Exceptional service in the national interest

Sandia National Laboratories

SAND2018-3464C

March 29, 2018

Structured grid algorithms in MueLu

Luc Berger-Vergiat, Ray Tuminaro and Peter Ohm

Center for Computing Research
Sandia National Laboratories
Albuquerque, New Mexico USA

SAND 2018-XXXXX YY

U.S. DEPARTMENT OF ENERGY | National Nuclear Security Administration | CCR Center for Computing Research

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc. for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Outline

Some notations

Sandia National Laboratories

A linear system is denoted

$$Ax = b, \tag{1}$$

we assume that A has size $n \times n$, x is the solution and $\bar{x}$ the exact solution.
The residual vector is defined as

$$r = b - Ax. \tag{2}$$

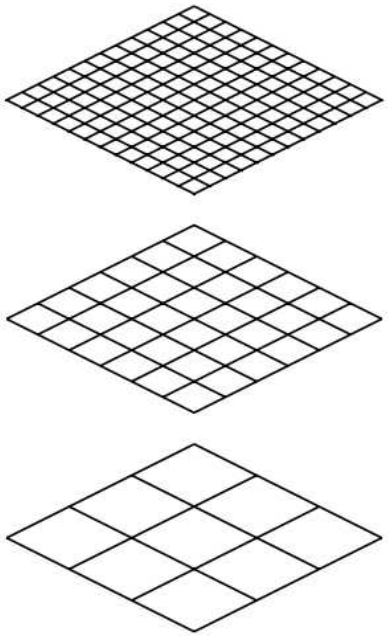
Noticing that $b = A\bar{x}$ we get

$$r = A\bar{x} - Ax. \tag{3}$$

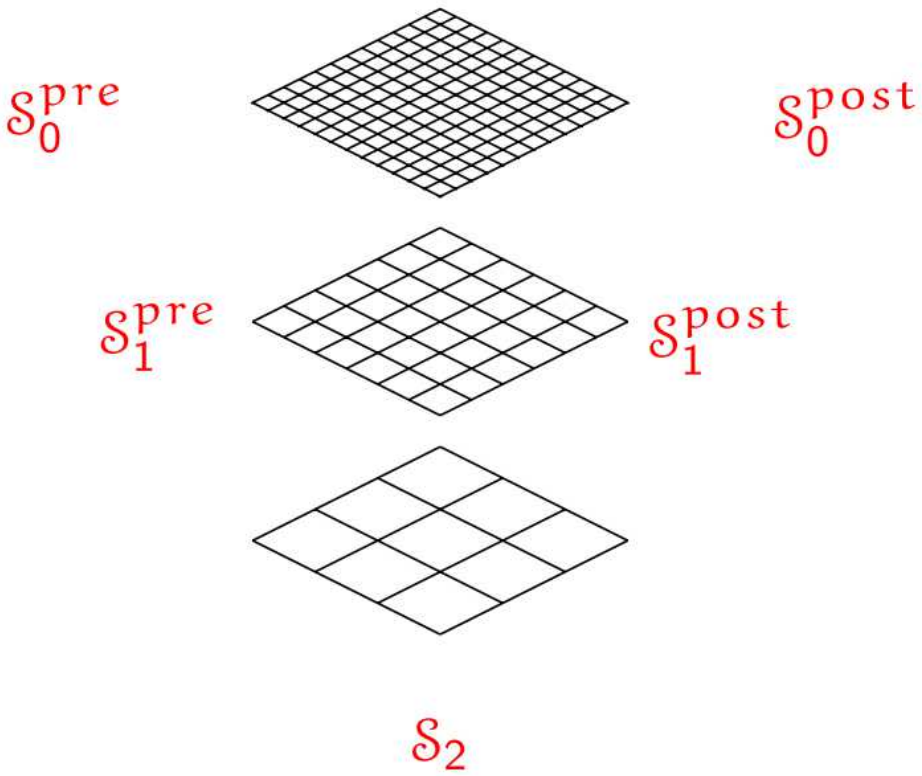
Which prompts us to introduce the error $e = \bar{x} - x$ and the correction equation

$$Ae = r. \tag{4}$$

Multigrid method



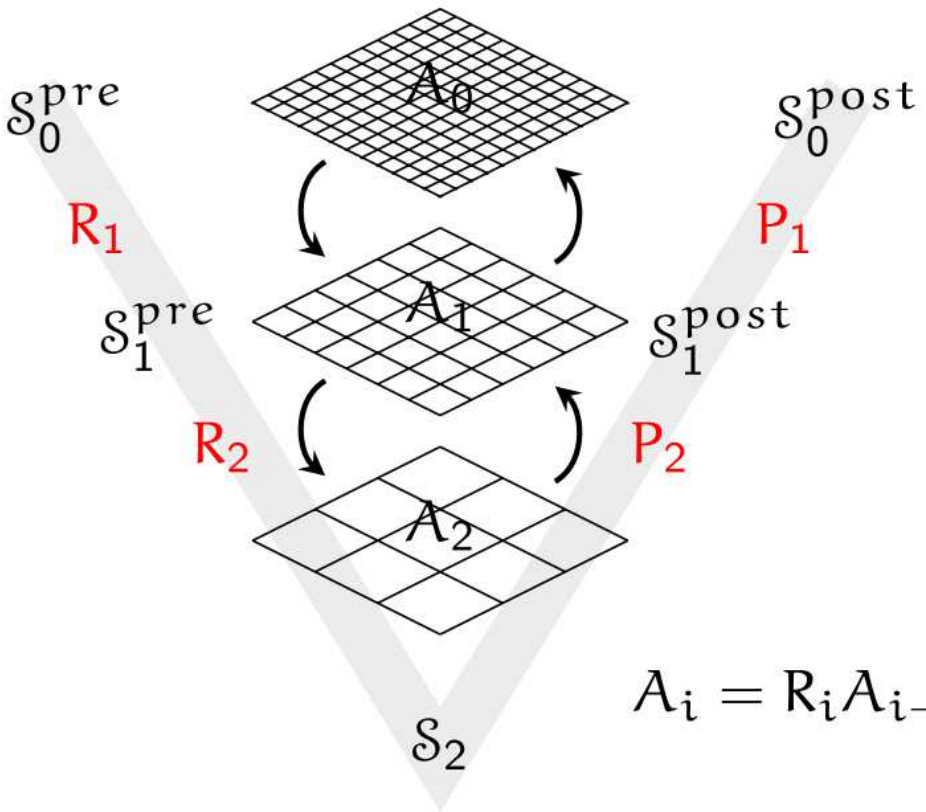
Multigrid method



Two main components

- Smoothers
  - "Cheap" reduction of oscillatory error (high energy)
  - $\mathcal{S}_L \approx A_L^{-1}$ on the coarsest level L

Multigrid method



$$A_i = R_i A_{i-1} P_i$$

Two main components

- Smoothers
  - "Cheap" reduction of oscillatory error (high energy)
  - $S_L \approx A_L^{-1}$ on the coarsest level L
- Grid transfers (prolongators and restrictors)
  - Definition of coarse level matrices (setup phase)
  - Data movement between levels (solve phase).

