Smoothed Aggregation (SA)

- Two most common types of AMG
  - Classic F/C-style AMG, covered previously
  - Smoothed Aggregation-Based AMG (SA)
- Smoothed Aggregation-Based AMG
  - New setup phase, but goal is same, i.e., build coarse grids
    - Select coarse grids based on "aggregation", not C-points
    - Define "smoothed" interpolation: $I_{m+1}^{m},\ m = 1, 2, \ldots$
    - Define coarse-grid operators as before:
      $A^{m+1} = (I_{m+1}^{m})^T A^m I_{m+1}^{m},\ m = 1, 2, \ldots$
  - Identical solve phase
    - Same old V-cycles, W-cycles, F-cycles, etc...

Smoothed Aggregation (SA)

- New setup phase, but same goal of building coarse grids
  - Construct prolongation to capture algebraically smooth error
  - SA assumes a priori knowledge of algebraically smooth error
    - Given user-provided "near null-space" mode(s) denoted B_1
    - Assuming these modes is not "cheating"
      - F/C-style AMG assumes slowly varying smooth error
  - For many problems, these modes are known
    - Look to null-space of PDE with no boundary conditions
    - For diffusion, $B_1 = 1$
    - For elasticity, B_1 represents rigid body modes

Constructing Prolongation

Algorithm Step	Desired Properties of I_{m+1}^m	SA Algorithm
1	Sparse	Sparsity outline determined by aggregation
2	Each block column describes smooth error locally for a neighborhood of dofs	Construct I_{m+1}^m by injecting B_m into sparsity outline
3	$\mathrm{span}(I_{m+1}^m)$ globally describes algebraically smooth error	Globally smooth I_{m+1}^m e.g., with weighted-Jacobi

Basic Aggregation Algorithm

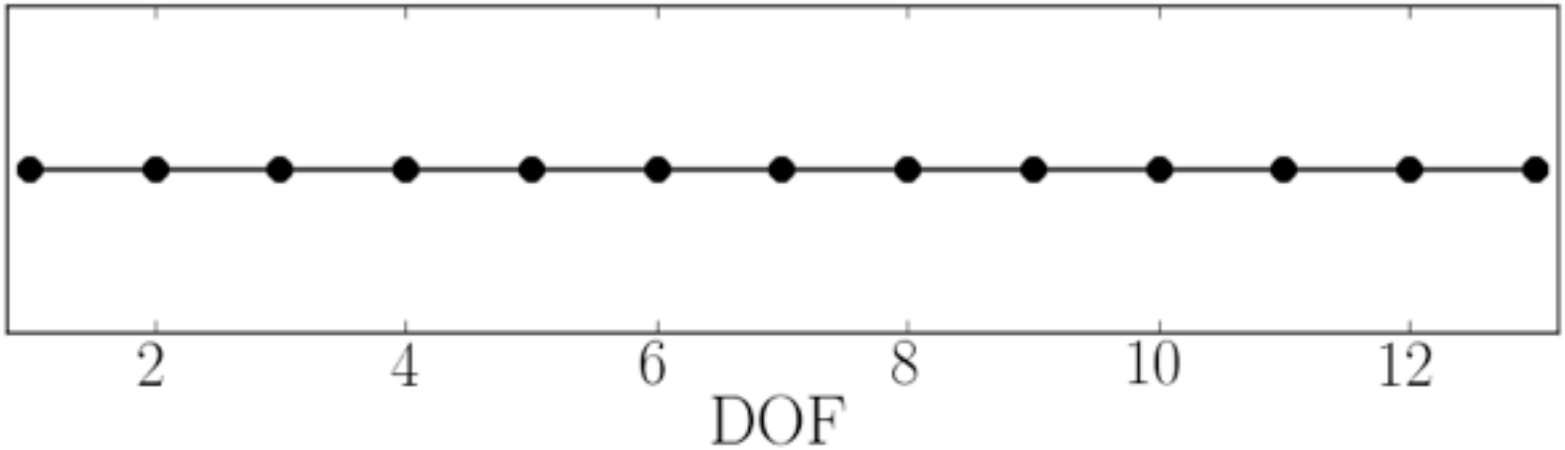
- Aggregate by applying greedy graph algorithm to strength-of-connection graph
- Each aggregate Ω_j^m is a set of locally connected dofs
  - Aggregates are disjoint, $\Omega_j^m \cap \Omega_k^m = \emptyset, \text{ if } j \neq k$
  - Aggregates cover the set of all dofs on level m

    $$\bigcup_j \Omega_j^m = \{0, 1, \ldots, n_m\}$$

    - where n_m is the number of dofs on level m
- Each aggregate defines a local interpolation neighborhood

