Coarse Space Correction and Preconditioning

Martin J. Gander
martin.gander@unige.ch

University of Geneva

DD Summer School, St. John's, July 22nd, 2018

On the Meaning of Optimal

- best or most favorable (Oxford Dictionaries)
- best or most effective in a particular situation (Cambridge Dictionary)
- most desirable or satisfactory (Merriam-Webster)

On the Meaning of Optimal

- best or most favorable (Oxford Dictionaries)
- best or most effective in a particular situation (Cambridge Dictionary)
- most desirable or satisfactory (Merriam-Webster)

Optimality in Domain Decomposition:

Definition 1.2 (Optimality). *An iterative method for the solution of a linear system is said to be optimal, if its rate of convergence to the exact solution is independent of the size of the system.*

$\Longrightarrow$ "optimal" in the classical Domain Decomposition literature does not mean there are no faster methods.

On the Meaning of Optimal

- best or most favorable (Oxford Dictionaries)
- best or most effective in a particular situation (Cambridge Dictionary)
- most desirable or satisfactory (Merriam-Webster)

Optimality in Domain Decomposition:

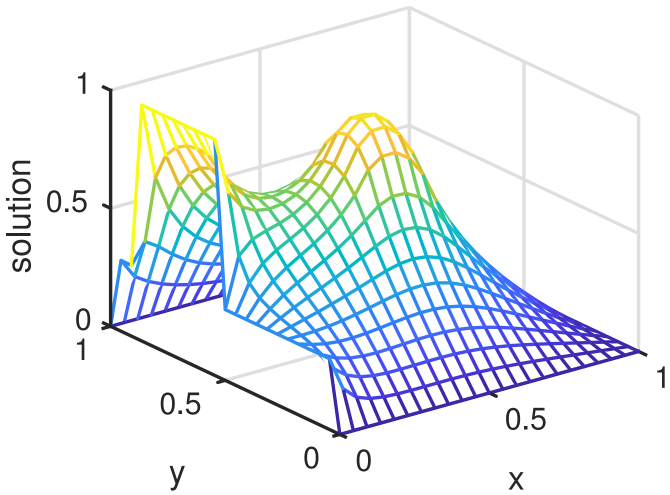
Definition 1.2 (Optimality). *An iterative method for the solution of a linear system is said to be optimal, if its rate of convergence to the exact solution is independent of the size of the system.*

$\Longrightarrow$ "optimal" in the classical Domain Decomposition literature does not mean there are no faster methods.

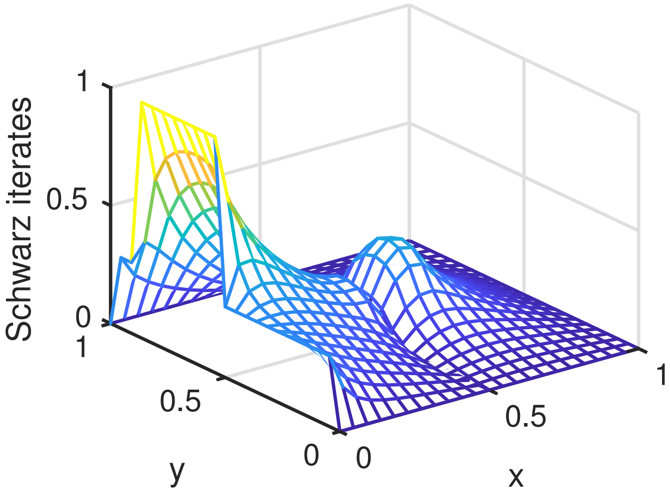
Three ways to increase the size of the linear system:

1. One keeps the domain and decomposition fixed and refines the mesh
2. One keeps the domain fixed, refines the mesh and adds proportionally subdomains
3. One increases the domain by adding subdomains

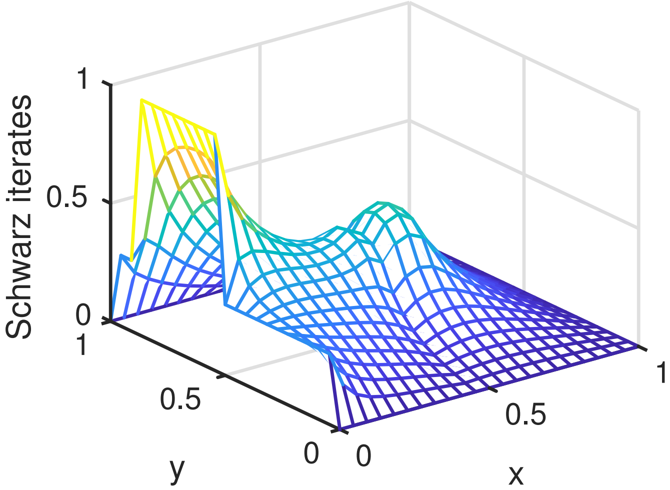
Case 1: Parallel Schwarz for Heating a Room