Classic smoothers

Split the numerical operator

$$A = M - N \tag{5}$$

taking care of choosing an M easily invertible.
Use a fixed point iteration based on the above split

$$Mx_{n+1} = Nx_n + b, \tag{6}$$

additionally some damping may be introduced:

$$x_{n+1} = \omega M^{-1}(Nx_n + b) + (1 - \omega)x_n. \tag{7}$$

Smoothing 2D problems

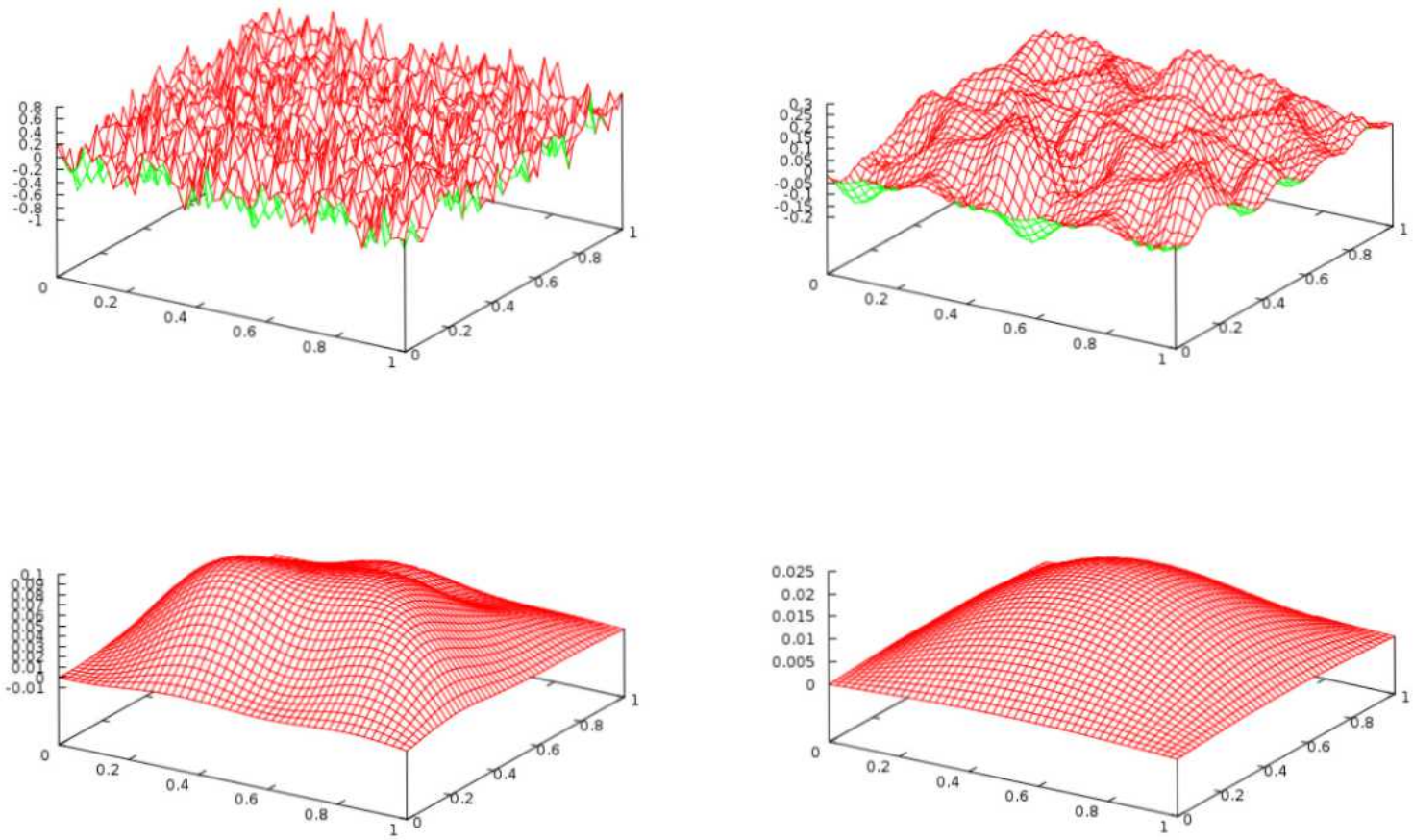


Figure: Damped Jacobi in 2D: images courtesy of the MueLu tutorial

The effect of coarsening

Jacobi iterations remove high frequencies quickly but stall on low frequencies. This happens to most smoothers...
The smooth error is well represented with much fewer grid points!