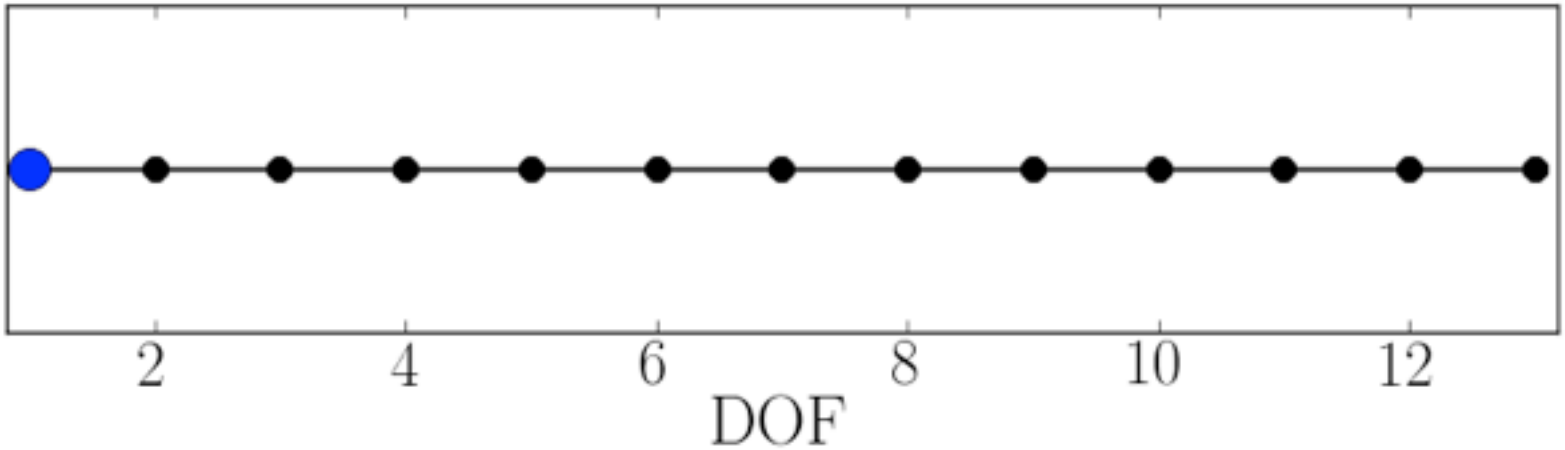
Basic Aggregation Algorithm

Sample 1D Laplace Strength-of-Connection Graph



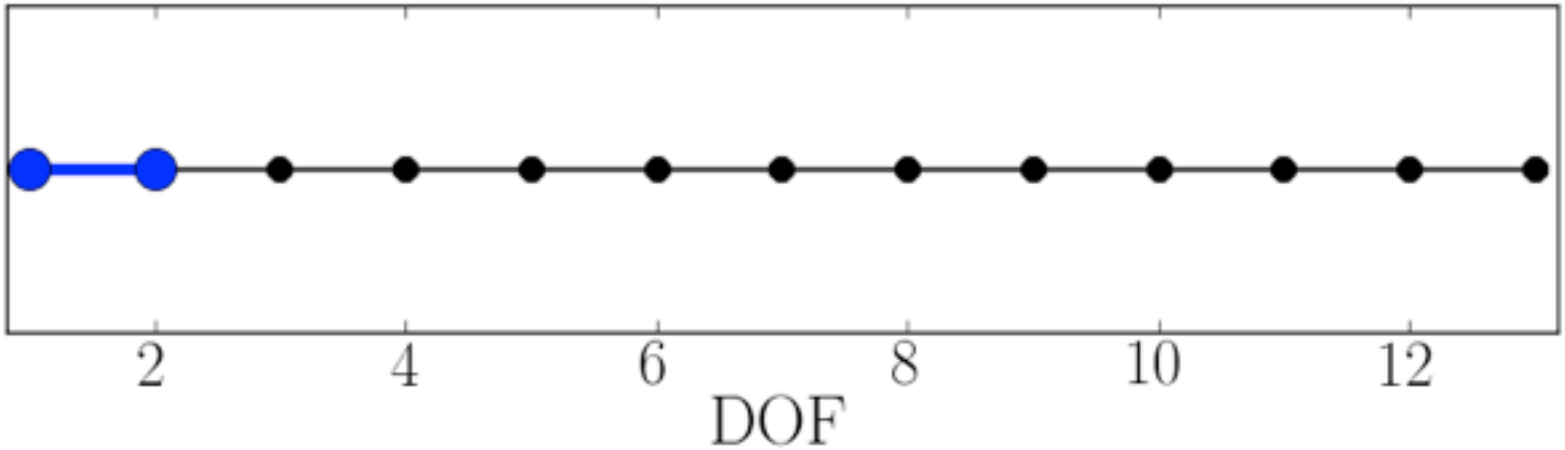
Basic Aggregation Algorithm

Choose initial unaggregated dof



Basic Aggregation Algorithm

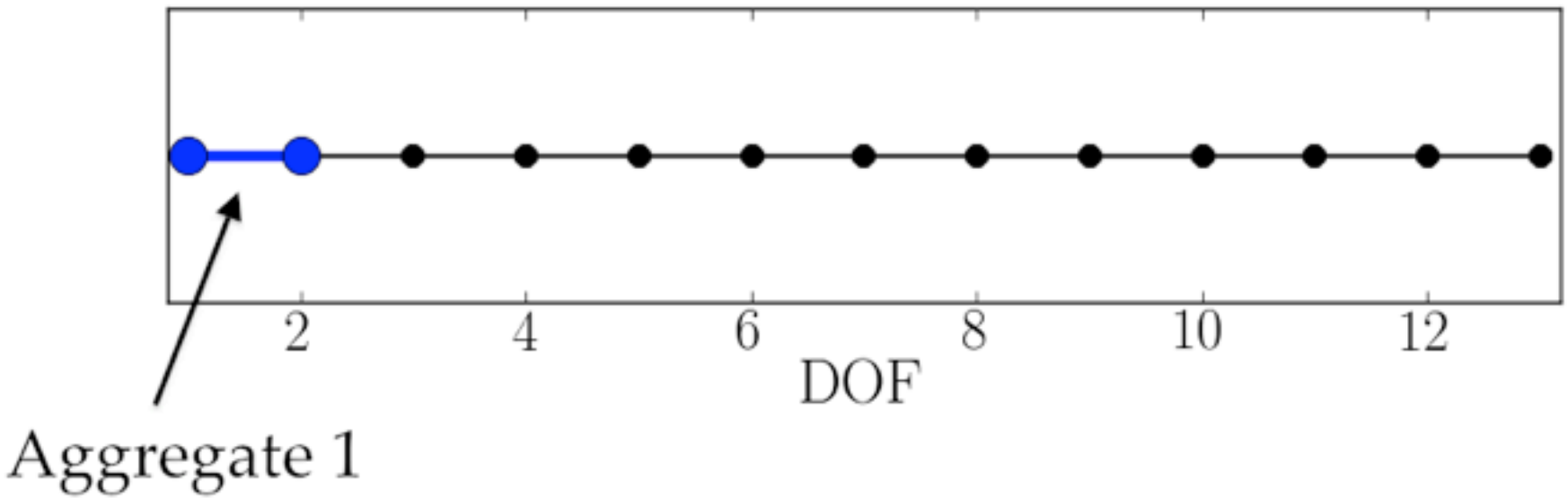
Place neighbor(s) into first aggregate



Basic Aggregation Algorithm

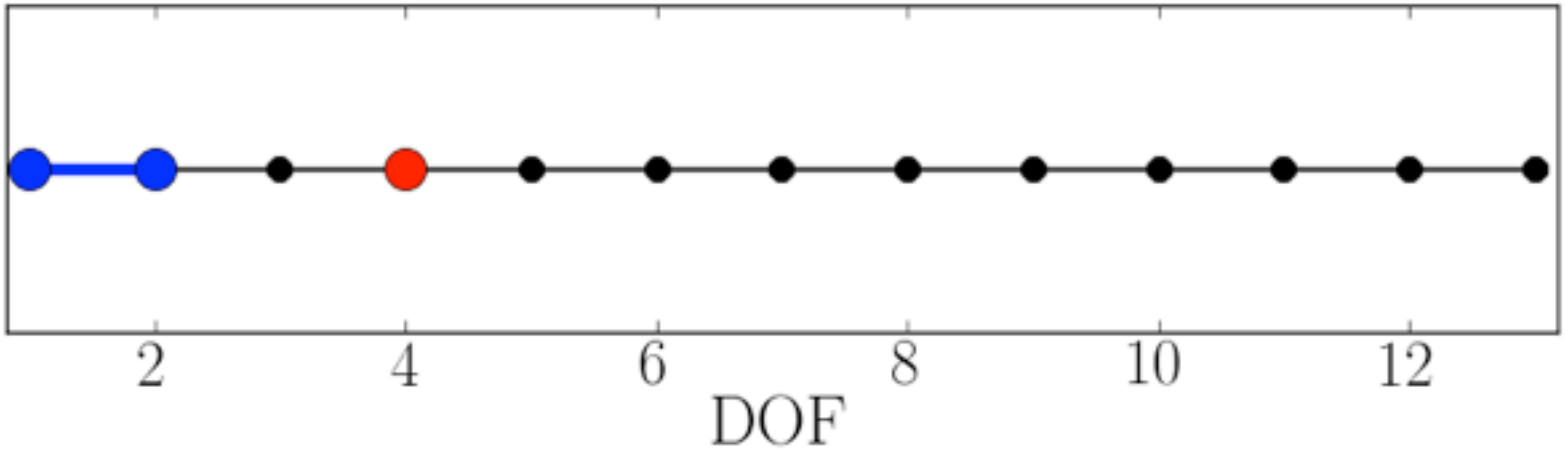
$$\Omega_1^1 = \{1,\ 2\}$$

First aggregate contains dofs 1 and 2



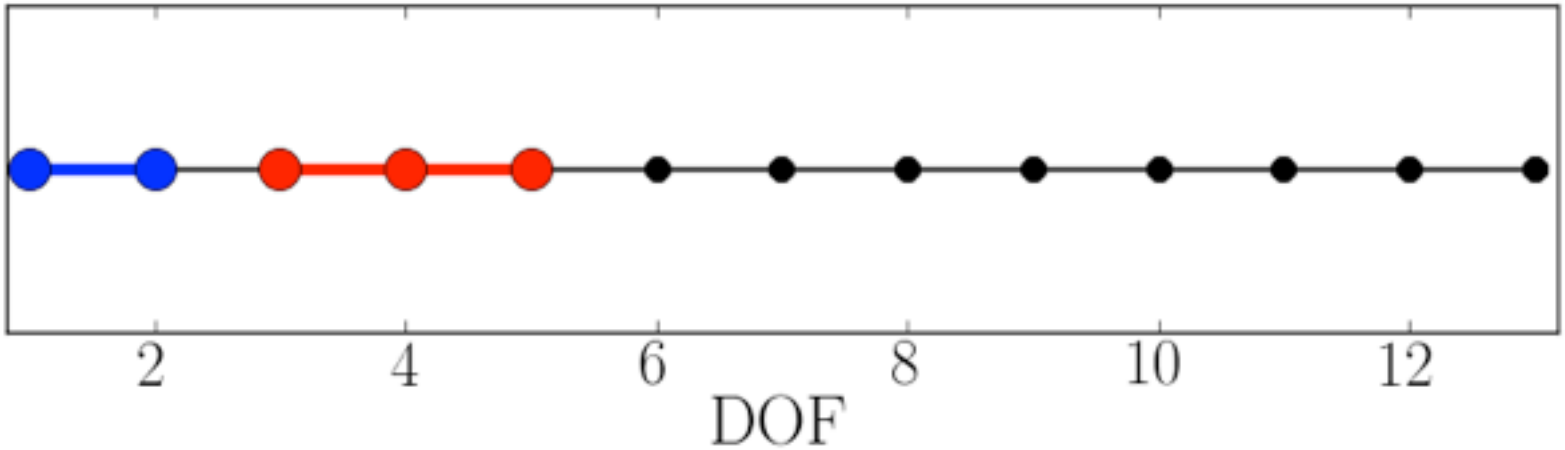
Basic Aggregation Algorithm

Choose next unaggregated dof, that has all unaggregated neighbors



Basic Aggregation Algorithm

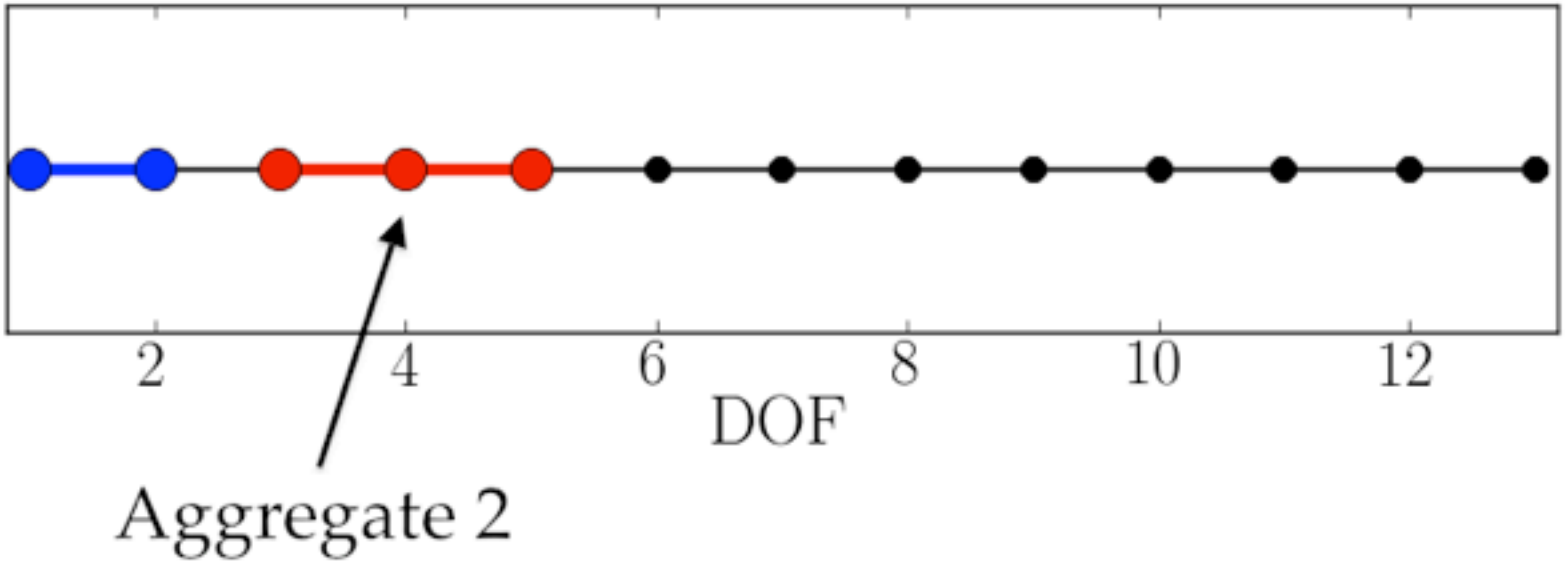
Place neighbor(s) into second aggregate



Basic Aggregation Algorithm

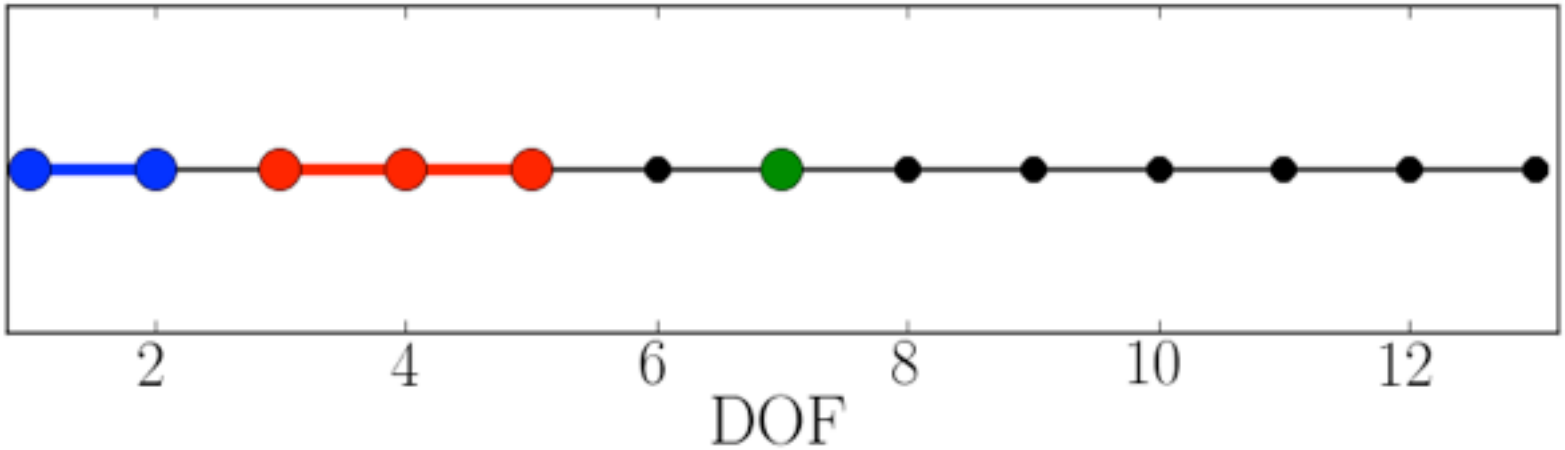
$$\Omega_2^1 = \{3,\ 4,\ 5\}$$

Second aggregate contains dofs 3, 4 and 5



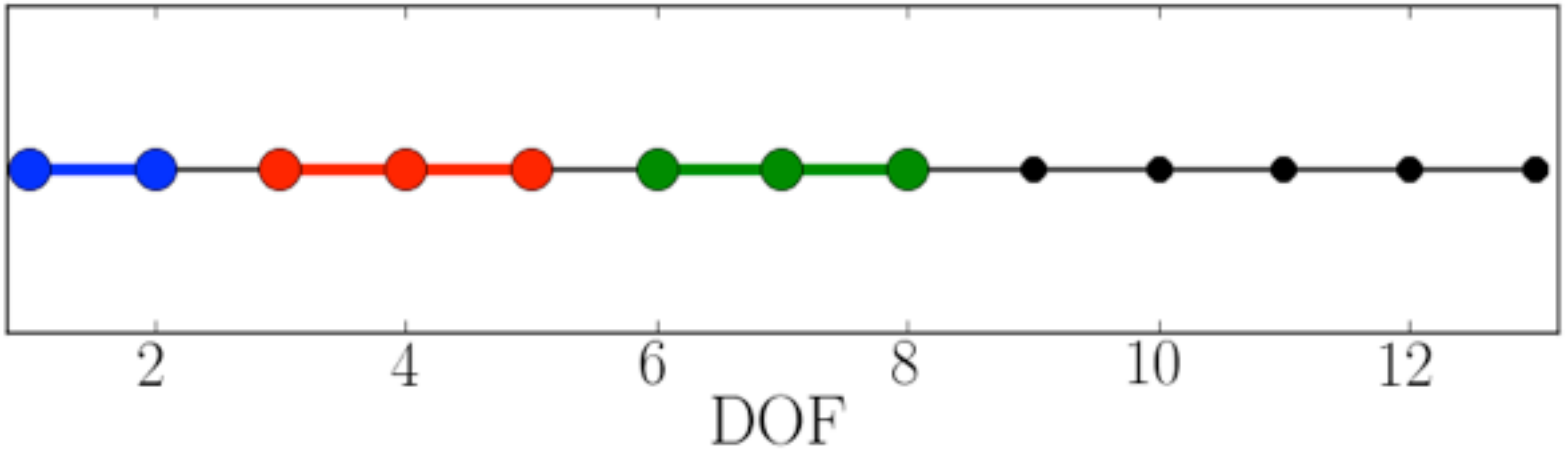
Basic Aggregation Algorithm

Choose next unaggregated dof, that has all unaggregated neighbors



Basic Aggregation Algorithm

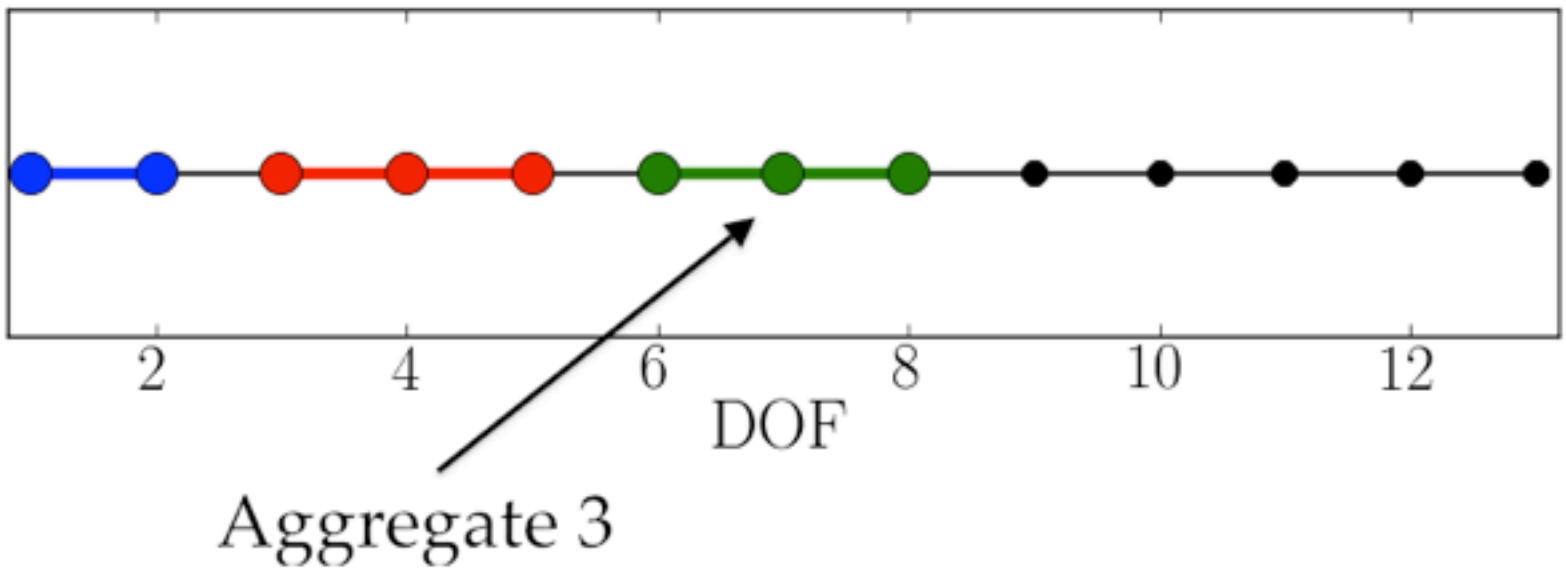
Place neighbor(s) into third aggregate



Basic Aggregation Algorithm

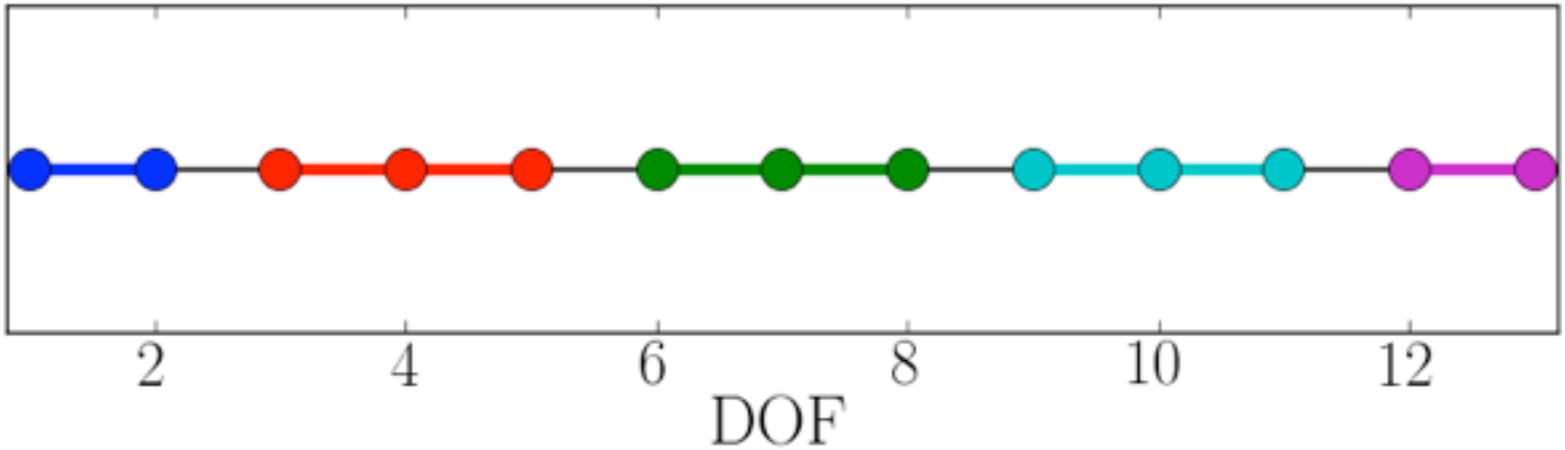
$$\Omega_3^1 = \{6,\ 7,\ 8\}$$

Third aggregate contains dofs 6, 7 and 8



Basic Aggregation Algorithm

Repeat process to obtain last two aggregates