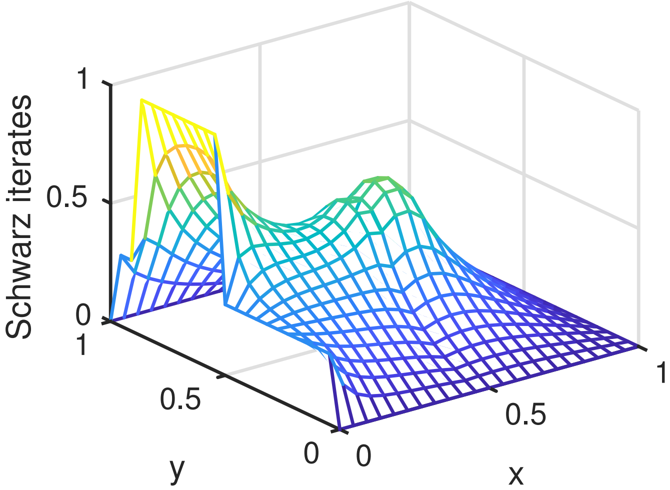
Iteration 1



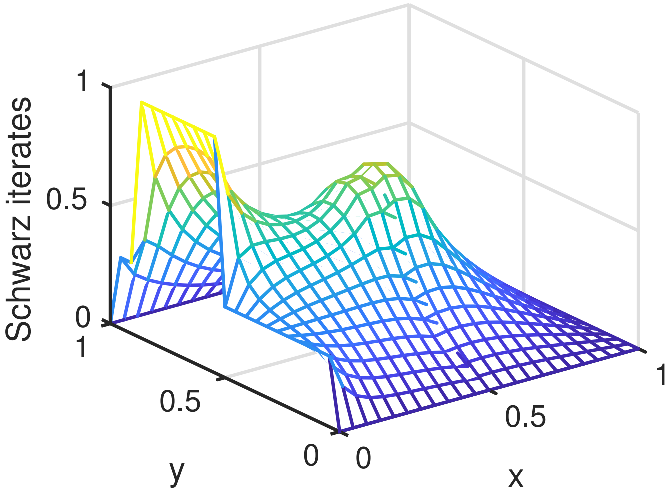
Iteration 2



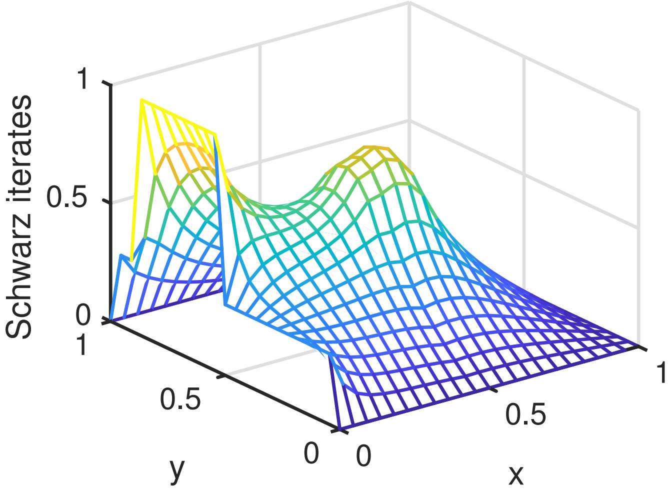
Iteration 3



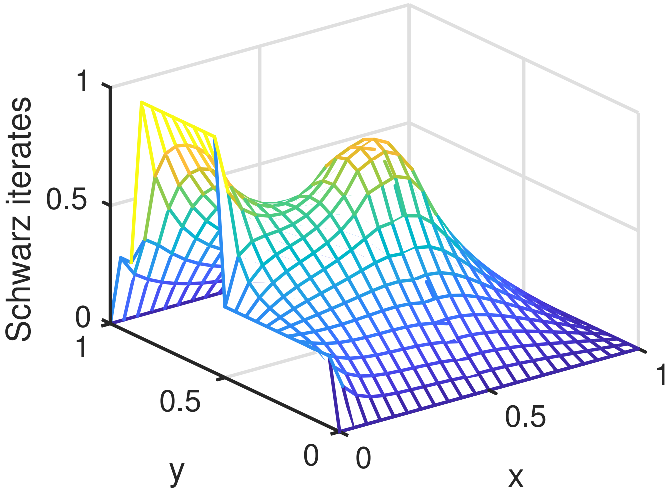
Iteration 4



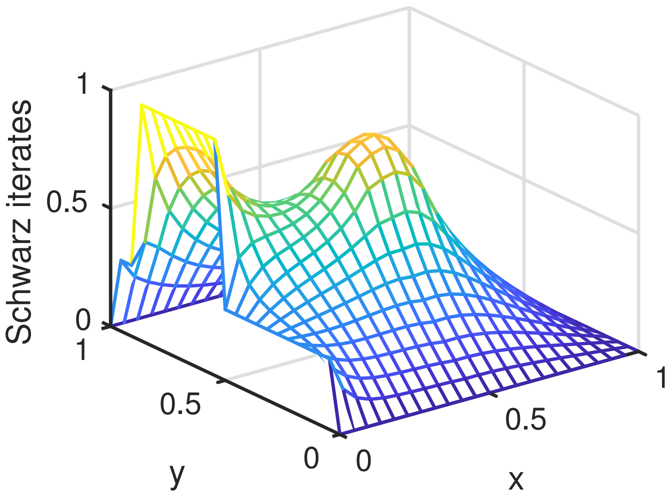
Iteration 5



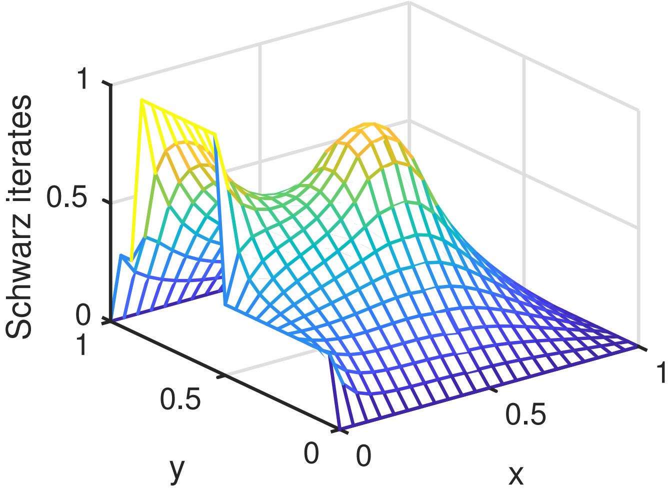
Iteration 6



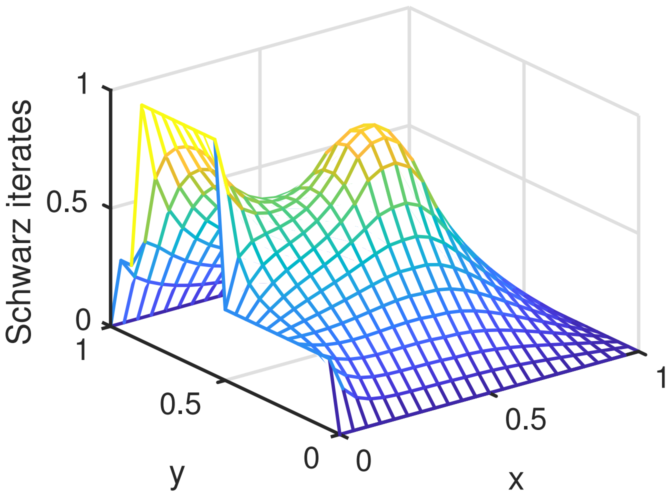
Iteration 7



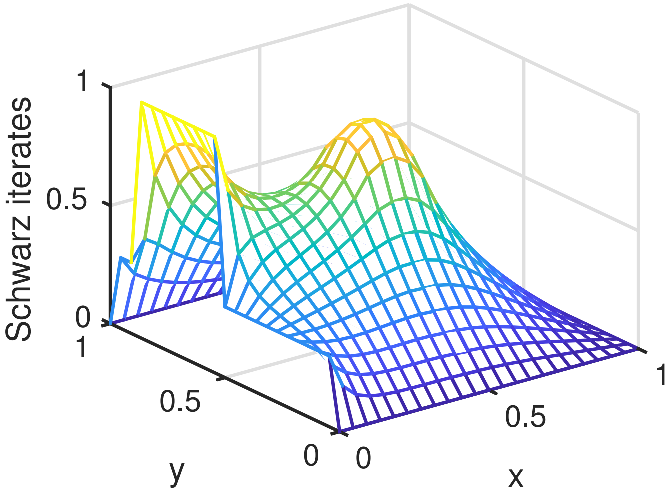
Iteration 8



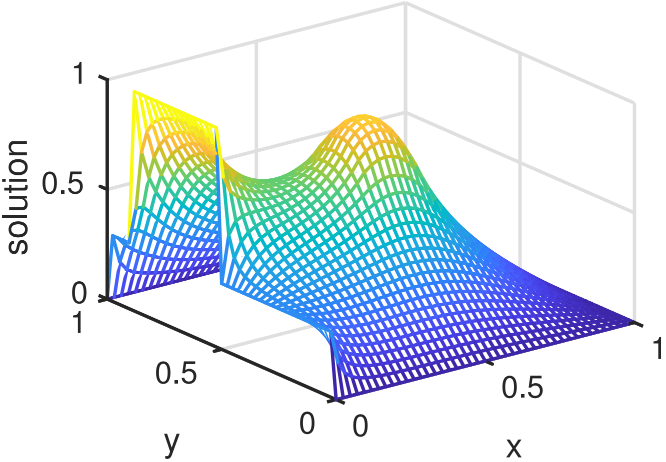
Iteration 9



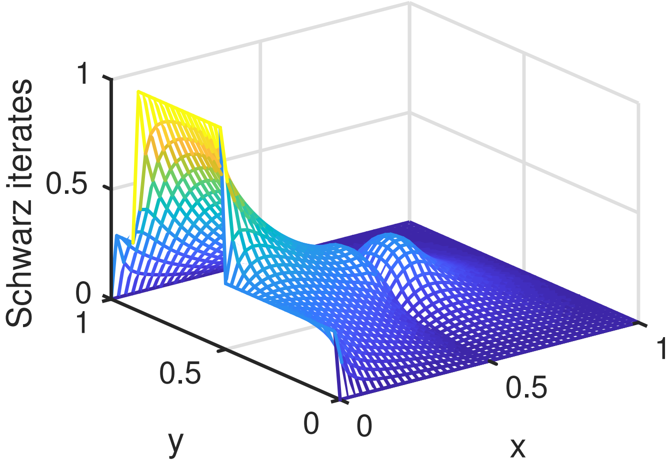
Iteration 10



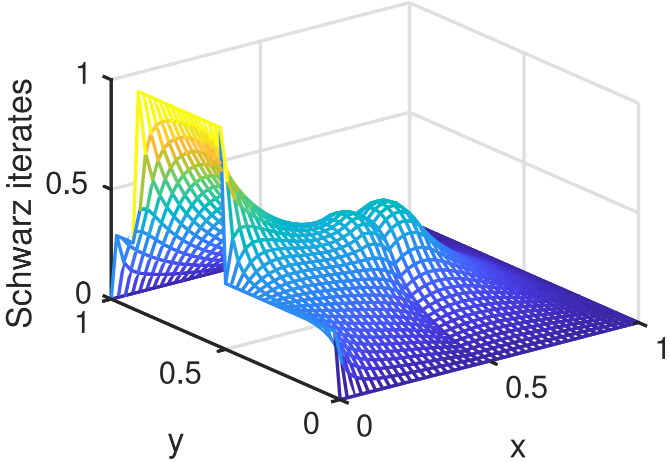
Refined Parallel Schwarz for Heating a Room