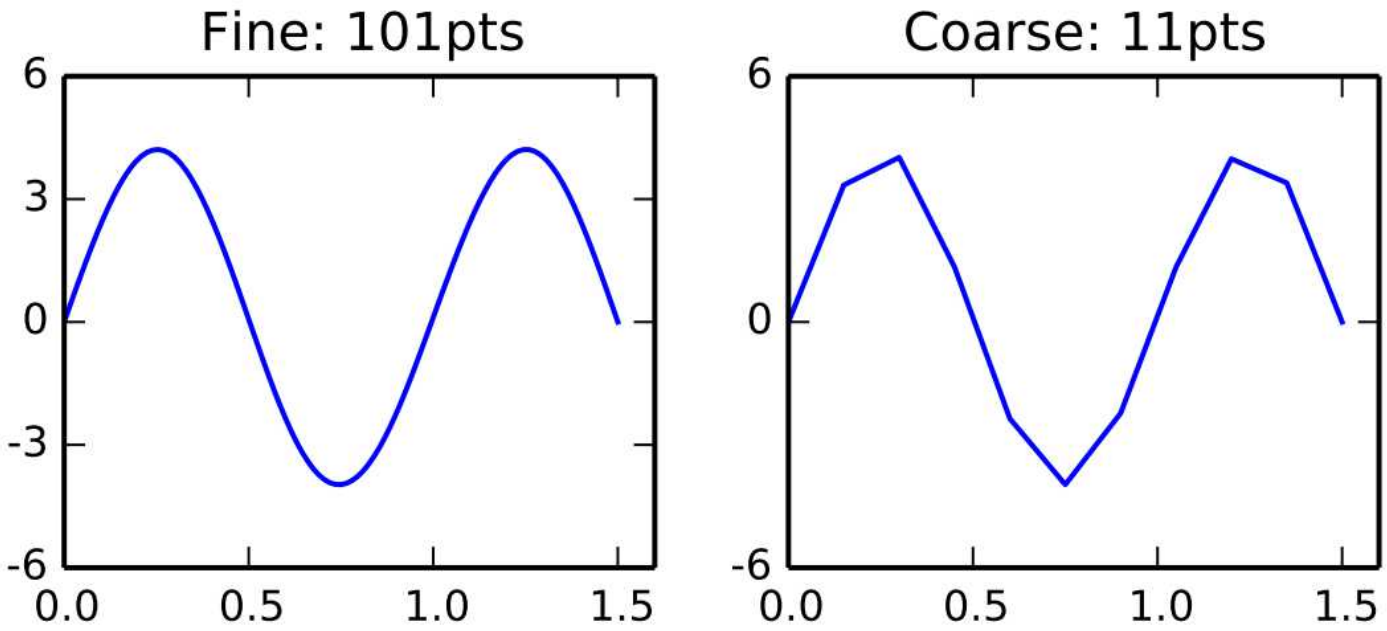


Figure: Representation of the error on two grids

Now smooth coarse problem or solve with LU if small enough.

Multigrid properties

Sandia National Laboratories

Mutligrid

- solves/accelerates the solution of linear problems,
- has a theoretical complexity $\mathrm{O(N)}$, with N the $\#$ unknowns,
- can be implemented in parallel but requires a lot of programing effort,
- comes in two main flavors: Geometric and Algebraic.

For Poisson's equation discretized on a uniform grid you can expect the number of iterations until convergence to remain constant for varying mesh sizes.

Convergence might depend on mesh size for more complex equations/discretizations.

Outline

What is MueLu?

Sandia National Laboratories

MueLu:

- is the multigrid package succeeding ML within Trilinos[1],
- provides an extendable set of grid transfer and smoother algorithms,
- is programmed in C++11 and relies heavily on polymorphism via templating (compile time polymorphism) and object oriented progamming (run time polymorphism),
- is available on multiple platforms: Linux, OsX, Windows, ...
- and runs on multiple architectures: multiple cores, many cores, GPU,
- can use 64bits indexing to solve problems with more than 4B equations.

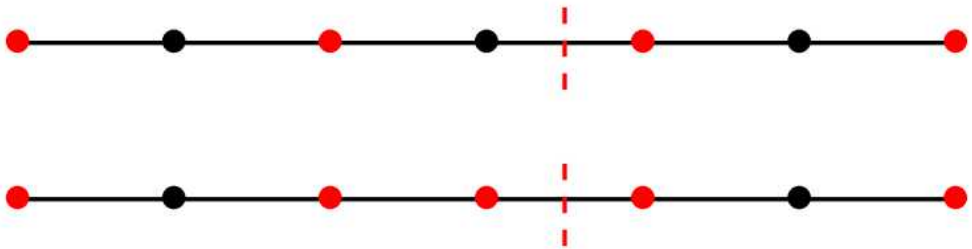
[1] The Trilinos Project is an effort to develop algorithms and enabling technologies within an object-oriented software framework for the solution of large-scale, complex multi-physics engineering and scientific problems.

Structured grid coarsening

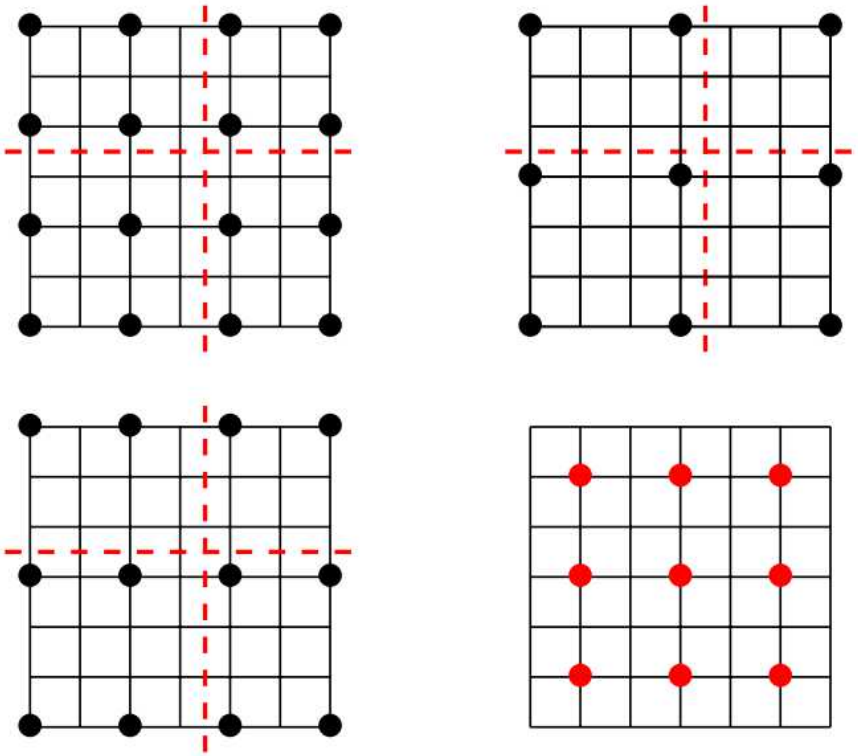
Sandia National Laboratories

Two approaches can be used to perform structured coarsening on a mesh:

1. coupled coarsening: the sampling of coarse points ignores rank bonundaries,
2. uncoupled coarsening: the sampling of coarse points is done on a per rank basis.