Sparsity Outline for I_{m+1}^{m}

- Aggregation induces sparsity outline
  - Each aggregate corresponds to one block column
  - Each block column is nonzero only for dofs in that aggregate
  - Example yields sparsity outline with 5 block columns

$$\begin{bmatrix}
* \cdots * & & & & \\
* \cdots * & & & & \\
& * \cdots * & & & \\
& * \cdots * & & & \\
& * \cdots * & & & \\
& & * \cdots * & & \\
& & * \cdots * & & \\
& & * \cdots * & & \\
& & & * \cdots * & \\
& & & * \cdots * & \\
& & & * \cdots * & \\
& & & & * \cdots * \\
& & & & * \cdots *
\end{bmatrix}$$

Construct Tentative I_{m+1}^{m}

- Goal: each block column locally describes smooth error over its aggregate
- Solution: inject B_m into sparsity outline
- Let $b_{i,:}$ be ith row of B_m

$$B_m \quad \Rightarrow \quad I_{m+1}^{m}$$

$$\begin{bmatrix} b_{1,:} \\ b_{2,:} \\ b_{3,:} \\ b_{4,:} \\ b_{5,:} \\ b_{6,:} \\ b_{7,:} \\ b_{8,:} \\ b_{9,:} \\ b_{10,:} \\ b_{11,:} \\ b_{12,:} \\ b_{13,:} \end{bmatrix} \Rightarrow \begin{bmatrix} b_{1,:} & 0 & 0 & 0 & 0 \\ b_{2,:} & 0 & 0 & 0 & 0 \\ 0 & b_{3,:} & 0 & 0 & 0 \\ 0 & b_{4,:} & 0 & 0 & 0 \\ 0 & b_{5,:} & 0 & 0 & 0 \\ 0 & 0 & b_{6,:} & 0 & 0 \\ 0 & 0 & b_{7,:} & 0 & 0 \\ 0 & 0 & b_{8,:} & 0 & 0 \\ 0 & 0 & 0 & b_{9,:} & 0 \\ 0 & 0 & 0 & b_{10,:} & 0 \\ 0 & 0 & 0 & b_{11,:} & 0 \\ 0 & 0 & 0 & 0 & b_{12,:} \\ 0 & 0 & 0 & 0 & b_{13,:} \end{bmatrix}$$

