = 3$		$k = 4$		$k = 5$		$k = 6$		$k = 7$		$k = 8$	
Grid	it	cpu	it	cpu	it	cpu	it	cpu	it	cpu	it	cpu	it	cpu
128^2	4	0.06	4	0.10	7	0.22	4	0.62	5	0.99	3	2.22	4	3.35
256^2	4	0.21	4	0.29	7	0.69	4	1.40	5	2.86	3	6.03	4	9.24
512^2	4	0.68	4	0.99	7	2.40	4	6.09	5	9.33	3	17.15	4	26.01
1024^2	4	2.60	4	3.80	7	9.13	3	16.49	5	33.05	3	53.69	4	80.74

Poisson equation: Additive Schwarz smoothers

Additive Schwarz smoother:

Multigrid method based on V(1,1)-cycles using additive Schwarz methods as smoothers with maximum overlapping. In this problem, we consider $9(3 \times 3)$-, $25(5 \times 5)$- and $49(7 \times 7)$-point additive Schwarz smoothers.

	9p ad Schwarz $\omega = 1/9$	25p ad Schwarz $\omega = 1/25$	49p ad Schwarz $\omega = 1/49$
$k = 2$	0.22	0.11	0.06
$k = 3$	0.48	0.27	0.17
$k = 4$	0.72	0.45	0.29
$k = 5$	–	0.62	0.43
$k = 6$	–	0.76	0.56
$k = 7$	–	0.85	0.67
$k = 8$	–	0.96	0.78

Poisson equation: Additive Schwarz smoothers

Element-based Additive Schwarz Smoother

- Size of the blocks is $k = (p+1)^2$
- Minimum overlaping $ov = 1$ is assumed in both directions
- two smoothing steps

	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$	$k=8$
AS	0.15	0.21	0.25	0.31	0.36	0.40	0.43
RAS	0.15	0.21	0.25	0.31	0.36	0.40	0.43

Poisson equation: Additive Schwarz smoothers

In order to achieve a determined performance for the solver, we show the asymptotic convergence factors obtained by using $W(1,0)$, $W(1,1)$, $W(2,1)$ and $W(2,2)$ cycles of this approach:

k	$W(1,0)$	$W(1,1)$	$W(2,1)$	$W(2,2)$
2	0.40	0.15	0.06	0.02
3	0.45	0.21	0.09	0.04
4	0.50	0.25	0.13	0.06
5	0.56	0.31	0.18	0.10
6	0.60	0.36	0.21	0.13
7	0.63	0.40	0.25	0.16
8	0.66	0.43	0.28	0.19

Moreover, the asymptotic convergence factors obtained by applying $V-$cycles were substantially similar to the ones obtained with $W-$cycles.

Saddle-Point problem: Biot's model

Equilibrium equation: $\operatorname{div} \boldsymbol{\sigma}' - \alpha \nabla p = \rho \mathbf{g}, \quad \text{in } \Omega,$

Hooke's law: $\boldsymbol{\sigma}' = \lambda \operatorname{tr}(\boldsymbol{\epsilon}) \mathbf{I} + 2\mu \boldsymbol{\epsilon}, \quad \text{in } \Omega,$

Compatibility equation: $\boldsymbol{\epsilon}(\mathbf{u}) = \frac{1}{2}(\nabla \mathbf{u} + \nabla \mathbf{u}^t), \quad \text{in } \Omega.$

Darcy's law: $$\mathbf{q} = -\frac{\boldsymbol{\kappa}}{\eta_f} \left(\nabla p - \rho_f \mathbf{g} \right), \quad \text{in } \Omega,$$

Continuity equation: $$\frac{\partial}{\partial t} \left(\frac{1}{M} p + \alpha \nabla \cdot \mathbf{u} \right) + \nabla \cdot \mathbf{q} = f, \quad \text{in } \Omega.$$

- λ and μ: Lamé coefficients
- α: Biot-Willis constant and M: Biot modulus
- $\boldsymbol{\kappa}$: Permeability tensor and ρ: bulk density
- η_f: viscosity of the fluid and ρ_f: density of the fluid phase
- $\mathbf{u}$: displacement vector and p: pore pressure
- $\boldsymbol{\sigma}'$ and $\boldsymbol{\epsilon}$: effective stress and strain tensors.
- f: a forced fluid extraction or injection process and g: gravity vector

Saddle-Point problem: Two field formulation for Biot

Two-field (displacement-pressure) formulation

$$
\begin{aligned}
-\nabla(\lambda+\mu)\nabla\cdot\boldsymbol{u} - \nabla\cdot\mu\nabla\boldsymbol{u} + \alpha\,\nabla\,p = \rho\boldsymbol{g},\\
\frac{1}{M}\frac{\partial p}{\partial t} + \alpha\frac{\partial}{\partial t}(\nabla\cdot\mathbf{u}) - \nabla\cdot\left(\frac{1}{\mu_f}K(\nabla p - \rho_f\boldsymbol{g})\right) = f.
\end{aligned}
$$

