SIAM LA'15

10/27/2015

Random-Order Alternating Schwarz for Sparse Triangular Solves

Hartwig Anzt, Edmond Chow, Daniel Szyld, Jack Dongarra

ICL UT
INNOVATIVE
COMPUTING LABORATORY
THE UNIVERSITY of TENNESSEE

Sparse Triangular Systems

- **Occur for approximate incomplete factorization preconditioners**
  - Low solution accuracy required as **LU ≈ A** typically only a rough approximation.
  - Replace forward/backward substitutions with iterative method.
  - Better scalability of iterative methods.

- **Jacobi iteration**

$$x^{k+1} = D^{-1}\left(b - (A - D)x^k\right)$$
$$x^{k+1} = D^{-1}b + Mx^k$$

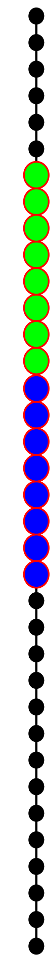
$$M_L = D_L^{-1}\left(D_L - L\right) = I - L$$
$$M_U = D_U^{-1}\left(D_U - U\right) = I - D_U^{-1}U$$

Sparse Triangular Systems

- **Block-Decomposition**
  - Typically, no information about the problem discretization.
  - Matrix partitioning, block sizes match hardware characteristics.
  - Over-decomposition for GPUs.

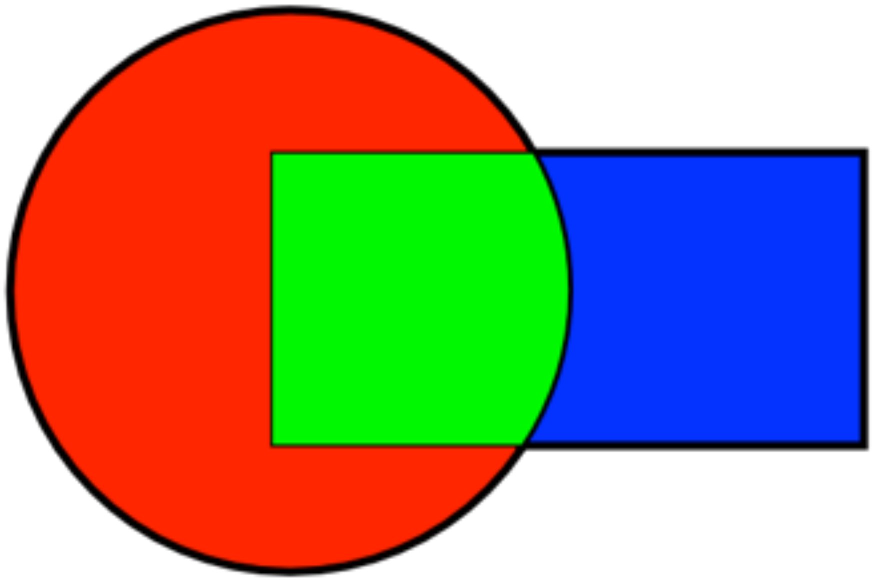
- **Clear information dependency**
  - Synchronous top-down subdomain scheduling results in (block-) substitution. For blocks containing one unknown, exact solve.
  - **Propagation** of **new information** in dependency direction is key.
  - Faster convergence expected for the scheduling:
    top-down in **Ly=b** and bottom-up in **Ux=y**.

  - GPUs do in general **not allow** to control the **scheduling**.



Domain Overlap Strategies

- **Global domain** is decomposed into **subdomains**.
- **A local problem** is solved for each subdomain.
- **Iterative process** generates the **global solution**.
- **Subdomains overlap** for faster **information propagation**.



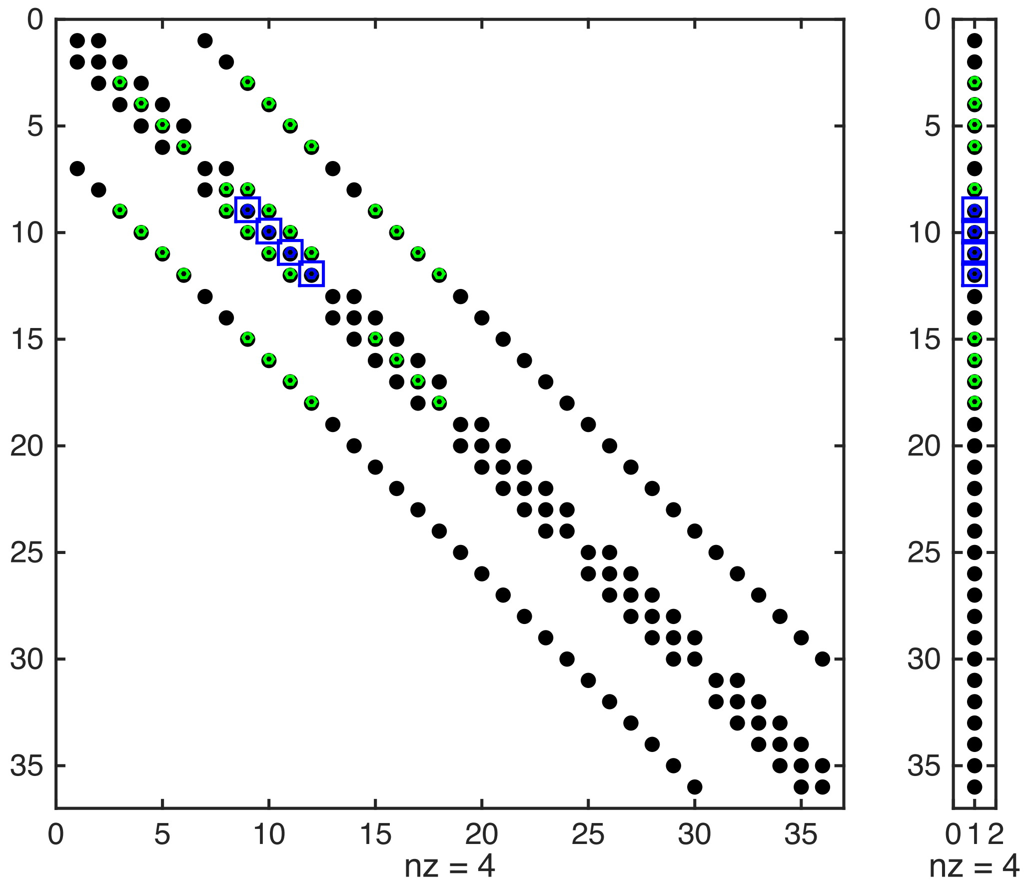
https://en.wikipedia.org/wiki/Domain_decomposition_methods

Domain Overlap Strategies

- **Alternating Schwarz**
  - Write back results for extended subdomain.
  - Sequential updates or multi-color ordering.
  - Fixed subdomain scheduling.
- **Restricted Additive Schwarz**
  - Write back results only for original subdomain.
  - Parallel update of all subdomains.
- **Random-Order Alternating Schwarz**
  - Write back results only for original subdomain.
  - Sequential subdomain updates.
  - Random subdomain scheduling.

Domain Overlap

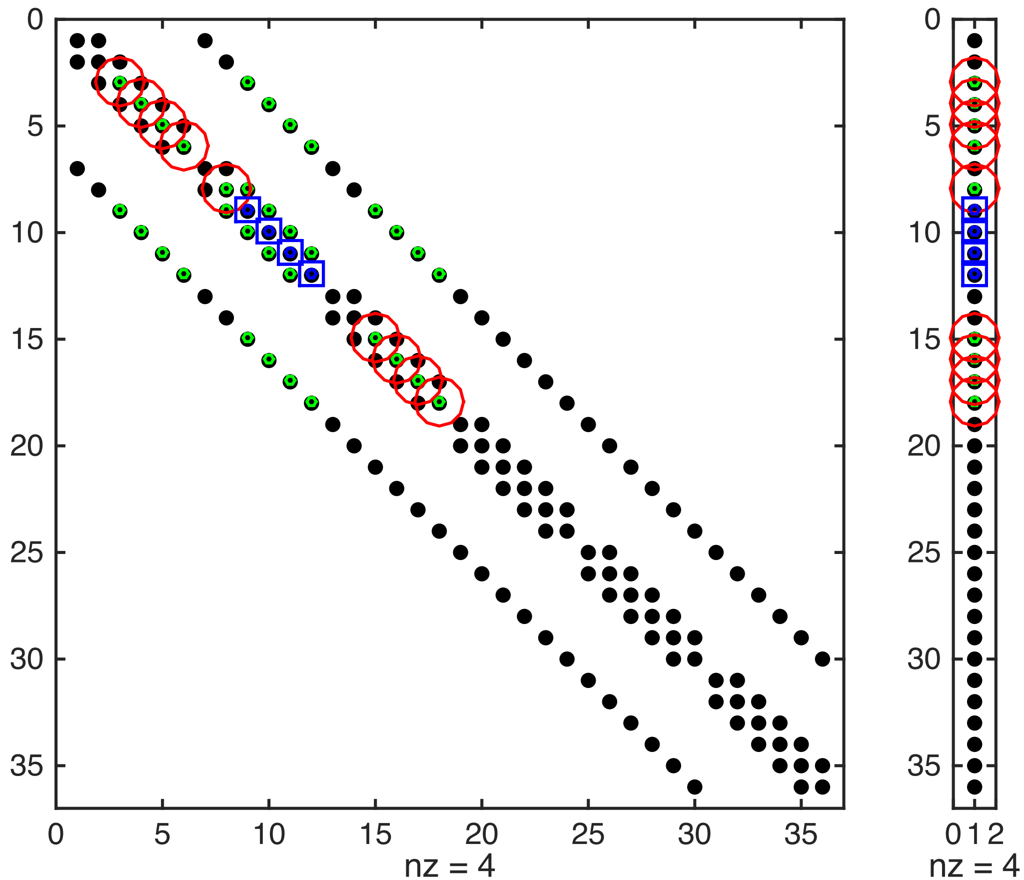
- **Domain overlap based on matrix partitioning**
  - Blocks are extended by components adjacent in the matrix.



Local solver is applied to problem on extended subdomain.

Domain Overlap

- **Domain overlap based on matrix partitioning**
  - Blocks are extended by components adjacent in the matrix.

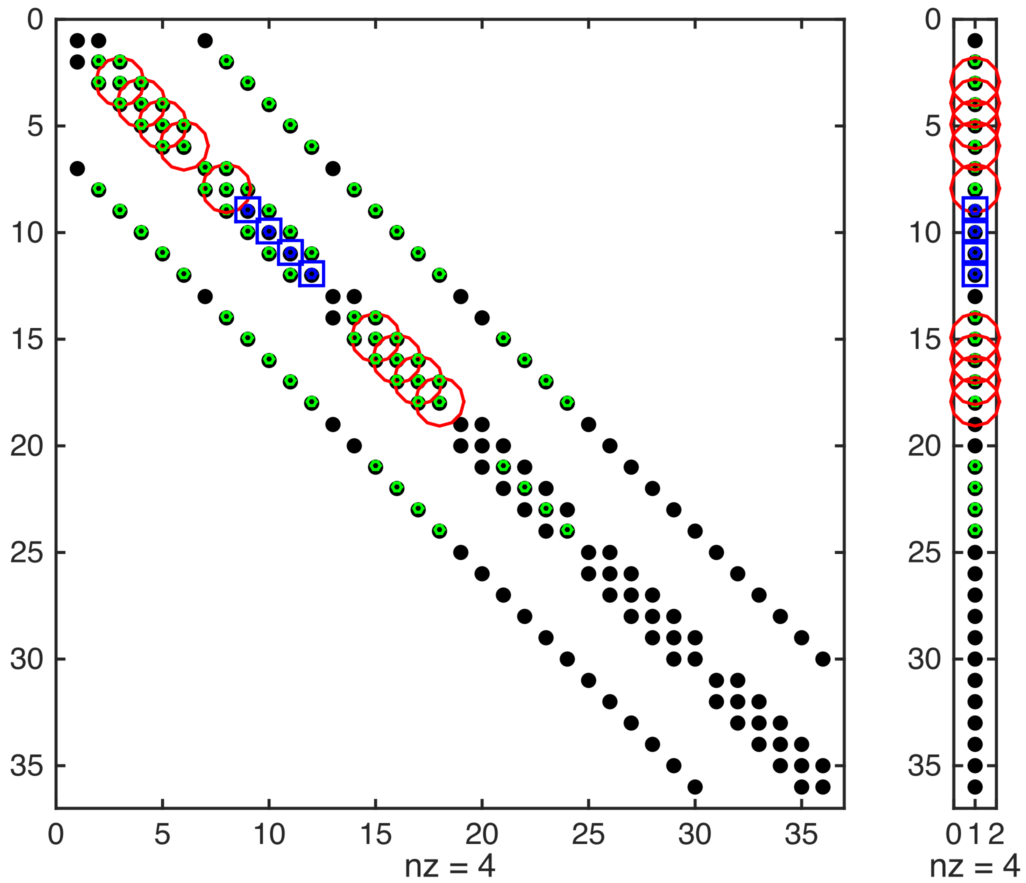


L-1 overlap

Local solver is applied to problem on extended subdomain.

Domain Overlap

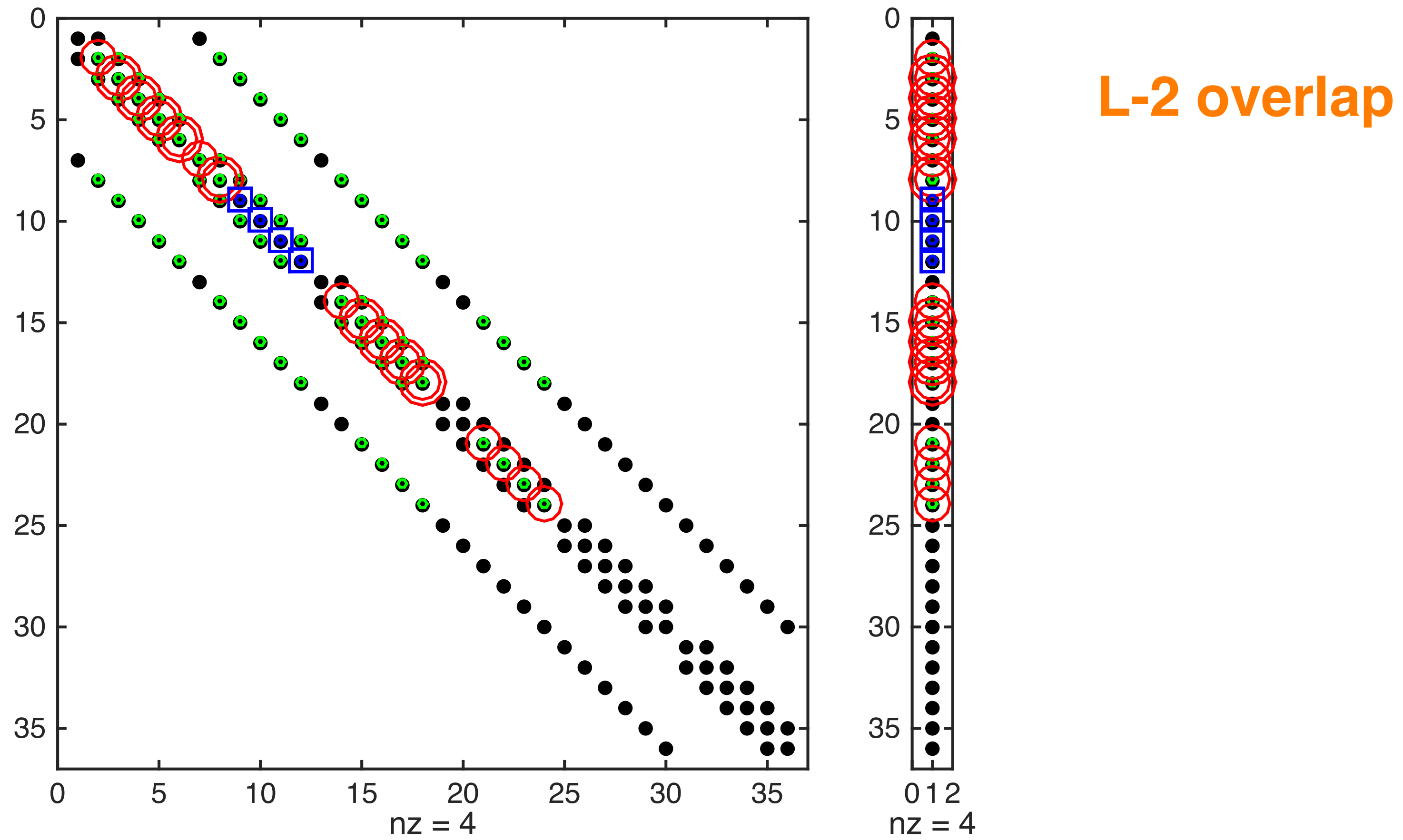
- **Domain overlap based on matrix partitioning**
  - Blocks are extended by components adjacent in the matrix.



Local solver is applied to problem on extended subdomain.

Domain Overlap

- **Domain overlap based on matrix partitioning**
  - Blocks are extended by components adjacent in the matrix.



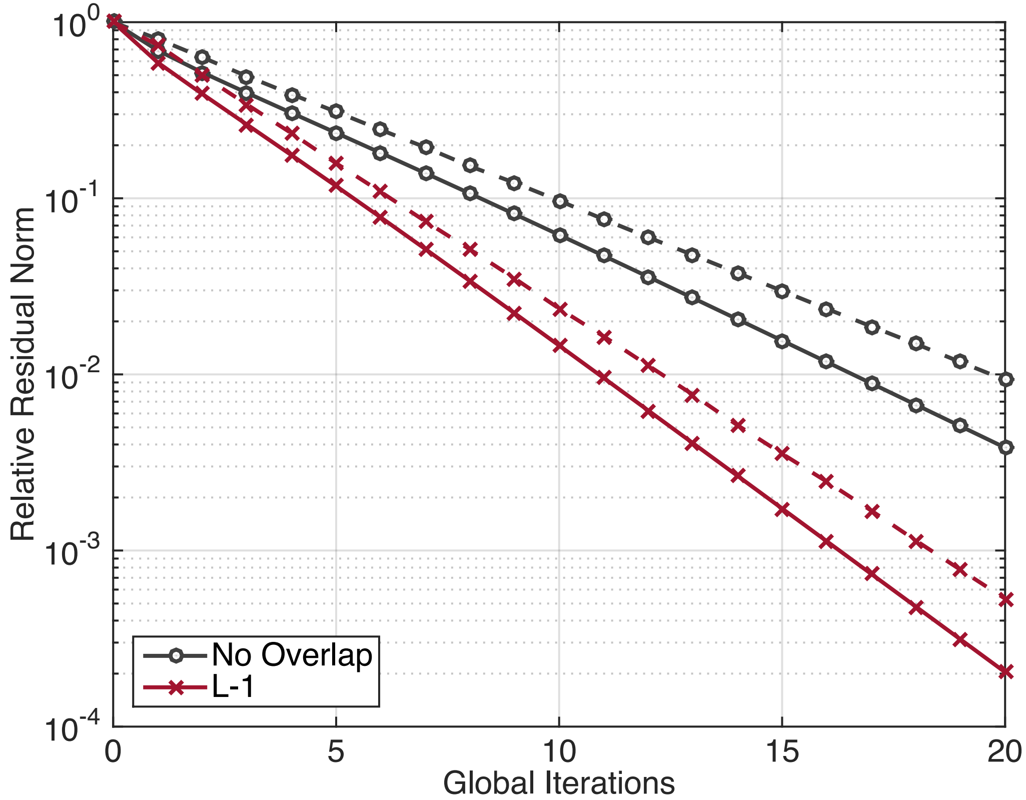
Local solver is applied to problem on extended subdomain.

Experiment Outline

- **Target Problems**
  - **Laplace Problem**, 3D, 27-point stencil, 8x8x8 grid, 47 blocks.
  - **Sparse triangular** systems from ILU(0) preconditioning.
- **Solver Setting**
  - 2 Jacobi sweeps as local solver on the blocks (subdomains).
  - Different update schemes in-between subdomains:
    - **Fixed Gauss-Seidel** top-down subdomain scheduling.
    - **Random** (subdomains are updated once per global iteration).
- **Overlap**
  - **Uniform** overlap derived from matrix partitioning.
  - **Non-uniform / directed** overlap derived from matrix characteristics.
- **Analyze convergence.**
  - **Normalized iterations** account for overhead of overlap.

Alternating Schwarz Convergence

Test case: L3D 27-pt stencil



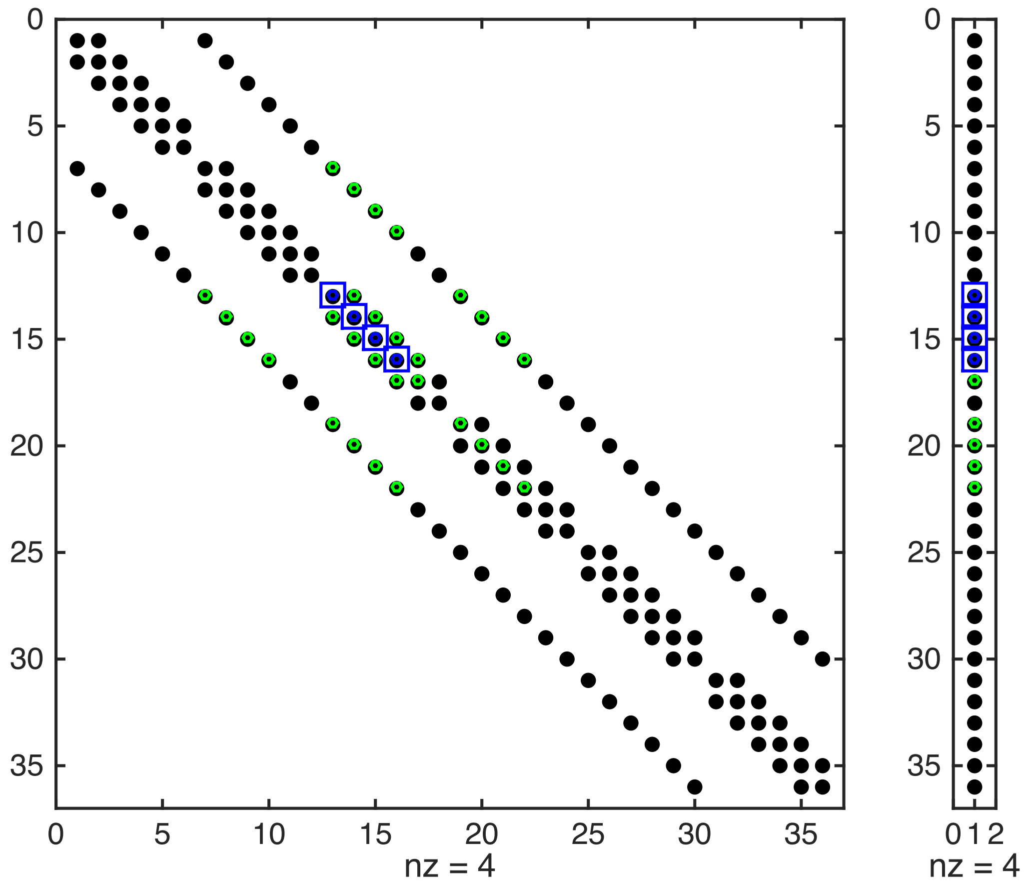
Normalized Iters L0: 1.00

Normalized Iters L1: 6.68

- **Overlap** improves convergence.
- **Random subdomain scheduling** (dashed lines) results in **slower** average **convergence**.

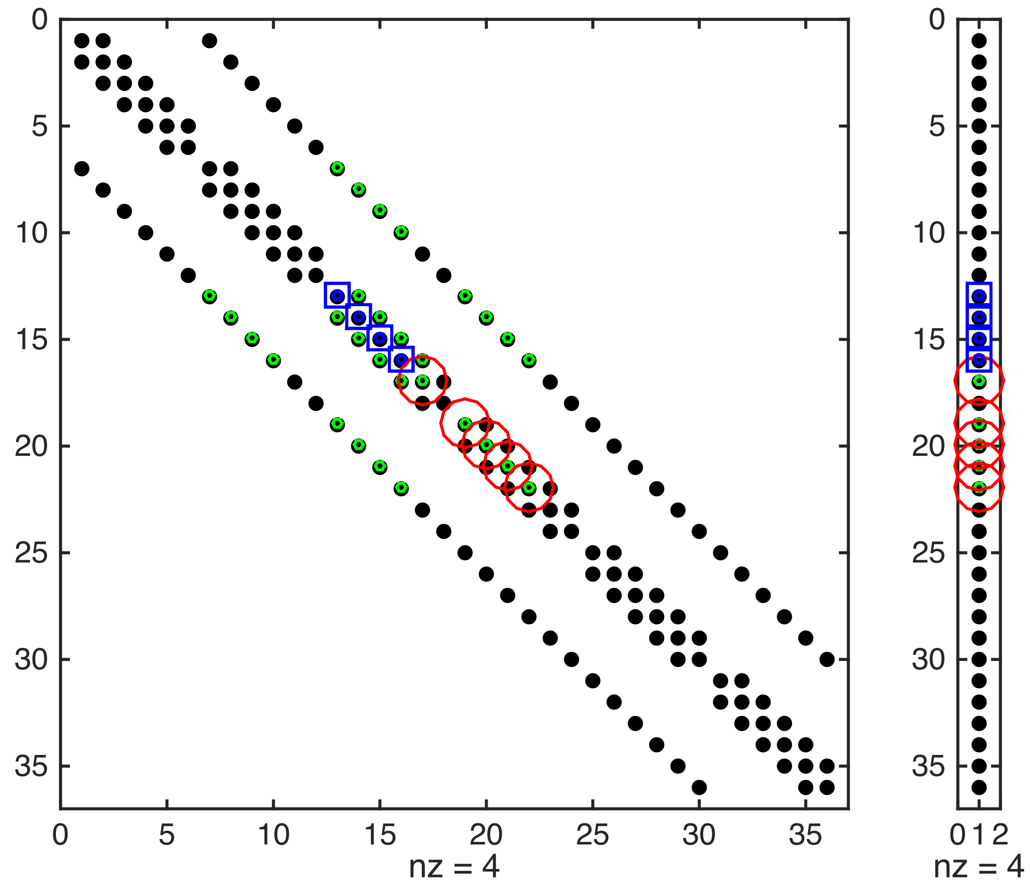
Directed Subdomain Extension

- **Overlap only in one direction, e.g. Top-Down overlap:**



Directed Subdomain Extension

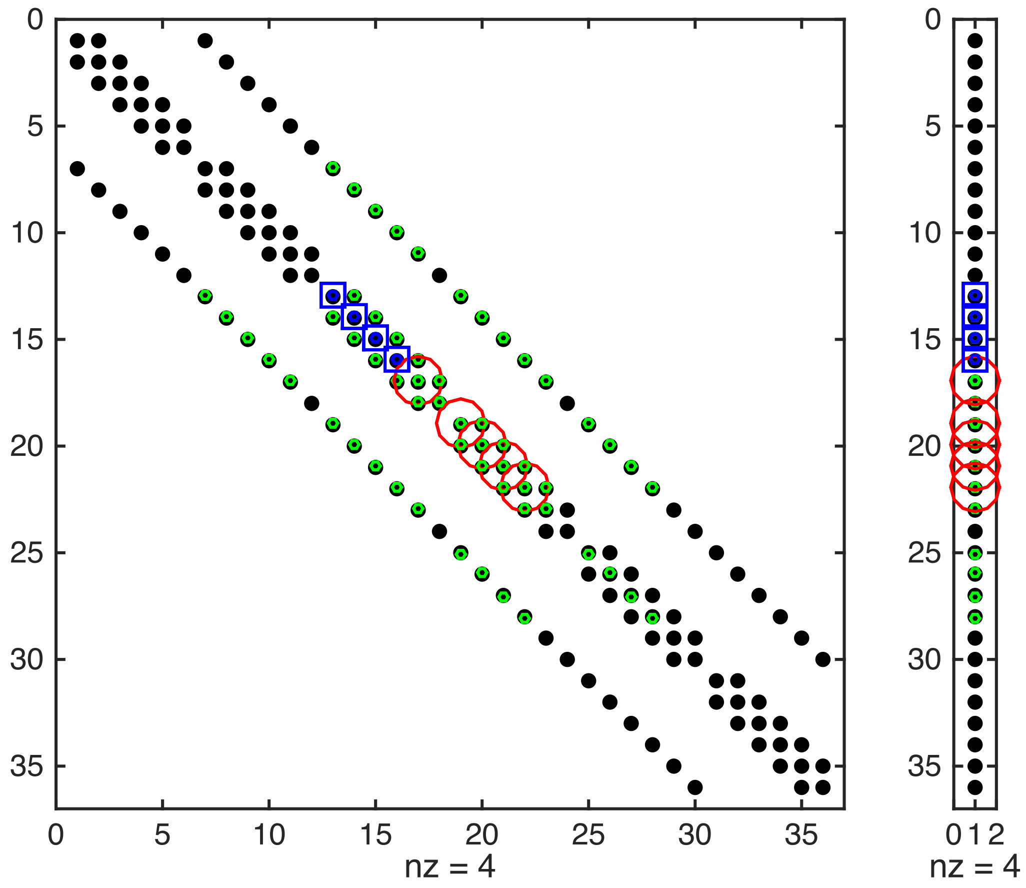
- **Overlap only in one direction, e.g. Top-Down overlap:**



L-1 overlap

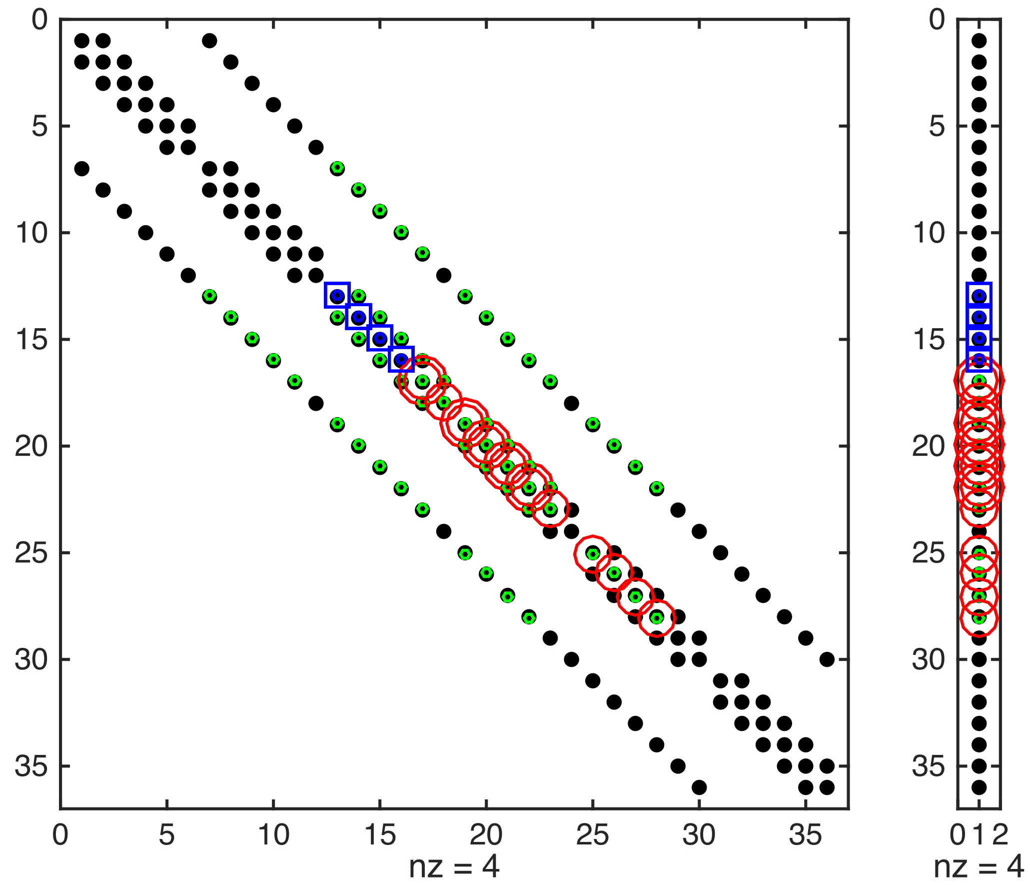
Directed Subdomain Extension

- **Overlap only in one direction, e.g. Top-Down overlap:**



Directed Subdomain Extension

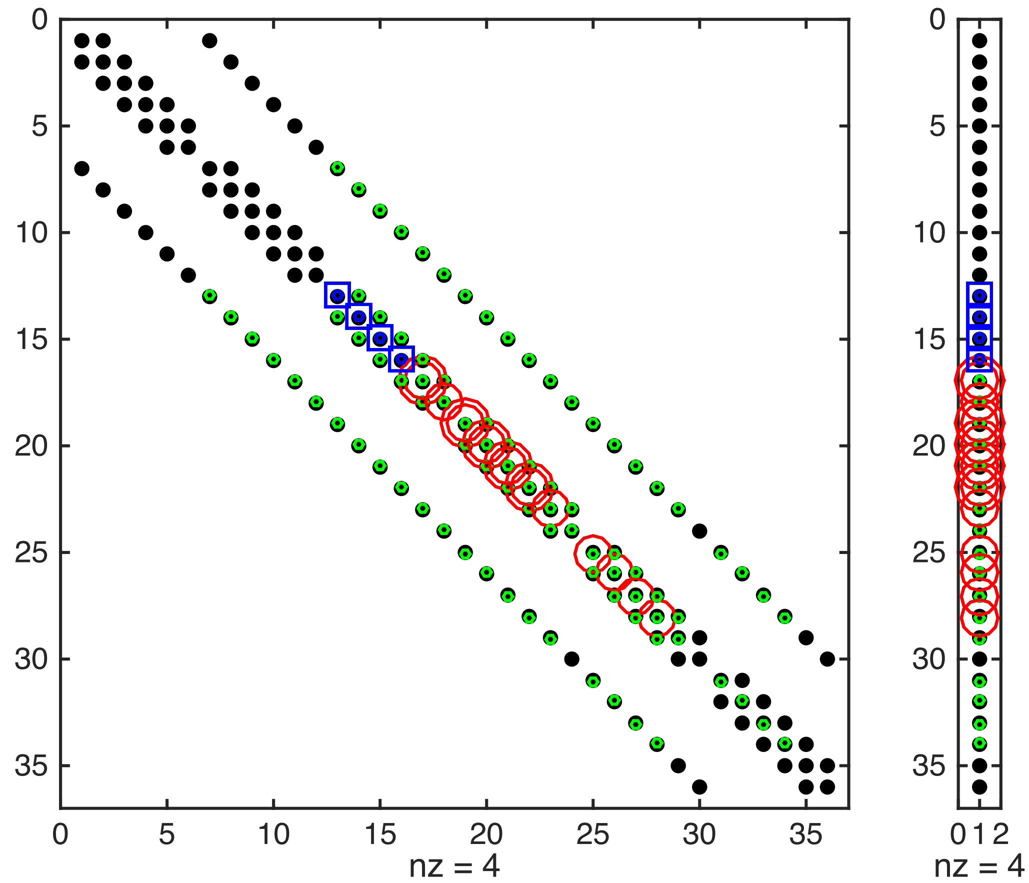
- **Overlap only in one direction, e.g. Top-Down overlap:**



L-2 overlap

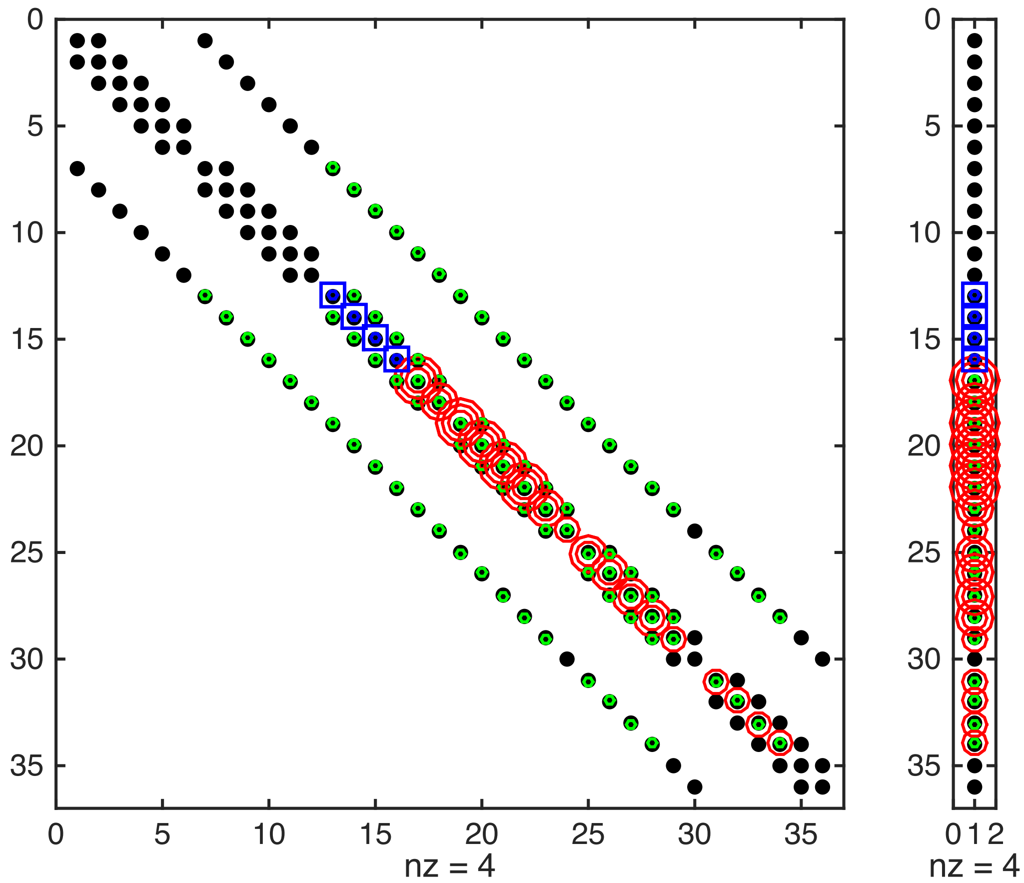
Directed Subdomain Extension

- **Overlap only in one direction, e.g. Top-Down overlap:**



Directed Subdomain Extension

- **Overlap only in one direction, e.g. Top-Down overlap:**

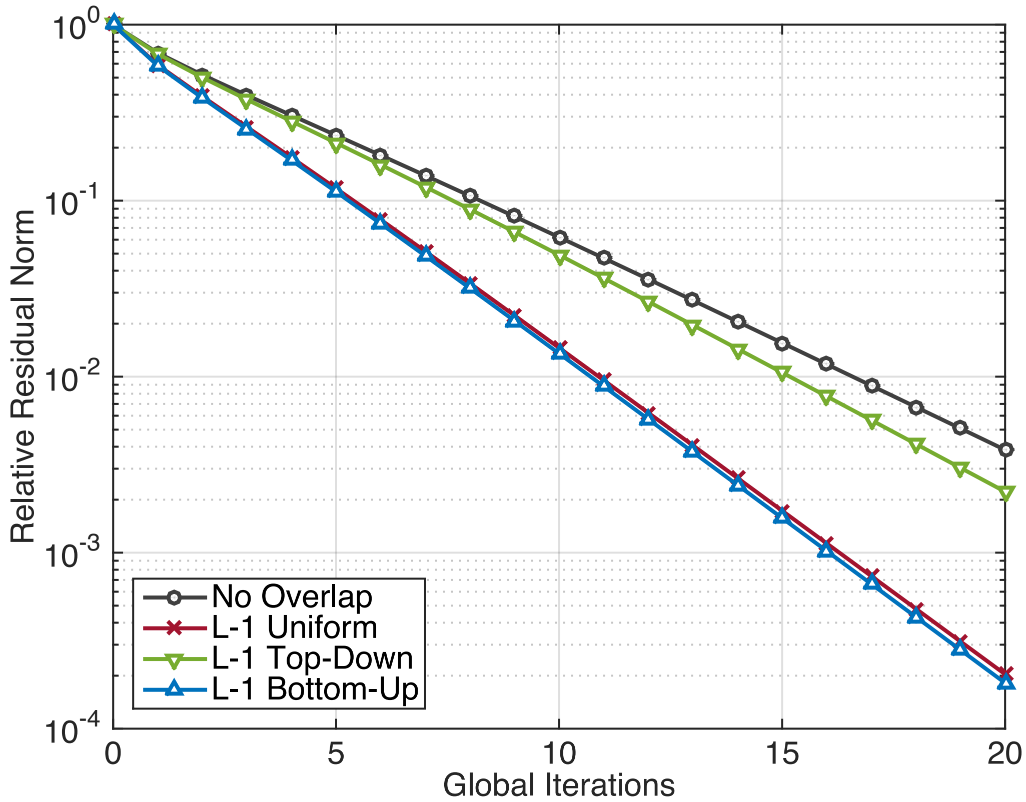


L-3 overlap

Subdomain only grows in one direction!

Directed Subdomain Extension

Test case: L3D 27-pt stencil



Normalized Iters L0: 1.00

Normalized Iters L1: 6.68

Normalized Iters L1: 3.83

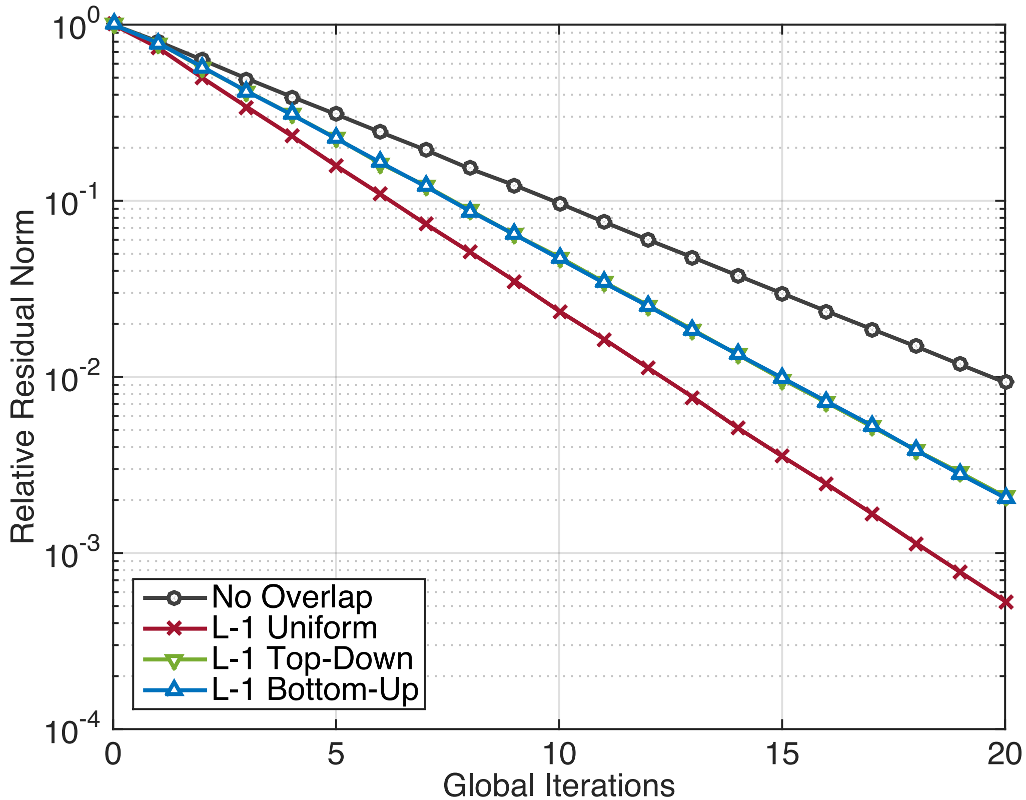
Normalized Iters L1: 3.83

For top-down subdomain scheduling:

- Bottom-Up overlap propagates all new information available in the uniform extension - at lower computational cost.
- Top-Down overlap provides almost no convergence benefit.

Directed Subdomain Extension

Test case: L3D 27-pt stencil



Normalized Iters L0: 1.00
Normalized Iters L1: 6.68
Normalized Iters L1: 3.83
Normalized Iters L1: 3.83

For random subdomain scheduling:

- No difference between Top-Down and Bottom-Up overlap.
- Symmetric matrix properties in combination with random update scheduling removes advantage of non-uniform extensions.

Sparse Triangular Systems

- **Clear information dependency**
  - Synchronous top-down subdomain scheduling results in (block-) substitution. For blocks containing one unknown, exact solve.
  - **Propagation** of **new information** in dependency direction is key.
  - Faster convergence expected for the scheduling:

    top-down in **Ly=b** and bottom-up in **Ux=y**.

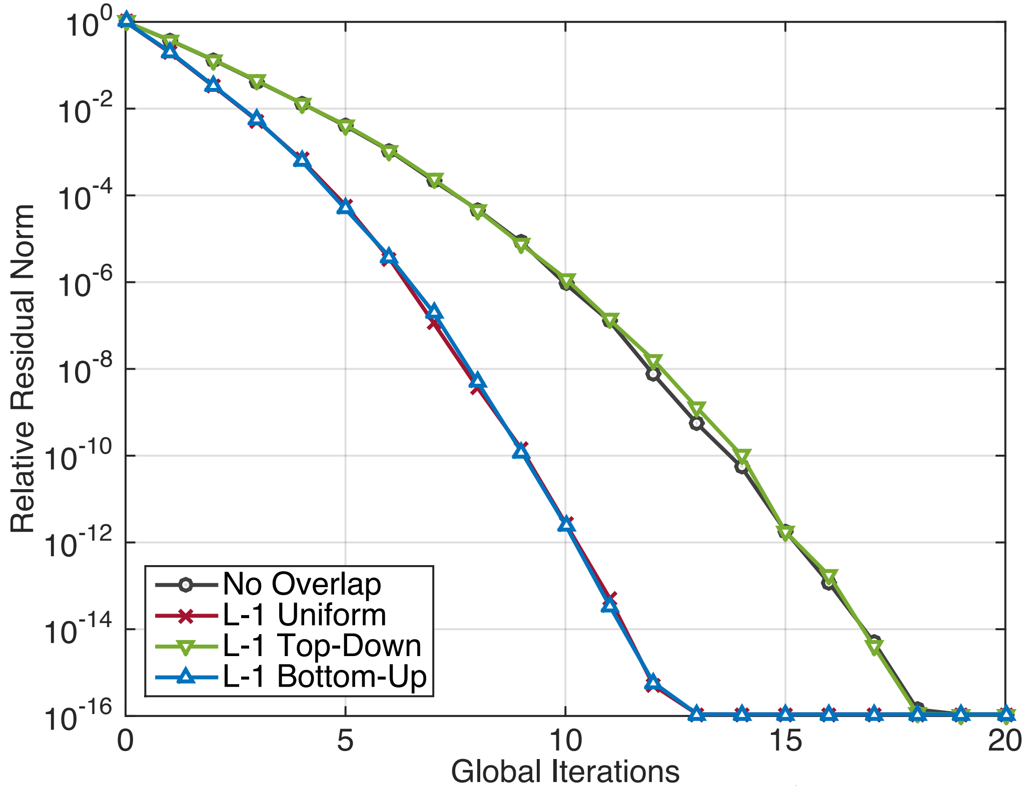
Sparse Triangular Systems

- **Clear information dependency**
  - Synchronous top-down subdomain scheduling results in (block-) substitution. For blocks containing one unknown, exact solve.
  - **Propagation** of **new information** in dependency direction is key.
  - Faster convergence expected for the scheduling:
    top-down in **Ly=b** and bottom-up in **Ux=y**.

- **Random subdomain scheduling**
  - No information on subdomain scheduling**.**
  - **Overlap useful** if it propagates **new information**.
  - **Directed subdomain extension opposite dependency direction**.

Directed Subdomain Extension

Test case: ILU(0) for L3D 27-pt stencil

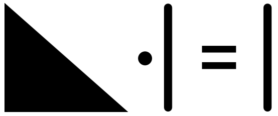


Normalized Iters L0: 1.00

Normalized Iters L1: 6.68

Normalized Iters L1: 3.83

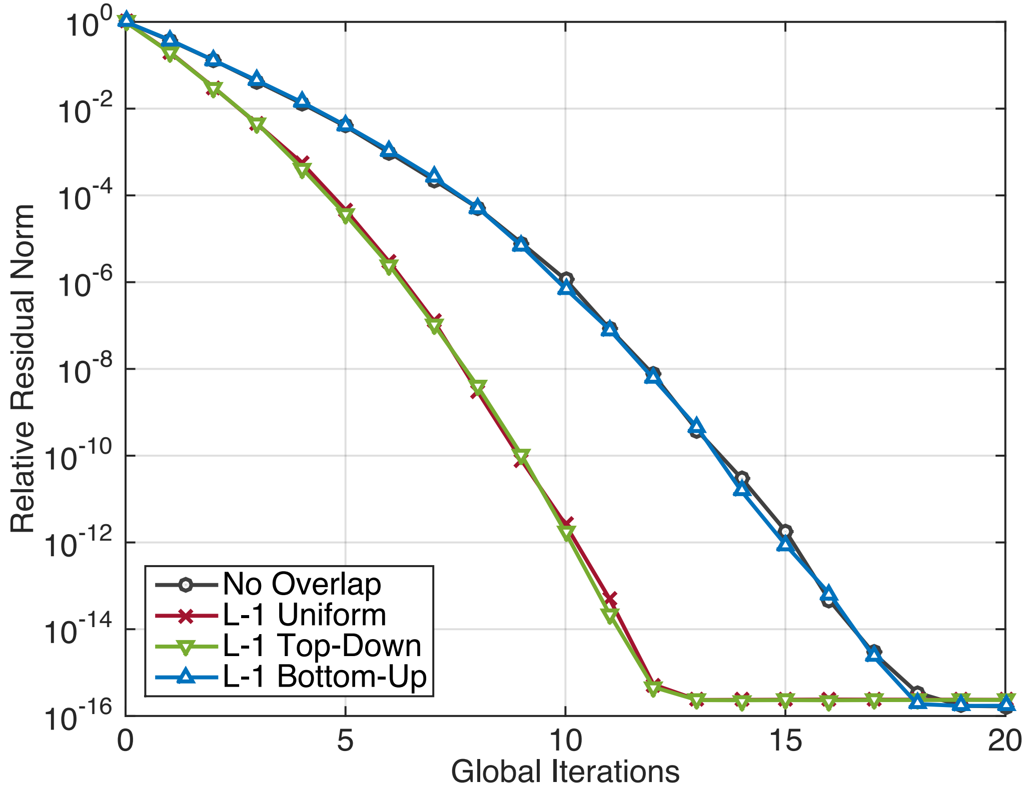
Normalized Iters L1: 3.83

For lower triangular system: 

- L-1 Bottom-Up overlap propagates all new information available in the uniform extension (left) - at lower computational cost (right).
- L-1 Top-Down overlap useless due to dependency in linear system.

Directed Subdomain Extension

Test case: ILU(0) for L3D 27-pt stencil

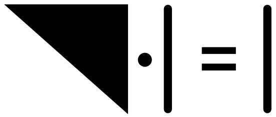


Normalized Iters L0: 1.00

Normalized Iters L1: 6.68

Normalized Iters L1: 3.83

Normalized Iters L1: 3.83

For upper triangular system: 

- L-1 Top-Down overlap propagates all new information available in the uniform extension (left) - at lower computational cost (right).
- L-1 Bottom-Up overlap useless due to dependency in linear system.

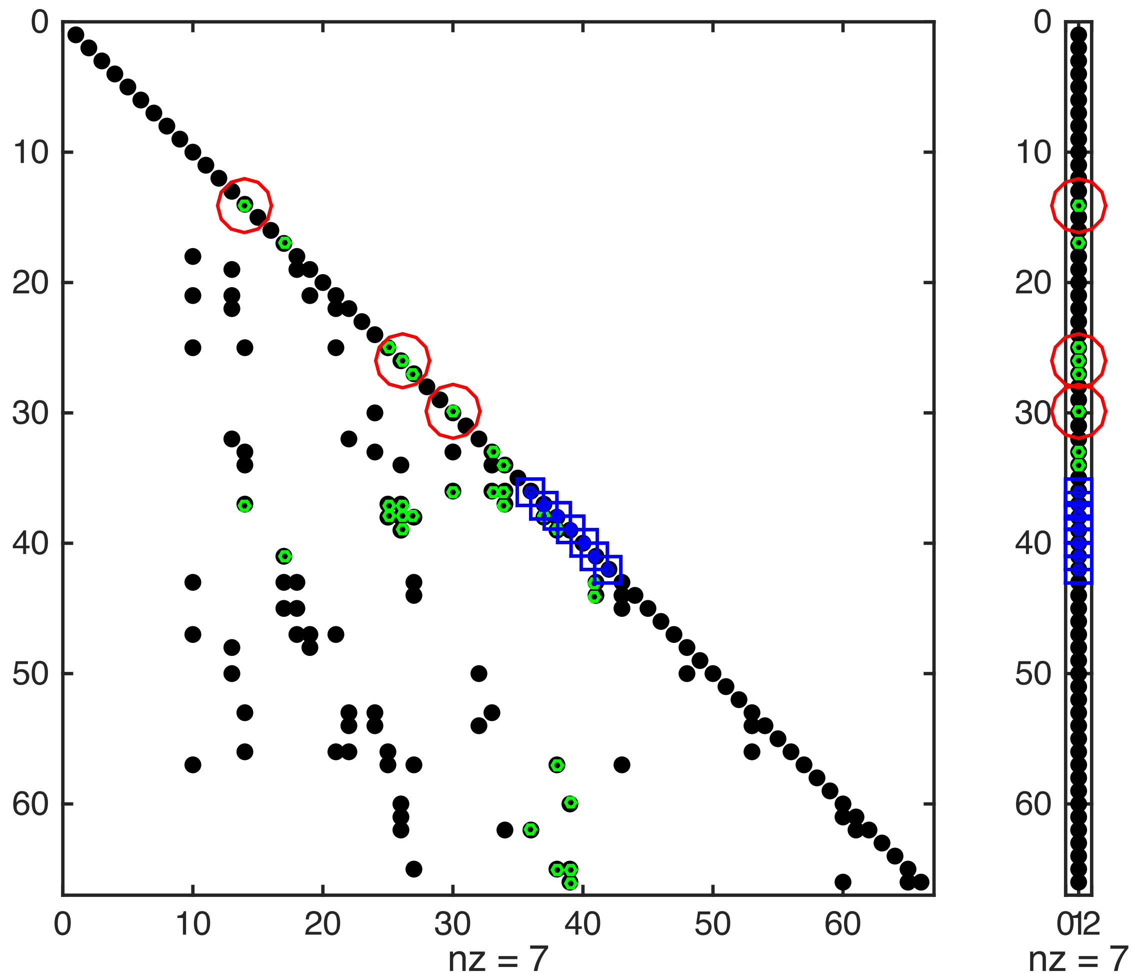
Anisotropic Problems

- **Anisotropic Fluid Flow**
  - E.g. groundwater flow.
  - Non-symmetric convection.
  - Information gets propagated faster in one than another direction.
  - Represented by different magnitude of matrix entries connecting unknowns.

- **Non-Uniform overlap accounting for Anisotropy**
  - Subdomain extended only by some candidates (largest matrix entries).
  - Recursive application of this strategy potentially results in subdomains that are different to any uniform extension.

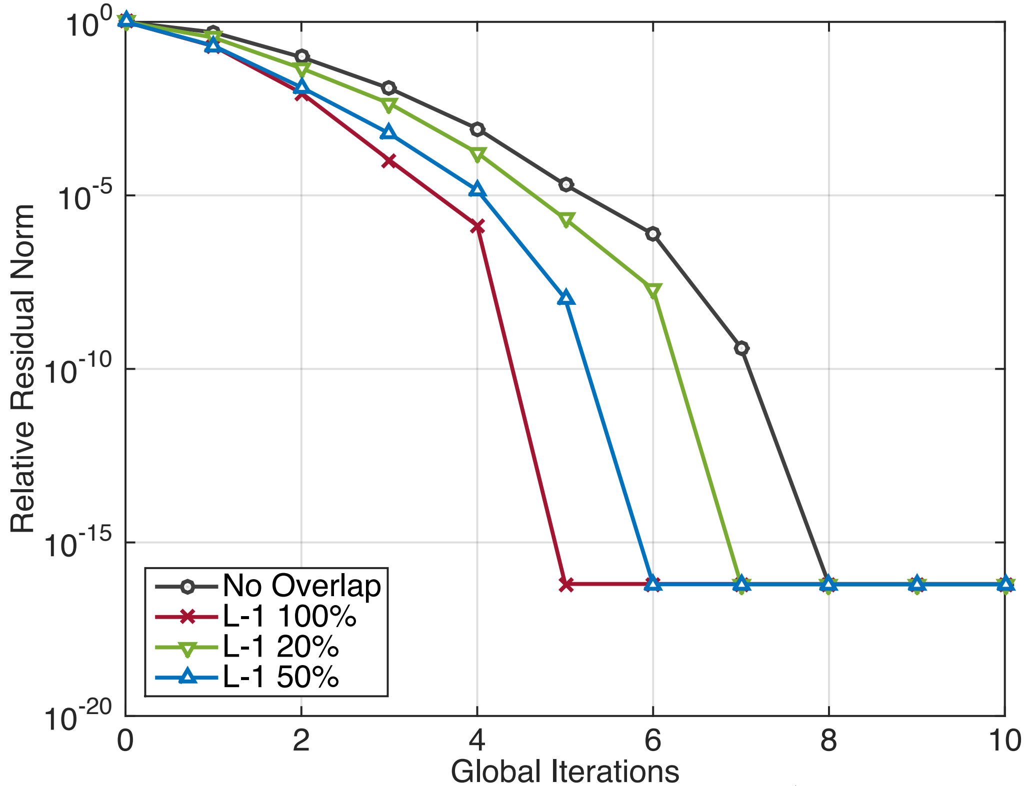
Anisotropic Problems

- Sparse triangular systems of ILU(0) for anisotropic fluid flow problem.
- Directed overlap opposite propagation direction.
- Recursive domain extension with only with some of the candidates.



Anisotropic Problems

Test case: ILU(0) for anisotropic problem



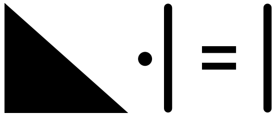
Normalized Iters L0: 1.00

Normalized Iters L1: 2.38

Normalized Iters L1: 1.33

Normalized Iters L1: 1.83

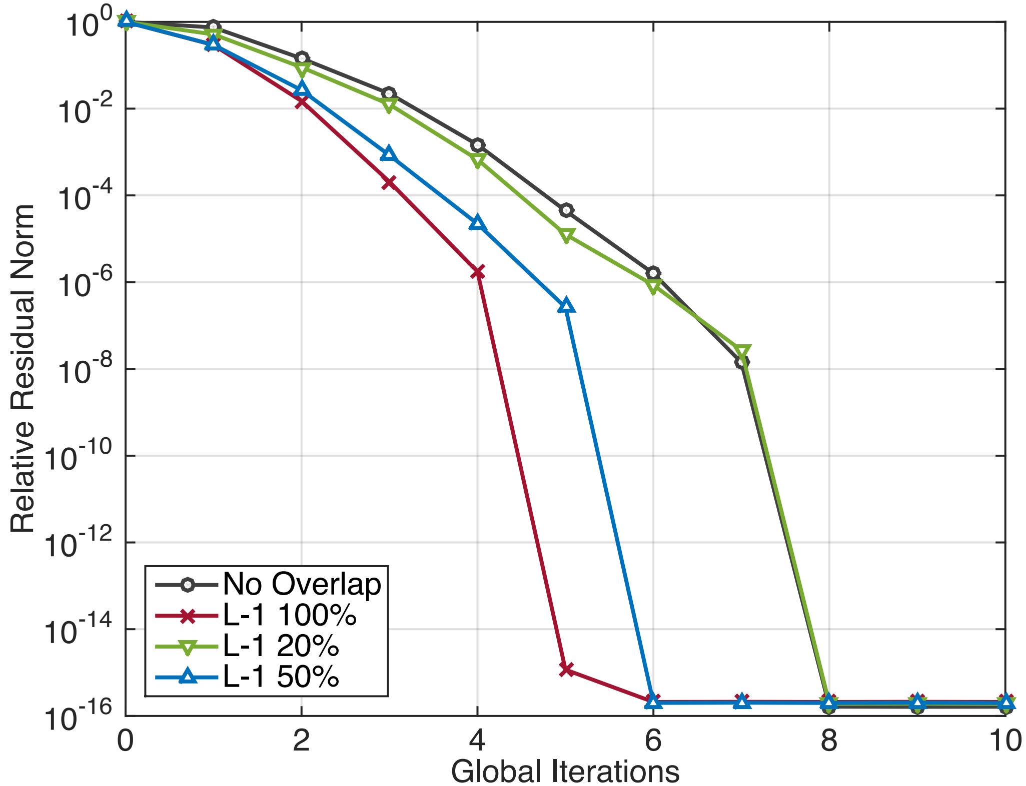
For lower triangular system:



- L-1 50% propagates all new information fast (left) and at low computational cost (right). (All Bottom-Up overlap.)
- L-1 20% is computationally cheaper, needs more global iterations.

Anisotropic Problems

Test case: ILU(0) for anisotropic problem



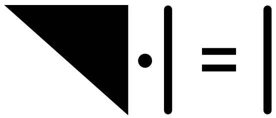
Normalized Iters L0: 1.00

Normalized Iters L1: 2.81

Normalized Iters L1: 1.35

Normalized Iters L1: 1.93

For upper triangular system:



- L-1 50% / L-1 80% matches L-1 100% in global iterations. (All Top-Down overlap.)
- L-1 50% computationally cheaper than L-1 100%.

Conclusion

- **Directed** overlap propagates information in a **certain direction**.
- Directed overlap **opposite** subdomain **scheduling** direction propagates only **new information**.
- For **triangular systems**, directed overlap **opposite dependency** works also for **random** subdomain **scheduling**.
- **Non-Uniform overlap** accounting for **anisotropy** propagates most **important information**.

This research is based on a cooperation with Edmond Chow from Georgia Institute of Technology, Daniel Szyld from the Temple University in Philadelphia, and supported by the U.S. Department of Energy.

U.S. DEPARTMENT OF ENERGY

Normalized Iters Iterations

Ω_i Subdomains without overlap

$\bar{\Omega}_i$ Subdomains with overlap

$FP\left(\Omega_i\right)$ Floating point operations in local solver for subdomain

k Iterations

Normalized Iters Iterations:

$$\tilde{k} = k \cdot \frac{\sum_i FP\left(\bar{\Omega}_i\right)}{\sum_i FP\left(\Omega_i\right)}$$