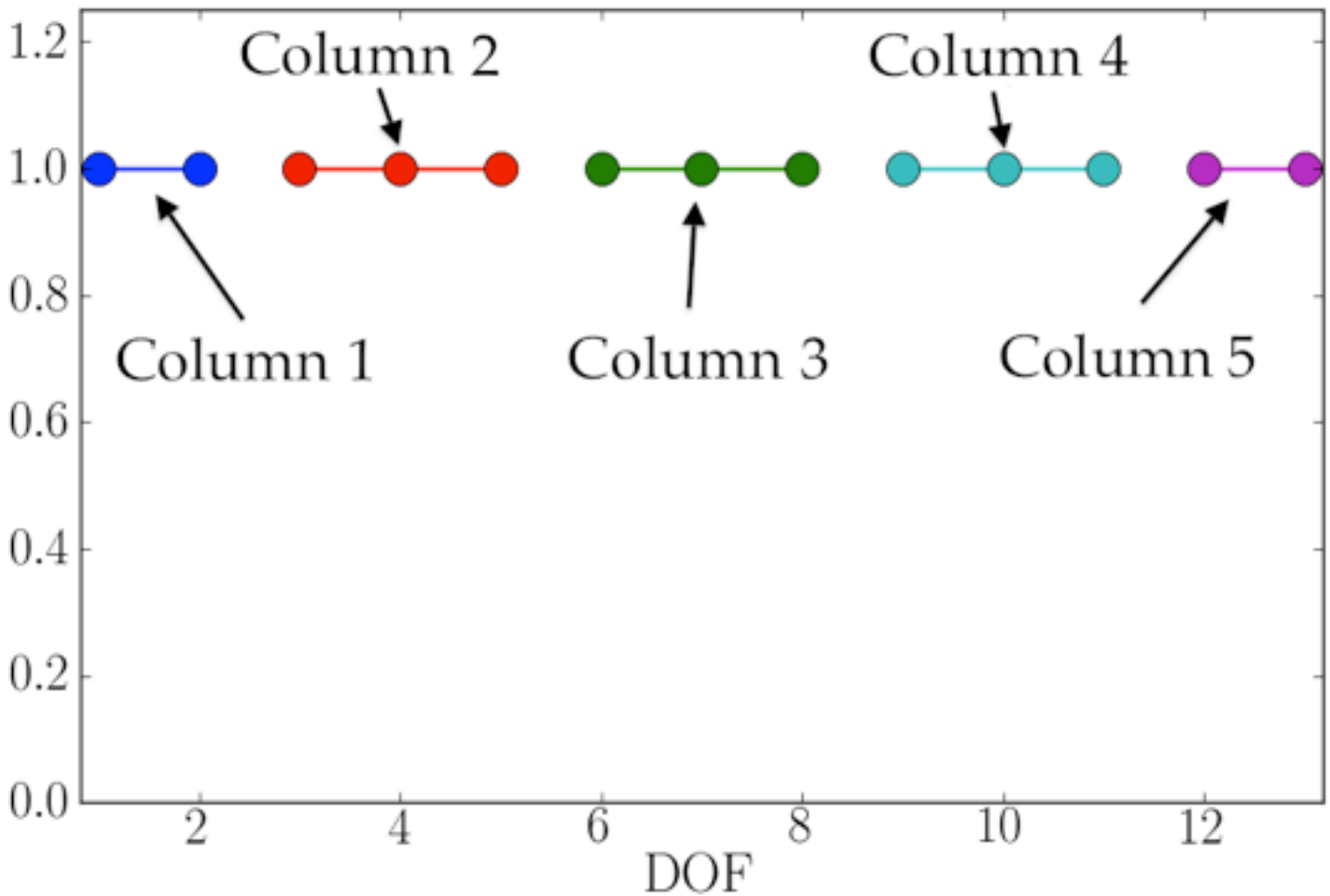
Construct Tentative I_{m+1}^{m}

- For diffusion, B_1 is all ones
- Yielding this tentative I_2^1

$$\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

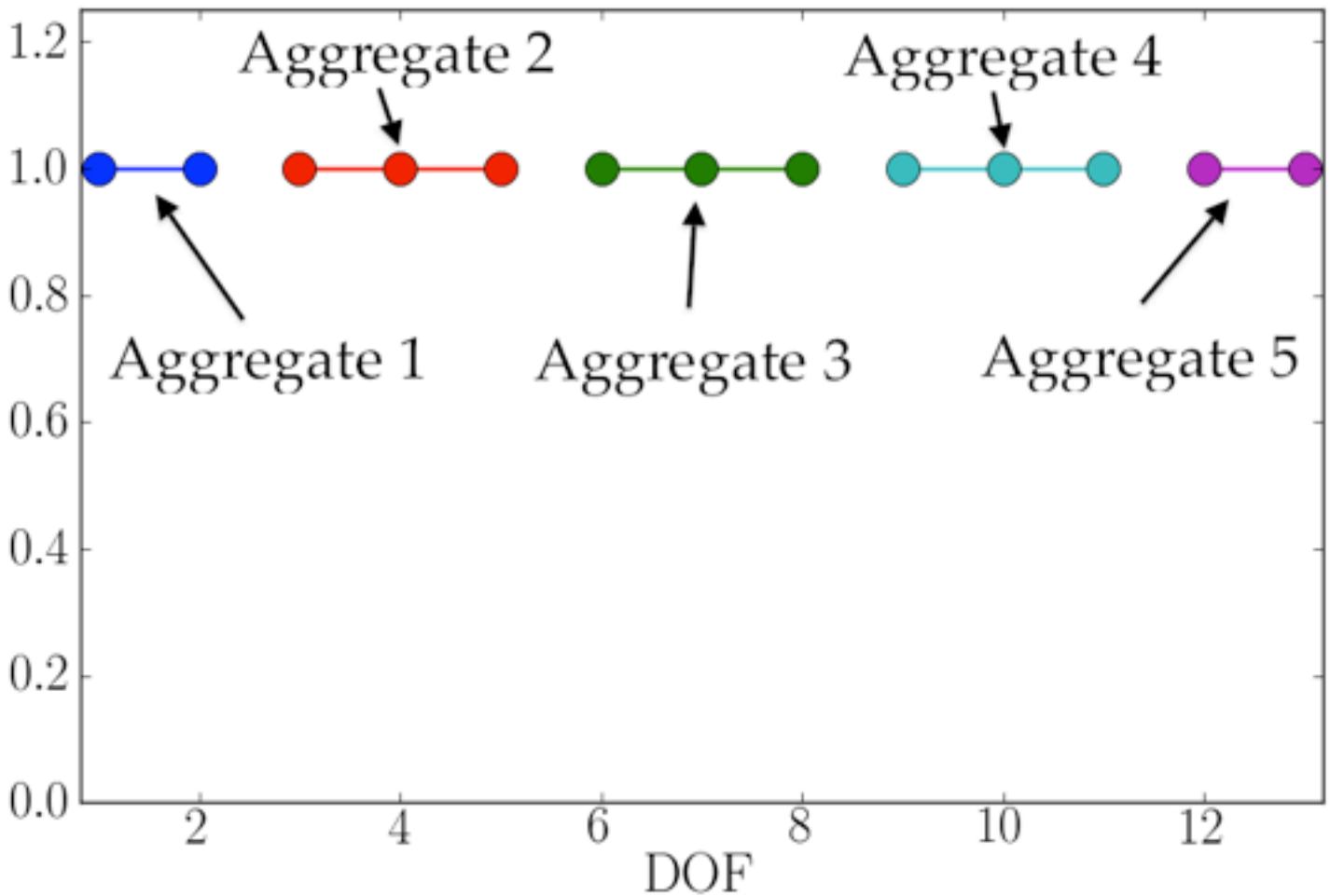
Construct Tentative I_{m+1}^{m}

- Plotting only the nonzero portion of each column



Construct Tentative I_{m+1}^{m}

- Plotting only the nonzero portion of each column



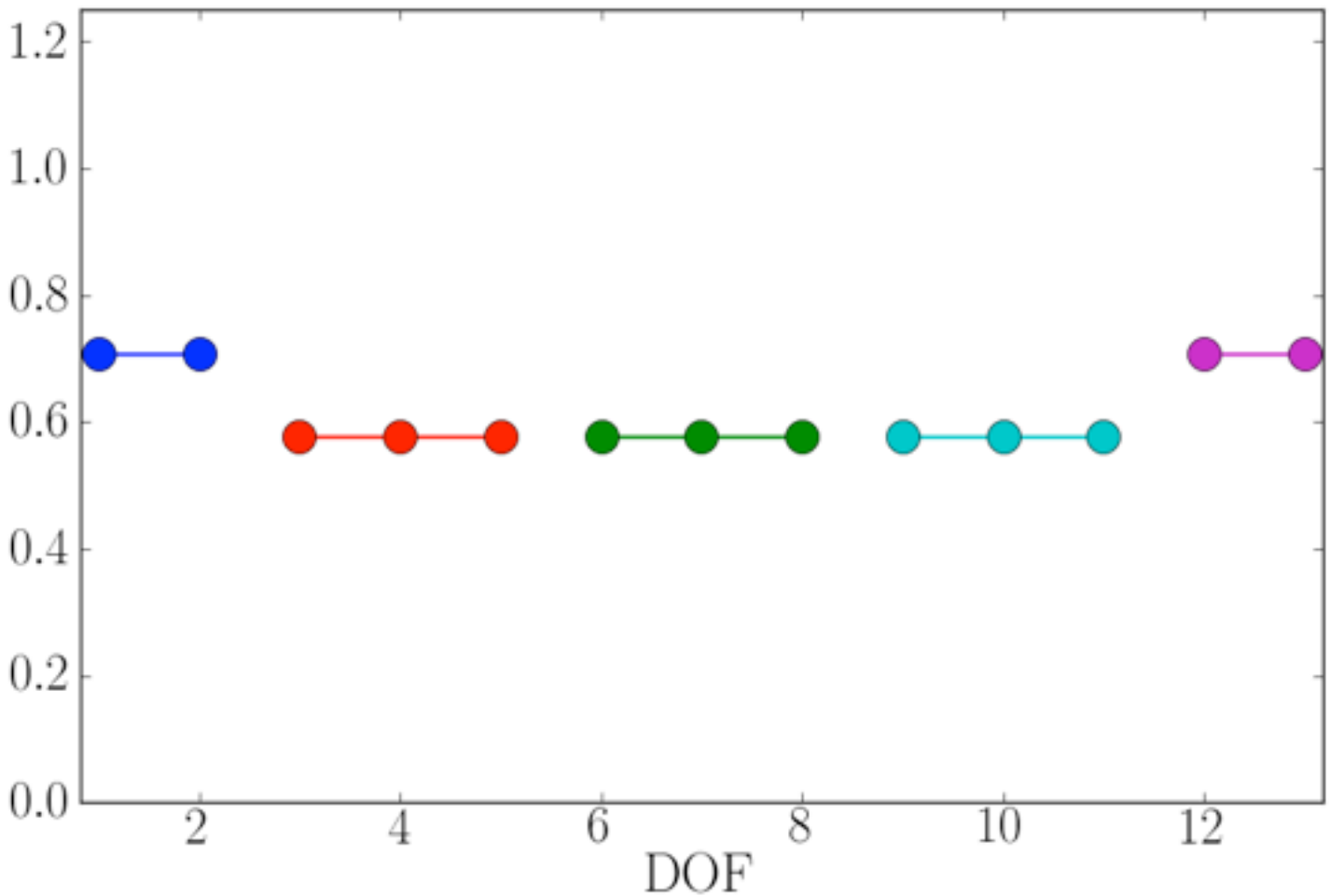
Construct Tentative I_{m+1}^{m}

- Compute QR factorization of each nonzero block
  - Let $Q^{(k)}R^{(k)}$ be factorization of kth nonzero block
- Replace each nonzero block with corresponding $Q^{(k)}$
- Improves conditioning of interpolation functions
- B_{m+1} becomes concatenation of all $R^{(k)}$
  - Resulting in this powerful relationship $I_{m+1}^{m}B_{m+1} = B_m$

$$B_{m+1} = \begin{bmatrix} R^{(1)} \\ R^{(2)} \\ R^{(3)} \\ R^{(4)} \\ R^{(5)} \end{bmatrix}, \quad I_{m+1}^{m} = \begin{bmatrix} Q^{(1)} & 0 & 0 & 0 & 0 \\ 0 & Q^{(2)} & 0 & 0 & 0 \\ 0 & 0 & Q^{(3)} & 0 & 0 \\ 0 & 0 & 0 & Q^{(4)} & 0 \\ 0 & 0 & 0 & 0 & Q^{(5)} \end{bmatrix}$$

Construct Tentative I_{m+1}^{m}

- Plotting only the nonzero portion of each column
- For diffusion example, QR normalizes each column and each $R^{(k)}$ is scalar

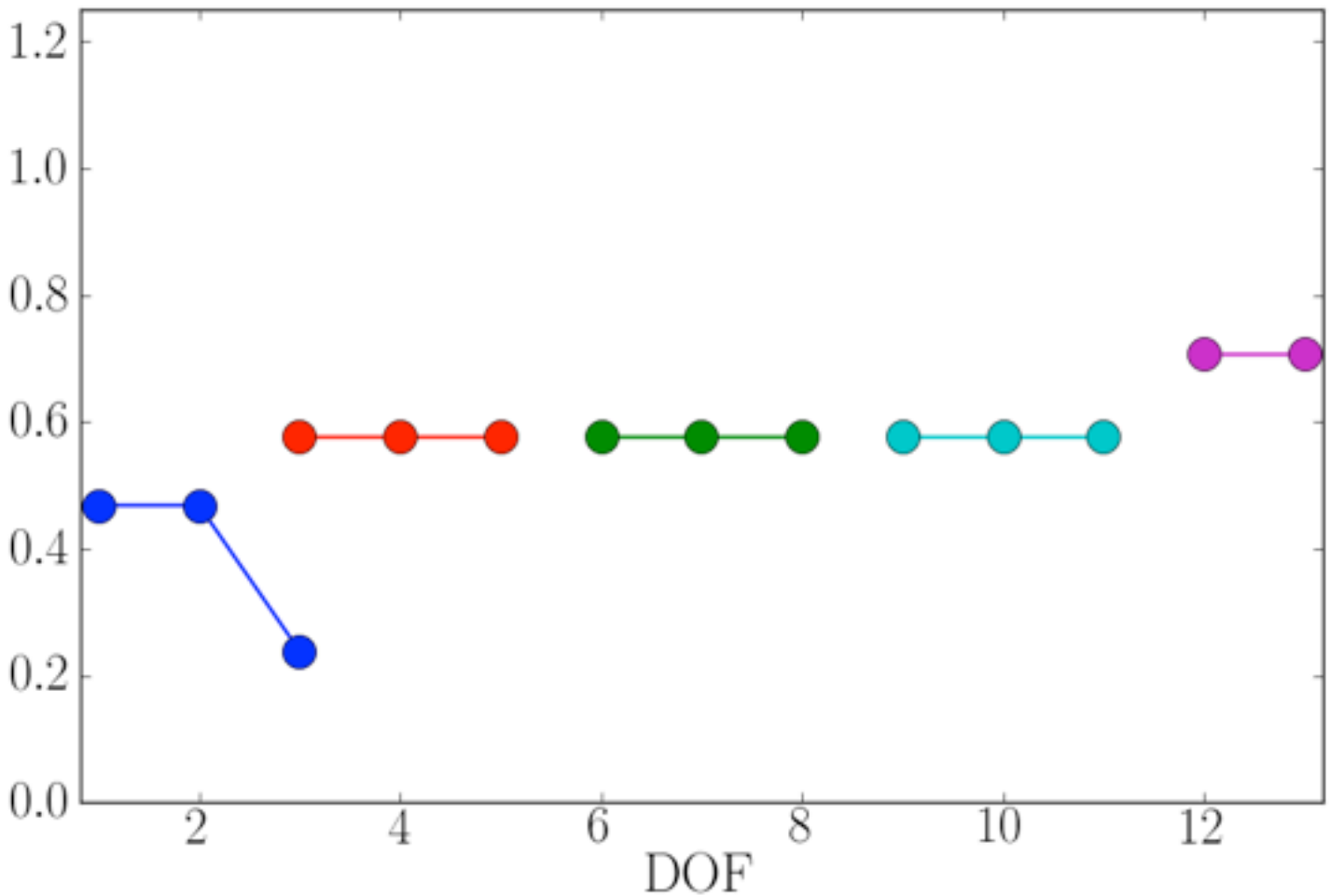


Smooth Tentative I_{m+1}^{m}

- Apply a smoother to tentative prolongation
- $\mathrm{span}(I_{m+1}^{m})$ better describes globally smooth error
  - Widens interpolation stencil
  - Smoothes out jumps in tentative prolongation
  - Lowers the energy of each column, i.e., each column better approximates smooth error
- Classic prolongation smoothing is weighted-Jacobi
  $I_{m+1}^{m} = (I - \omega D_m^{-1} A_m) I_{m+1}^{m}$

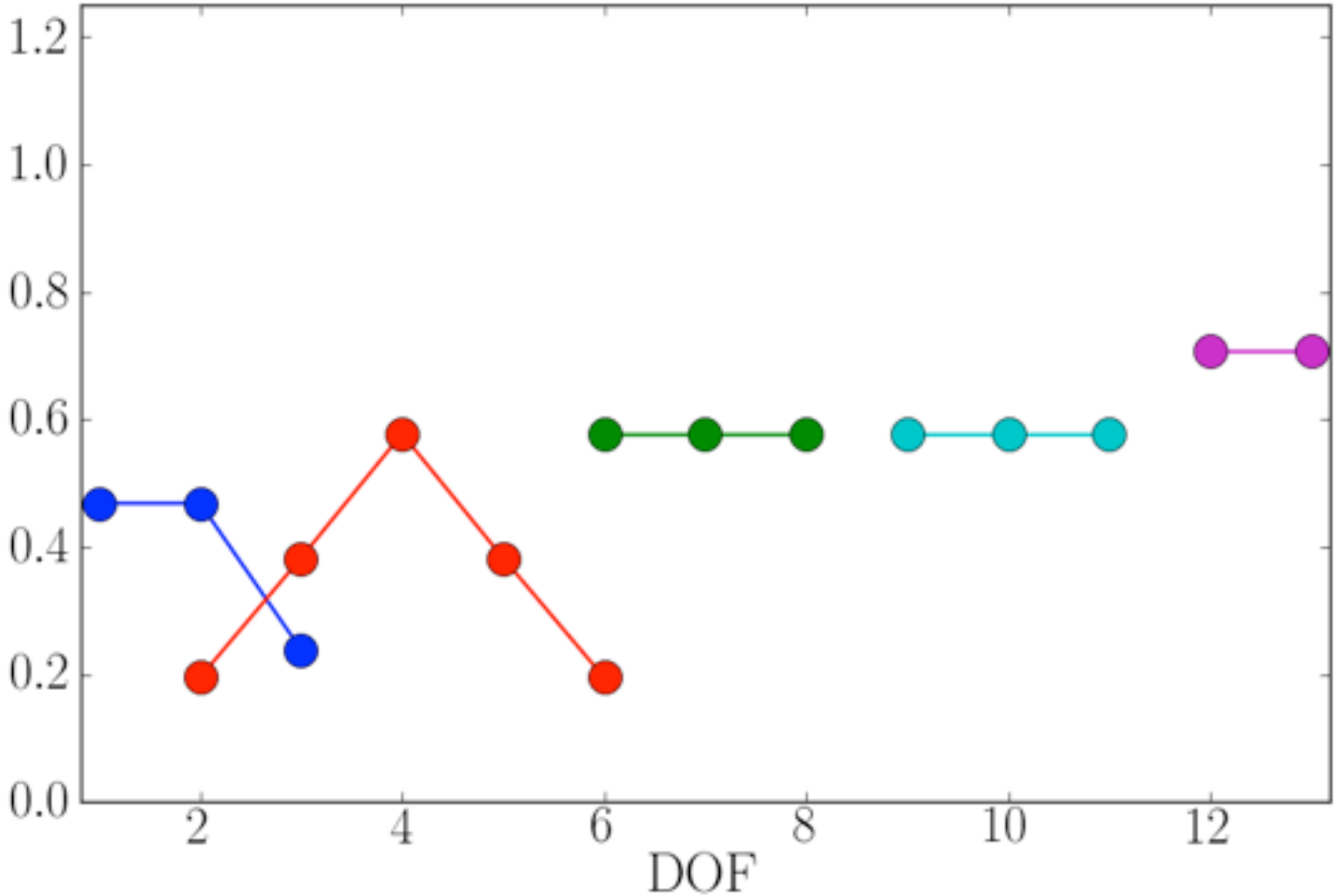
Smooth Tentative I_{m+1}^{m}

- Plotting only the nonzero portion of each column
- Apply weighted-Jacobi to only first column