We complete the formulation of this problem with the following boundary conditions:

$$
\begin{aligned}
\boldsymbol{u} = \mathbf{0}, \quad \frac{K}{\mu_f}(\nabla p - \rho_f\boldsymbol{g})\cdot\boldsymbol{n} = 0, \quad \text{on } \Gamma_c,\\
p = 0, \quad \boldsymbol{\sigma}'\cdot\boldsymbol{n} = \mathbf{0}, \quad \text{on } \Gamma_t,
\end{aligned}
$$

where $\boldsymbol{n}$ is the unit outward normal to the boundary and $\partial\Omega = \Gamma_t \cup \Gamma_c$.
Also, we consider the initial condition

$$
\left(\frac{1}{M}p + \alpha\nabla\cdot\boldsymbol{u}\right)(\boldsymbol{x},0) = 0, \quad \text{on } \boldsymbol{\Omega}.
$$

Saddle-Point problem: Variational formulation

Let us introduce the variational spaces

$$\boldsymbol{V} = \{\boldsymbol{u} \in H^1(\Omega)^n \mid \boldsymbol{u}|_{\Gamma_c} = \boldsymbol{0}\}, \quad Q = \{p \in H^1(\Omega) \mid p|_{\Gamma_t} = 0\}.$$

Then, the variational formulation is given as follows

Variational formulation

For each $t \in (0, T)$, find $(\boldsymbol{u}(t), p(t)) \in \boldsymbol{V} \times Q$ such that:

$$a(\boldsymbol{u}, \boldsymbol{v}) - \alpha(p, \mathrm{div}\boldsymbol{v}) = (\rho \boldsymbol{g}, \boldsymbol{v}), \forall\ \boldsymbol{v} \in \boldsymbol{V},$$

$$\alpha(\mathrm{div}\ \partial_t \boldsymbol{u}, q) + \frac{1}{M}(\partial_t p, q) + b(p, q) = (f, q) + (\boldsymbol{K}\mu_f^{-1}\rho_f \boldsymbol{g}, \nabla q), \forall\ q \in Q$$

where the bilinear forms $a(\cdot,\cdot)$ and $b(\cdot,\cdot)$ are given by

$$\begin{aligned} a(\boldsymbol{u}, \boldsymbol{v}) &= 2\int_\Omega \mu \epsilon(\boldsymbol{u}) : \epsilon(\boldsymbol{v}) d\Omega + \int_\Omega \lambda \mathrm{div}\boldsymbol{u}\ \mathrm{div}\boldsymbol{v} d\Omega, \\ b(p, q) &= \int_\Omega \frac{\boldsymbol{K}}{\mu_f} \nabla p \cdot \nabla q d\Omega. \end{aligned}$$

Saddle-Point problem: Discrete scheme

- Implicit Euler scheme for discretization in time
- Taylor-Hood elements $(\boldsymbol{V}_h, Q_h)$ for discretization in space.

Fully discretized scheme at time t_m

Find $(\boldsymbol{u}_h^m, p_h^m) \in \boldsymbol{V}_h \times Q_h$ such that

$$a(\boldsymbol{u}_h^m, \boldsymbol{v}_h) - \alpha(p_h^m, \operatorname{div} \boldsymbol{v}_h) = (\rho \boldsymbol{g}, \boldsymbol{v}_h), \quad \forall\ \boldsymbol{v}_h \in \boldsymbol{V}_h,$$

$$\alpha(\operatorname{div} \bar{\partial}_t \boldsymbol{u}_h^m, q_h) + \frac{1}{M}(\bar{\partial}_t p_h^m, q_h) + b(p_h^m, q_h) = (f_h^m, q_h) + (\boldsymbol{K} \mu_f^{-1} \rho_f \boldsymbol{g}, \nabla q_h), \forall\ q_h \in Q_h,$$

where $\bar{\partial}_t \boldsymbol{u}_h^m := (\boldsymbol{u}_h^m - \boldsymbol{u}_h^{m-1})/\tau$, $\bar{\partial}_t p_h^m := (p_h^m - p_h^{m-1})/\tau$,

After discretization, for solving the problem on each time step we have the following saddle point problem

$$\begin{bmatrix} E & B^T \\ -B & \tau C + \frac{1}{M} M_p \end{bmatrix} \begin{bmatrix} \mathbf{u}^m \\ p^m \end{bmatrix} = \begin{bmatrix} g^m \\ f^m \end{bmatrix}.$$

Saddle-Point problem: Schwarz smoother

SCHWARZ TYPE SMOOTHER for Biot:

- Blocks consists of one pressure unknown and all the velocity unknowns that are connected to it.
- Local system of size 51×51 and 76×76 for two-dimensional and three-dimensional problems

High computational cost ➡ Additive Schwarz smoothers

Saddle-Point problem: Schwarz smoother

SCHWARZ TYPE SMOOTHER for Biot:

- Blocks consists of one pressure unknown and all the velocity unknowns that are connected to it.
- Local system of size 51×51 and 76×76 for two-dimensional and three-dimensional problems

High computational cost ➡ Additive Schwarz smoothers

κ	$W(1,0)$	$W(1,1)$	$W(2,1)$	$W(2,2)$
1	0.49	0.22	0.11	0.06
10^{-3}	0.49	0.21	0.11	0.06
10^{-6}	0.49	0.21	0.11	0.06
10^{-9}	0.65	0.42	0.27	0.18
10^{-12}	0.72	0.52	0.56	0.28
10^{-15}	0.72	0.52	0.56	0.28

Table: Asymptotic convergence factors considering natural weights for the $Q2-Q1$ discretization on the unit square domain.