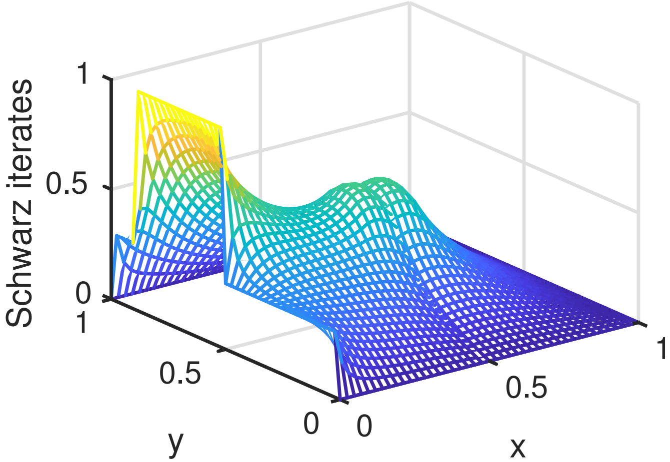
Iteration 1



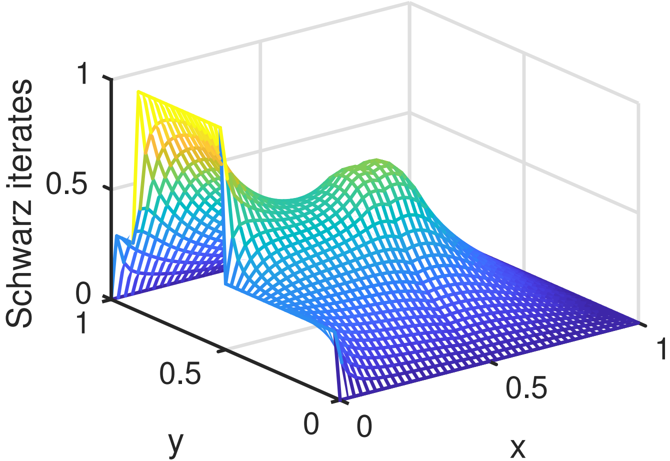
Iteration 2



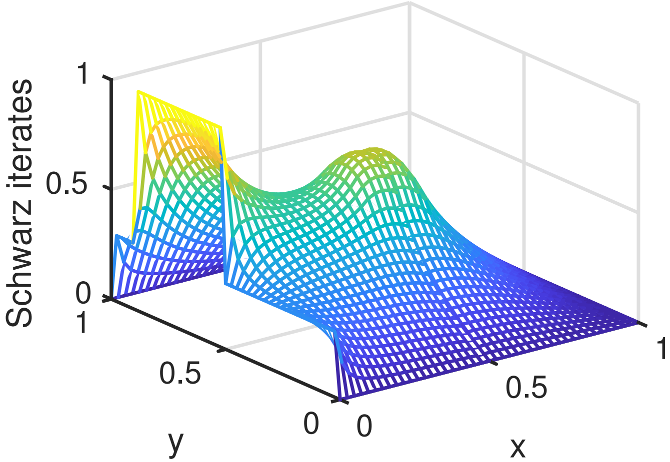
Iteration 3



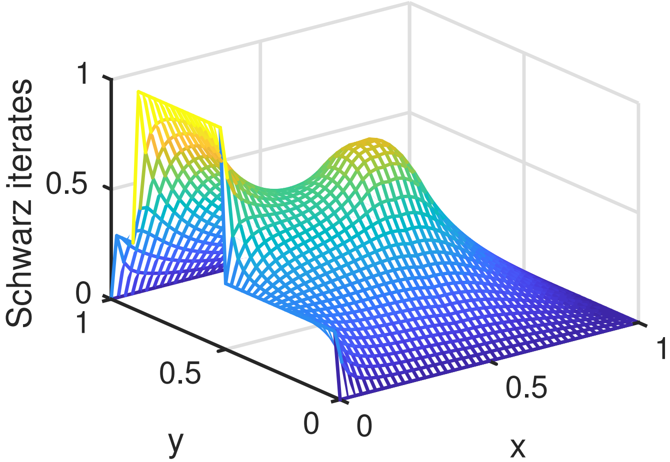
Iteration 4



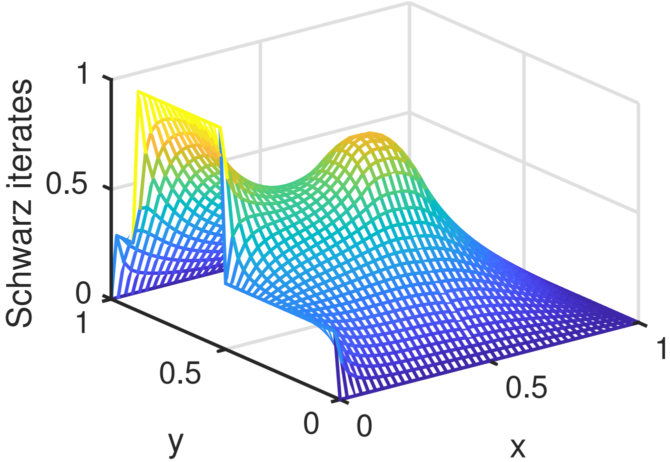
Iteration 5



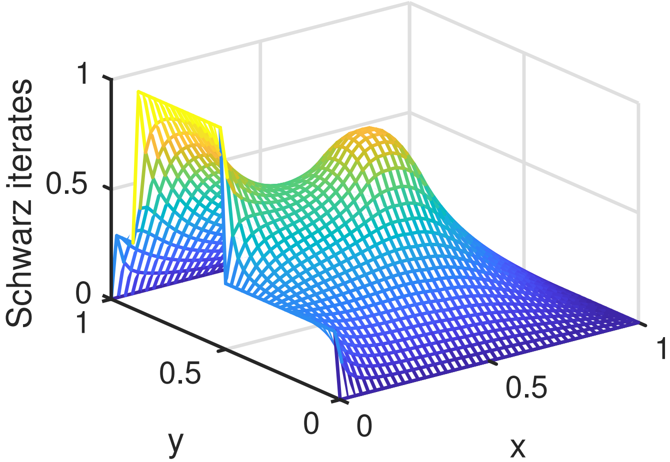
Iteration 6



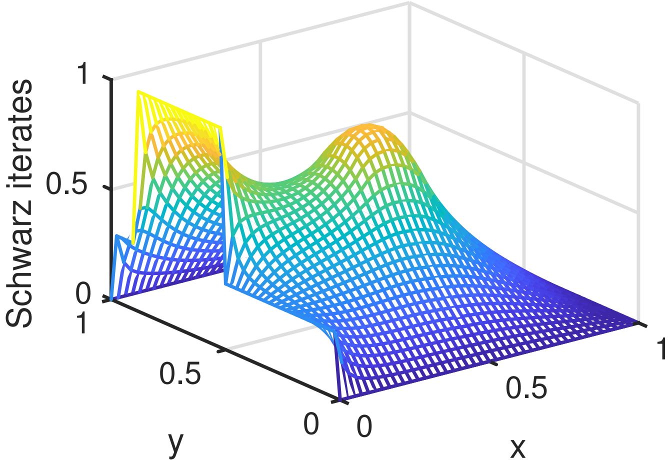
Iteration 7



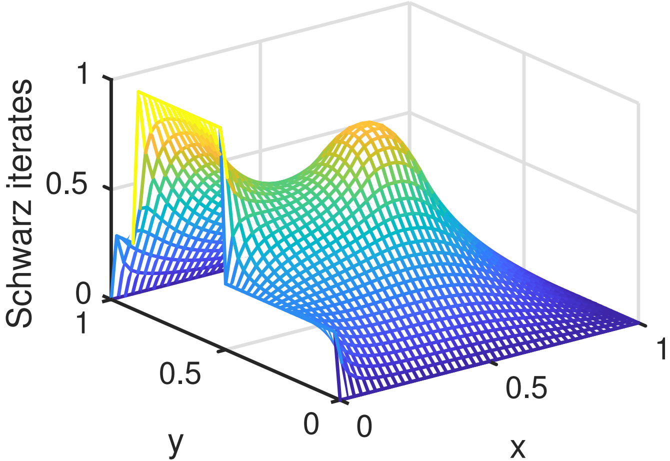
Iteration 8



Iteration 9



Iteration 10



Cases 2-3: Scalability in Domain Decomposition

Want to be "optimal" also when using more and more processors, i.e. more and more subdomains.

Dryja and Widlund (1987):

> *"... we have shown that if we only have next neighbors communication, the minimum number of iterations required grows at least as fast as $N^{\frac{1}{2}}$, where N is the number of substructures."*

Mandel and Brezina (1993):

> *"It is well known that the absence of a coarse problem results in deterioration of convergence of the iteration with increasing number of subdomains."*

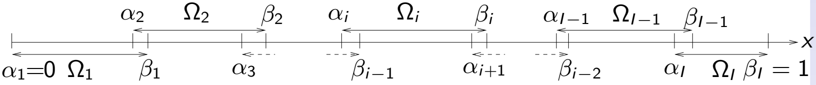
Case 2: when Coarse Spaces are Needed

Study of a model problem:

$$(\eta - \partial_{xx})u = 0, \quad u(0) \text{ and } u(1) \text{ given}$$

Parallel Schwarz method (Lions 1988), equivalent to RAS (Cai and Sarkis 1999) for I subdomains $\Omega_i := (\alpha_i, \beta_i)$:

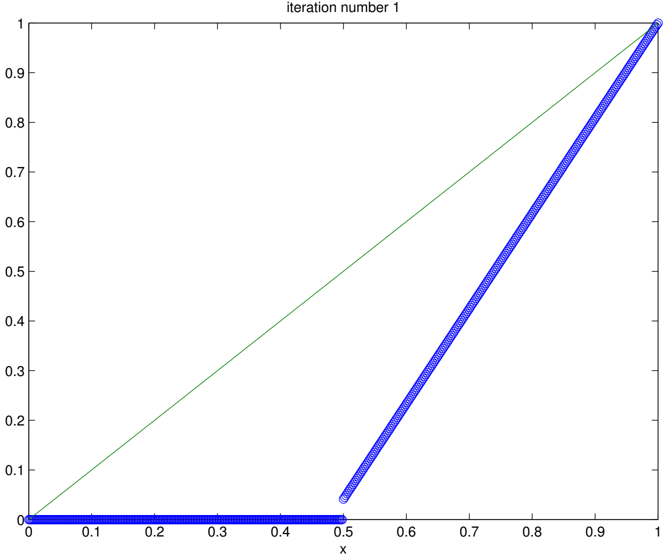
$$\begin{aligned}(\eta - \partial_{xx})u_i^n &= 0 \quad \text{in } \Omega_i,\\ u_i^n(\alpha_i) = u_{i-1}^{n-1}(\alpha_i), &\quad u_i^n(\beta_i) = u_{i+1}^{n-1}(\beta_i),\end{aligned}$$



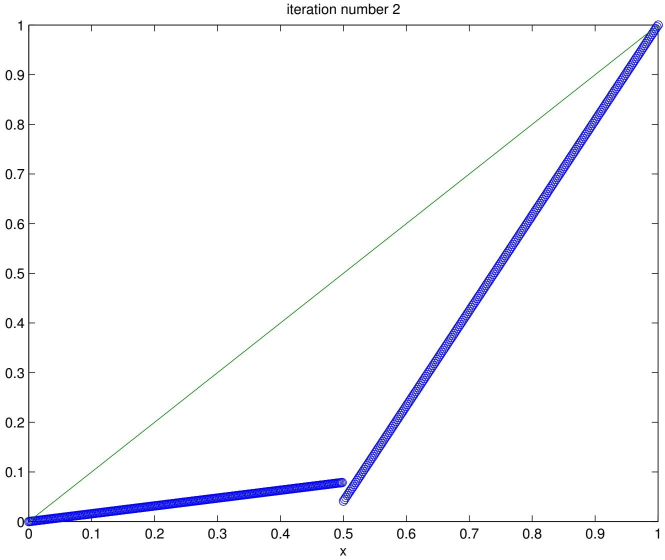
Overlap $L_i := \beta_i - \alpha_{i+1}$, $i = 1..I-1$