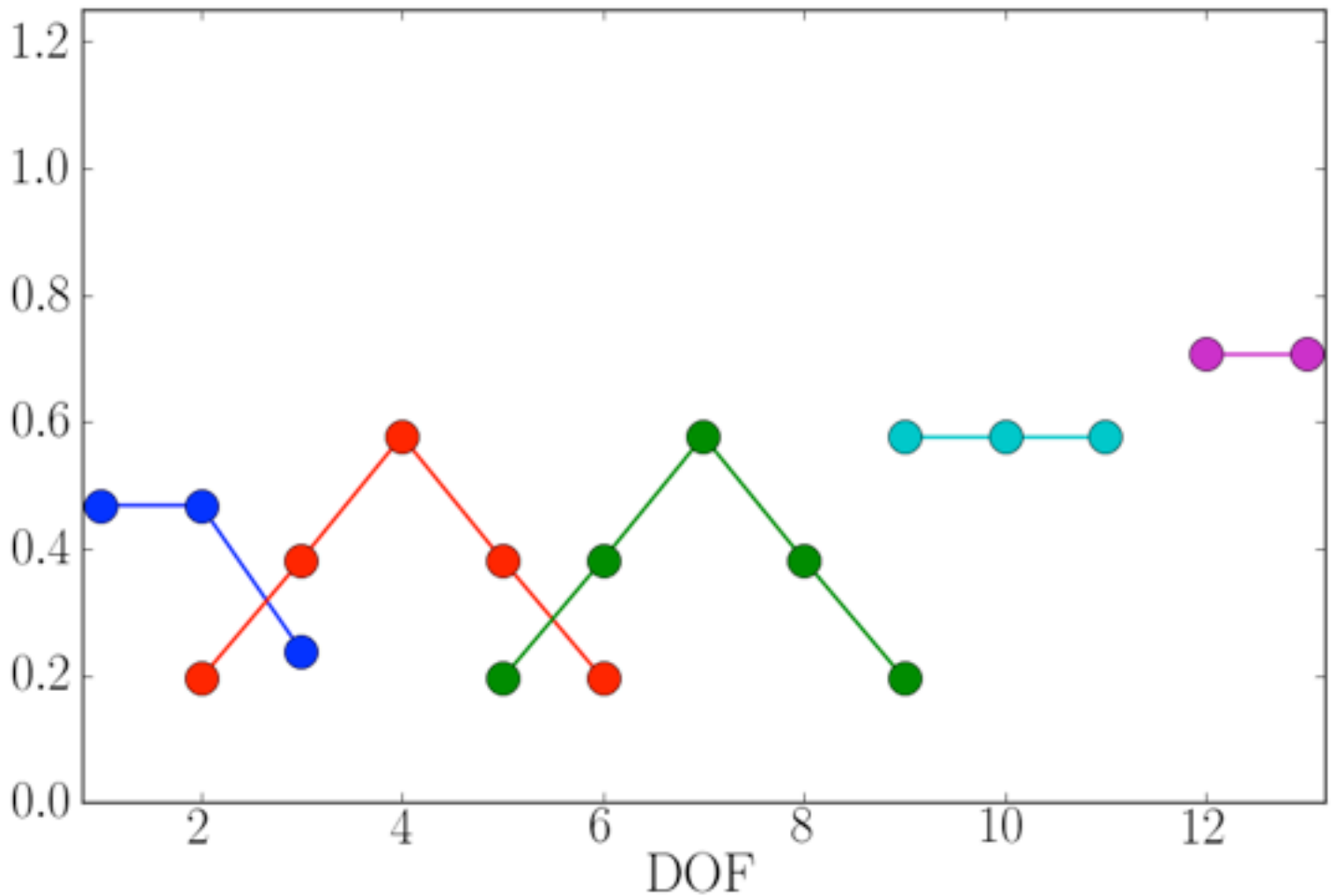
Smooth Tentative I_{m+1}^{m}

- Plotting only the nonzero portion of each column
- Apply weighted-Jacobi to only second column



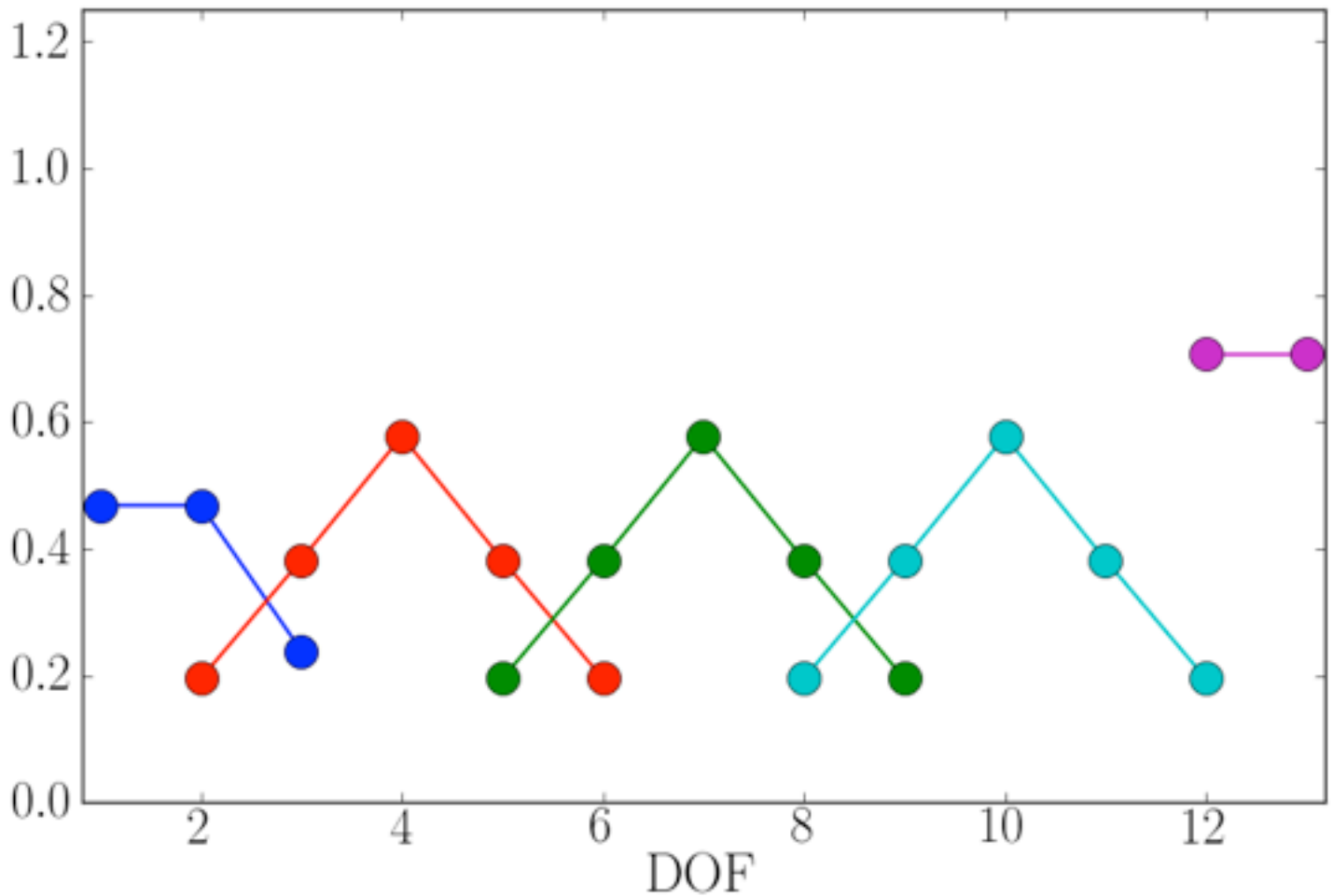
Smooth Tentative I_{m+1}^{m}

- Plotting only the nonzero portion of each column
- Apply weighted-Jacobi to only third column



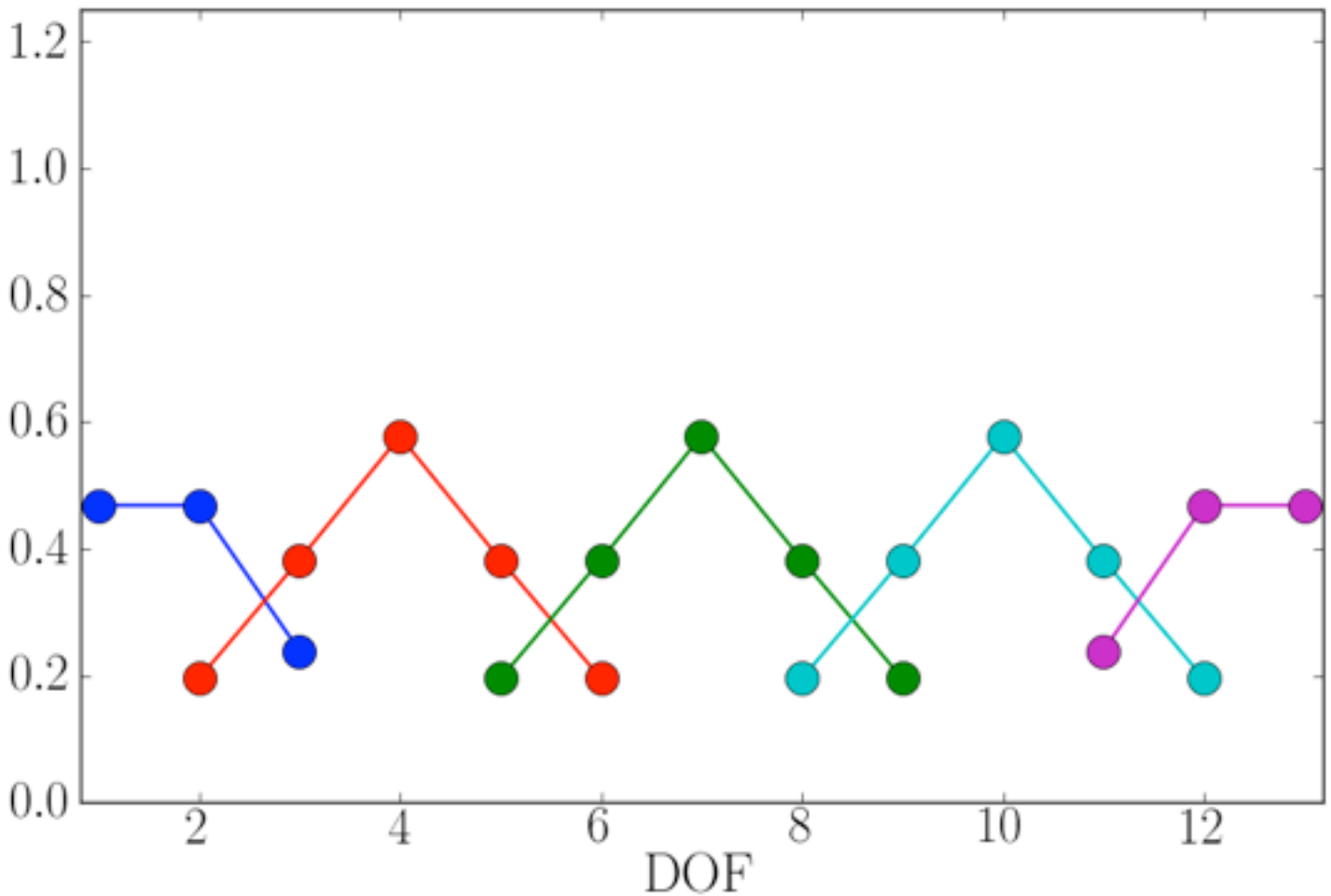
Smooth Tentative I_{m+1}^{m}

- Plotting only the nonzero portion of each column
- Apply weighted-Jacobi to only fourth column



Smooth Tentative I_{m+1}^{m}

- Plotting only the nonzero portion of each column
- Apply weighted-Jacobi to only fifth column