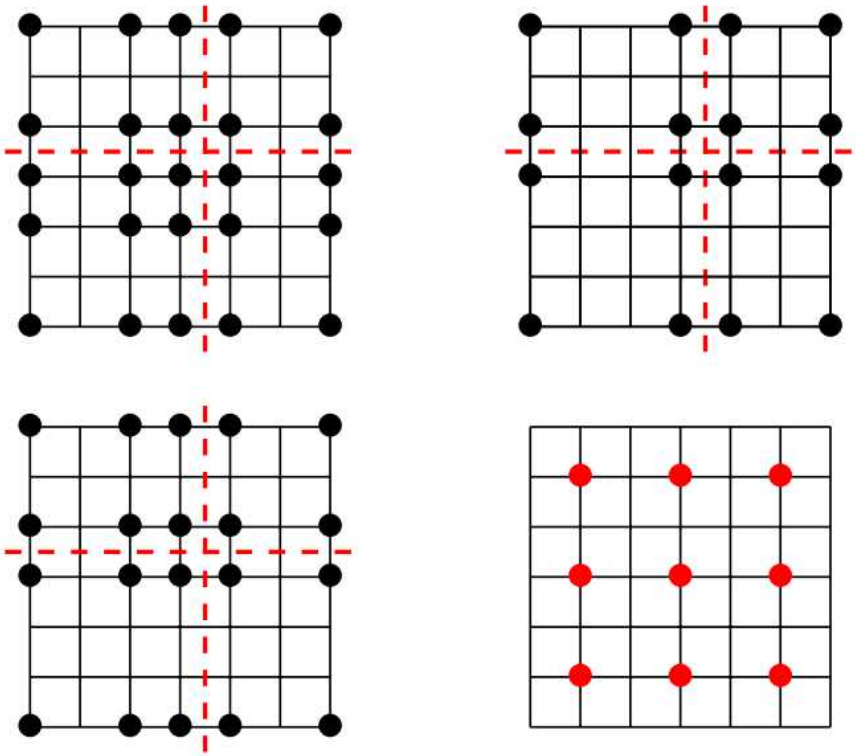
Coupled coarsening



Red dashed lines - -, represent processor boundaries.

1. Coarsening rates are variable and independent in each direction,
2. coarse points are chosen to include boundary points (better for BC),
3. coarsening is continuous across processor boundary for uniform coarse point distribution
4. coarsening rate is automatically reduced at mesh boundary.

Uncoupled coarsening



Red dashed lines - -, represent processor boundaries.

1. Coarsening rates are variable and independent in each direction,
2. coarse points are chosen to include rank boundary points (better for BC),
3. coarsening is continuous within a rank
4. coarsening rate is automatically reduced at rank boundary.

Characteristic of two approaches

Sandia National Laboratories

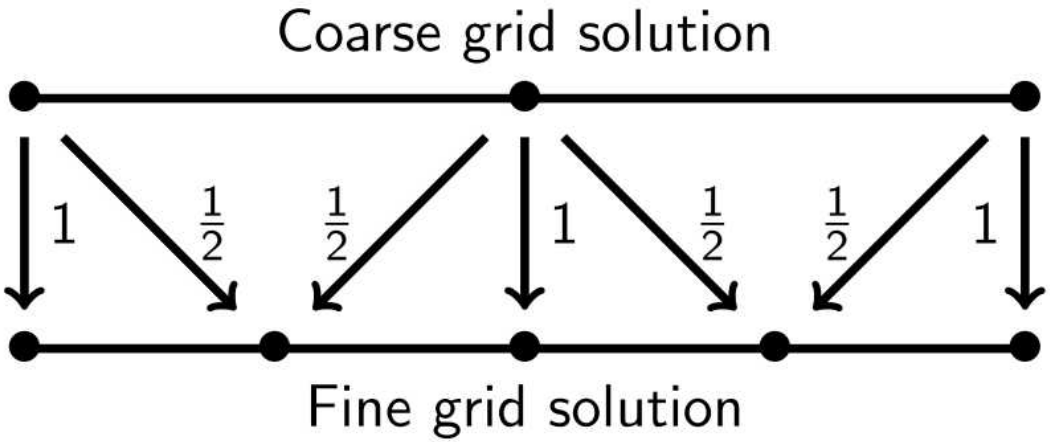
- **coupled coarsening**: more uniform, less coarse points, improved performance with linear interpolation, sensitive to global node ordering.
- **uncoupled coarsening**: requires less user input and less communication during setup, easier to implement.

Both implemented in MueLu and availble to all transfer operators: geometric interpolation, black box and smoothed-aggregation.

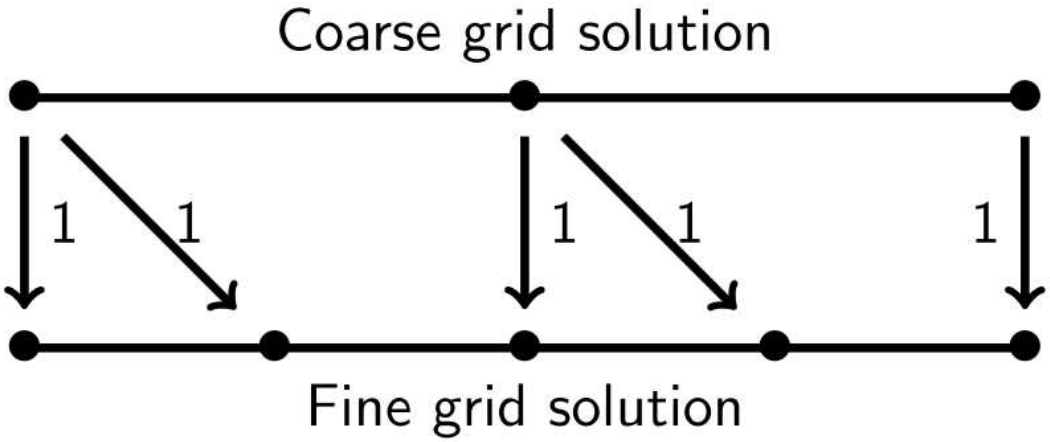
Interpolation based grid transfer

Going from coarse grid to fine grid is easily achieved using:

- Linear interpolation



- Piece-wise constant:



Properties of transfer operators

Sandia National Laboratories

Both piece-wise constant and linear interpolation between coarse points are available.
Linear interpolation:

- relies on Newton iteration to find interpolation weights (more fragile),
- takes geometric distances into account (requires coordinates).