Subdomain size $H_i := \beta_i - \alpha_i$, $i = 1..I$

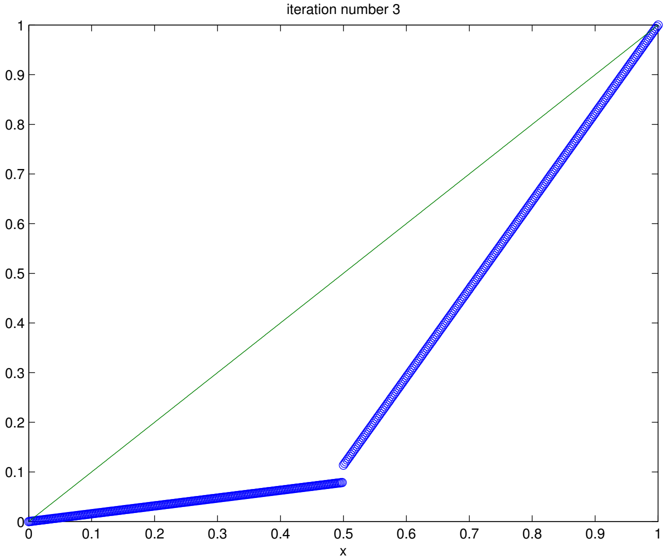
Two Subdomains, Iteration 1



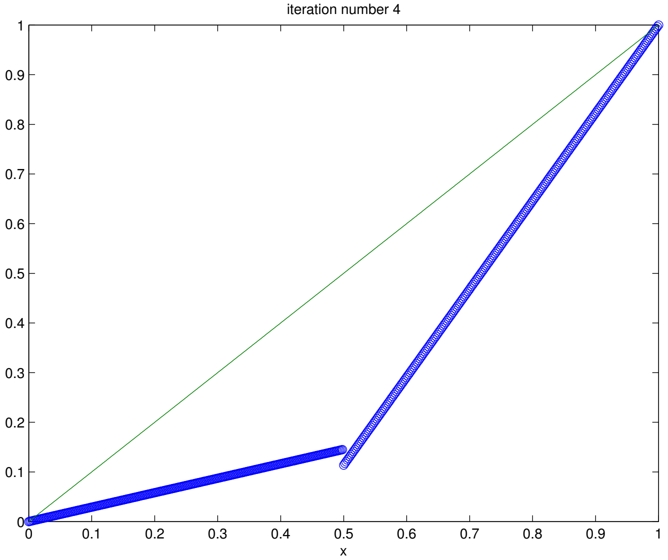
Two Subdomains, Iteration 2



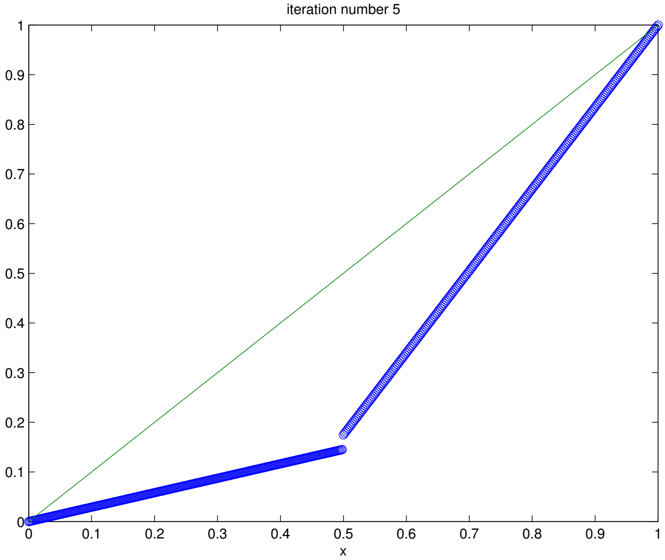
Two Subdomains, Iteration 3



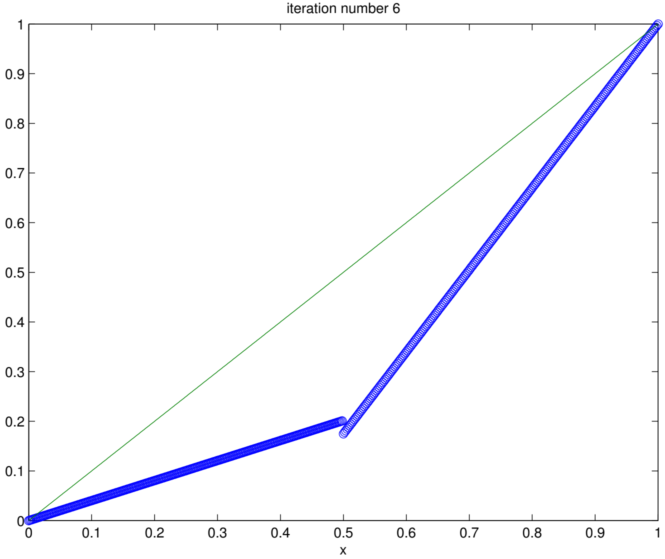
Two Subdomains, Iteration 4



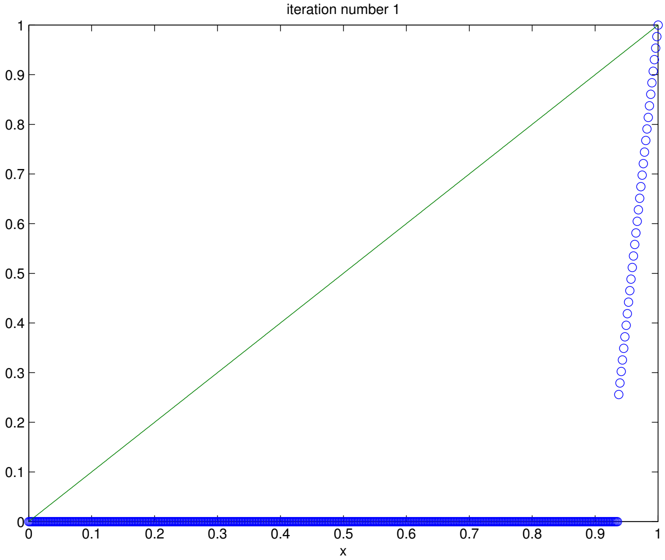
Two Subdomains, Iteration 5



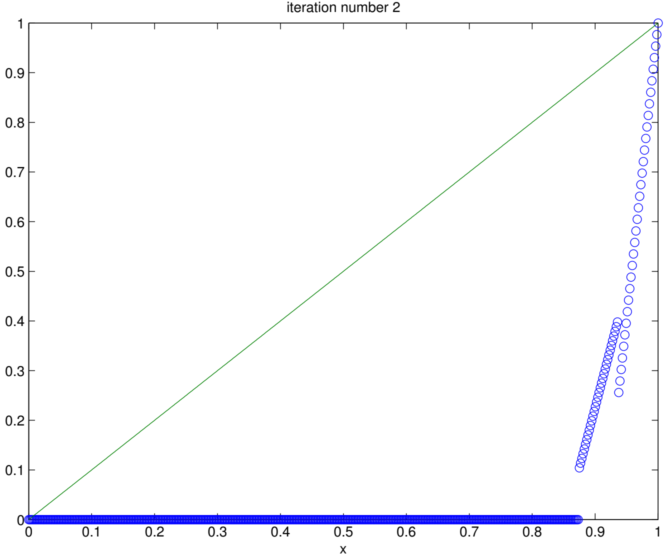
Two Subdomains, Iteration 6



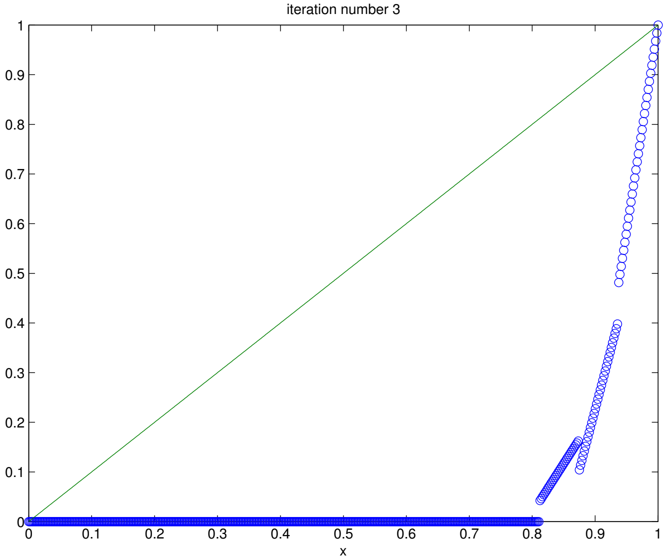
16 Subdomains, Iteration 1



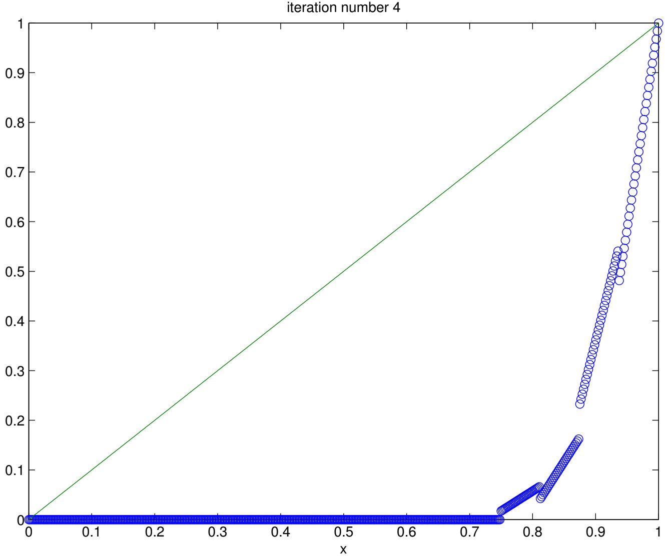
16 Subdomains, Iteration 2



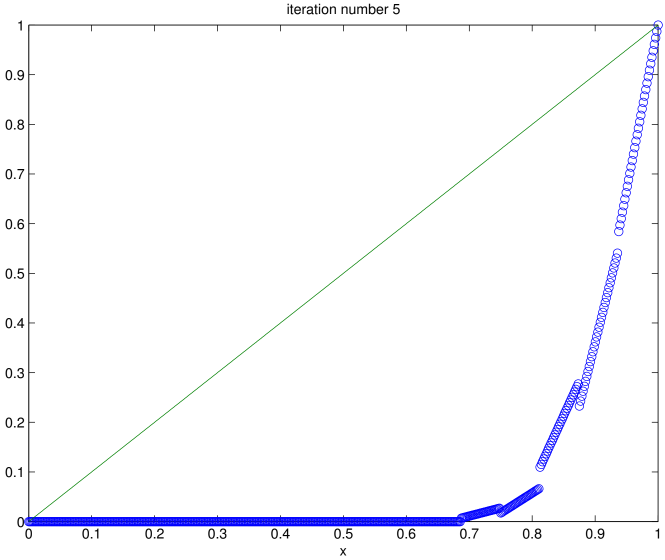
16 Subdomains, Iteration 3



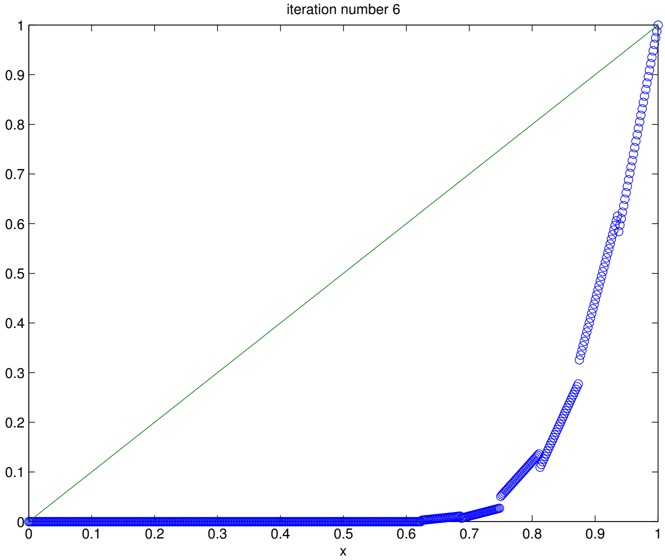
16 Subdomains, Iteration 4



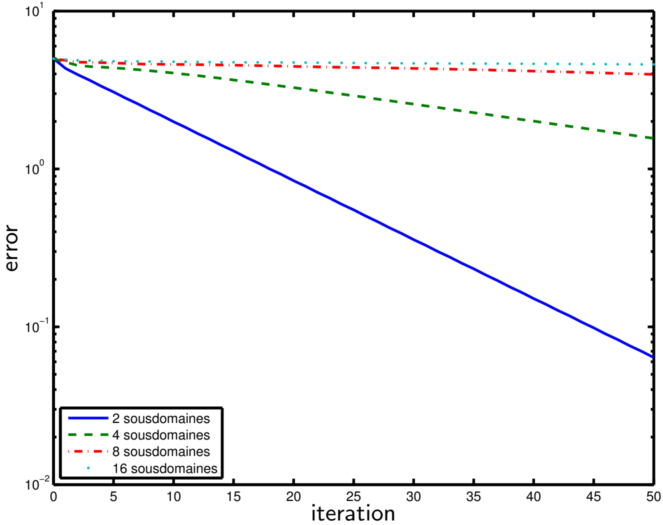
16 Subdomains, Iteration 5



16 Subdomains, Iteration 6



No Scalability in This Case



(Overlap $L = L_i$ diminishes with subdomain size $H = H_i$)

Introduce a Coarse Correction Like in Multigrid

Need to define a global approximate solution $\mathbf{u}_n$.