Basic SA Setup Phase

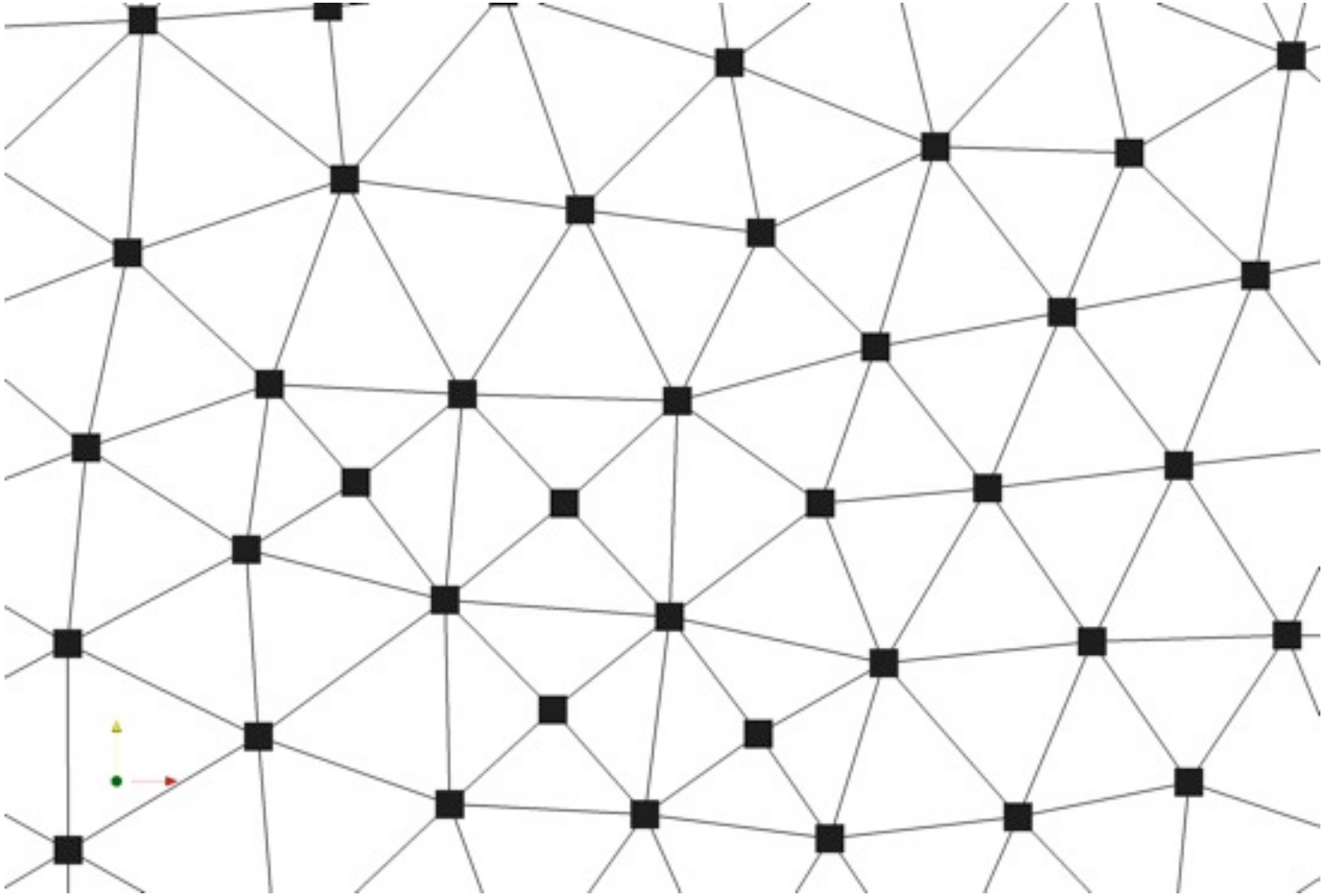
Algorithm 1: `sa_setup`(A, B)

```
A_1 ⇐ A
B_1 ⇐ B
for m = 1, 2, ...
    S_m            ⇐ strength(A_m)
    Agg_m          ⇐ aggregate(S_m)
    I^m_{m+1}, B_{m+1} ⇐ inject(Agg_m, B_m)
    I^m_{m+1}      ⇐ smooth_prolongator(A_m, I^m_{m+1})
    A_{m+1}        ⇐ (I^m_{m+1})^T A_m I^m_{m+1}

return A_0 ... A_m, P_0 ... P_{m-1}
```