Piece-wise constant interpolation:

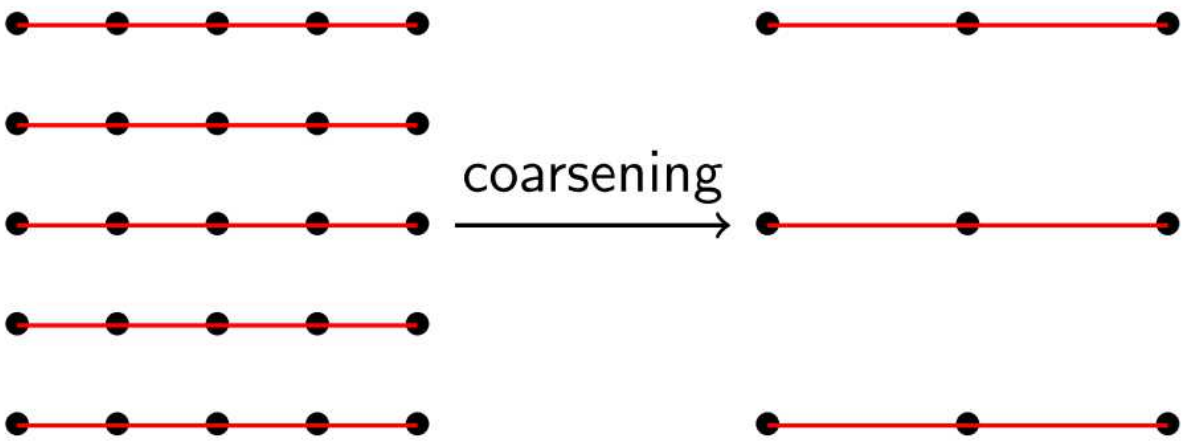
- uses (i,j,k) indexes for coarsening,
- distance based variant relies on coordinates (not tested so far),
- produces less fill on coarse grids.

Structured line detection

Sandia National Laboratories

Structured grid coarsening allows for line/plane detection on each grid level allowing for:

- semi-coarsening with plane relaxation
- line smoothing on anisotropic mesh/problem
- variable coarsening on anisotropic meshes



Outline

Poisson on a cube

Start with an easy problem: $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} = f$

Using the exact solution:
$u(x, y, z) = \sin(\pi x)\sin(\pi y)\sin(\pi z)\exp(x + y + z)$ to compute the forcing term.

In all the following example we use a geometric multigrid (GMG) algorithm with a single sweep of damped (0.9) Jacobi pre- and post-smoother and an LU coarse grid solver.

Serial experimentations

Sandia National Laboratories

For serial examples 3D uniform grids are used and FEM is chosen as the discretization method, the grid transfer operators are constructed using linear interpolation (piece-wise constant gives same results for Poisson).

Iterations	Grid levels			
	2	3	4	5
1	0.0062253	0.006397	0.006397	0.006397
2	4.4829e-05	4.6156e-05	4.6156e-05	4.6156e-05
3	3.4685e-07	3.5036e-07	3.5036e-07	3.5036e-07
4	2.4203e-09	2.4898e-09	2.4898e-09	2.4898e-09
5	1.5014e-11	1.6925e-11	1.6925e-11	1.6925e-11
6	9.9889e-14	1.262e-13	1.2619e-13	1.262e-13
7	2.3459e-15	2.4433e-15	2.4363e-15	2.4445e-15

Table: Using GMG as fixed point iteration solver with varying number of levels on a mesh with 4913 points (17^3).

Serial experimentations

Same problem using varying coarsening rate

Iterations	Coarsening rate			
	2	3	4	5
1	5.3674e-04	1.2750e-03	1.8536e-03	2.0950e-03
2	6.2792e-06	4.3705e-05	1.9476e-04	2.3626e-04
3	1.2070e-07	2.1998e-06	2.8050e-05	4.3682e-05
4	2.6778e-09	1.2962e-07	4.5258e-06	9.7886e-06
5	6.0525e-11	8.4691e-09	7.7184e-07	2.4465e-06
6	1.3386e-12	5.8442e-10	1.3501e-07	6.5032e-07
7	3.0480e-14	4.1147e-11	2.3897e-08	1.7805e-07
8	7.9183e-16	2.9033e-12	4.2500e-09	4.9413e-08
9	1.0104e-16	2.0357e-13	7.5659e-10	1.3802e-08
10	9.2781e-17	1.4138e-14	1.3454e-10	3.8693e-09

Table: The finest grid is discretized with $n = (1 + c^2)^3$ points, where c is the coarsening rate, two grid levels are used and the coarse grid contains 8 points.

Parallel results

Here we use GMG as a preconditioner for a GMRES linear solver, the coarsening rate is set to 2 and the coarse grid contains 10 or less points. The convergence criteria for these simulations is $\frac{||R||}{||R_0||} \leqslant 10^{-15}$.

MPI ranks	Mesh size			
	8,000	64,000	512,000	4,096,000
1	6	7	7	7
2	7	7	7	7
4	7	7	7	7
8	7	7	8	8

Table: Parallel behavior of the multigrid algorithm.

Note: one could use CG instead of GMRES for Poisson but GMRES is our default solver.

Outline

Concluding remarks

Sandia National Laboratories

We have achieved:

- the implementation of a parallel geometric multigrid algorithm in MueLu using variable coarsening rate and line smoothing on all levels,
- said algorithm performance is tested on simple Poisson 3D problem in serial and parallel,
- finally initial experiments are conducted on the blunt wedge problem to assess the performance of the preconditioner on a problem of interest.

Concluding remarks

Sandia National Laboratories

We are working on:

- the implementation of a black box multigrid algorithm with variable coarsening rate: code generates correct grid transfer operators on simple meshes, more tests are needed before production runs,
- both geometric and black box algorithms are extended to block meshes,
- the algorithm need to be tested on more challenging CFD problems: complex geometries, higher velocities, reacting gas problem, etc...