Saddle-Point problem: Schwarz smoother

We can find the optimal weights that provide a robust monolithic multigrid method based on the 51−point additive Schwarz smoother:

- $w_u^{opt} = 0.09$ for displacements at vertices and edges
- $w_u^{opt} = 0.22$ for displacements at interior points
- $w_p^{opt} = 1.02$ for pressure

κ	$W(1,0)$	$W(1,1)$	$W(2,1)$	$W(2,2)$
1	0.58	0.34	0.19	0.11
10^{-3}	0.58	0.34	0.19	0.11
10^{-6}	0.58	0.34	0.19	0.11
10^{-9}	0.58	0.34	0.19	0.11
10^{-12}	0.60	0.36	0.21	0.13
10^{-15}	0.60	0.36	0.21	0.13

Table: Asymptotic convergence factors considering optimal weights w_u^{opt} and w_p^{opt} for the $Q2 - Q1$ discretization on the unit square domain.

Saddle-Point problem: Schwarz smoother

Furthermore, we compared the performance of $V-$cycles and $W-$cycles:

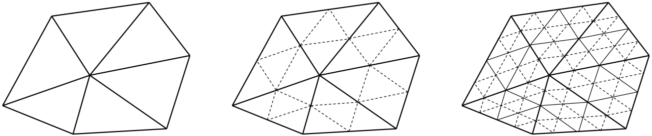
	Smoothing steps (ν_1, ν_2)							
	$(1,0)$		$(1,1)$		$(2,1)$		$(2,2)$	
κ	V	W	V	W	V	W	V	W
1	39	40	20	20	13	14	10	10
10^{-3}	40	40	20	20	14	13	10	10
10^{-6}	39	40	20	20	13	14	10	10
10^{-9}	40	40	23	20	14	13	10	10
10^{-12}	50	43	40	22	18	15	12	12
10^{-15}	50	45	40	23	18	15	12	12

Table: Number of iterations using the optimal weights w_u^{opt}, w_p^{opt}.

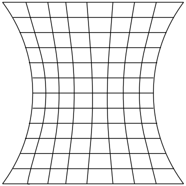
We conclude that $V-$cycles are also robust with respect to κ when applying $V(2,1)$ or $V(2,2)-$cycles yielding a similar number of iterations than $W-$cycles.

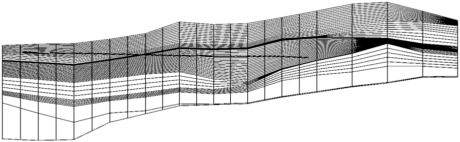
Extension to complex domains

- SEMI-STRUCTURED GRIDS



- LOGICALLY RECTANGULAR GRIDS





- ISOGEOMETRIC ANALYSIS (IGA)

References

- A. Brandt, “Multi-level Adaptive Solutions to Boundary Value Problems”, Math. Comp., 31, 1977, pp 333-390.
- W. Briggs, V. Henson and S. McCormick, “A multigrid tutorial”. Siam, 2000.
- U. Trottemberg, C. Oosterlee and A. Schüller, “Multigrid”, Academic Press, 2000.
- P. Wesseling, “An introduction to Multigrid methods”, Wiley, 2004.
- W. Hackbusch, “Multi-Grid Methods and Applications”, Springer, 2003.
- R. Wienands and W. Joppich, “Practical Fourier Analysis for Multigrid Methods”, Chapman 2005.
- Y. Saad, “Iterative Methods for Sparse Linear Systems”, 2nd edition, SIAM, 2003.
- X.-C Cai, M. Sarkis, “A restricted additive Schwarz preconditioner”, SIAM Journal on Scientific Computing, 21(2), 792-797, 1999.
- A. Pé de la Riva, C. Rodrigo, F.J. Gaspar, “A Robust Multigrid Solver for Isogeometric Analysis Based on Multiplicative Schwarz Smoothers”, SIAM Journal on Scientific Computing, 41 (5), pp. S321-S345, 2019.
- A. Pé de la Riva, C. Rodrigo, F.J. Gaspar, J. Adler, X. Hu, L. Zikatanov, “A local Fourier analysis for additive Schwarz smoothers”, Computers & Mathematics with Applications, 158 (15), pp. 13-20, 2024.

Thank you for your attention!!