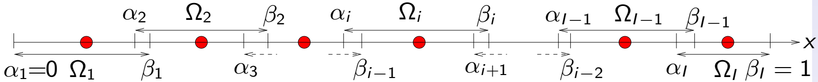
Then introduce a coarse grid and compute

$$
\begin{aligned}
\mathbf{r}_n &= \mathbf{f} - A\mathbf{u}_n\,; \\
\mathbf{r}_c &= R\mathbf{r}_n\,; \\
\mathbf{u}_c &= A_c^{-1}\mathbf{r}_c\,; \\
\mathbf{u}_n &= \mathbf{u}_n + P\mathbf{u}_c\,;
\end{aligned}
$$

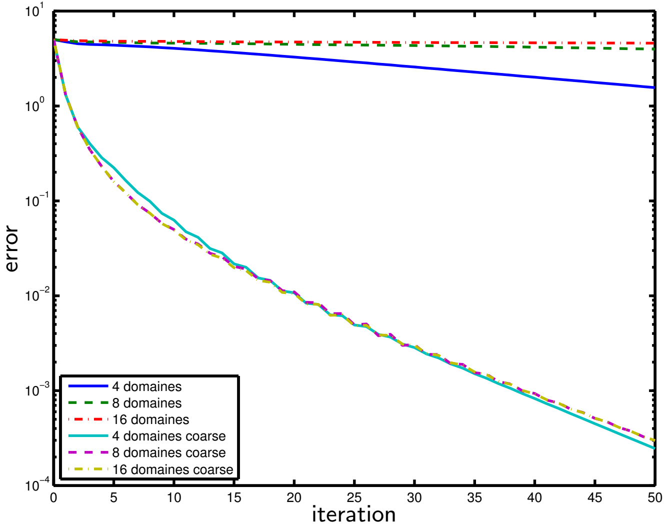
Standard components in multigrid:

- use for the prolongation P interpolation
- use for the restriction R the prolongation transposed
- use for the coarse matrix $A_c = RAP$ (Galerkin)

Classical coarse grid choice: one point in the middle of each subdomain



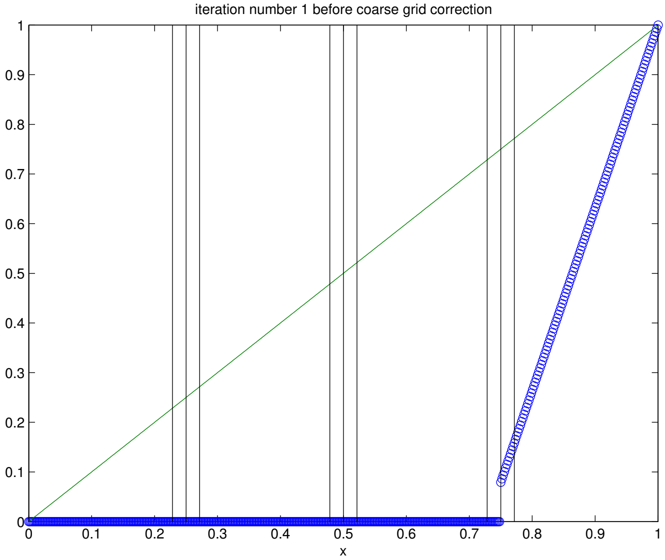
Scalability with Coarse Grid



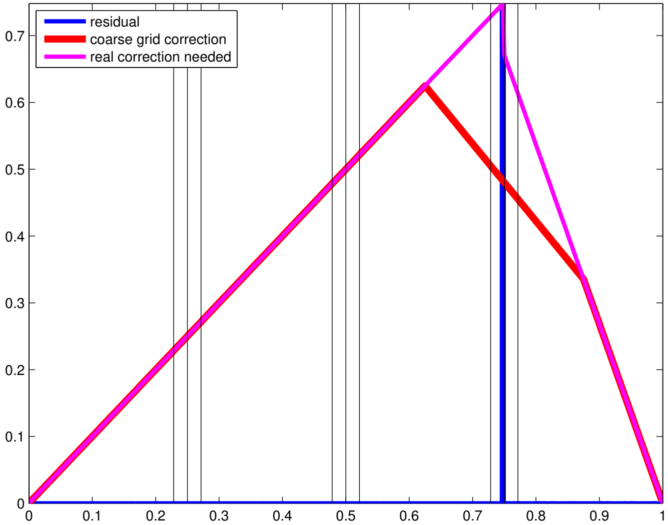
Weak scalability: $L = L_i$ diminishes with $H = H_i$

$\Longrightarrow$ This method is thus "optimal" in classical DD

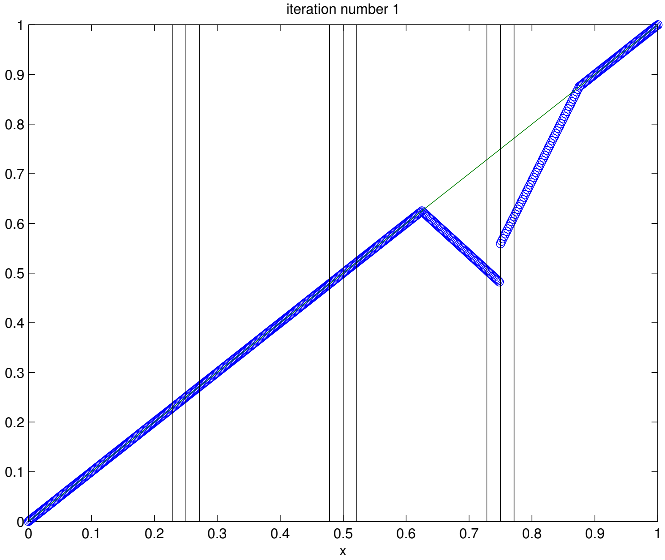
How Good is this Multigrid Coarse Space?



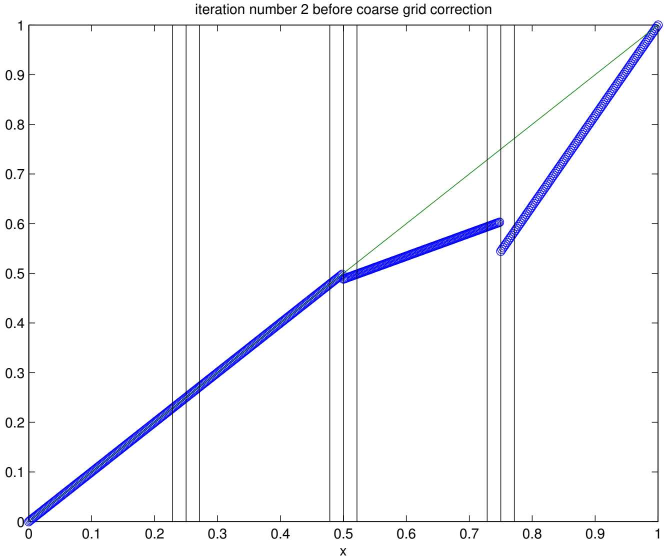
Iter 1: Residual, Error and Coarse Correction



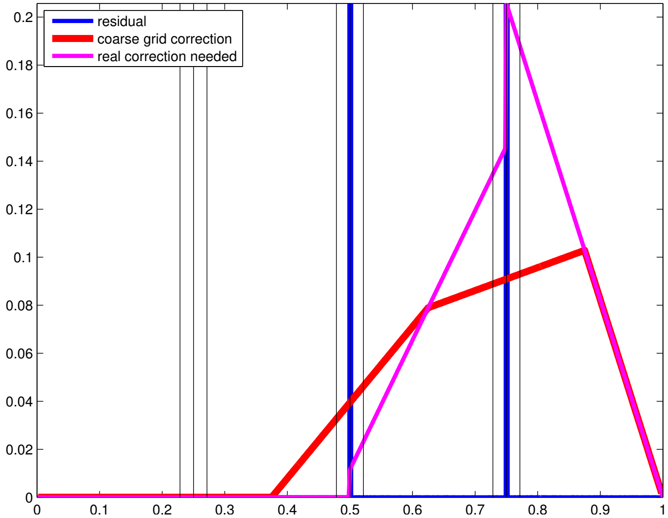
Solution after Coarse Correction 1



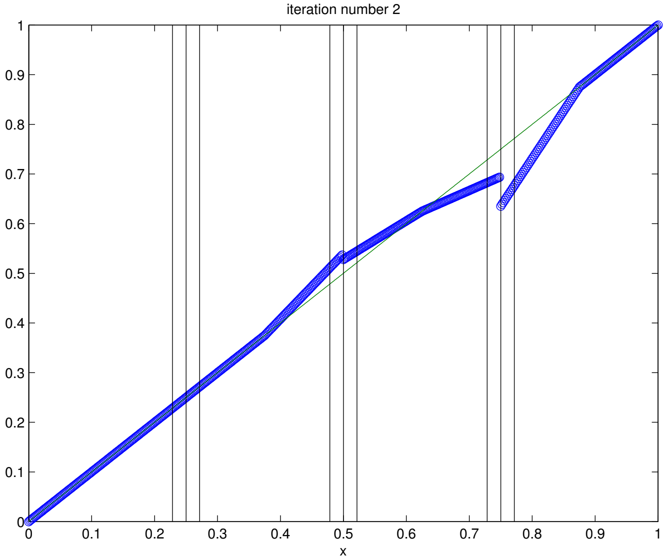
Solution after Subdomain Solve 2



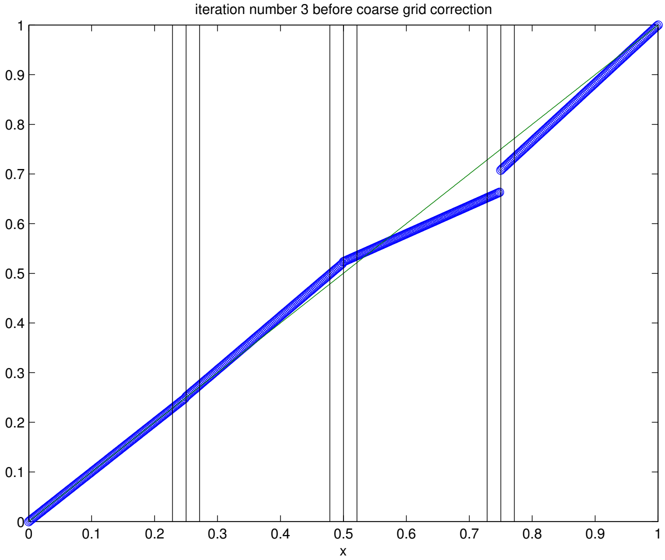
Iter 2: Residual, Error and Coarse Correction



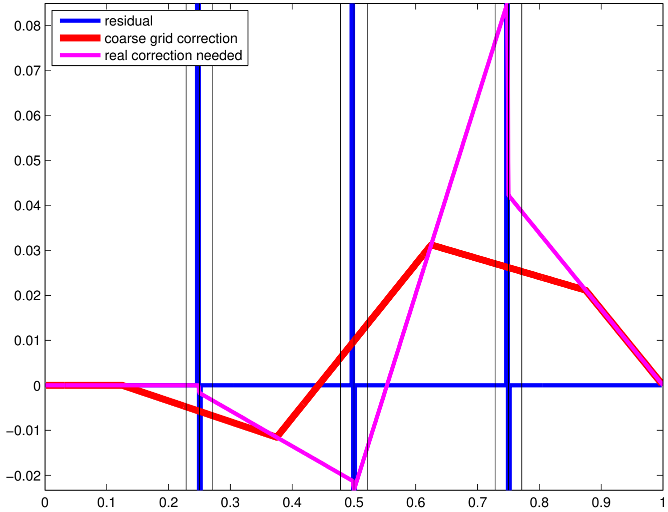
Solution after Coarse Correction 2



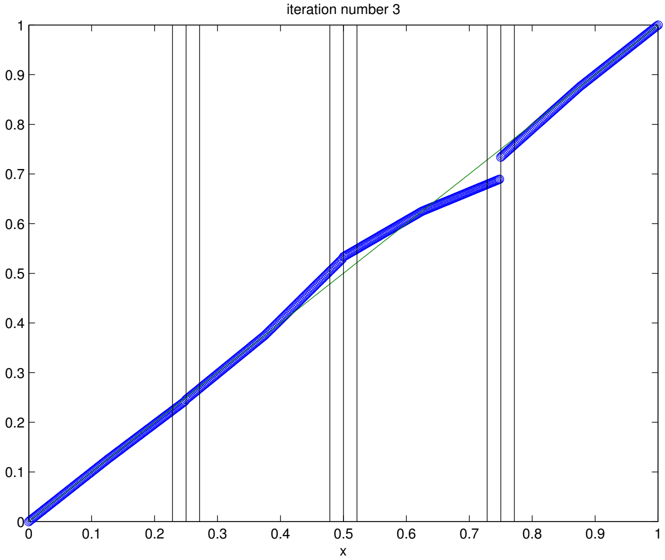
Solution after Subdomain Solve 3



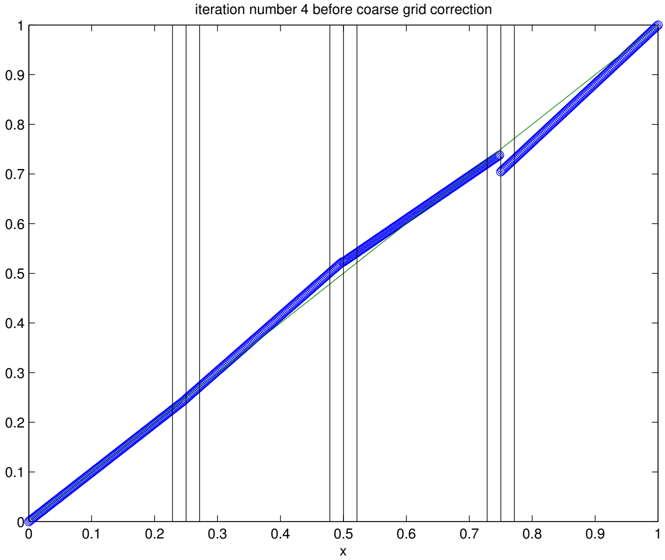
Iter 3: Residual, Error and Coarse Correction



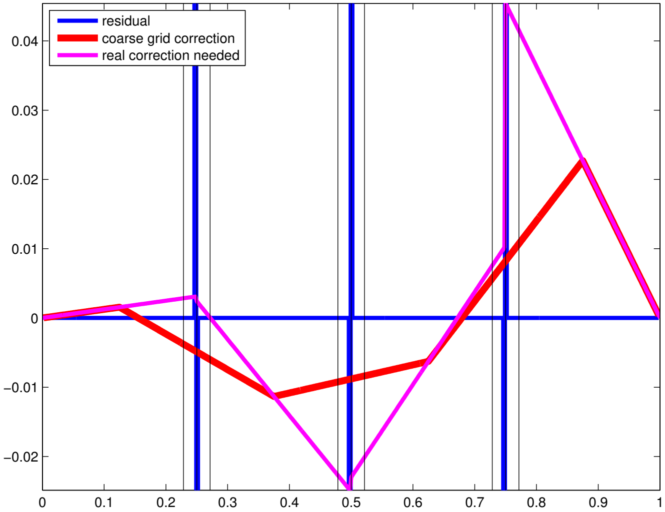
Solution after Coarse Correction 3



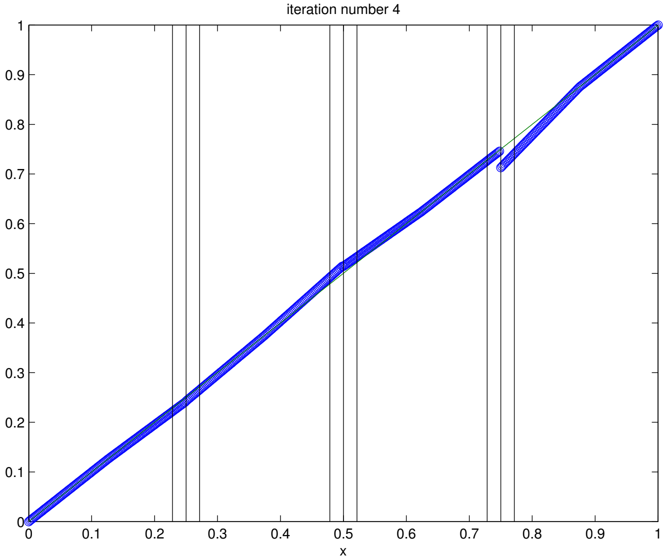
Solution after Subdomain Solve 4



Iter 4: Residual, Error and Coarse Correction



Solution after Coarse Correction 4



First Observation

The coarse space components should not destroy the work of the subdomain iteration

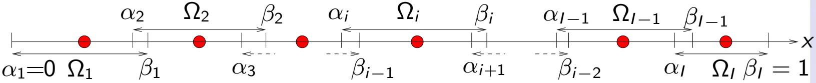
$\Longrightarrow$ **Coarse space components should be harmonic in the subdomains**

First Observation

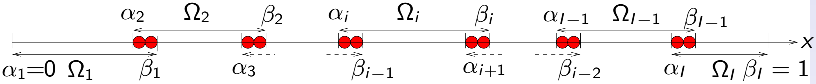
The coarse space components should not destroy the work of the subdomain iteration

$\Longrightarrow$ Coarse space components should be harmonic in the subdomains

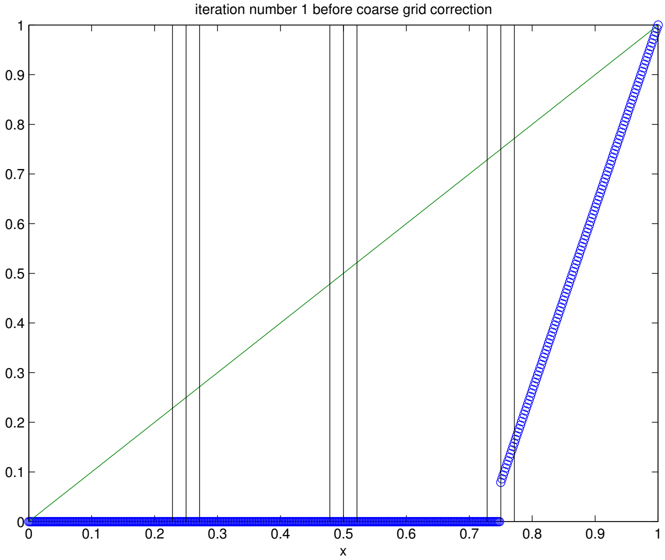
Idea: instead of choosing one degree of freedom in each subdomain,



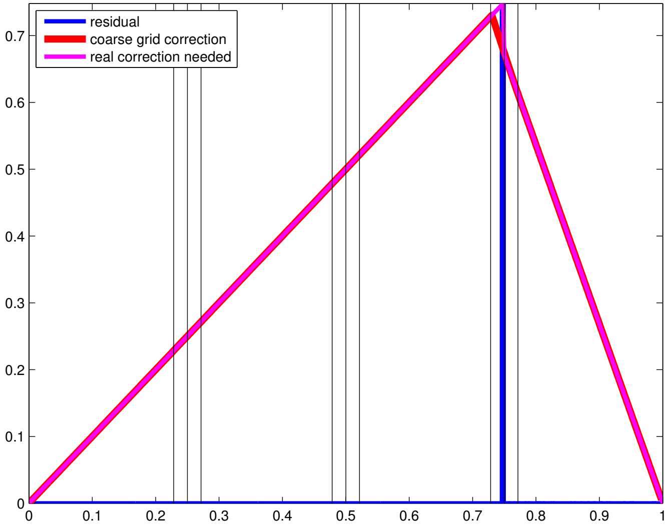
choose the support of the coarse grid in the overlap:



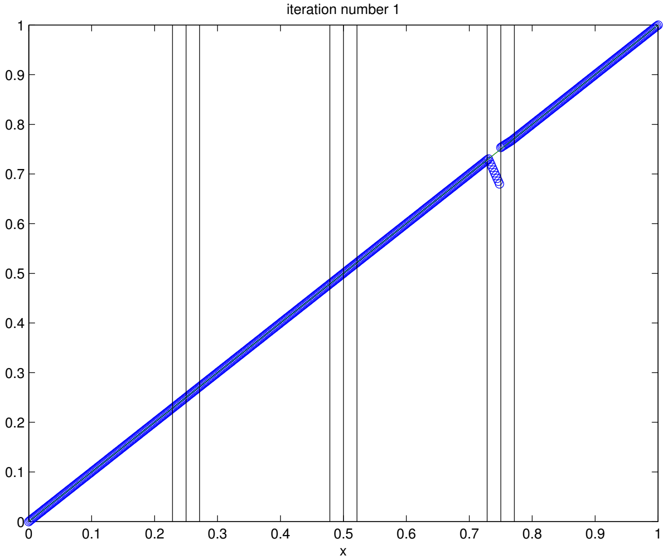
Coarse Grid Support in the Overlap



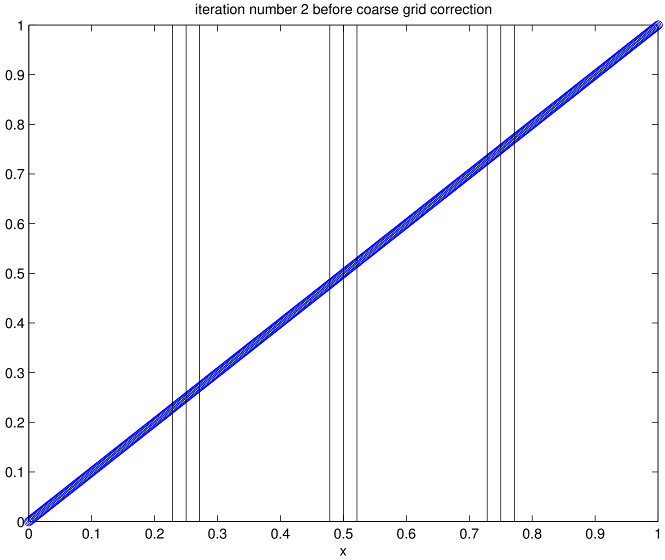
Iter 1: Residual, Error and Coarse Correction



Solution after Coarse Correction 1



Solution after Subdomain Solve 2



Second Observation

We found a coarse space which leads to convergence after one coarse correction (G, Halpern (2012), see also algebraic multigrid: Brandt, McCormick, Ruge (1982), Stüben (1983)).

$\Longrightarrow$ **The 2-level DD method becomes a direct solver**

Definition (Complete Coarse Space)

A complete coarse space is a coarse space such that the domain decomposition method converges after the coarse correction.

Definition (Optimal Coarse Space)

An optimal coarse space is a complete coarse space of smallest dimension.

Second Observation

We found a coarse space which leads to convergence after one coarse correction (G, Halpern (2012), see also algebraic multigrid: Brandt, McCormick, Ruge (1982), Stüben (1983)).

$\Longrightarrow$ **The 2-level DD method becomes a direct solver**

Definition (Complete Coarse Space)

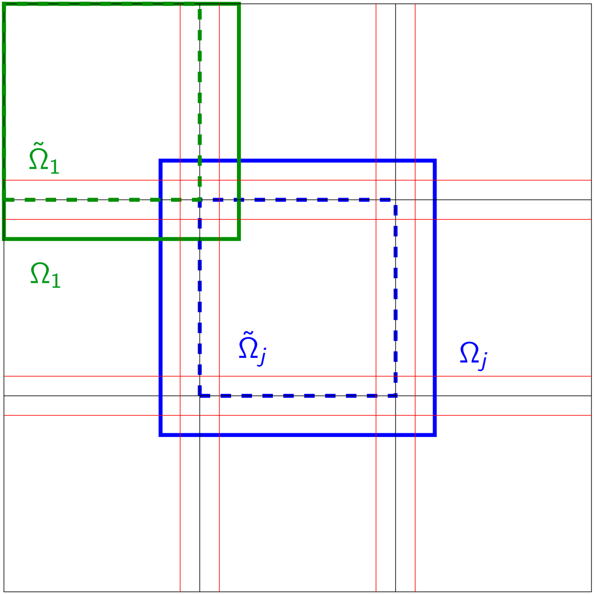
A complete coarse space is a coarse space such that the domain decomposition method converges after the coarse correction.

Definition (Optimal Coarse Space)

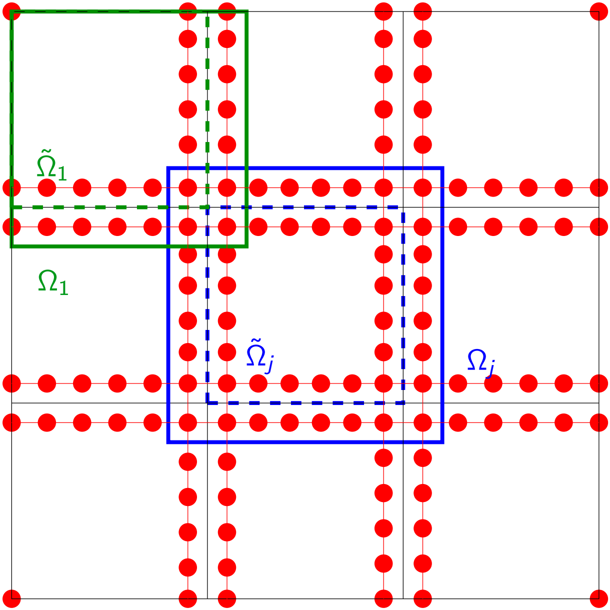
An optimal coarse space is a complete coarse space of smallest dimension.

Optimal here really means better is not possible!

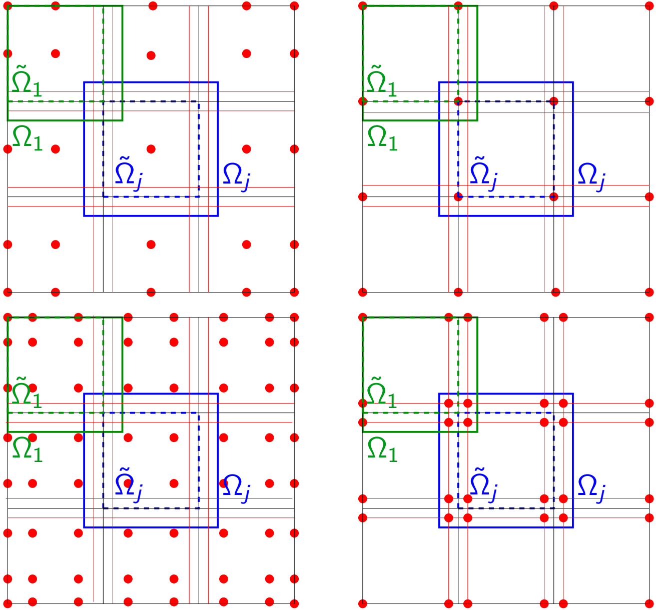
Optimal Coarse Space in Two Dimensions



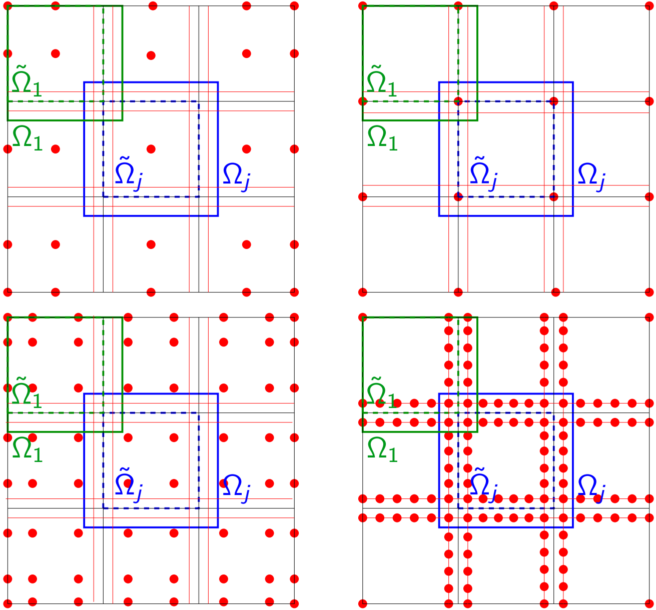
Optimal Coarse Space in Two Dimensions



Comparison of Various Q1 Coarse Spaces

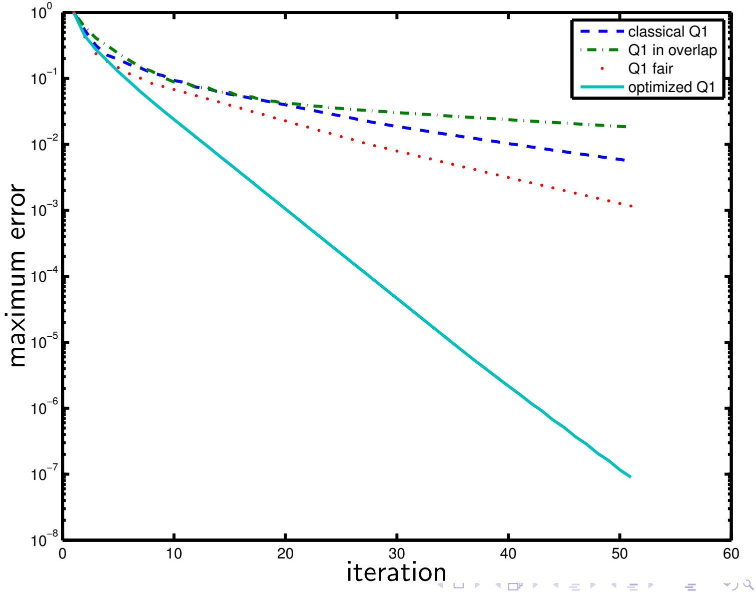


Only one Approximates the Optimal One!



Schwarz Method with Various Q1 Coarse Spaces

ASM, 2D, 16×16 subdomains, 256×256 gridpoints, h overlap



Preconditioning: What is it ?

Classical approach: if for a given linear system $A\mathbf{u} = \mathbf{f}$, Krylov methods like CG, GMRES do not converge well, try to find a *preconditioner* M such that they work better on the *preconditioned system*

$$M^{-1}A\mathbf{u} = M^{-1}\mathbf{f}.$$

How to find a good preconditioner ?

Suppose we have some stationary fixed point iteration

$$\mathbf{u}^{n+1} = \mathbf{u}^n + M^{-1}(\mathbf{f} - A\mathbf{u}^n).$$

If this method converges, we have

$$\mathbf{u} = \mathbf{u} + M^{-1}(\mathbf{f} - A\mathbf{u}) \Longleftrightarrow M^{-1}A\mathbf{u} = M^{-1}\mathbf{f}$$

Any fixed point iteration defines a preconditioner!

Why does this give good preconditioners?

For stationary iterative methods,

$$\mathbf{u}^{n+1} = \mathbf{u}^n + M^{-1}(\mathbf{f} - A\mathbf{u}^n) = (I - M^{-1}A)\mathbf{u}^n + M^{-1}\mathbf{f},$$

we need M such that

1. the spectral radius $\rho(I - M^{-1}A)$ is small
2. it should be inexpensive to apply M^{-1}

For a Krylov method, we need M such that

1. the spectrum of $M^{-1}A$ is clustered (around one)
2. it should be inexpensive to apply M^{-1}

Note that

$$\rho(I - M^{-1}A) \text{ small} \Longleftrightarrow \text{spectrum of } M^{-1}A \text{ close to one}$$

Result: Using a Krylov method with preconditioner M always gives you lower iteration counts than just using the stationary iteration.

Proof (for residual minimizing Krylov methods):

I) The stationary iterative method computes

$$\mathbf{u}^n = (I - M^{-1}A)\mathbf{u}^{n-1} + M^{-1}\mathbf{f} = \mathbf{u}^{n-1} + \mathbf{r}_{stat}^{n-1}$$

Multiply on the left and right by $M^{-1}A$:

$$M^{-1}A\mathbf{u}^n = M^{-1}A(\mathbf{u}^{n-1} + \mathbf{r}_{stat}^{n-1})$$

Subtract this from $M^{-1}\mathbf{f}$:

$$M^{-1}\mathbf{f} - M^{-1}A\mathbf{u}^n = M^{-1}\mathbf{f} - M^{-1}A(\mathbf{u}^{n-1} + \mathbf{r}_{stat}^{n-1})$$

This implies that the residual $\mathbf{r}_{stat}^n := M^{-1}\mathbf{f} - M^{-1}A\mathbf{u}^n$ satisfies

$$\boxed{\mathbf{r}_{stat}^n = (I - M^{-1}A)\mathbf{r}_{stat}^{n-1} = (I - M^{-1}A)^n\mathbf{r}^0}$$

Proof (continued):

II) The Krylov method will use the Krylov space

$$\mathcal{K}_n(M^{-1}A, \mathbf{r}^0) := \{\mathbf{r}^0, M^{-1}A\mathbf{r}^0, \dots, (M^{-1}A)^{n-1}\mathbf{r}^0\}$$

to search for $\mathbf{u}^n \in \mathbf{u}^0 + \mathcal{K}_n(M^{-1}A, \mathbf{r}^0)$, i.e.

$$\mathbf{u}^n = \mathbf{u}^0 + \sum_{i=1}^{n} \alpha_i (M^{-1}A)^{i-1}\mathbf{r}^0$$

Multiply on the left and right by $M^{-1}A$:

$$M^{-1}A\mathbf{u}^n = M^{-1}A\mathbf{u}^0 + \sum_{i=1}^{n} \alpha_i (M^{-1}A)^{i}\mathbf{r}^0$$

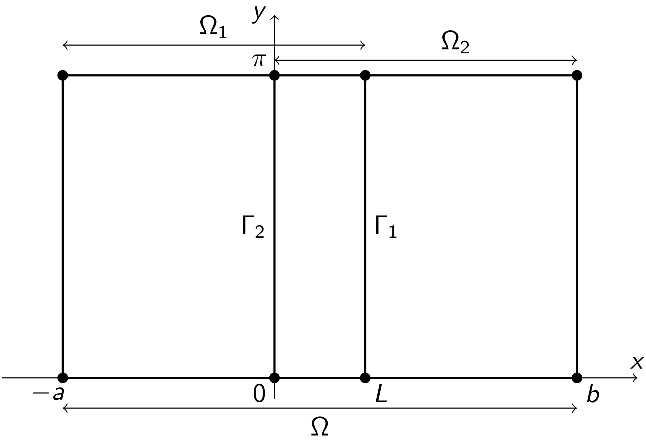
Subtract this from $M^{-1}\mathbf{f}$:

$$M^{-1}\mathbf{f} - M^{-1}A\mathbf{u}^n = M^{-1}\mathbf{f} - M^{-1}A\mathbf{u}^0 - \sum_{i=1}^{n} \alpha_i (M^{-1}A)^{i}\mathbf{r}^0$$

which means the residual $\mathbf{r}^n_{kry}$ satisfies

$$\boxed{\mathbf{r}^n_{kry} = p_n(M^{-1}A)\mathbf{r}^0, \quad p_n(0) = 1}$$

Using Parallel Schwarz as Preconditioner



$$\begin{aligned} \Delta u_1^n &= f && \text{in } \Omega_1 \\ u_1^n &= u_2^{n-1} && \text{on } \Gamma_1 \\ u_1^n &= g && \text{on } \partial\Omega \cap \partial\Omega_1 \end{aligned} \qquad \begin{aligned} \Delta u_2^n &= f && \text{in } \Omega_2 \\ u_2^n &= u_1^n && \text{on } \Gamma_2 \\ u_2^n &= g && \text{on } \partial\Omega \cap \partial\Omega_2 \end{aligned}$$

Using Parallel Schwarz as Preconditioner

We rewrite the parallel Schwarz iteration

$$\begin{array}{rcll} \Delta u_1^n & = & f & \text{in } \Omega_1 \\ u_1^n & = & u_2^{n-1} & \text{on } \Gamma_1 \\ u_1^n & = & g & \text{on } \partial\Omega \cap \partial\Omega_1 \end{array} \qquad \begin{array}{rcll} \Delta u_2^n & = & f & \text{in } \Omega_2 \\ u_2^n & = & u_1^{n-1} & \text{on } \Gamma_2 \\ u_2^n & = & g & \text{on } \partial\Omega \cap \partial\Omega_2 \end{array}$$

explicitly using the local solvers $\mathcal{L}_i^{-1}$,

$$\begin{array}{rcl} u_1^n(\Gamma_2) & = & \mathcal{L}_1^{-1}(f, g, u_2^{n-1}(\Gamma_1)) \\ u_2^n(\Gamma_1) & = & \mathcal{L}_2^{-1}(f, g, u_1^{n-1}(\Gamma_2)) \end{array}$$

At convergence, we have using linearity

$$\begin{array}{l} u_1(\Gamma_2) = \mathcal{L}_1^{-1}(f, g, u_2(\Gamma_1)) = \mathcal{L}_1^{-1}(f, g, 0) + \mathcal{L}_1^{-1}(0, 0, u_2(\Gamma_1)) \\ u_2(\Gamma_1) = \mathcal{L}_2^{-1}(f, g, u_1(\Gamma_2)) = \mathcal{L}_2^{-1}(f, g, 0) + \mathcal{L}_2^{-1}(0, 0, u_1(\Gamma_2)) \end{array}$$

and thus obtain the preconditioned system

$$\begin{array}{rcl} u_1(\Gamma_2) - \mathcal{L}_1^{-1}(0, 0, u_2(\Gamma_1)) & = & \mathcal{L}_1^{-1}(f, g, 0) \\ u_2(\Gamma_1) - \mathcal{L}_2^{-1}(0, 0, u_1(\Gamma_2)) & = & \mathcal{L}_2^{-1}(f, g, 0) \end{array}$$

Substructured System from Parallel Schwarz

This preconditioned system we obtained

$$\begin{aligned} u_1(\Gamma_2) - \mathcal{L}_1^{-1}(0,0,u_2(\Gamma_1)) &= \mathcal{L}_1^{-1}(f,g,0), \\ u_2(\Gamma_1) - \mathcal{L}_2^{-1}(0,0,u_1(\Gamma_2)) &= \mathcal{L}_2^{-1}(f,g,0). \end{aligned}$$

is a preconditioned interface system (a so called substructured system), which we can write explicitly as

$$\begin{pmatrix} Id & -\mathcal{L}_1^{-1}(0,0,\cdot) \\ -\mathcal{L}_2^{-1}(0,0,\cdot) & Id \end{pmatrix} \begin{pmatrix} u_1(\Gamma_2) \\ u_2(\Gamma_1) \end{pmatrix} = \begin{pmatrix} \mathcal{L}_1^{-1}(f,g,0) \\ \mathcal{L}_2^{-1}(f,g,0) \end{pmatrix}$$

Solving this system with Block Jacobi gives back the parallel Schwarz method, but it is more efficient to solve it using a Krylov method.

$\Longrightarrow$ **using Krylov acceleration for parallel Schwarz**

$\Longrightarrow$ **using parallel Schwarz as a preconditioner**

Conclusions

- Multigrid type coarse corrections are "optimal", but not very efficient (G, Halpern (2012))
- There exist optimal coarse spaces and optimized spectral approximations (G, Halpern, Santugini (2014))
- Coarse spaces can fix problems of DD methods (see DD25 conference)
- All domain decomposition methods should be used with Krylov acceleration, i.e. as preconditioners.