References

R.D. Falgout, *An Introduction to Algebraic Multigrid*, in Computing in Science & Engineering Volume 8, Issue 6 (2006) 24–33, 10.1109/MCSE.2006.105

Black box grid transfer

Sandia National Laboratories

Black box multigrid is an algorithm designed to take advantage of the actual operator to compute the grid transfer operators. I was first proposed by Dendy [**?**, **?**, **?**] This algorithm relies on two ideas:

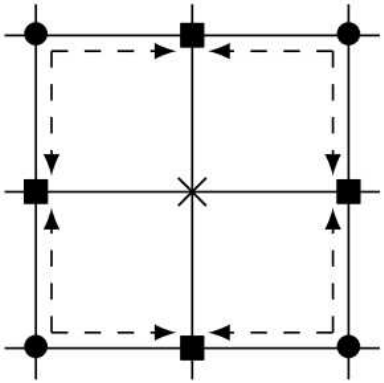
1. Schur complement provides "perfect" grid transfer operators

$$A_h = \begin{bmatrix} A_{ff} & A_{fc} \\ A_{cf} & A_{cc} \end{bmatrix}, \; P = \begin{bmatrix} -A_{ff}^{-1} A_{fc} \\ I_c \end{bmatrix}, \; A_c = A_{cc} - A_{cf} A_{ff}^{-1} A_{fc}$$

2. dimensional decoupling generates sparsity and triangular structure

Black box grid transfer

Macro element schematic representation:

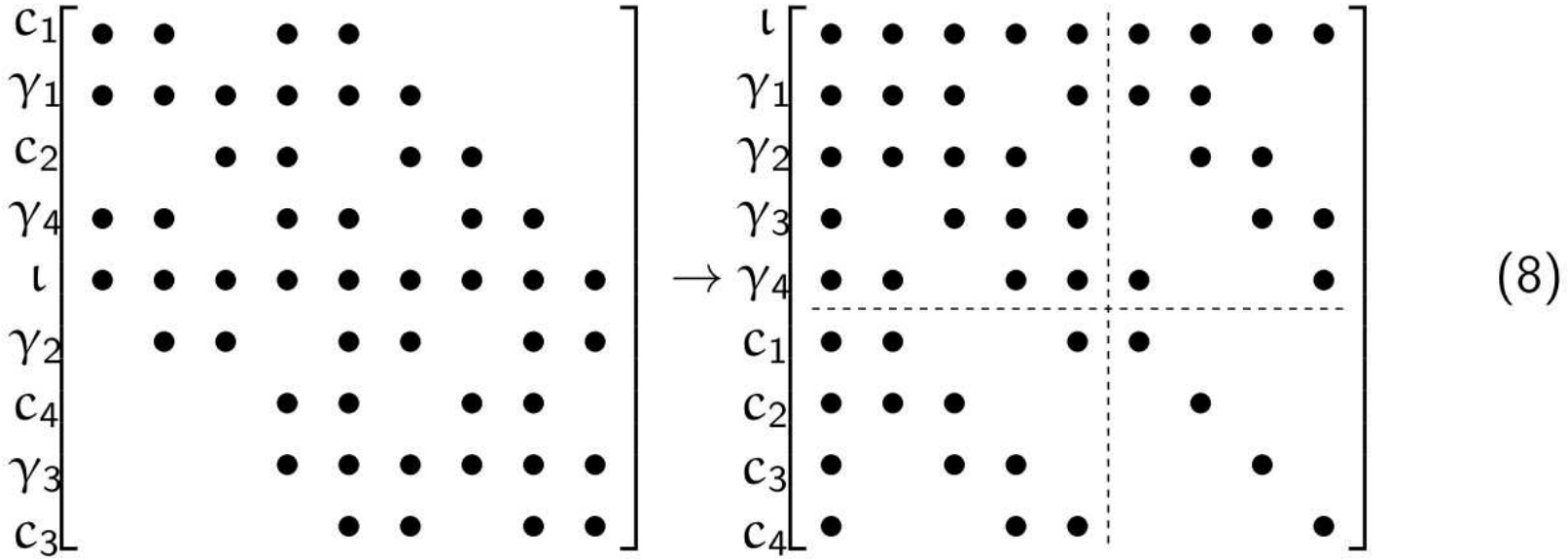


● : corner points

■ : edge points

× : interior points

Reorder nodes by type: interior, edge, corner



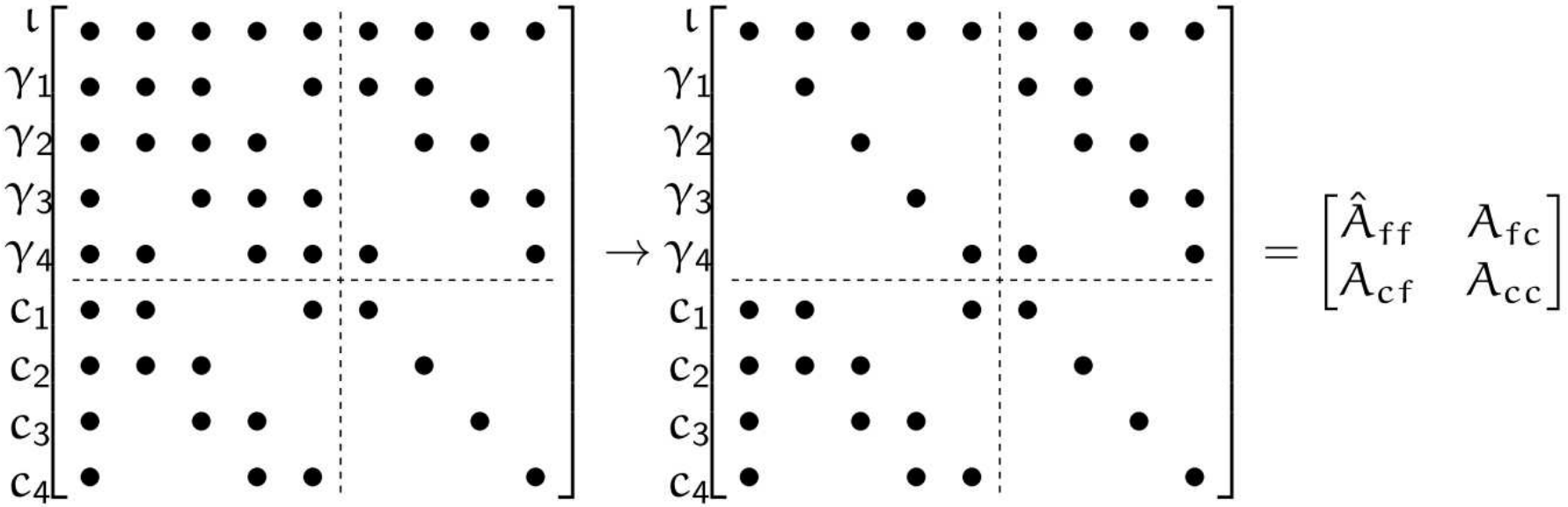
(8)