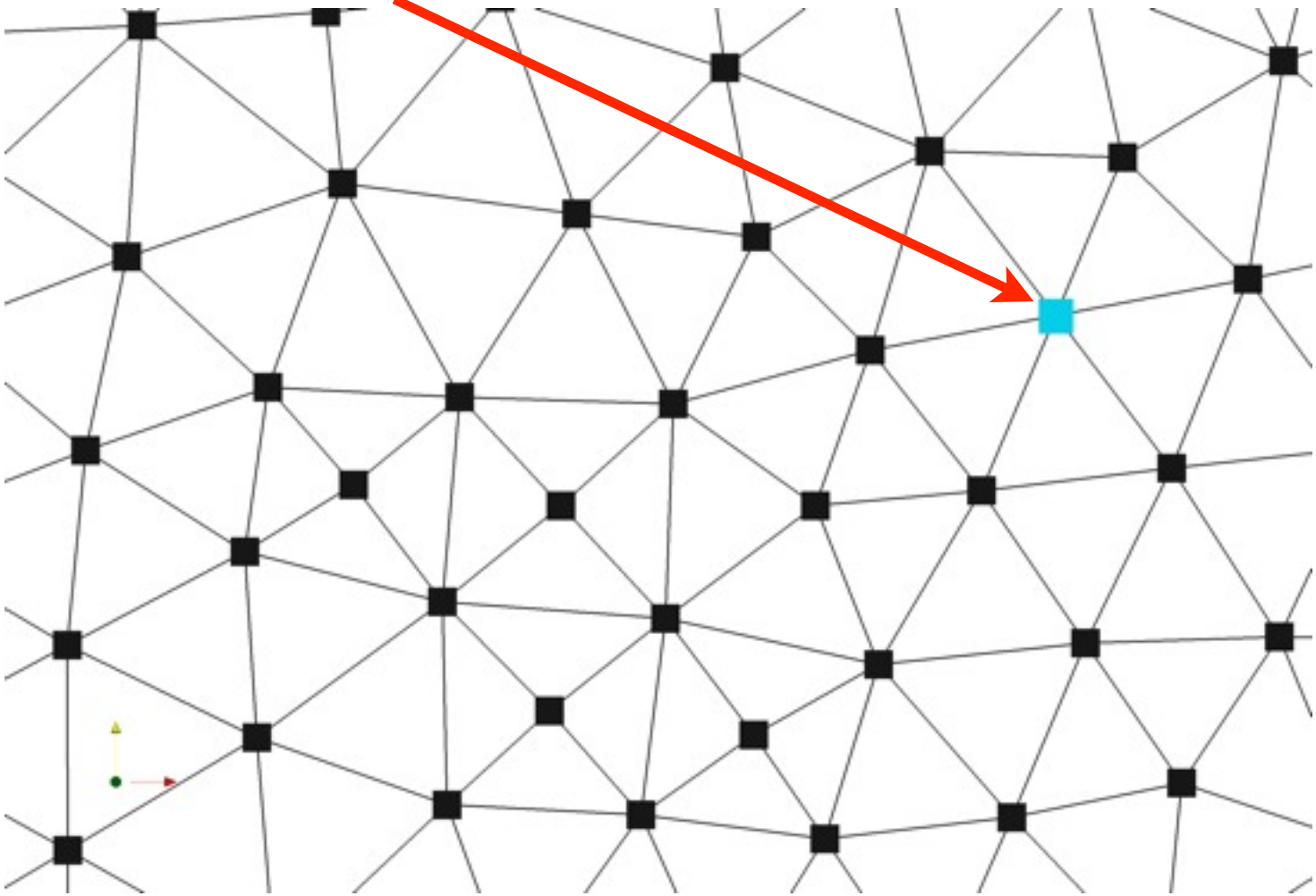
2D Example Aggregation

Example Section of Matrix Graph for 2D Diffusion



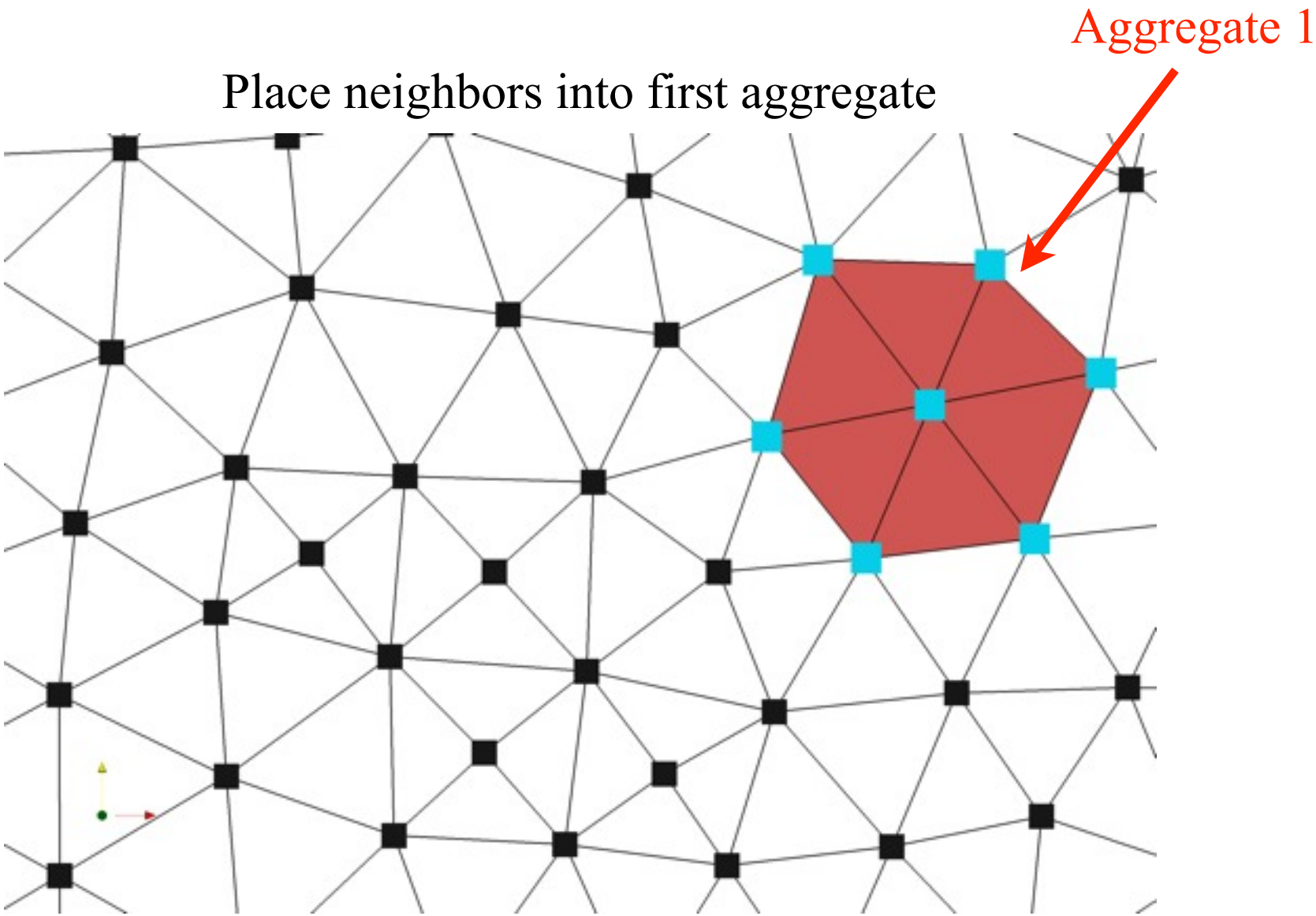
2D Example Aggregation

Choose initial unaggregated dof



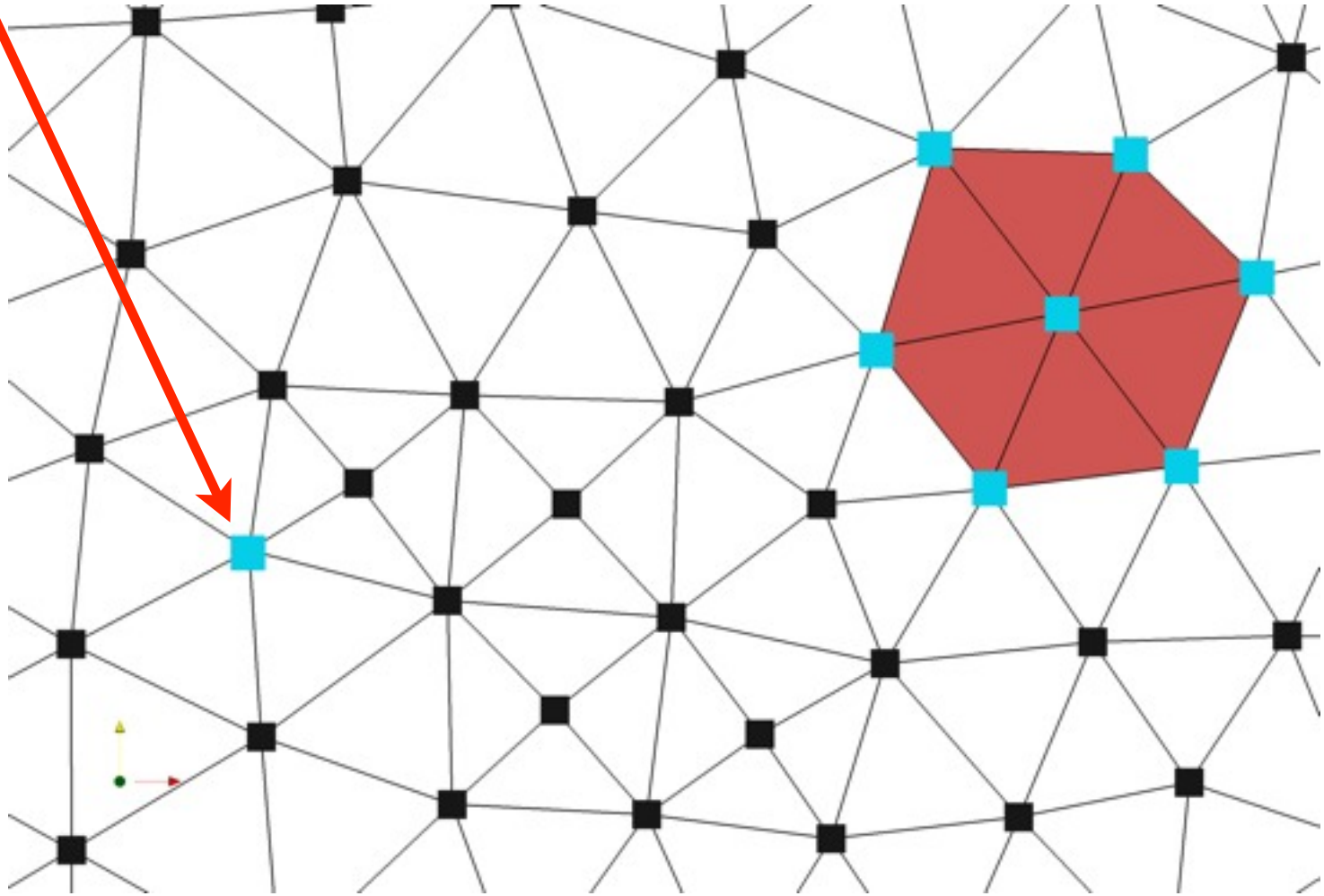
2D Example Aggregation

Place neighbors into first aggregate



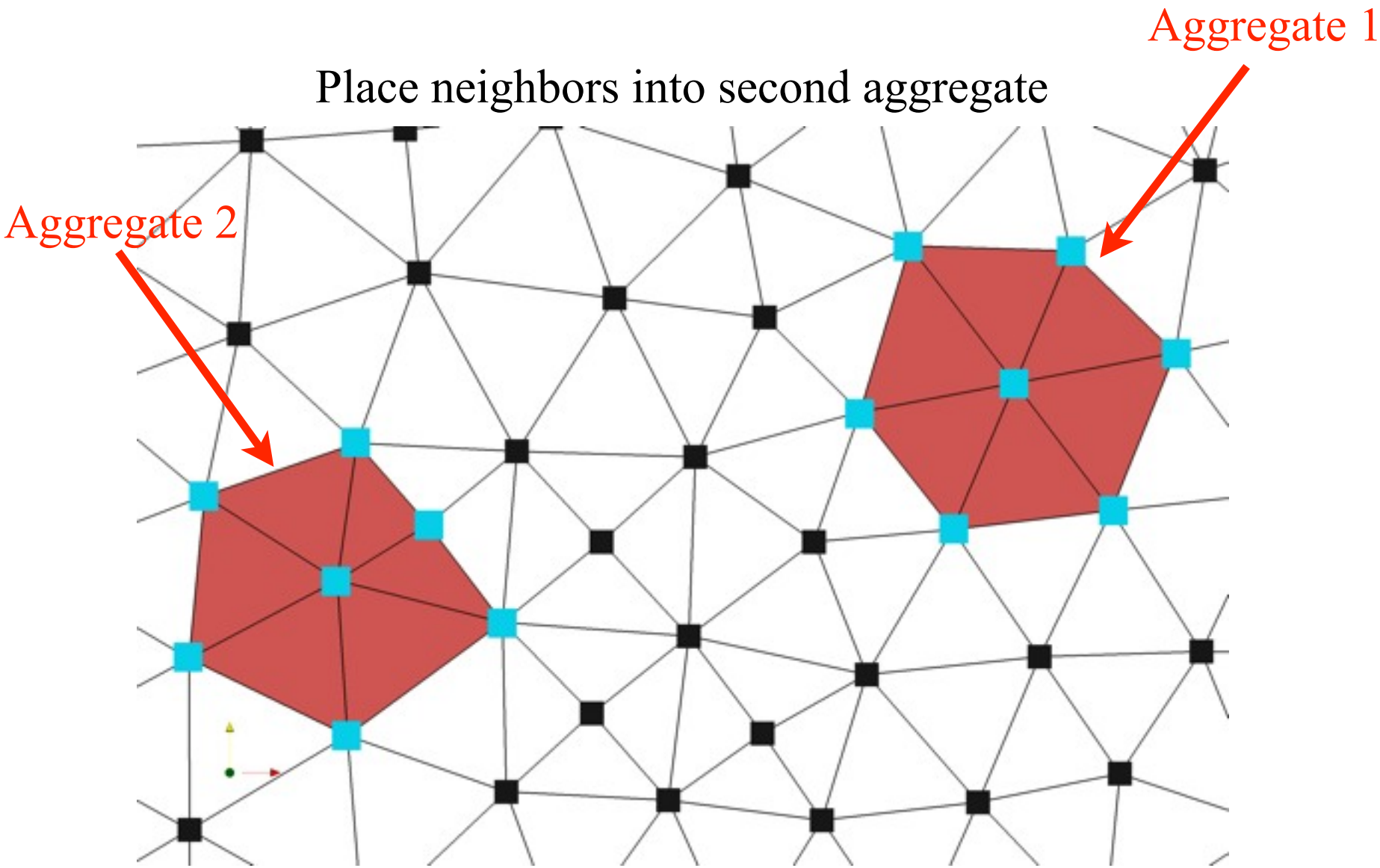
2D Example Aggregation

Choose next unaggregated dof, that has all unaggregated neighbors



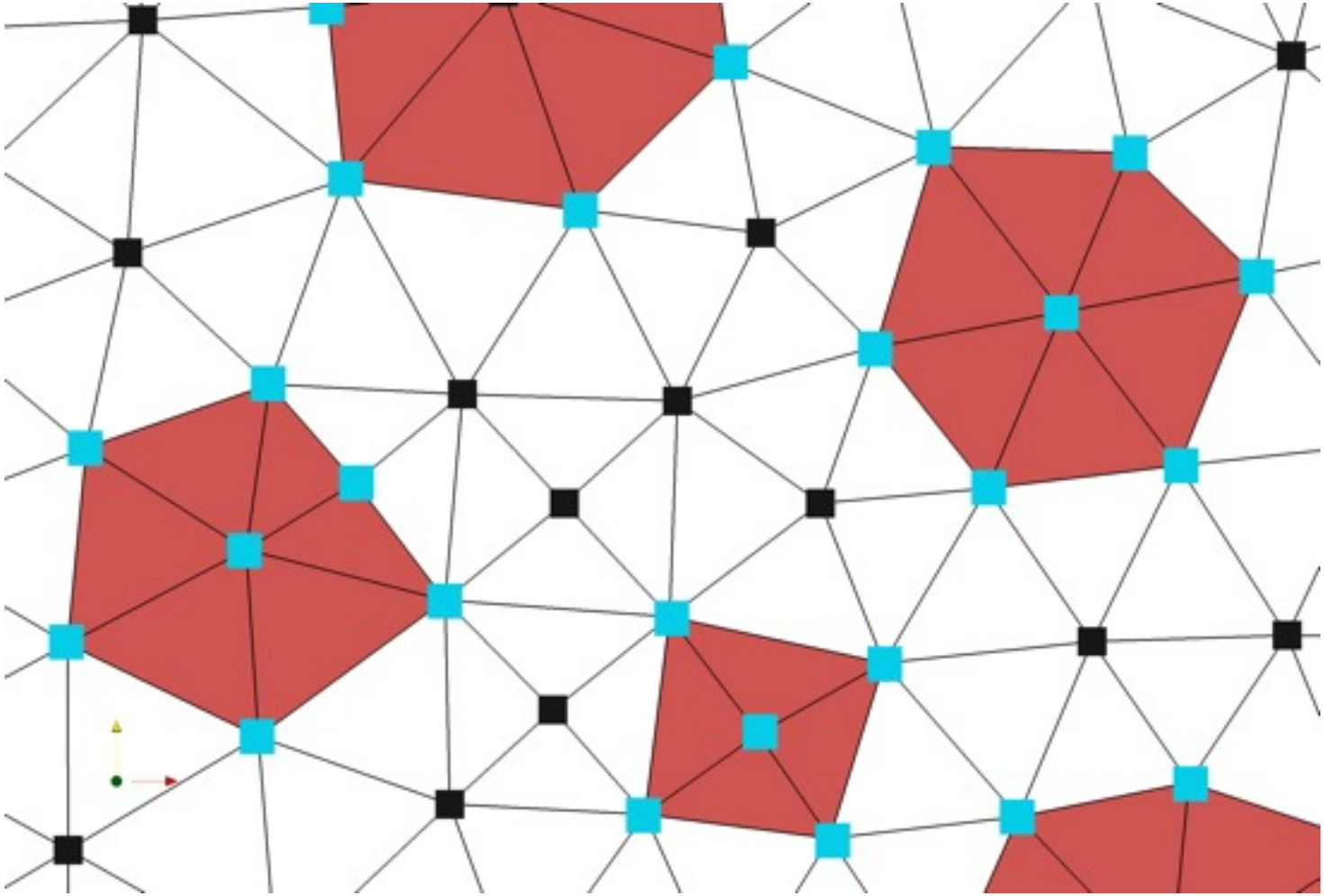
2D Example Aggregation

Place neighbors into second aggregate



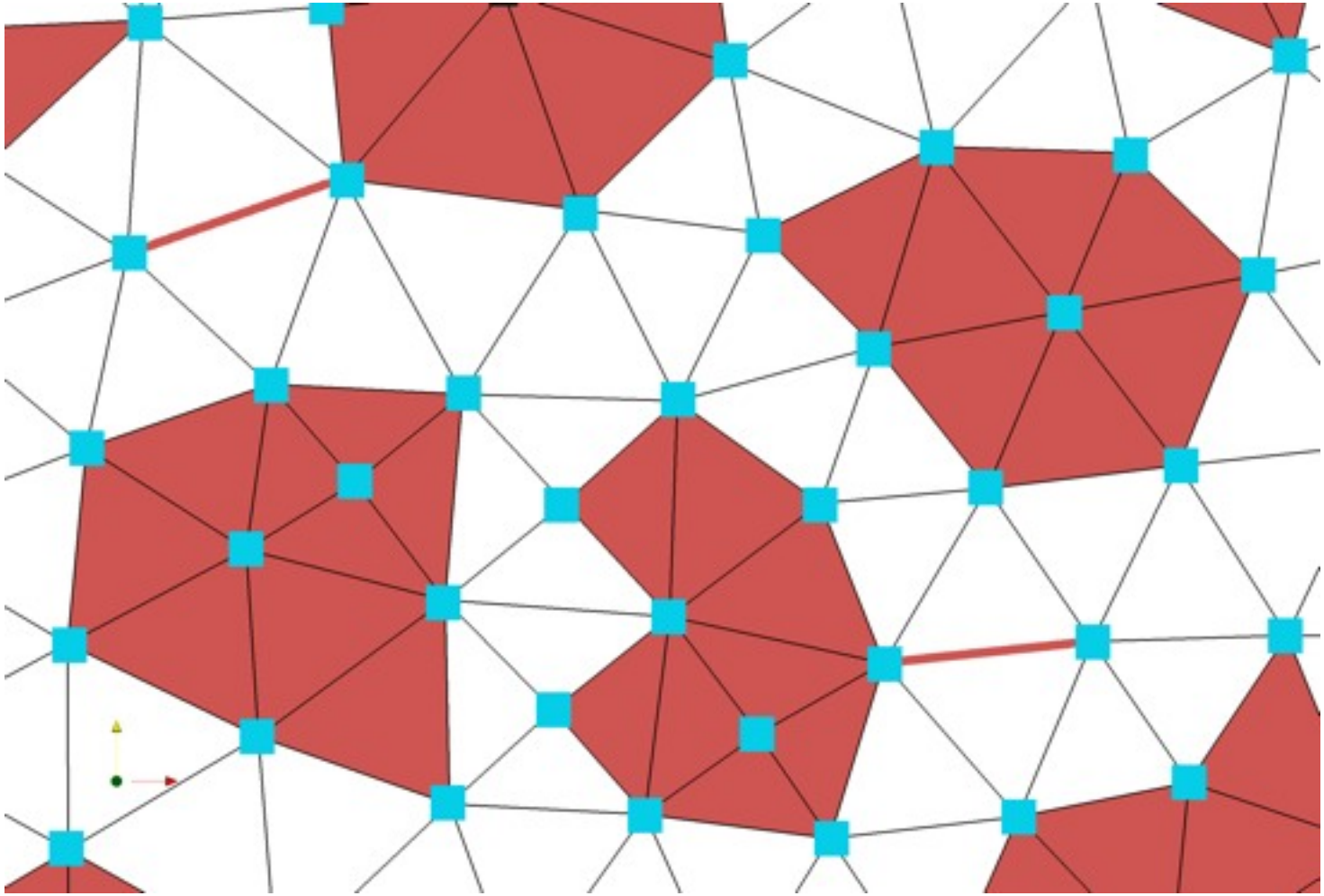
2D Example Aggregation

Repeat process, until no dof remains that has only unaggregated neighbors



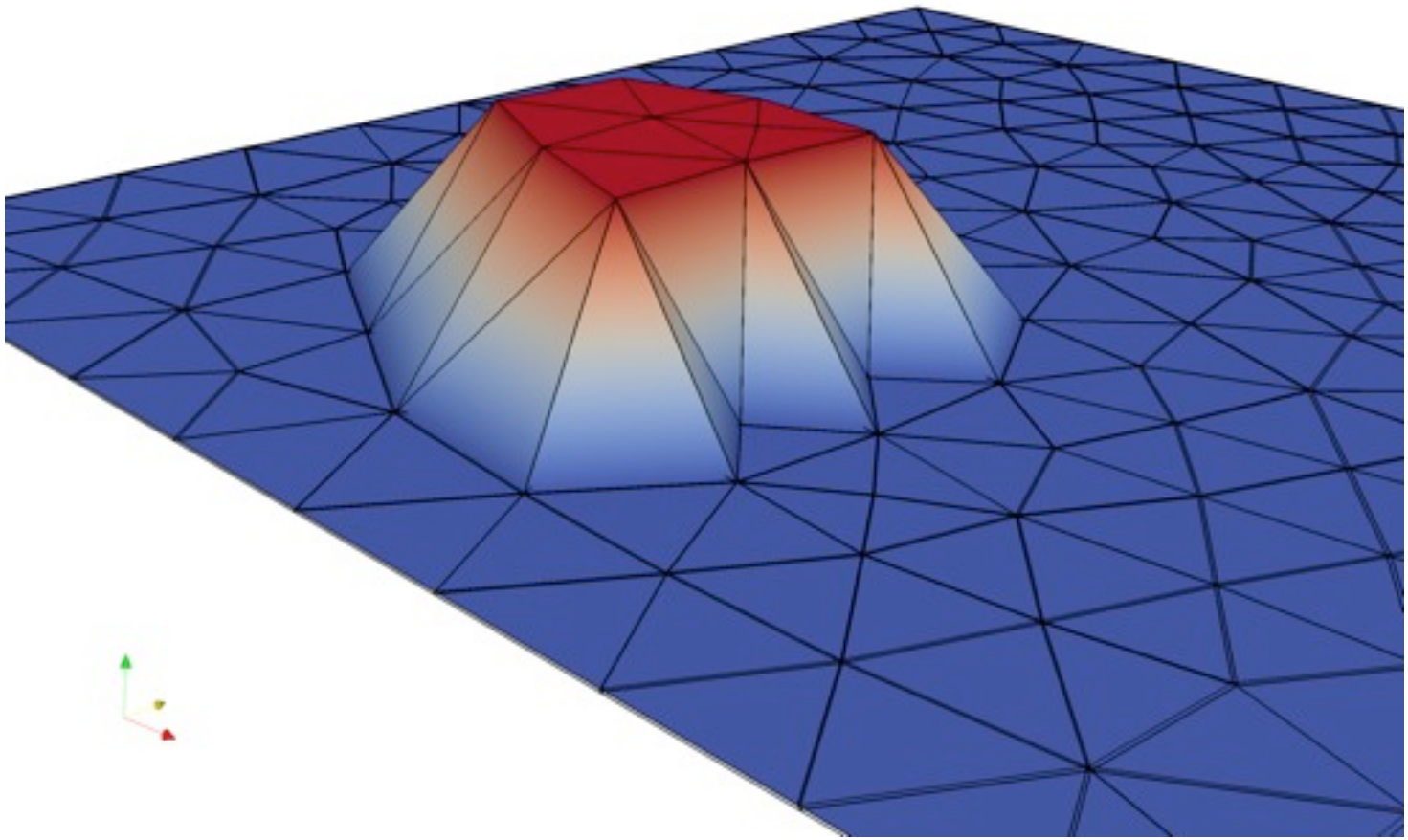
2D Example Aggregation

Cleanup phase, placing unaggregated dofs with nearest aggregate



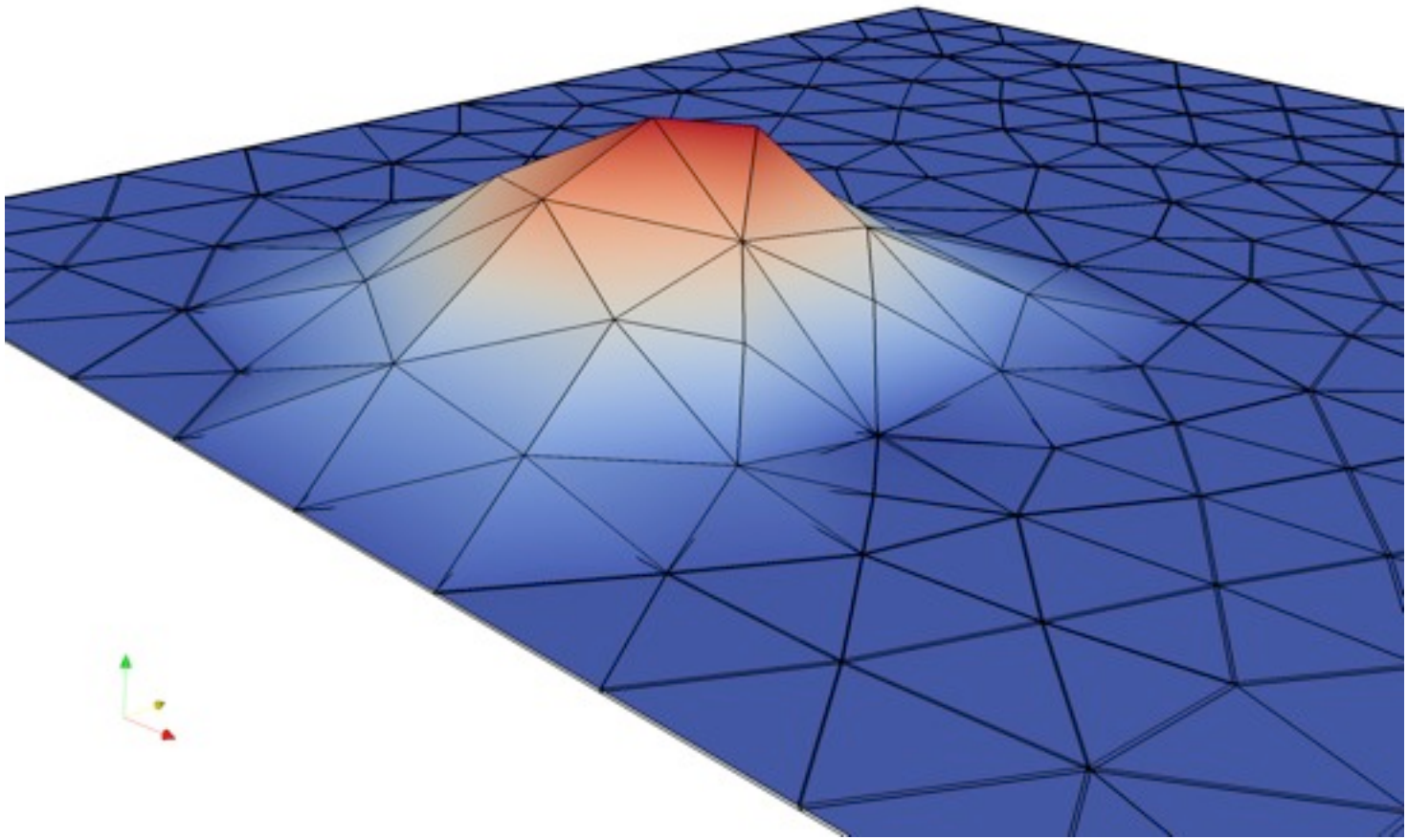
2D Example Prolongation Smoothing

Unsmoothed column of tentative prolongation



2D Example Prolongation Smoothing

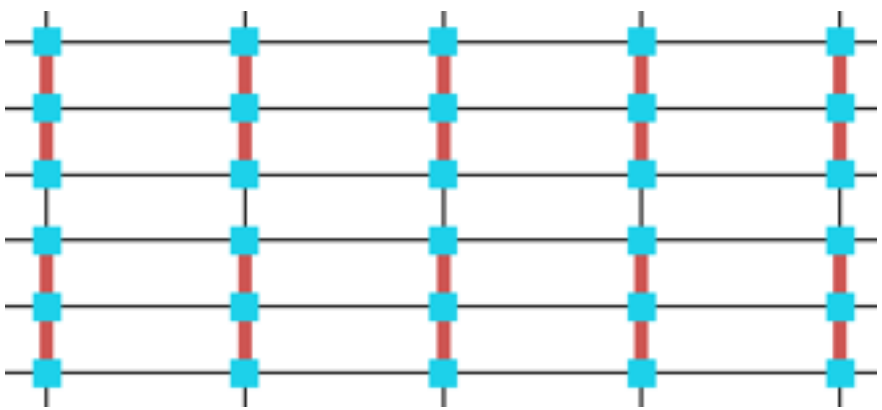
Smoothed column of prolongation



Sample aggregates for the Laplacian

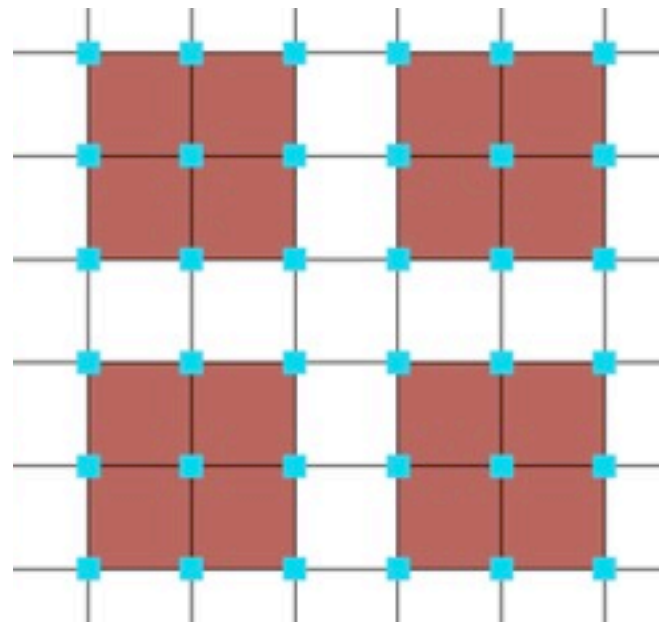
9-pt FE (quads), & 9-pt FE (stretched quads)

9-pt FE
(stretched quads)



$$\begin{pmatrix} -1 & -4 & -1 \\ 2 & 8 & 2 \\ -1 & -4 & -1 \end{pmatrix}$$

9-pt FE (quads)



$$\begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix}$$

SA Performance: Sometimes a Success Story

- For diffusion, SA broadly similar to F/C-style AMG
  - Optimal for model problem (Poisson's equation, regular grid)
  - Efficient and scalable for diffusion on unstructured grids
  - Handles anisotropic diffusion relatively well
- Typically, smaller operator complexity than F/C-style AMG

Regular grid, plain, old, vanilla problem,

unit square, n = 64, Dirichlet boundaries

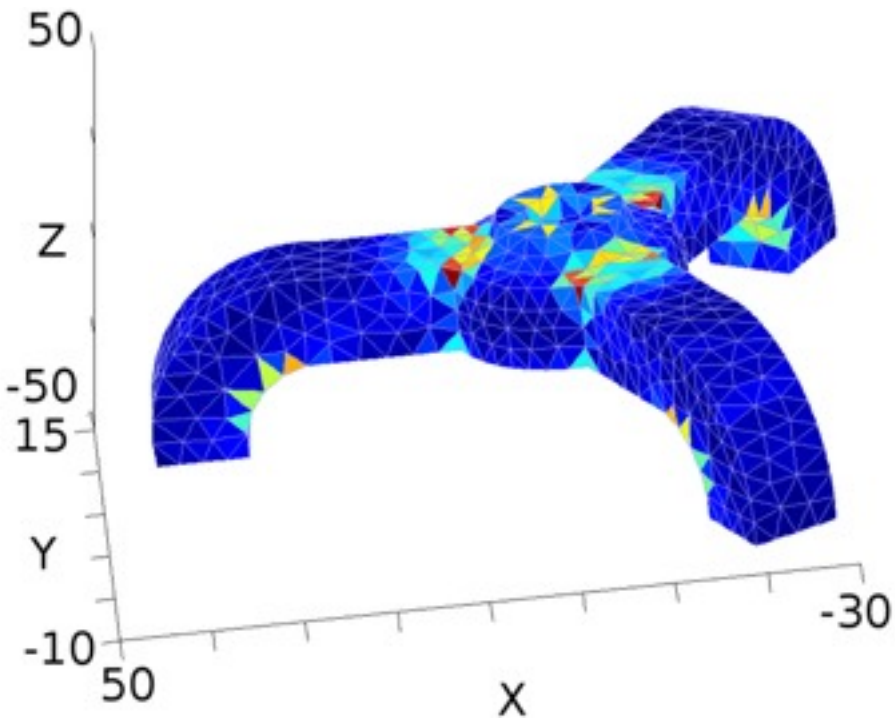
Stencil	Convergence per Cycle	Operator Complexity
5-pt	0.15	1.33
9-pt (-1, 8)	0.09	1.11

SA for Systems

- Solving PDE systems simple with SA framework
- k unknowns at each finest-level grid point
  - Group each set of k unknowns into a "supernode"
  - Coarsen only supernodes, not individual dofs
- B_1 typically contains multiple vectors, yielding k x j blocks in prolongation, where j is the number of vectors in B_1
  - For 3D elasticity, j=6 vectors in and k=3
- Coarse grids have supernodes of size j

Elasticity Performance

- 3D Isotropic Linearized Elasticity on a Tripod
  - Downward force applied
  - How does it deform?
  - 6 near null-space modes (i.e., rigid body modes)
  - Size 3 supernodes on finest level



Num. Dofs	Convergence per Cycle	Operator Complexity
2,757	0.70	1.56
16,341	0.82	1.54
109,551	0.86	1.54

SA Recap

- User-provided B_1 roughly describes smooth error
  - F/C-style AMG also makes smooth error assumptions
- SA approach directly allows multigrid to capture arbitrary near null-spaces
  - Just change B_1!
  - This flexibility is advantage over F/C-style AMG
- Define prolongation through aggregation, injection of B_1 into sparsity outline, and prolongation smoothing
  - No more C-points, and interpolation formulas
- Naturally handles systems of PDEs (arguably better than F/C-style AMG)
- Identical Solve phases for F/C-style AMG and SA

Classic SA References

(1) Vaněk, P. and Mandel, J. and Brezina, M. Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems. Computing, 1996. pp. 179-196.

(2) Vaněk, P. and Mandel, J. and Brezina, M. Convergence of algebraic multigrid based on smoothed aggregation. Numerische Mathematik, 2001. pp. 559-579.