Black box grid transfer

Collapse stencils on edge, i.e. for a vertical, respectively horizontal edge:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} -1 \\ 2 \\ -1 \end{bmatrix}; \qquad \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} -1 & 2 & -1 \end{bmatrix},$$

which leads to the following sparsity pattern:



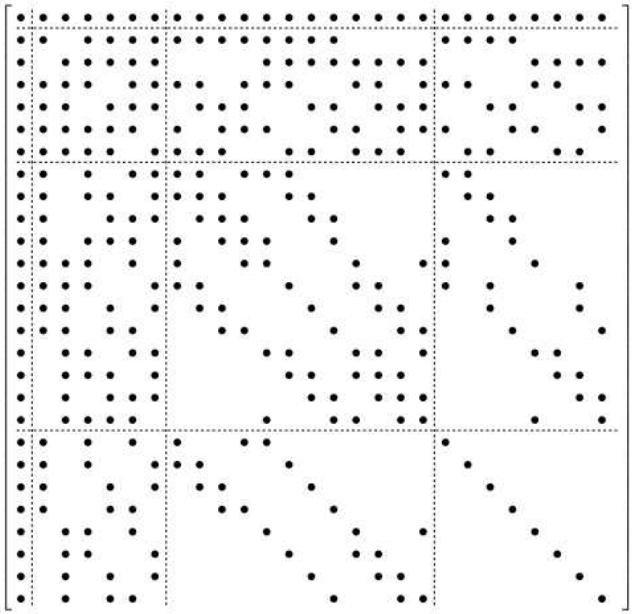
Black box grid transfer

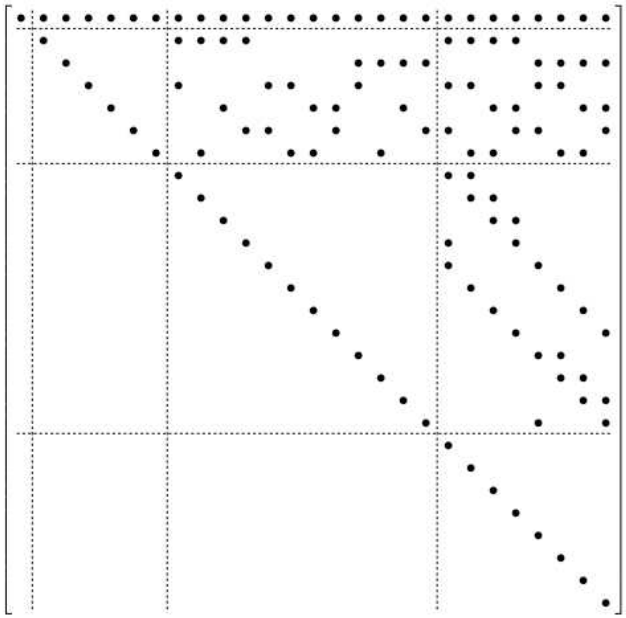
Sandia National Laboratories

Now $\hat{A}_{ff}$ is readily invertible:

$$\hat{A}_{ff} = \begin{bmatrix} \bullet & & & & \\ & \bullet & & & \\ & & \bullet & & \\ & & & \bullet & \\ & & & & \bullet \end{bmatrix} = \begin{bmatrix} A_{\iota\iota} & A_{\iota\gamma} \\ 0 & \hat{A}_{\gamma\gamma} \end{bmatrix} \rightarrow \hat{A}_{ff}^{-1} = - \begin{bmatrix} A_{\iota\iota}^{-1} & -A_{\iota\iota}^{-1} A_{\iota\gamma} \hat{A}_{\gamma\gamma}^{-1} \\ 0 & \hat{A}_{\gamma\gamma}^{-1} \end{bmatrix}$$

The same ideas also apply to 3D problem:





Black box grid transfer

Sandia National Laboratories

Black box multigrid is an algorithm designed to take advantage of the actual operator to compute the grid transfer operators. I was first proposed by Dendy [**?**, **?**, **?**] This algorithm relies on two ideas:

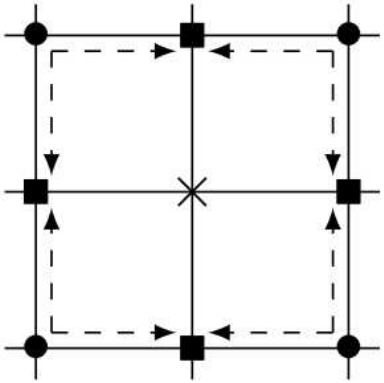
1. Schur complement provides "perfect" grid transfer operators

$$A_h = \begin{bmatrix} A_{ff} & A_{fc} \\ A_{cf} & A_{cc} \end{bmatrix}, \; P = \begin{bmatrix} -A_{ff}^{-1} A_{fc} \\ I_c \end{bmatrix}, \; A_c = A_{cc} - A_{cf} A_{ff}^{-1} A_{fc}$$

2. dimensional decoupling generates sparsity and triangular structure

Black box grid transfer

Macro element schematic representation:

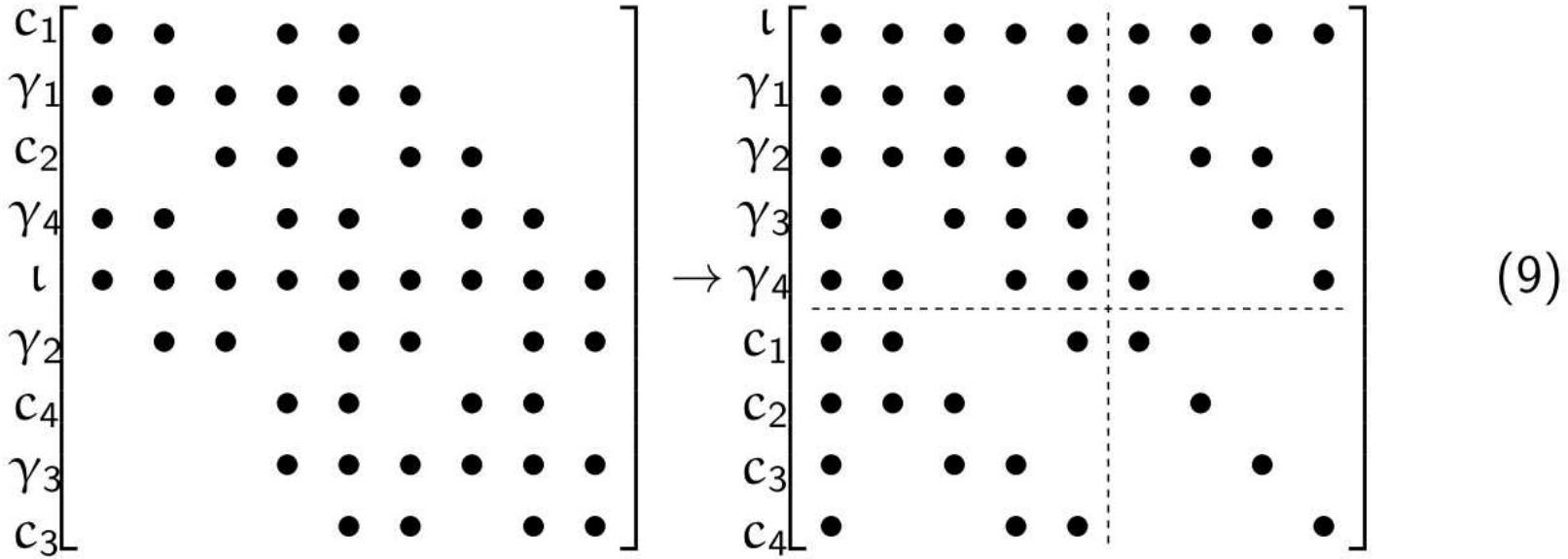


● : corner points

■ : edge points

× : interior points

Reorder nodes by type: interior, edge, corner



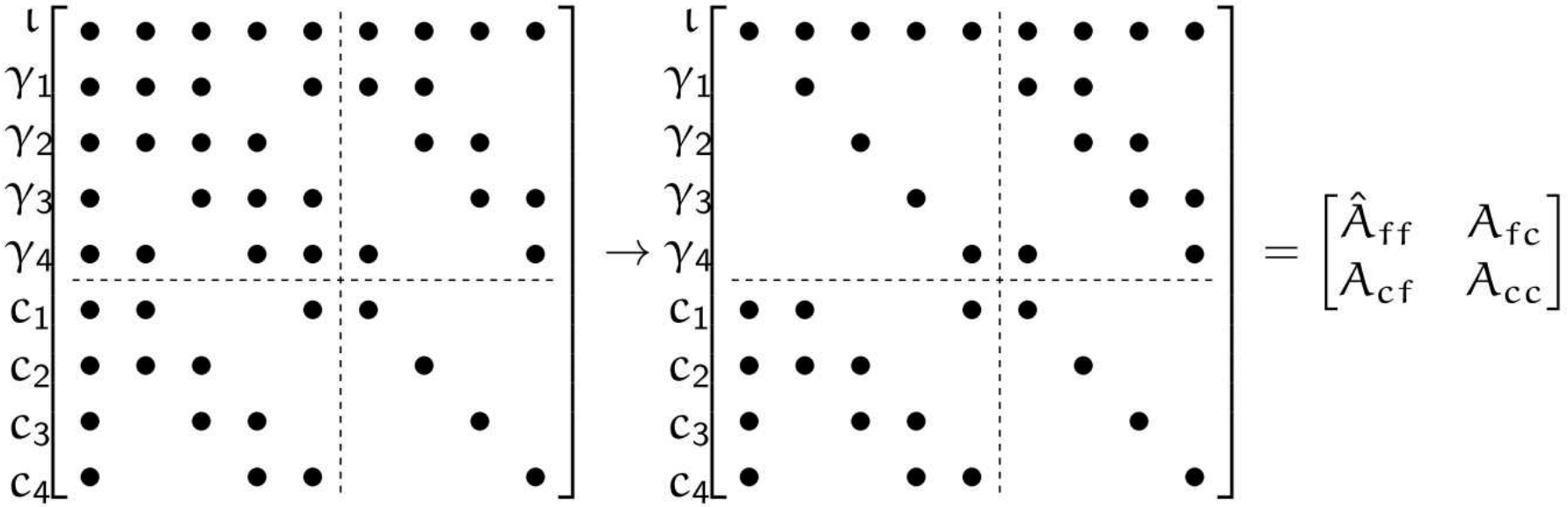
(9)

Black box grid transfer

Collapse stencils on edge, i.e. for a vertical, respectively horizontal edge:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} -1 \\ 2 \\ -1 \end{bmatrix}; \qquad \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} -1 & 2 & -1 \end{bmatrix},$$

which leads to the following sparsity pattern:



Black box grid transfer

Now $\hat{A}_{ff}$ is readily invertible:

$$\hat{A}_{ff} = \begin{bmatrix} \bullet & \bullet & \bullet & \bullet & \bullet \\ & \bullet & & & \\ & & \bullet & & \\ & & & \bullet & \\ & & & & \bullet \end{bmatrix} = \begin{bmatrix} A_{\iota\iota} & A_{\iota\gamma} \\ 0 & \hat{A}_{\gamma\gamma} \end{bmatrix} \rightarrow \hat{A}_{ff}^{-1} = - \begin{bmatrix} A_{\iota\iota}^{-1} & -A_{\iota\iota}^{-1} A_{\iota\gamma} \hat{A}_{\gamma\gamma}^{-1} \\ 0 & \hat{A}_{\gamma\gamma}^{-1} \end{bmatrix}$$

The same ideas also apply to 3D problem: