Localized solutions of sparse linear systems for geometry processing

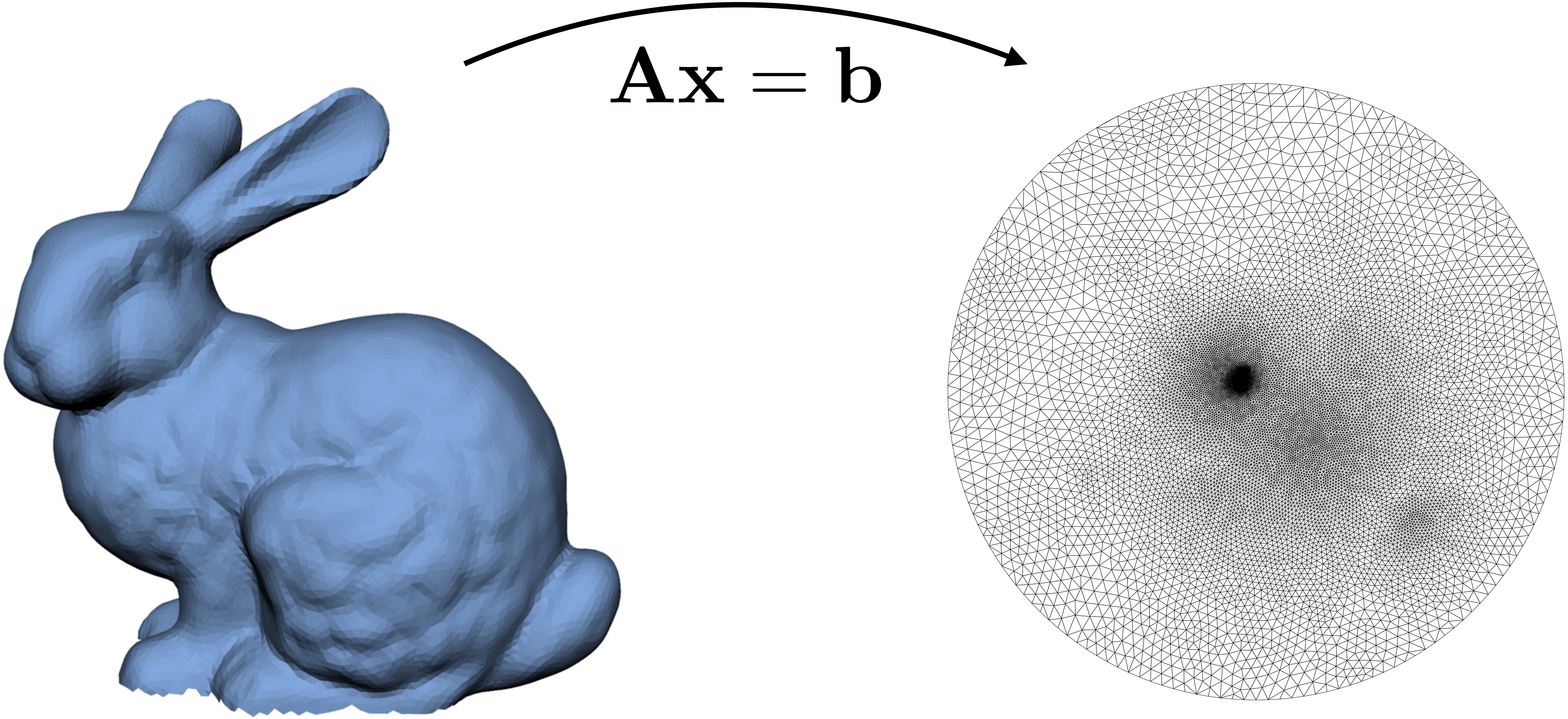
Philipp Herholz, Timothy A. Davis, Marc Alexa

Linear solvers in geometry processing

- Many applications require repeated solutions of linear systems.

Discrete conformal parameterization

$$\mathbf{Ax} = \mathbf{b}$$

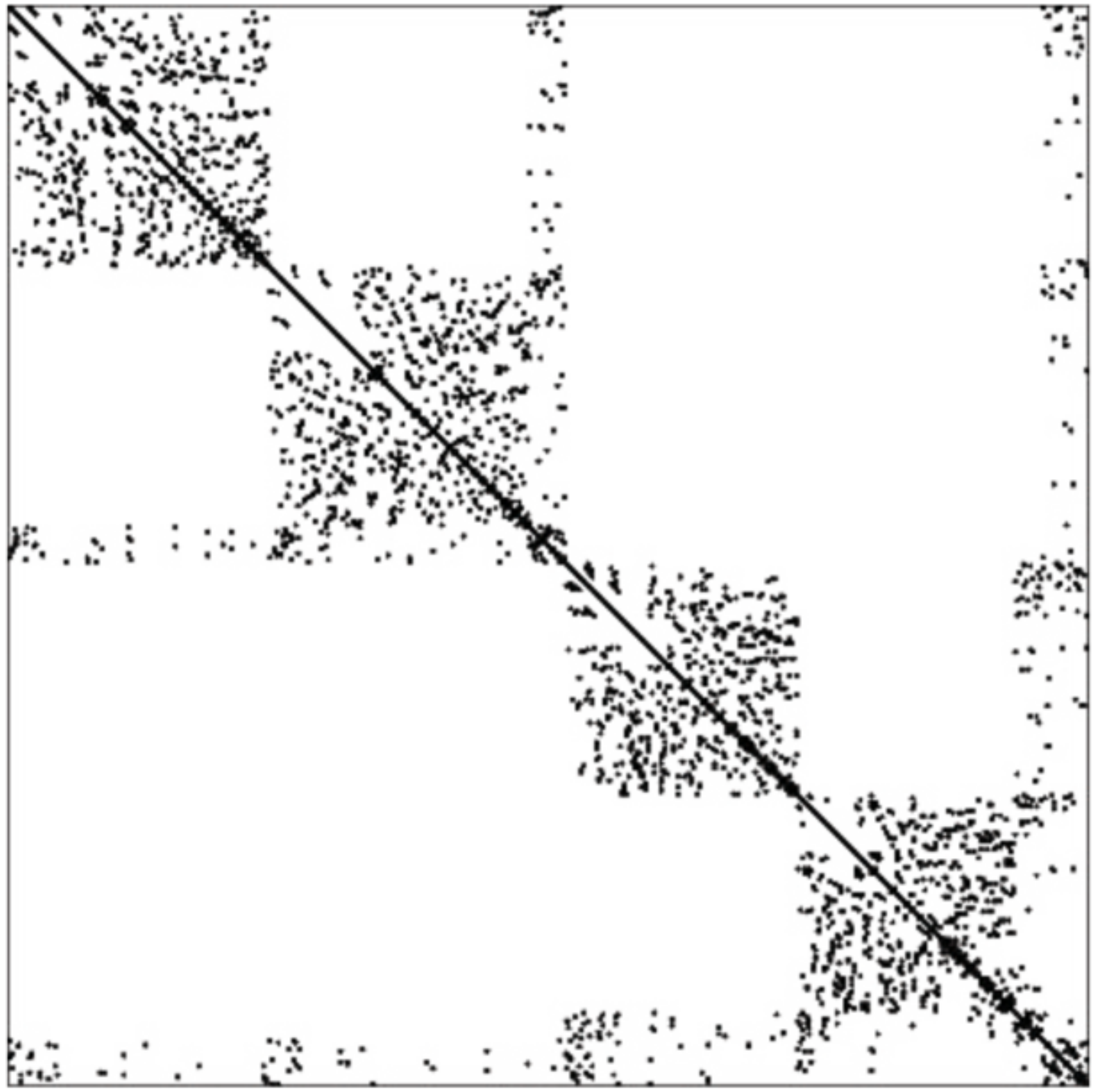


Discrete conformal parameterization

$$\mathbf{A}\mathbf{x} = \mathbf{b}$$

$\mathbf{A}$ is typically:

- sparse
- symmetric
- positive semi-definite

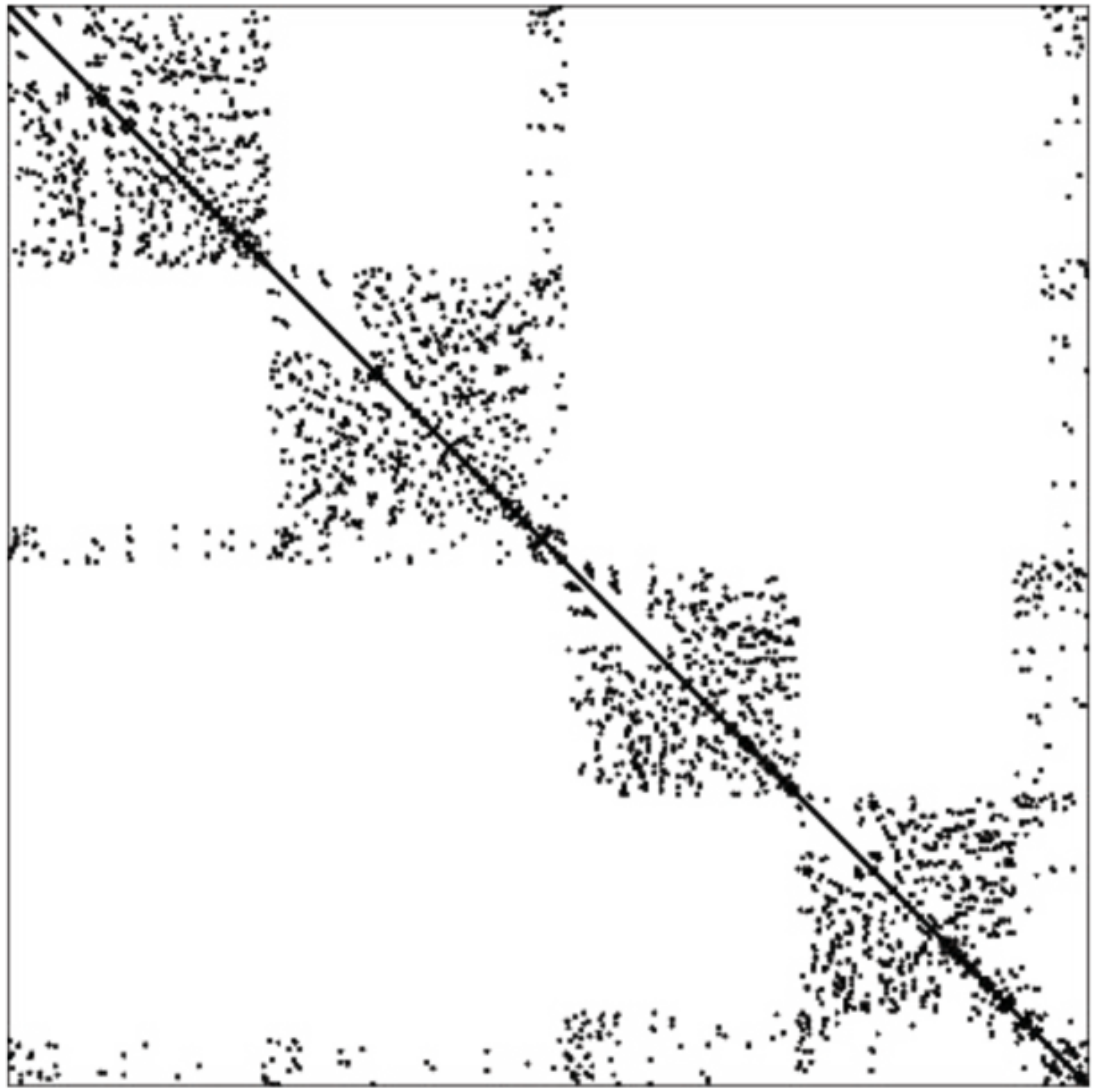


Cholesky factorization

$$\mathbf{Ax} = \mathbf{b}$$

$\mathbf{A}$ is typically:

- sparse
- symmetric
- positive semi-definite



Sparse Cholesky factorization can be applied!

Cholesky factorization

$$\mathbf{A}\mathbf{x} = \mathbf{b}$$

$$\mathbf{A} = \mathbf{L} \times \mathbf{D} \times \mathbf{L}^T$$

Cholesky factorization

$$\mathbf{A}\mathbf{x} = \mathbf{b}$$

$$\mathbf{L}\mathbf{D}\mathbf{L}^T\mathbf{x} = \mathbf{b}$$

Cholesky factorization

$$\mathbf{Ax} = \mathbf{b}$$

$$\mathbf{LD}\underbrace{\mathbf{L}^T\mathbf{x}}_{\mathbf{y}} = \mathbf{b}$$

Cholesky factorization

$$\mathbf{A}\mathbf{x} = \mathbf{b}$$

$$\mathbf{L}\mathbf{D}\underbrace{\mathbf{L}^T\mathbf{x}}_{\mathbf{y}} = \mathbf{b}$$

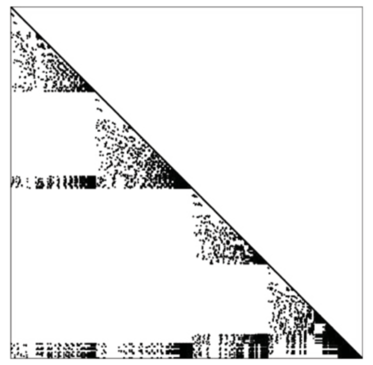
$$\mathbf{L}\mathbf{D}\mathbf{y} = \mathbf{b} \; (1)\text{Forward solve}$$

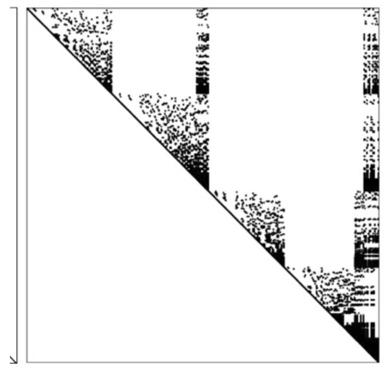
$$\mathbf{L}^T\mathbf{x} = \mathbf{y} \; (2)\text{Back solve}$$

Cholesky factorization

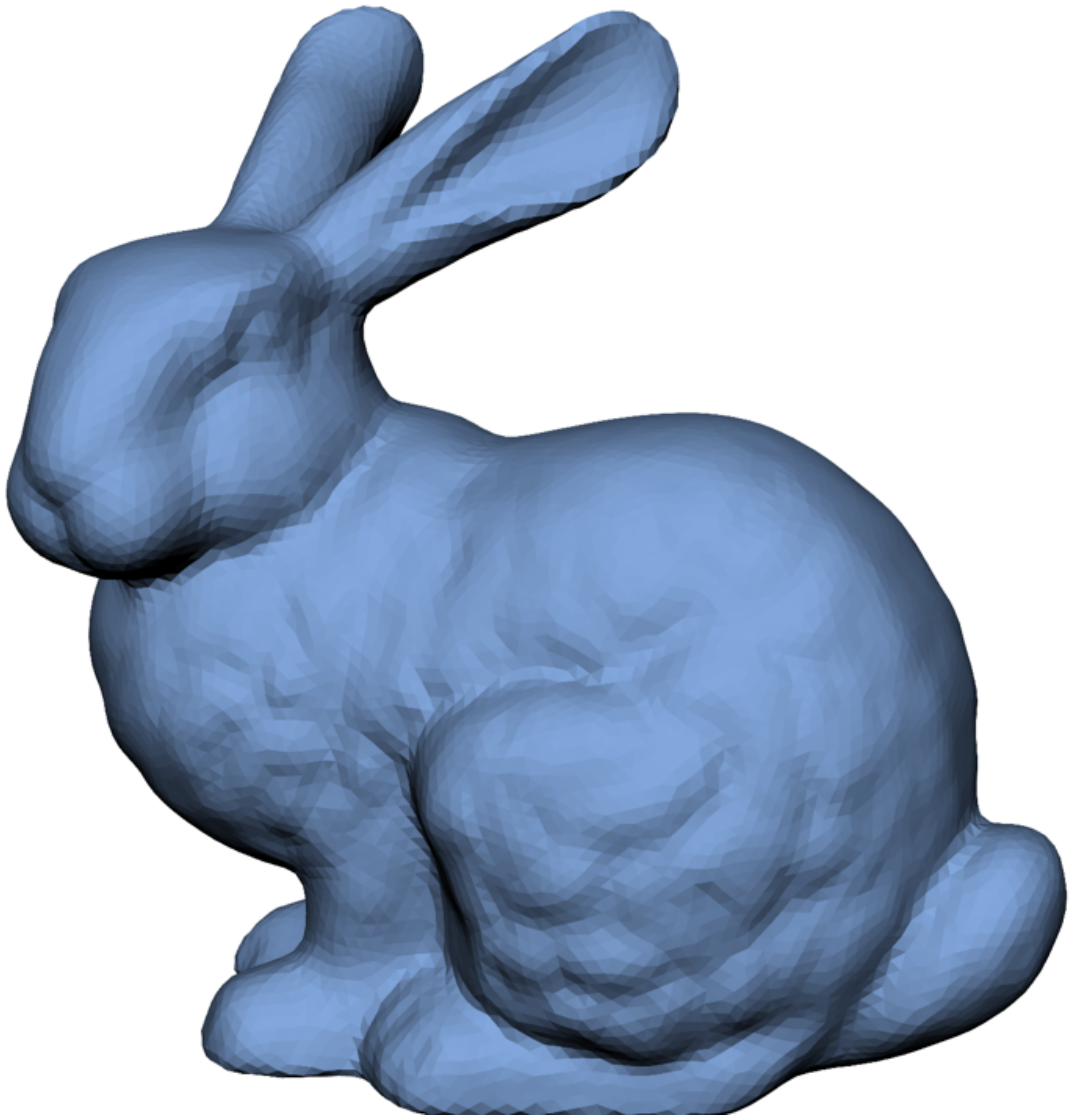
$$\mathbf{Ax} = \mathbf{b}$$

$$\mathbf{LD}\underbrace{\mathbf{L}^T\mathbf{x}}_{\mathbf{y}} = \mathbf{b}$$

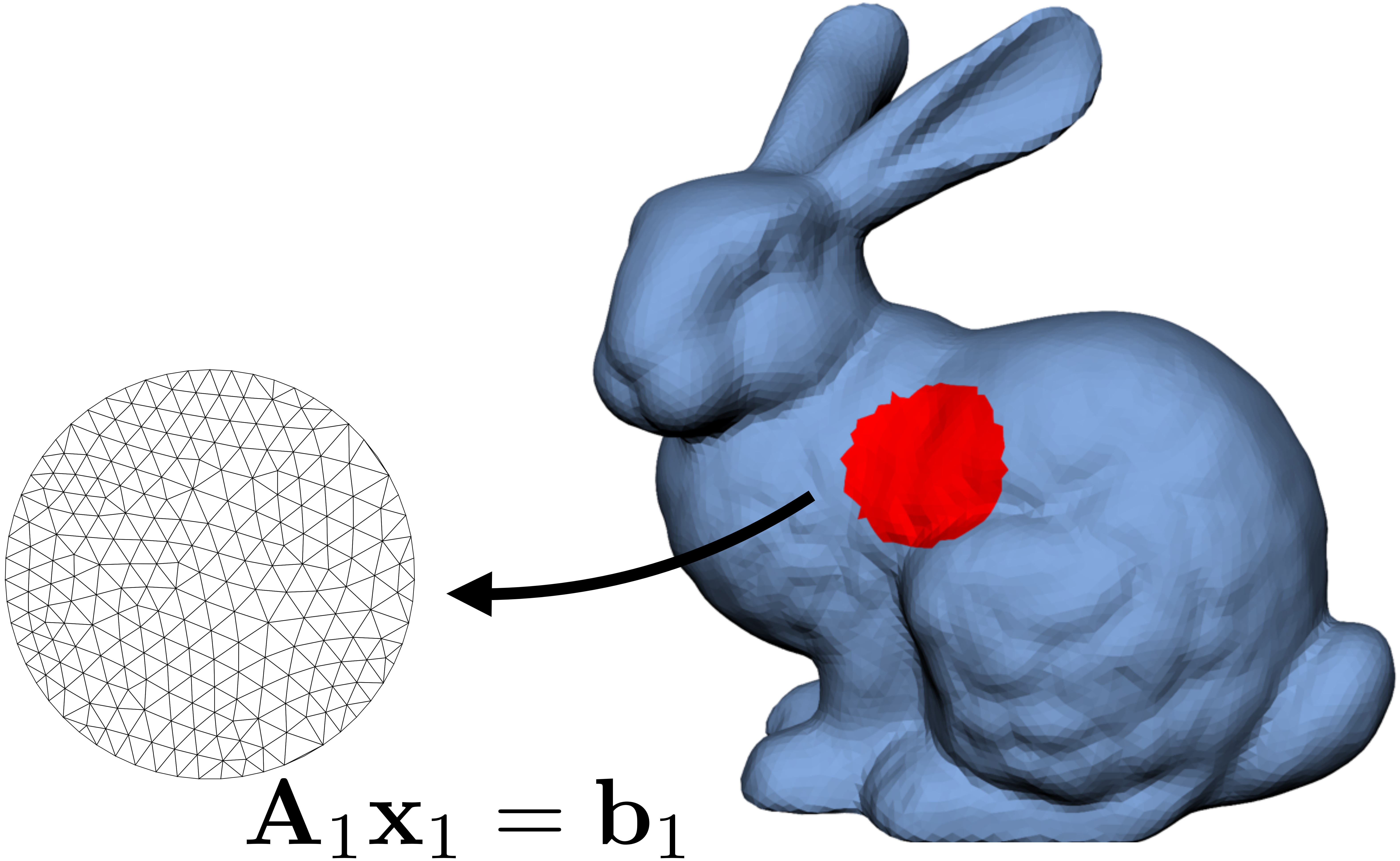
 $\mathbf{y} = \mathbf{b}$ (1)Forward solve

 $\mathbf{x} = \mathbf{y}$ (2)Back solve

Local discrete conformal parameterization



Local discrete conformal parameterization



$$\mathbf{A}_1 \mathbf{x}_1 = \mathbf{b}_1$$

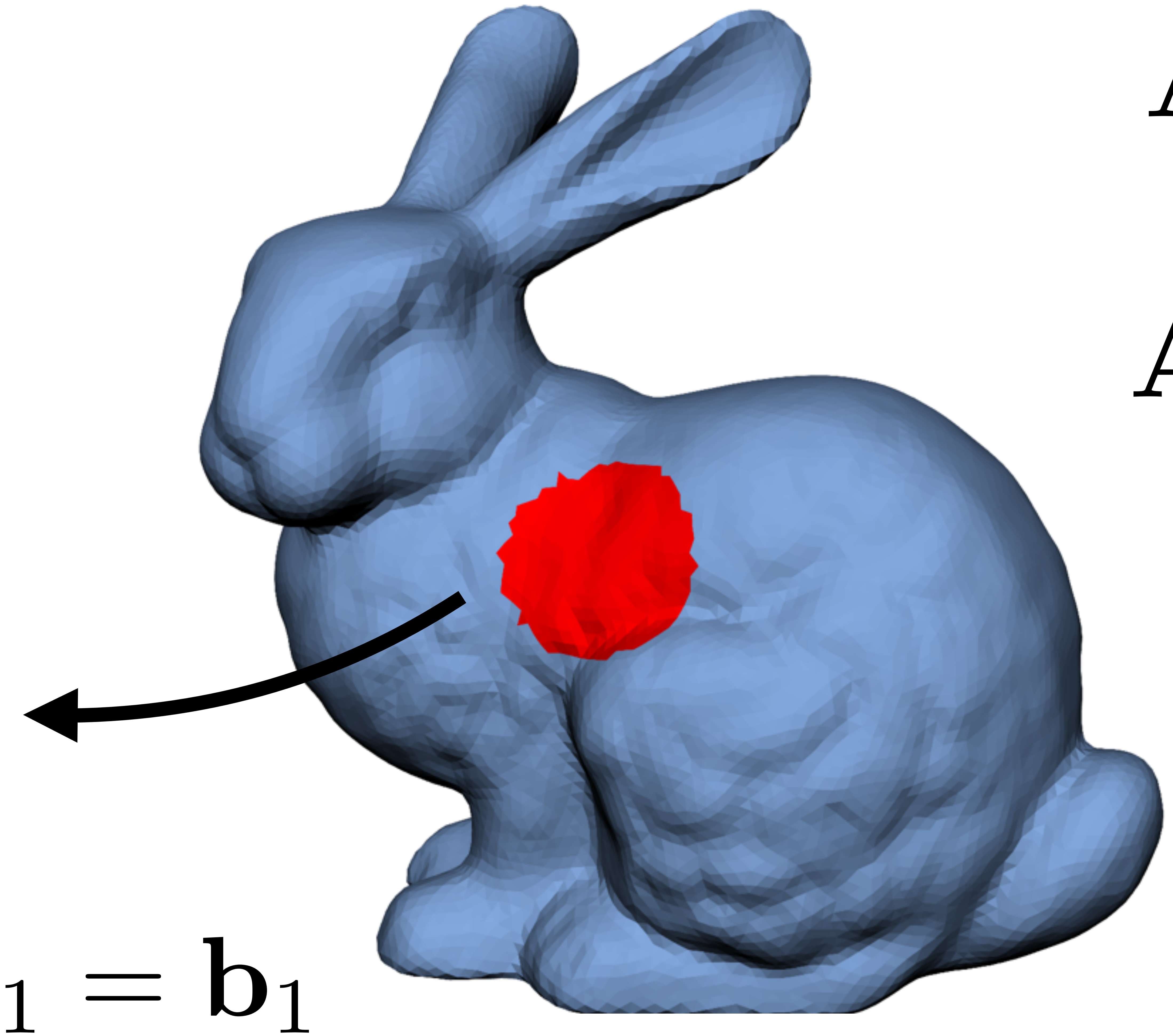
Local discrete conformal parameterization

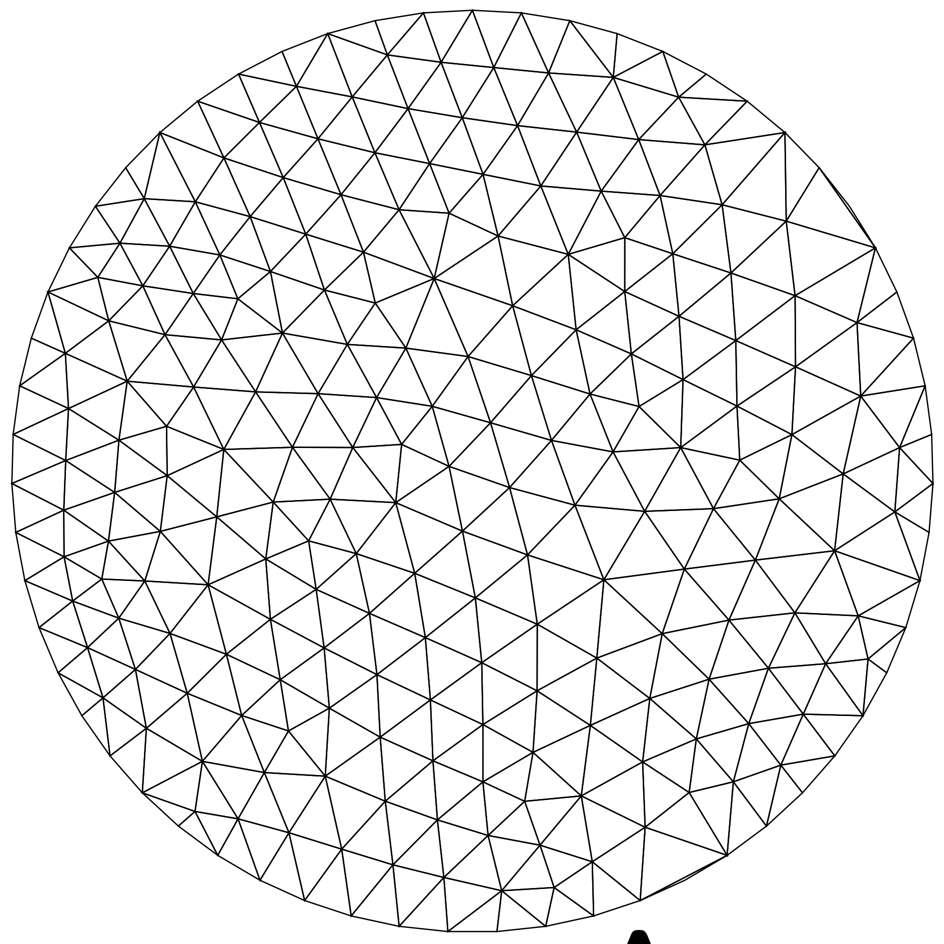
$$\mathbf{A}_1 = \mathbf{A}\Big($$  $$\Big)$$

$$\mathbf{A}_1 = \mathbf{LDL}^T$$

$$\mathbf{LDy} = \mathbf{b}_1$$

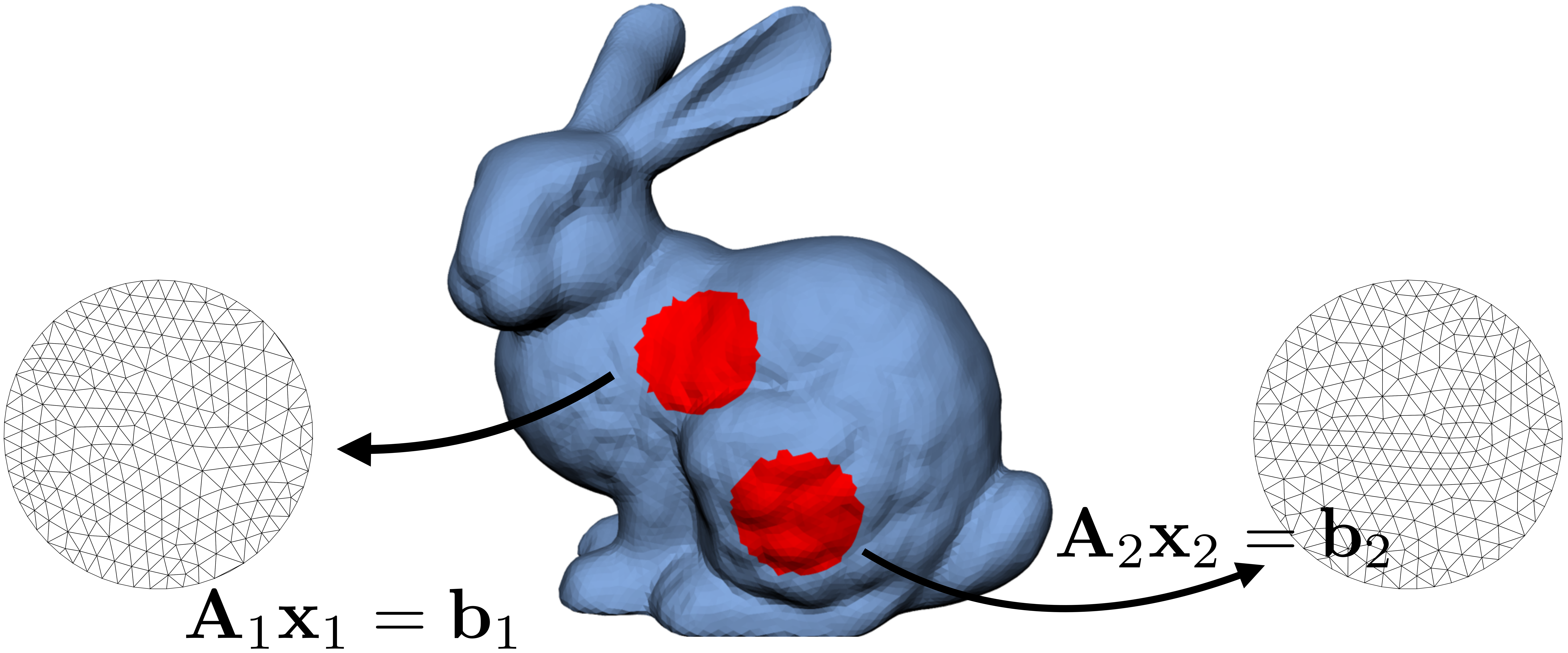
$$\mathbf{L}^T\mathbf{x}_1 = \mathbf{y}$$



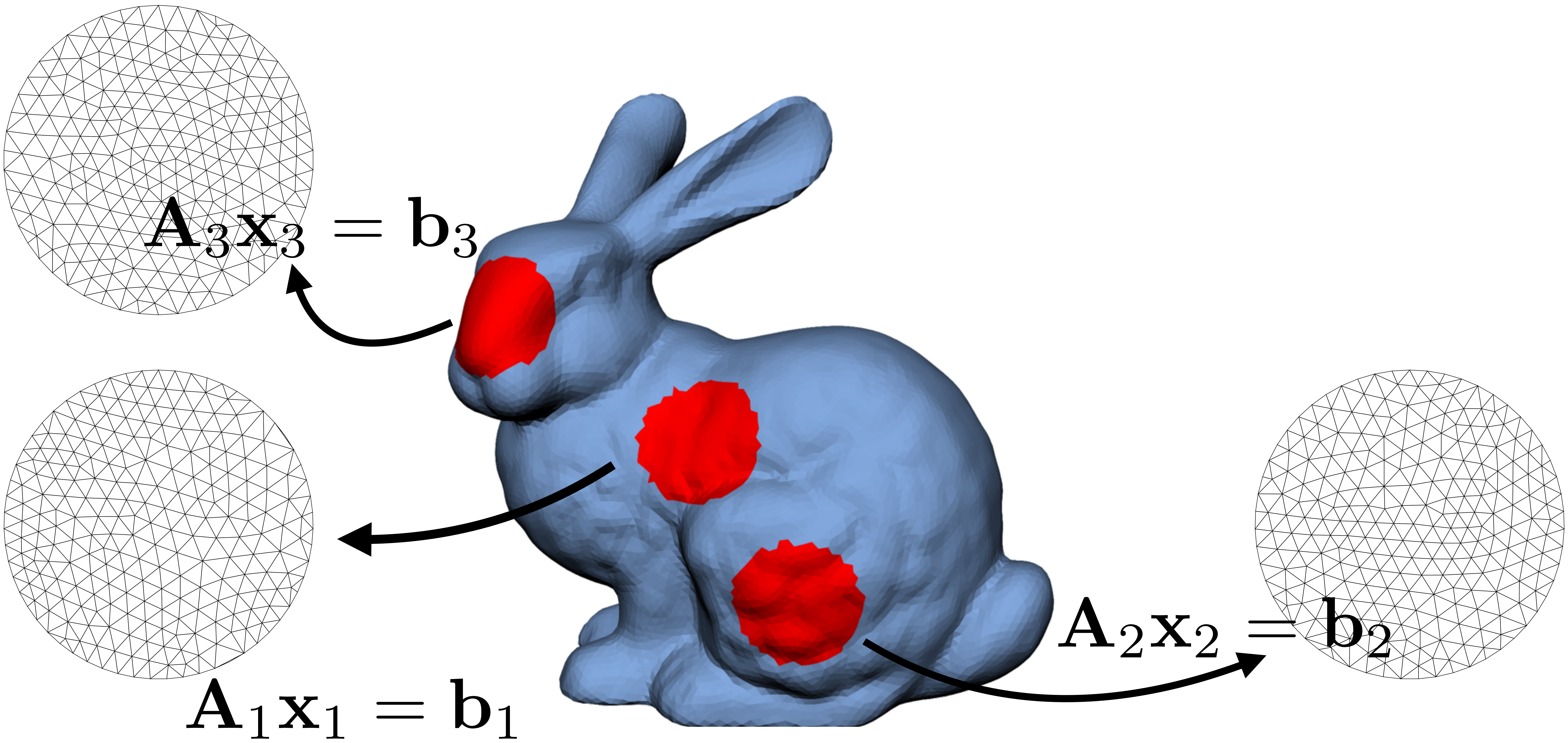


$$\mathbf{A}_1\mathbf{x}_1 = \mathbf{b}_1$$

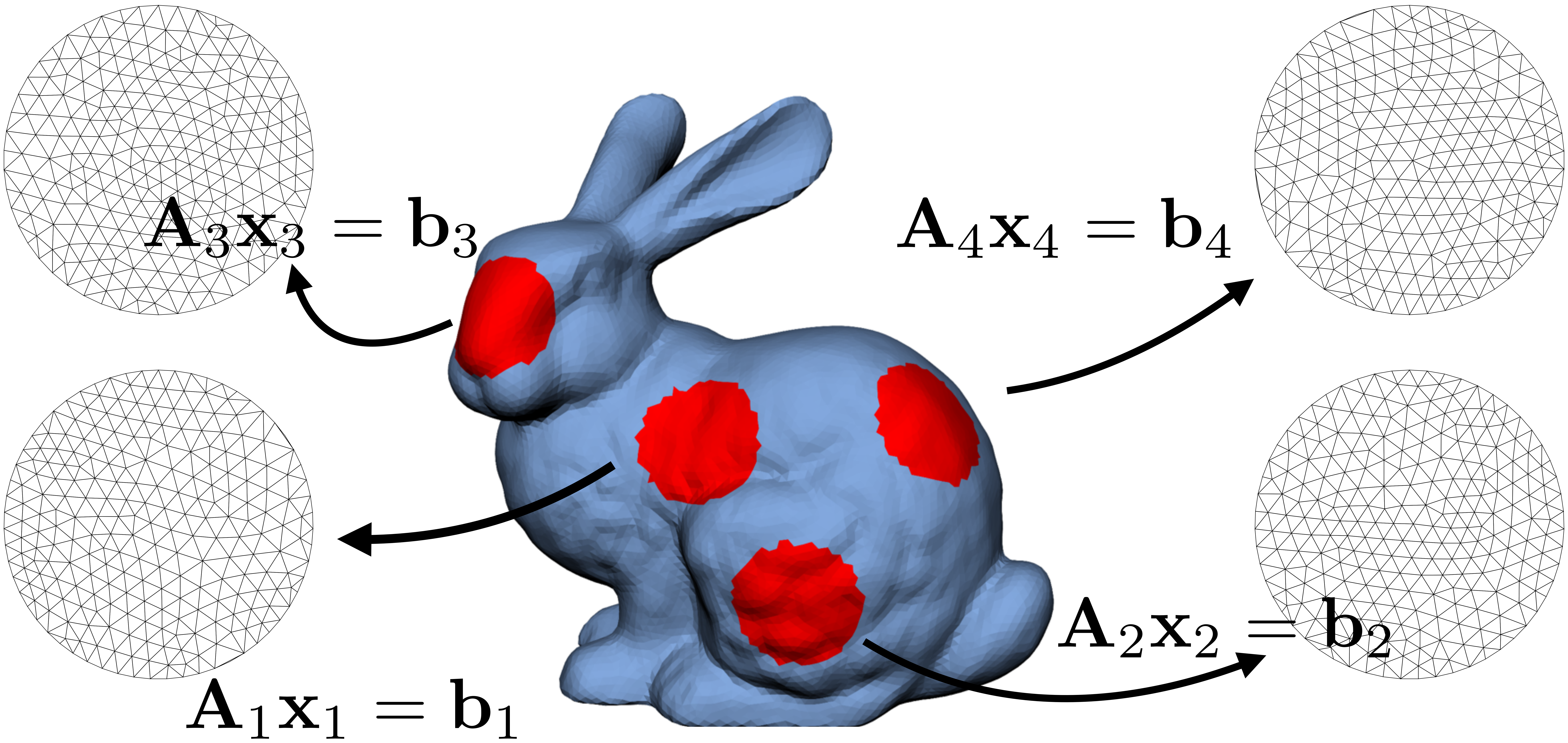
Local discrete conformal parameterization



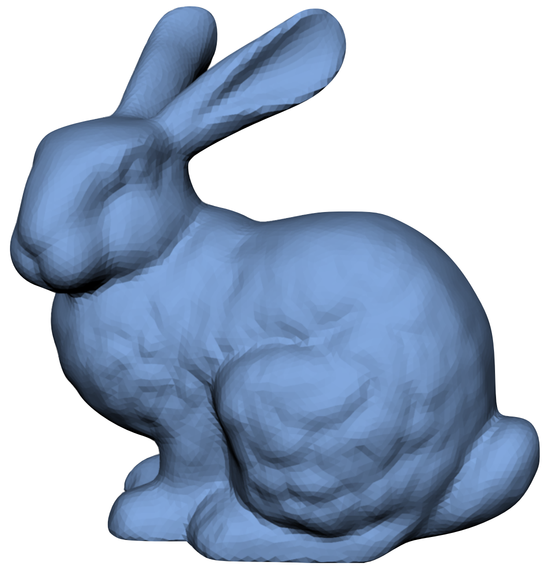
Local discrete conformal parameterization

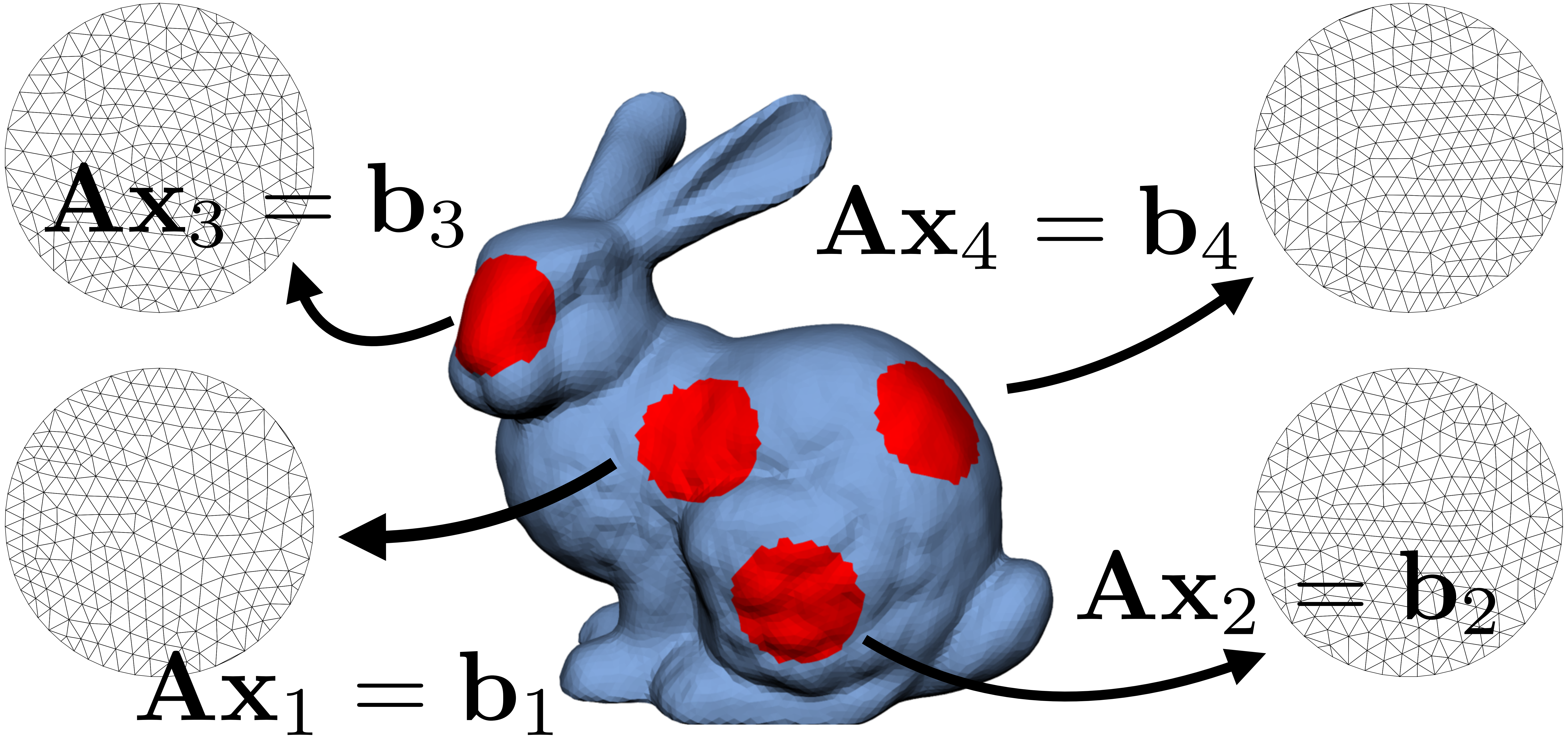


Local discrete conformal parameterization



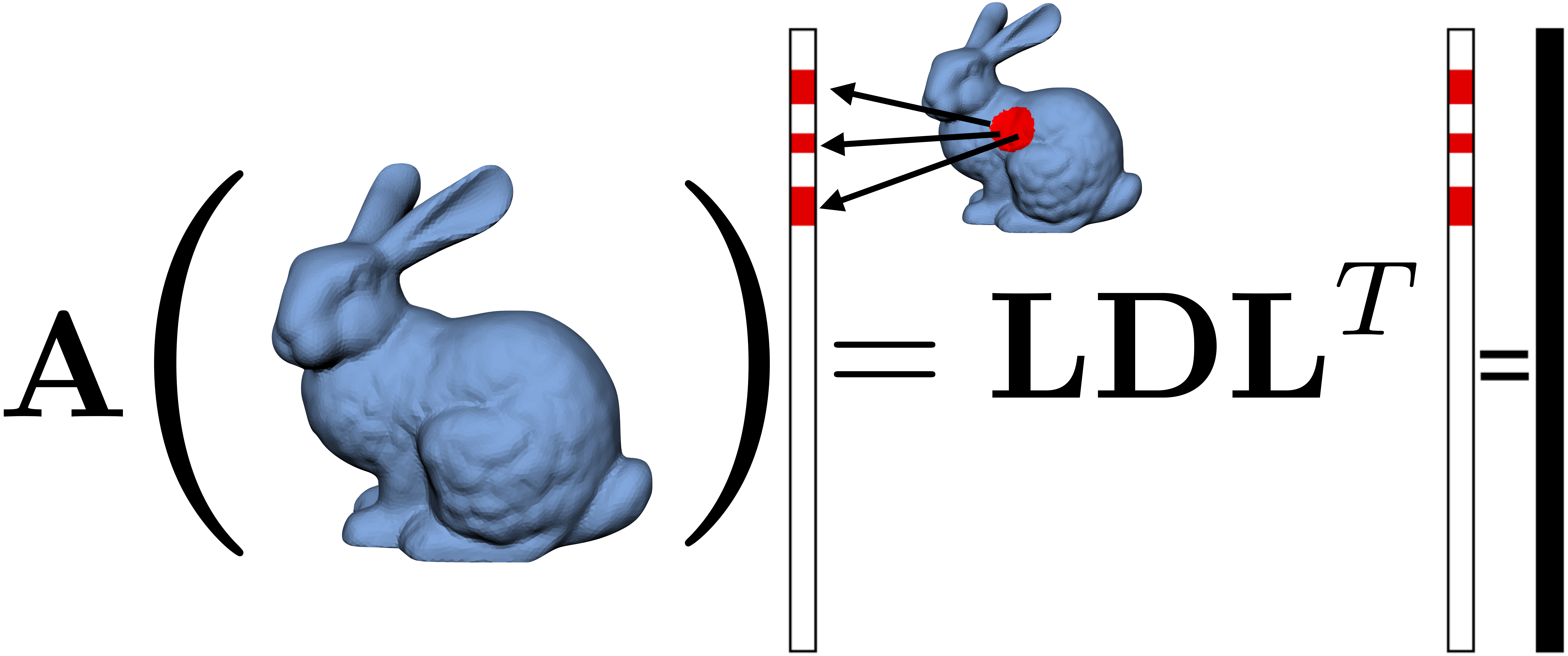
Contribution

- Idea: Reuse a global factorization: $\mathbf{A}\Bigg($  $\Bigg) = \mathbf{LDL}^T$



Contribution

- Question: Can we quickly compute subset of solution?



Sparse solution vector

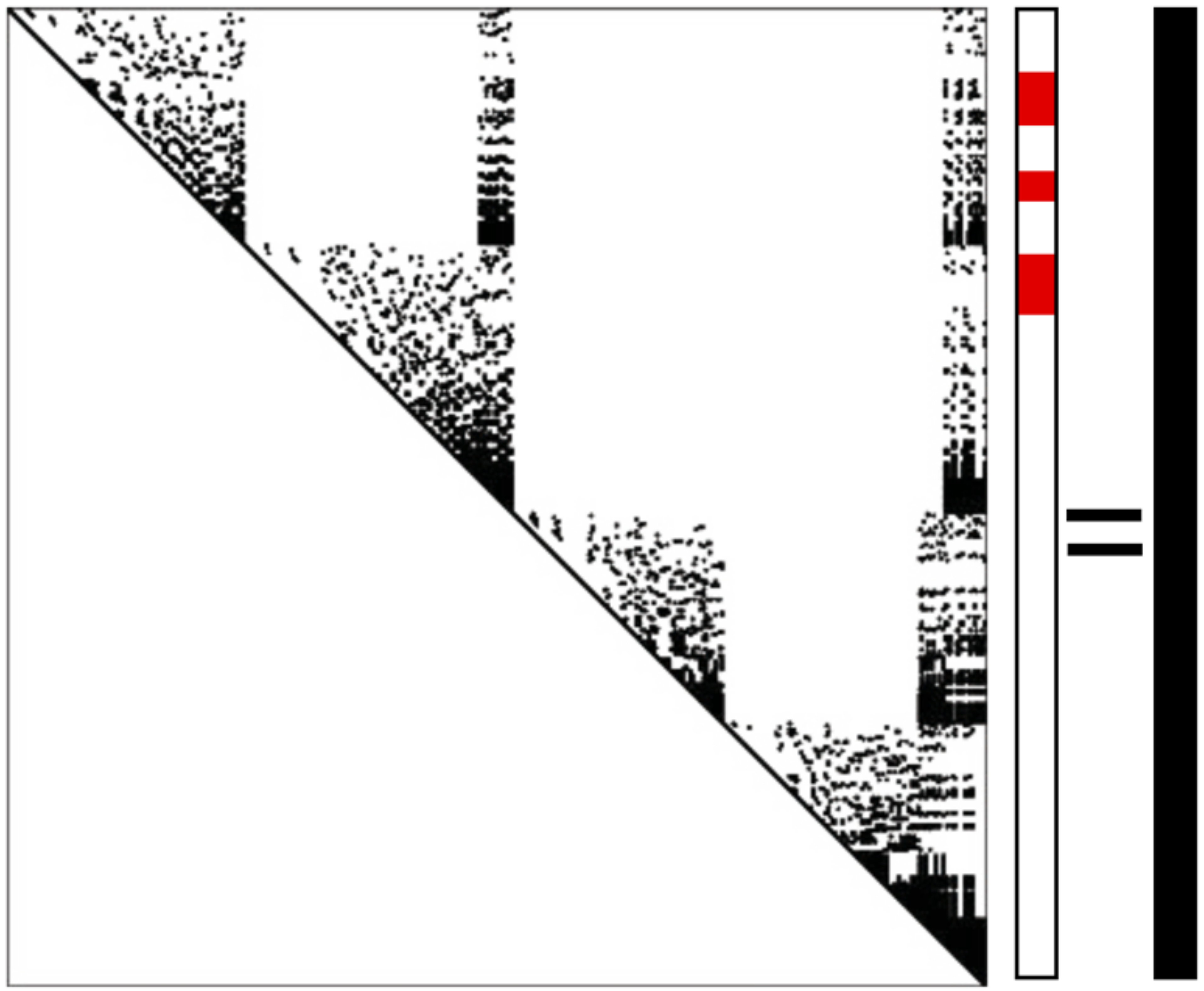
$$\mathbf{LDy} = \mathbf{b} \ (1) \text{Forward solve}$$

$$\mathbf{L}^T\mathbf{x} = \mathbf{y} \ (2) \text{Back solve}$$

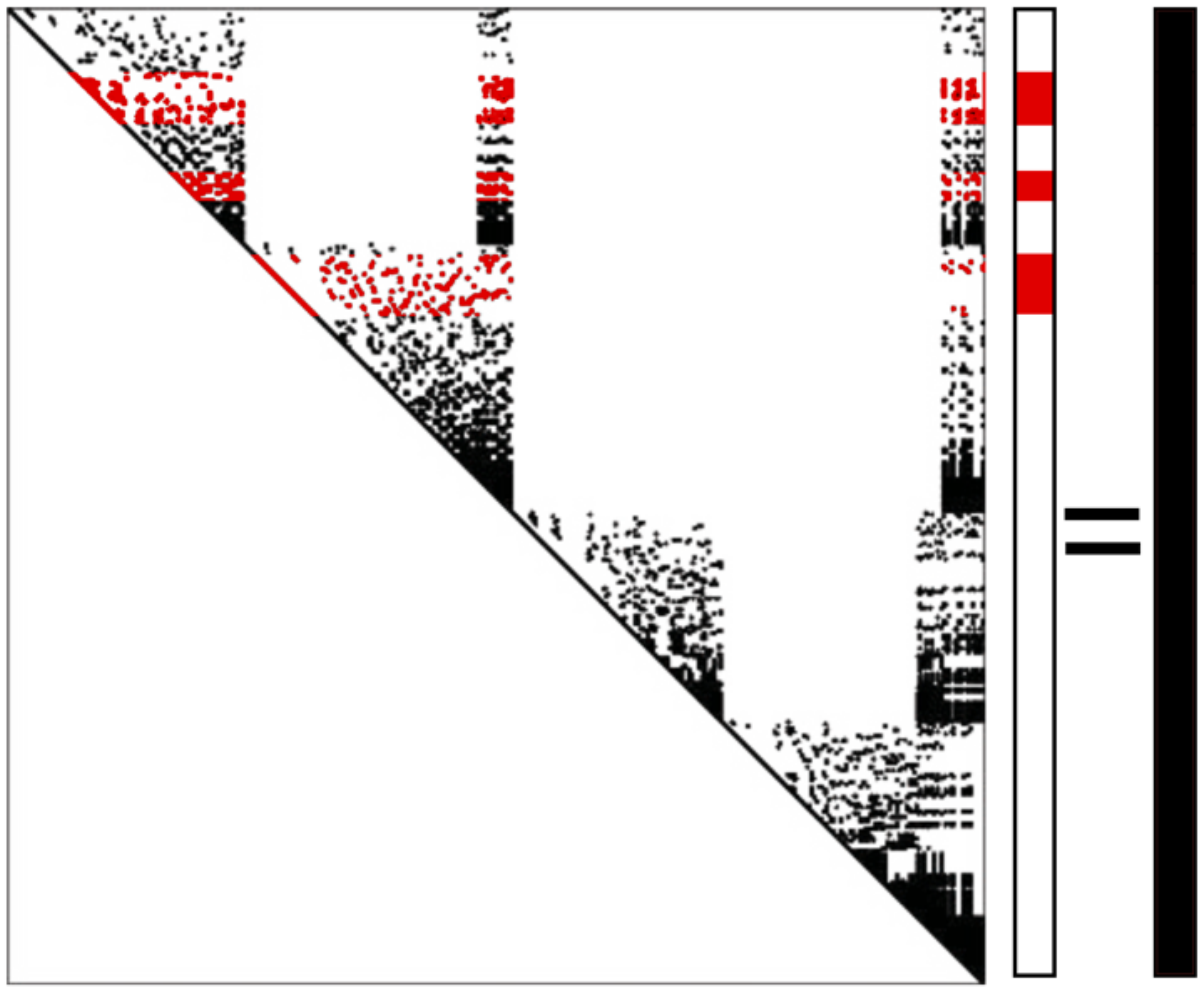
- $\mathbf{x}$ will generally be dense.

- **Central insight:** If we are interested in only a subset of values in $\mathbf{x}$ we do not have to compute all values during the back solve!

Sparse solution vector

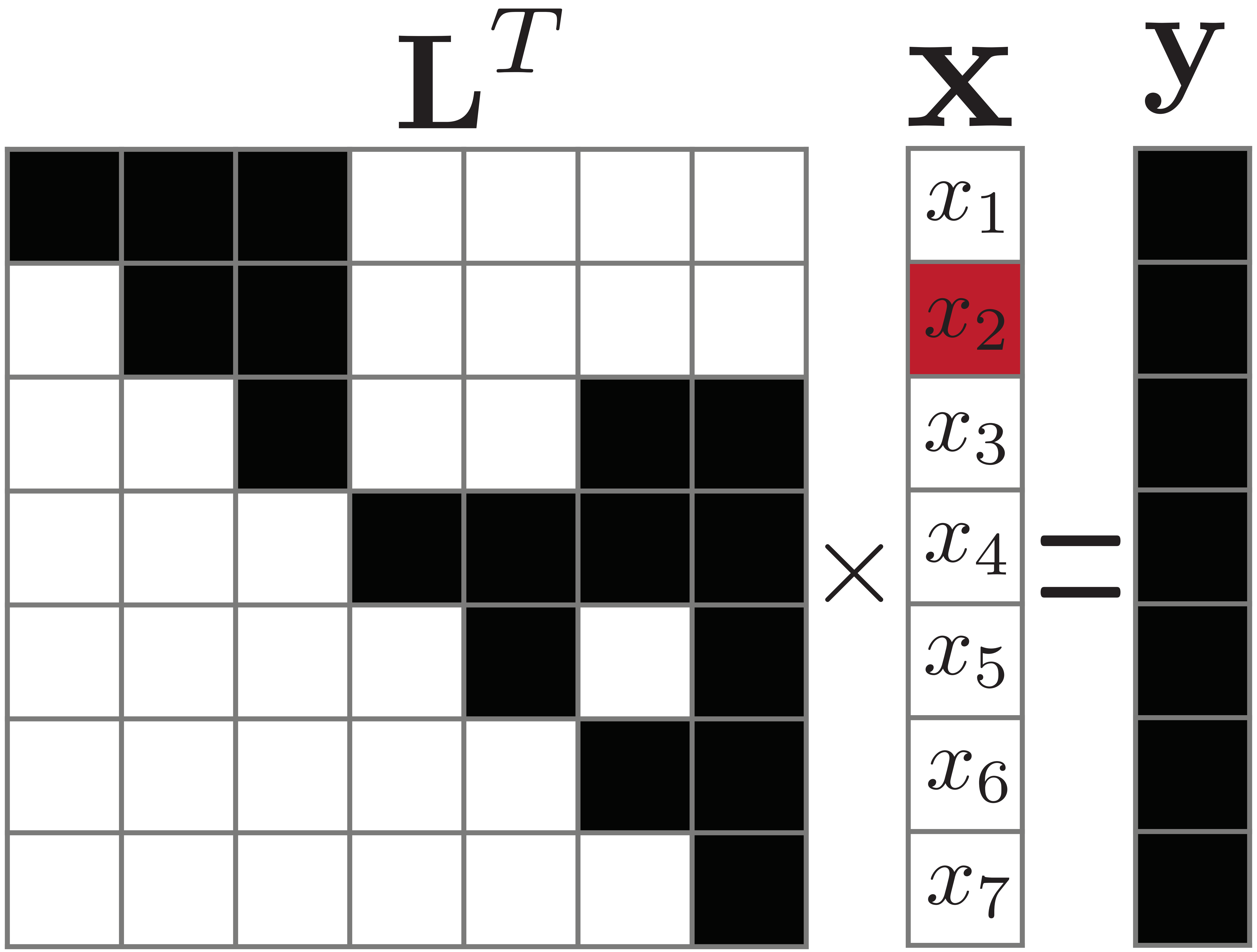


Sparse solution vector

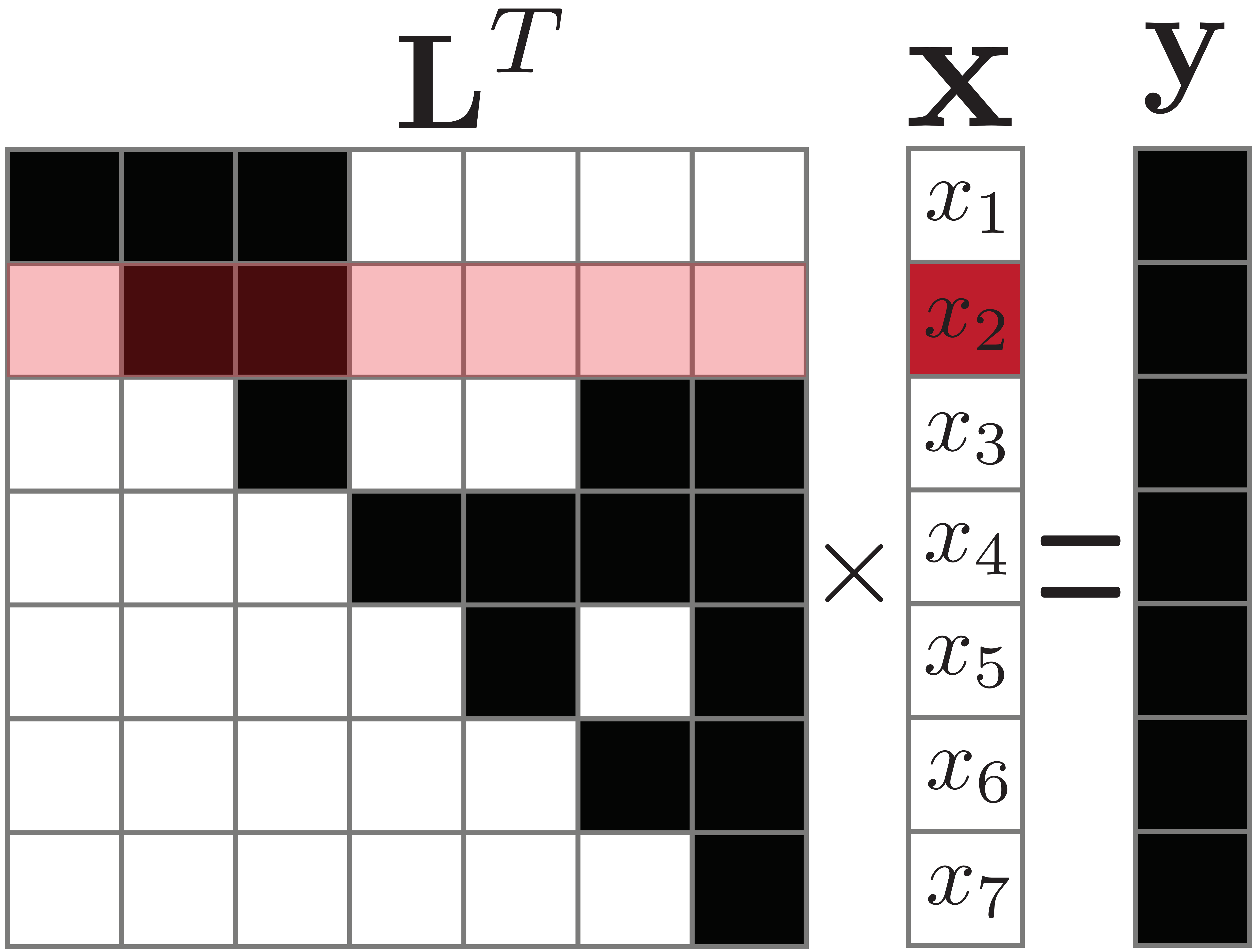


- How many values do we have to compute additionally?
- Can we identify them efficiently?

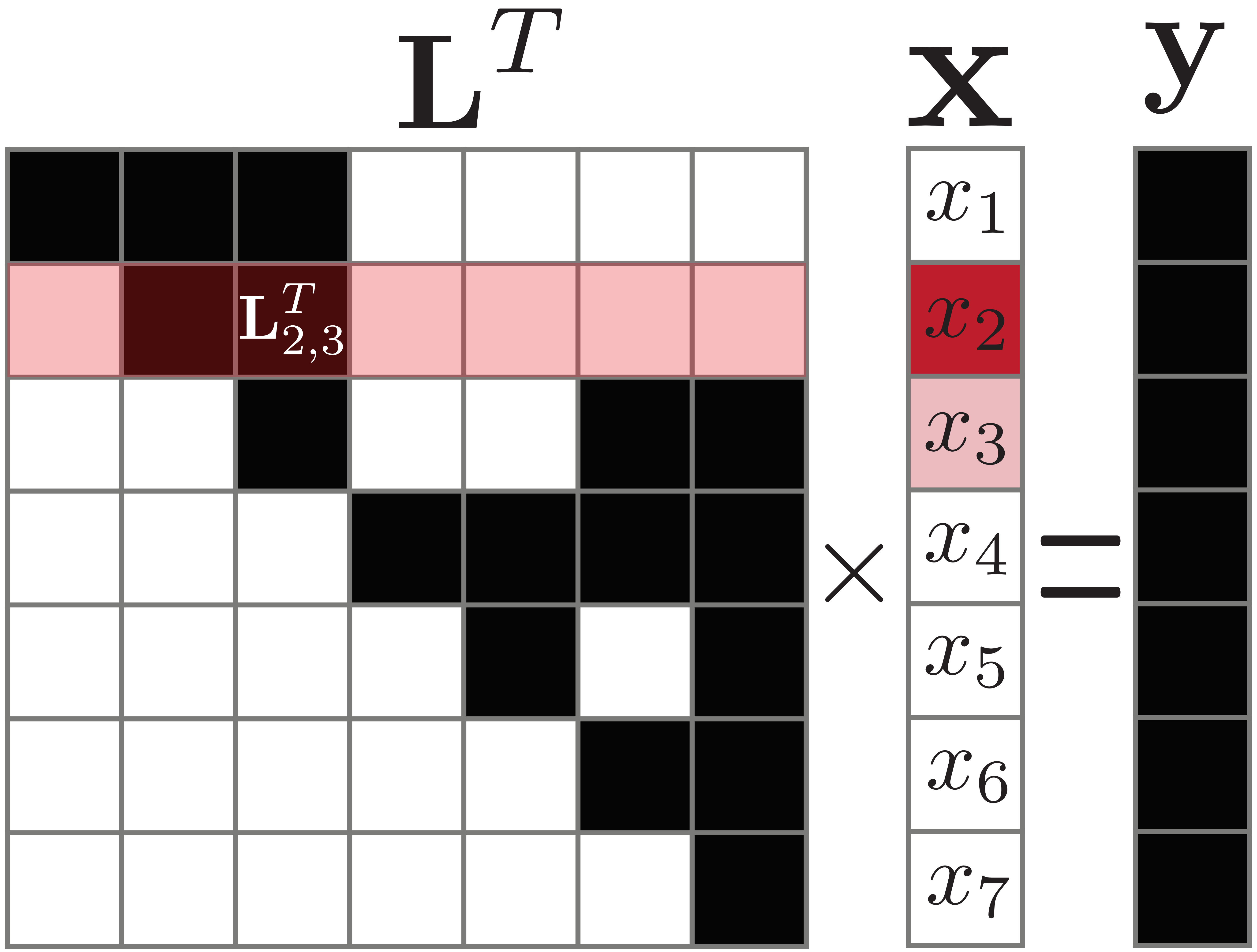
Sparse back-solve



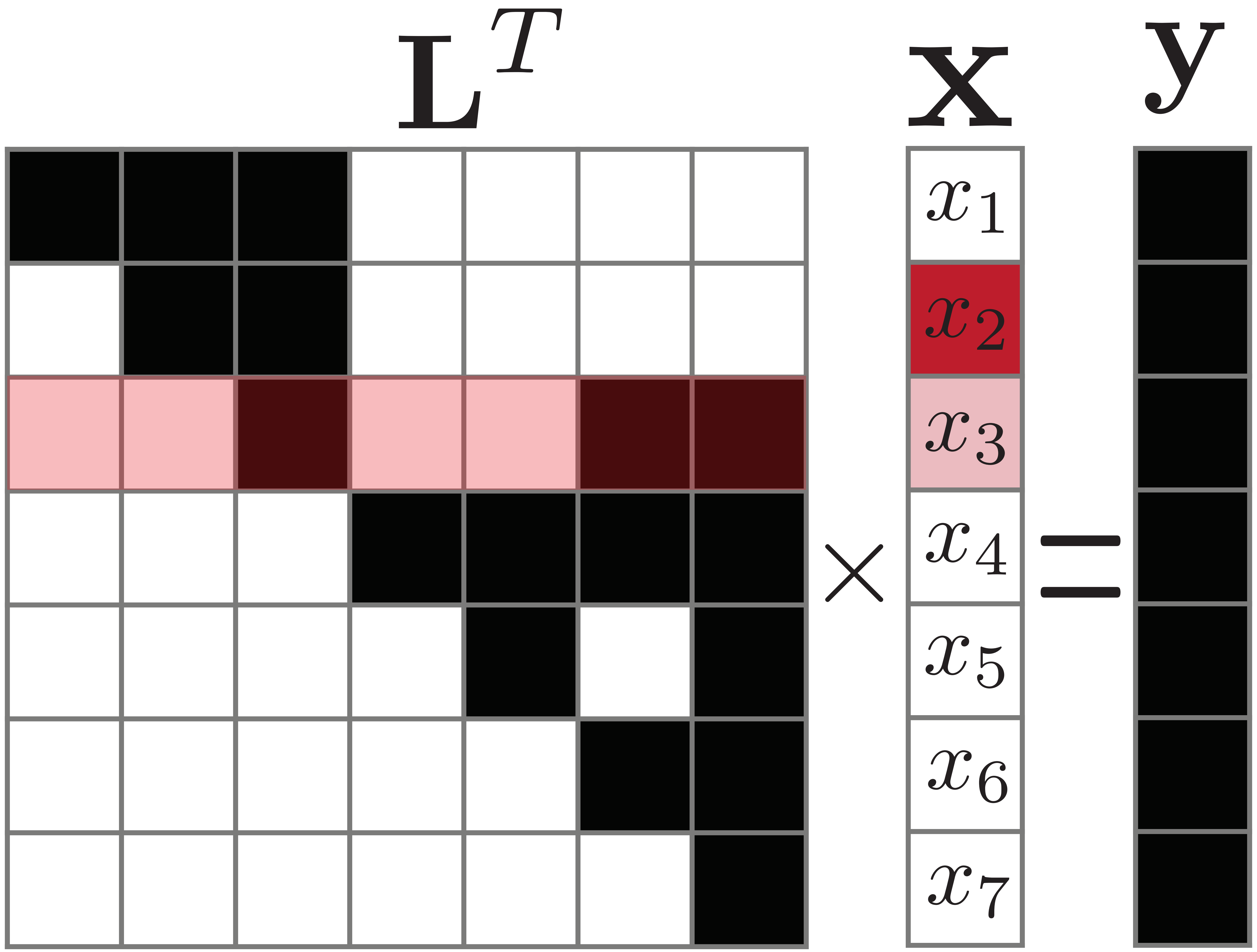
Sparse back-solve



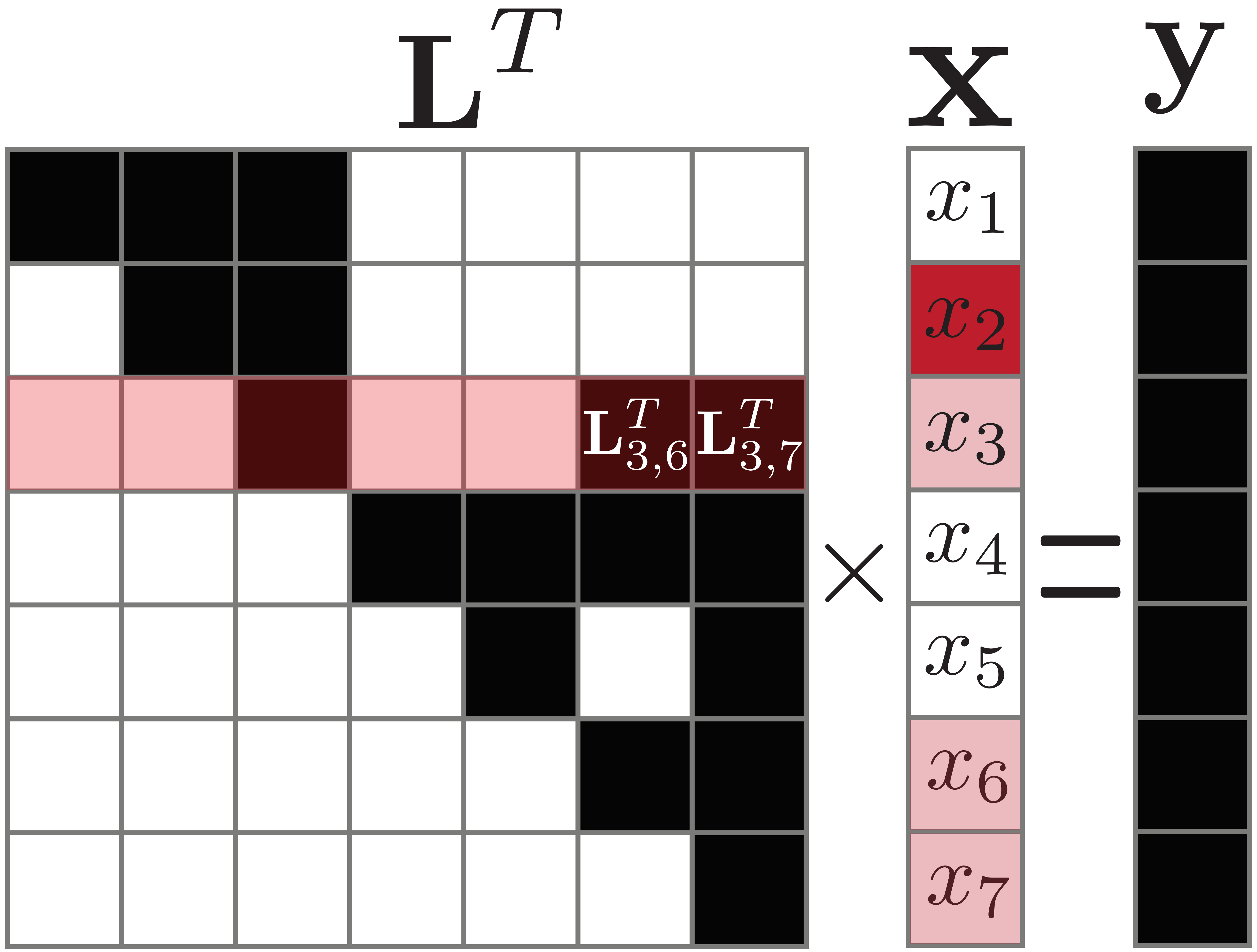
Sparse back-solve



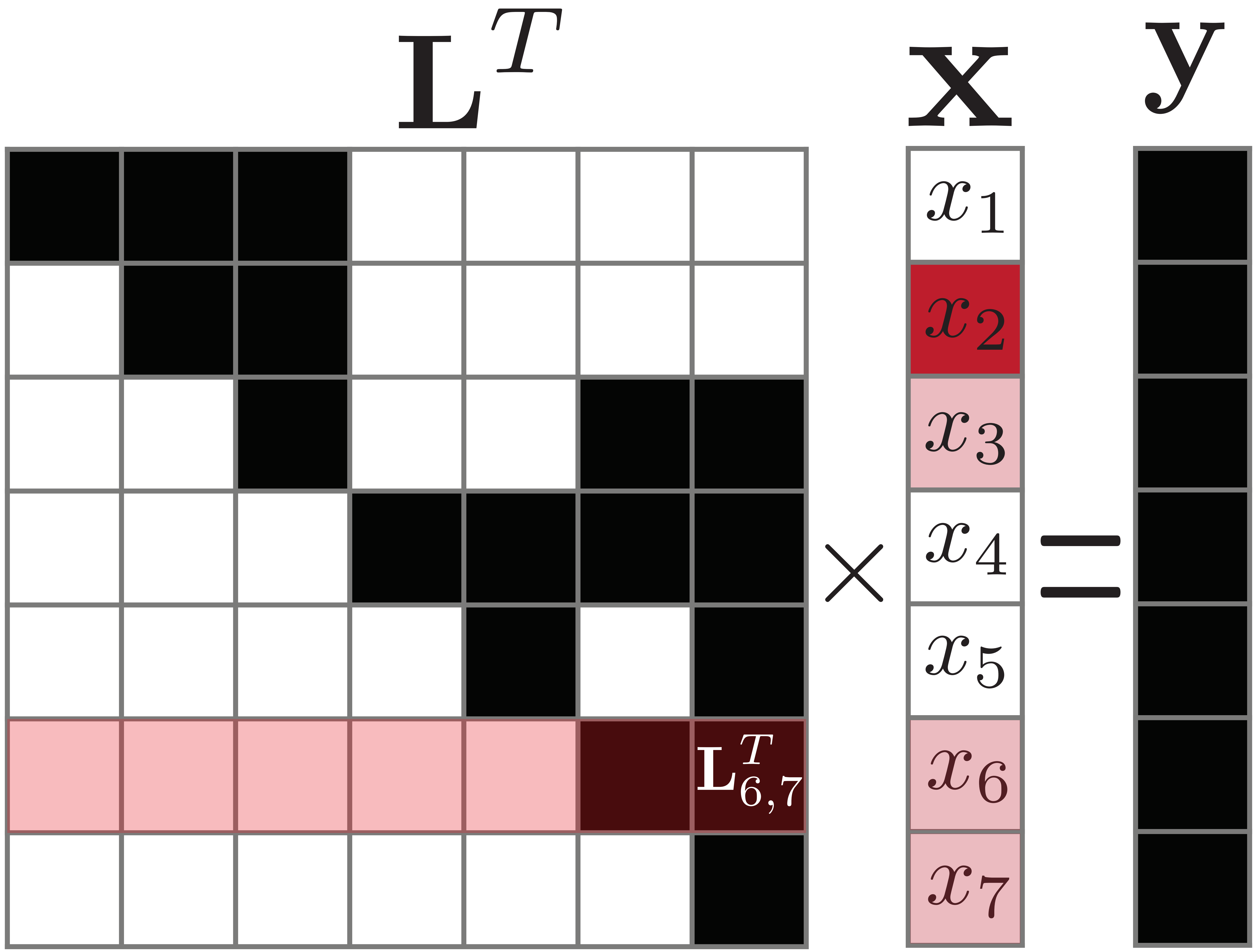
Sparse back-solve



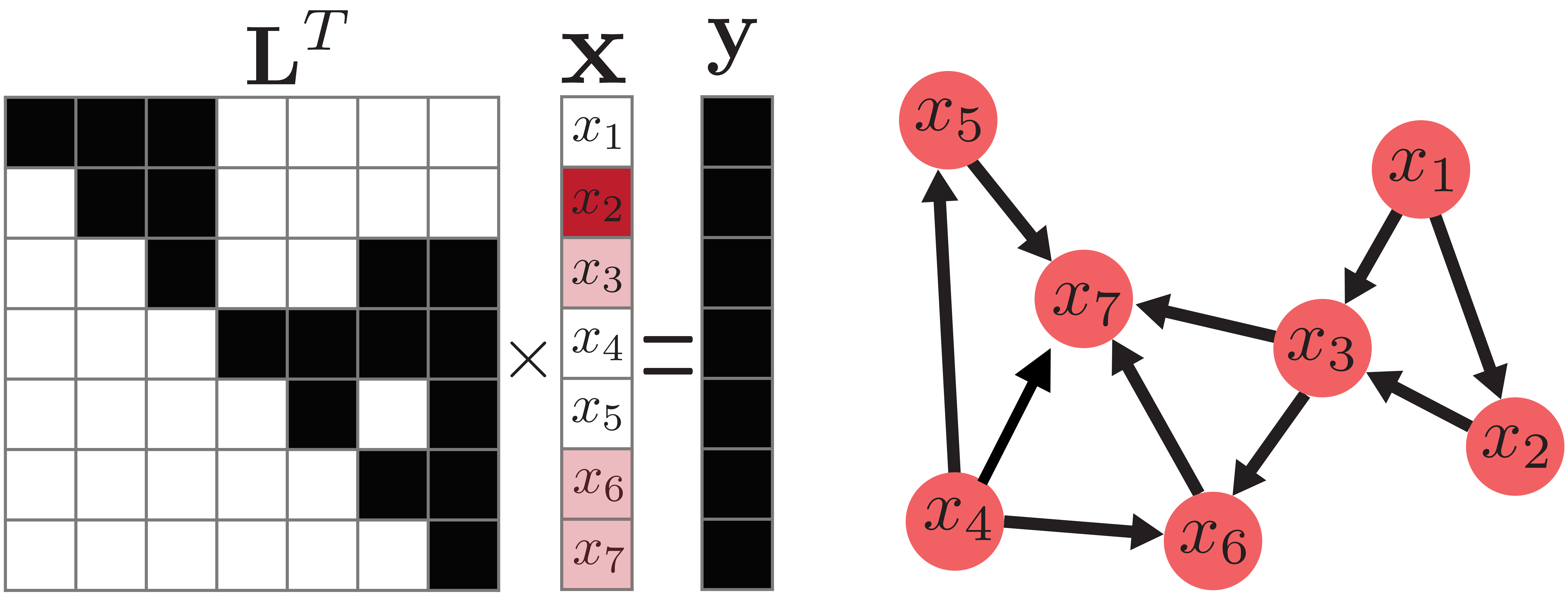
Sparse back-solve



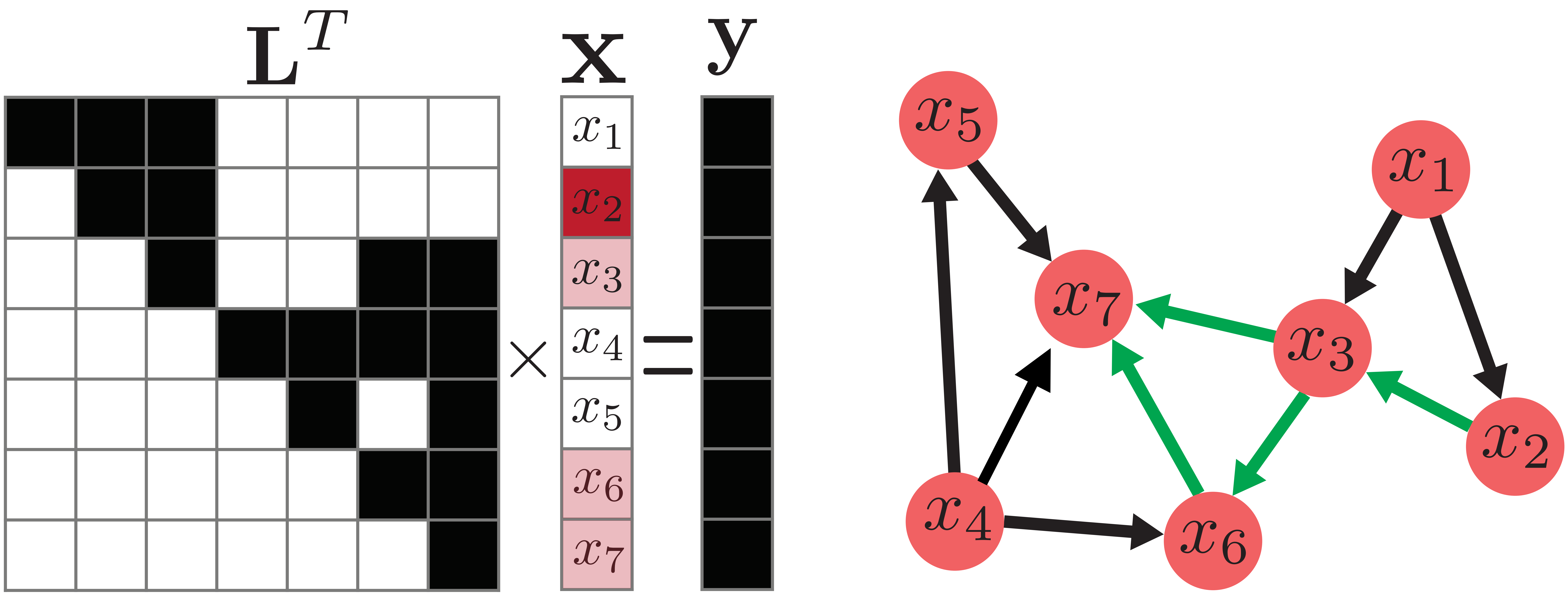
Sparse back-solve



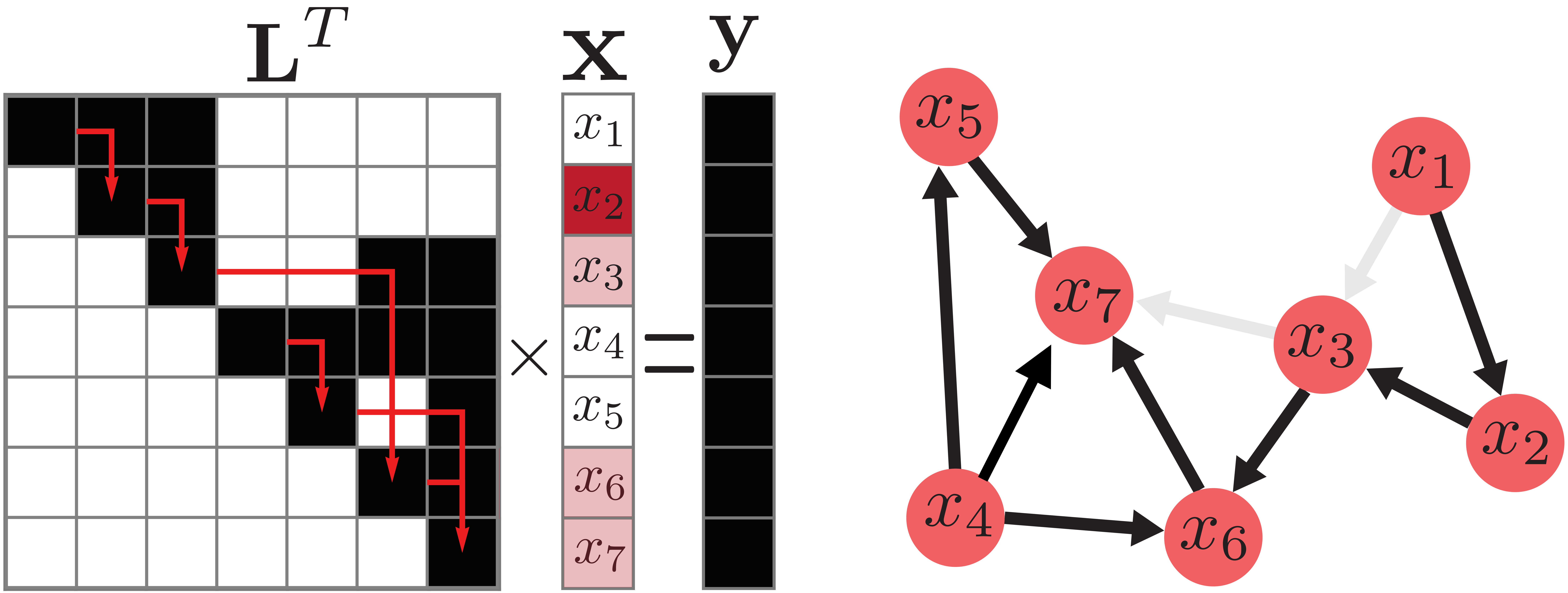
Sparse back-solve



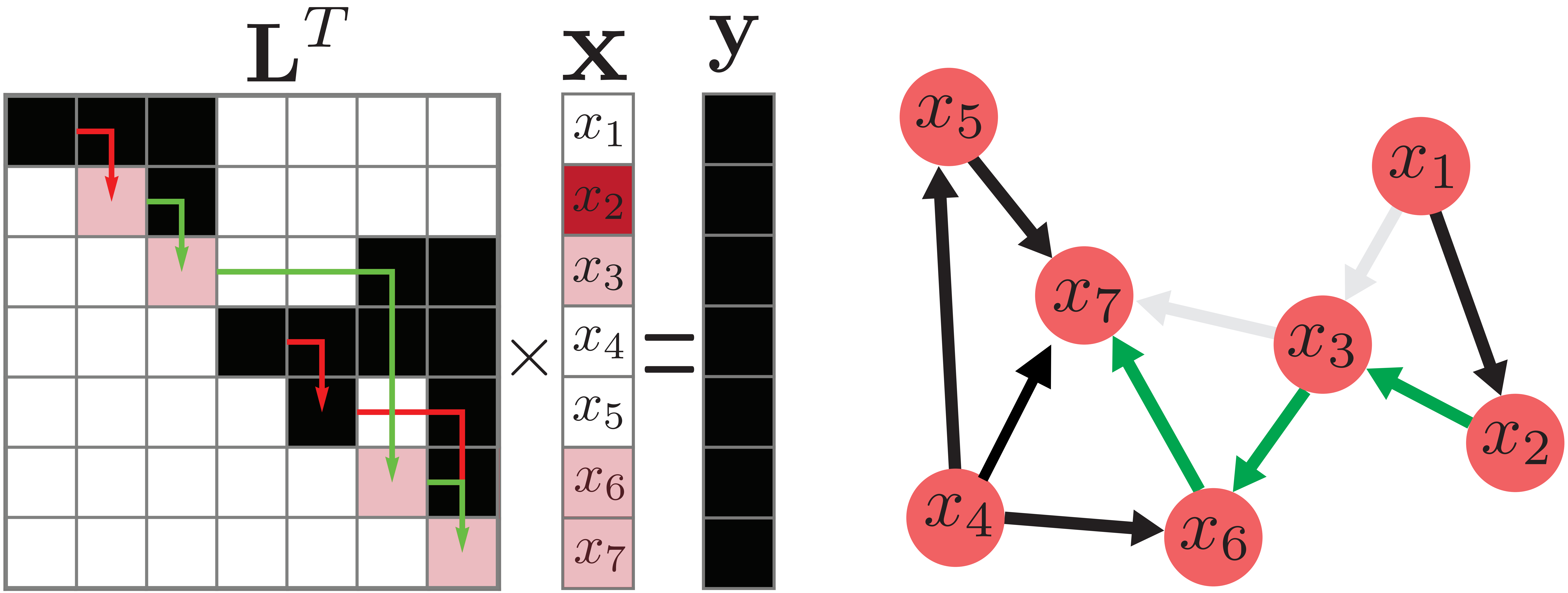
Sparse back-solve



Sparse back-solve

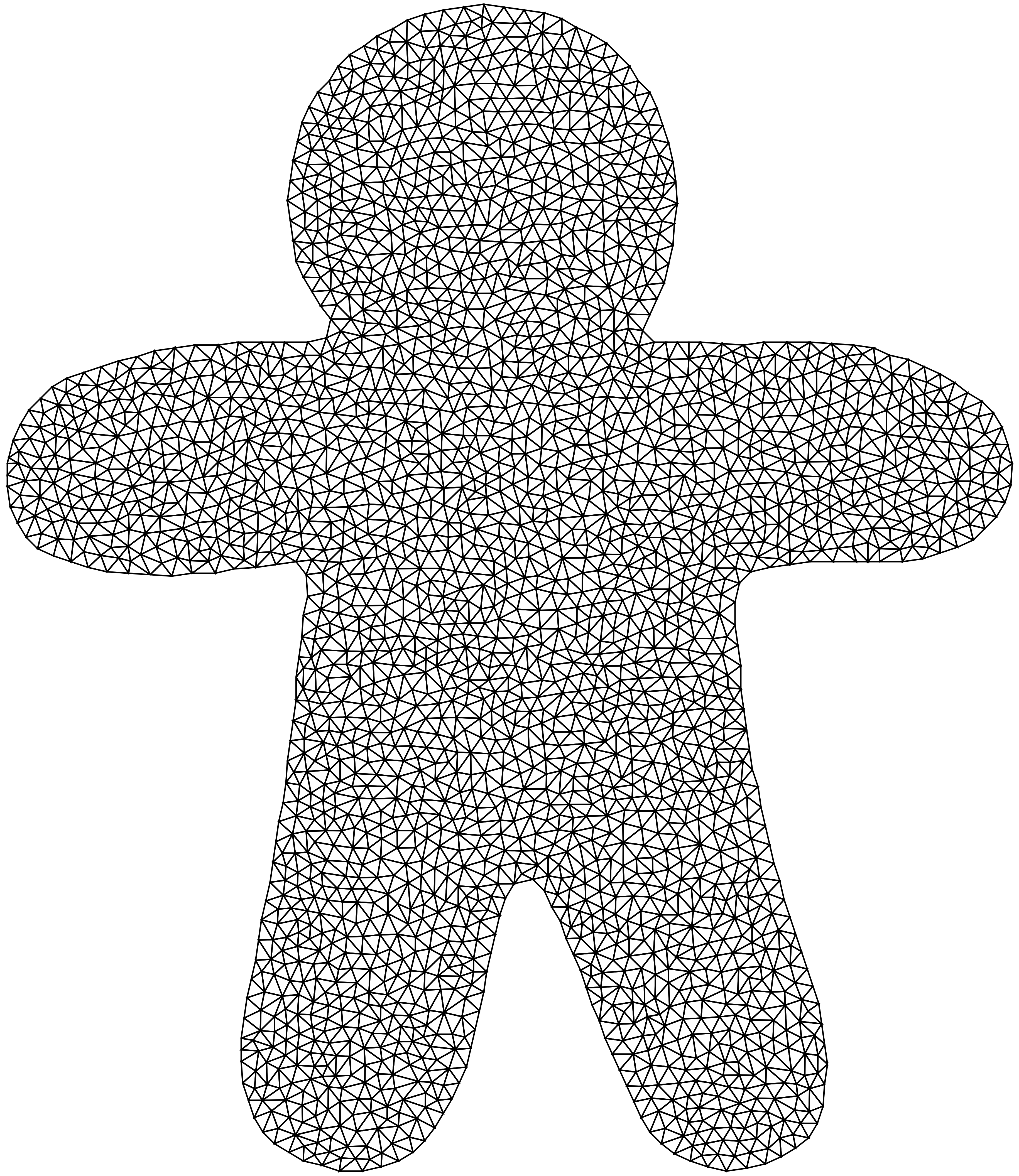


Sparse back-solve



Nested dissection reordering

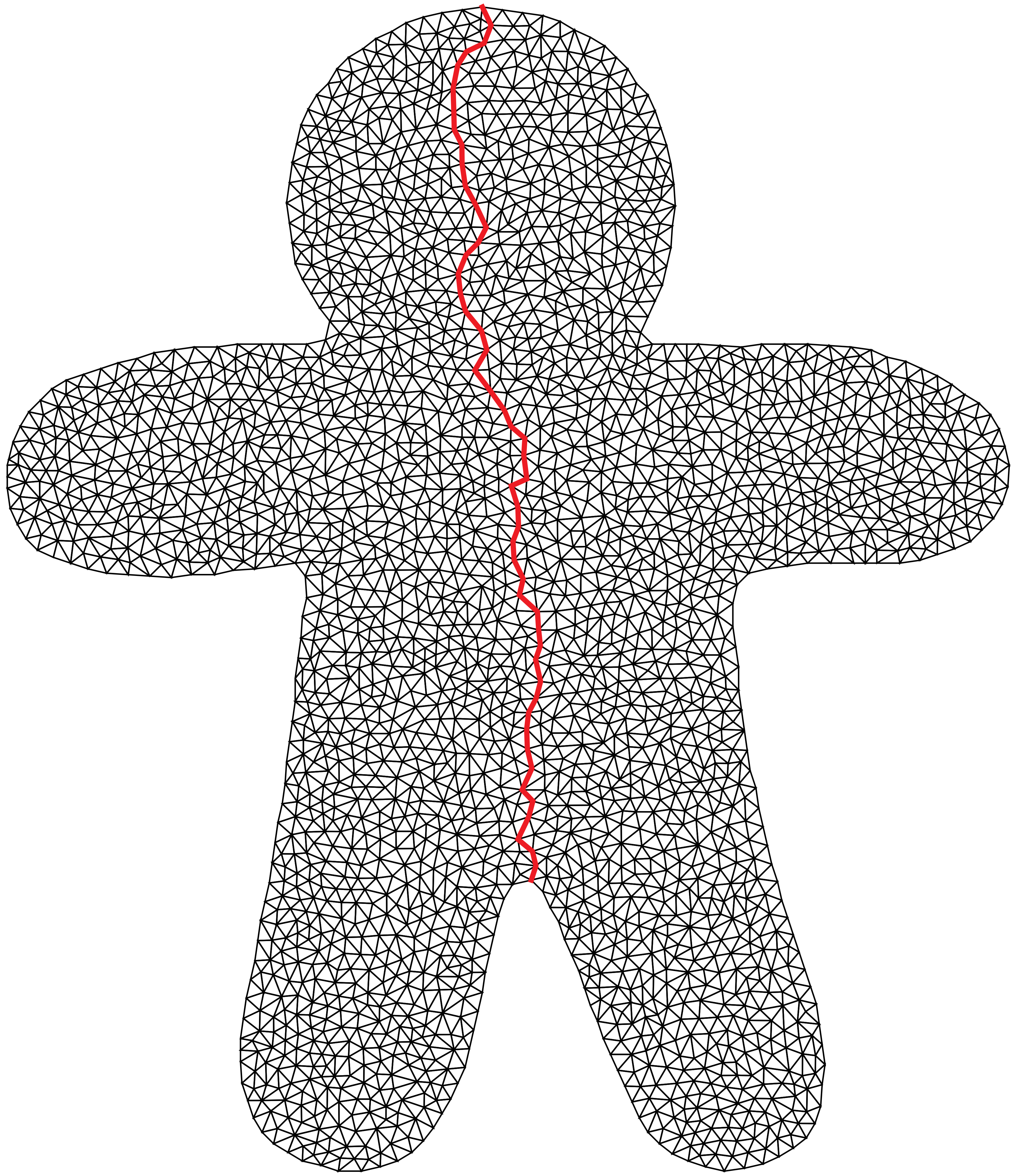
Input mesh



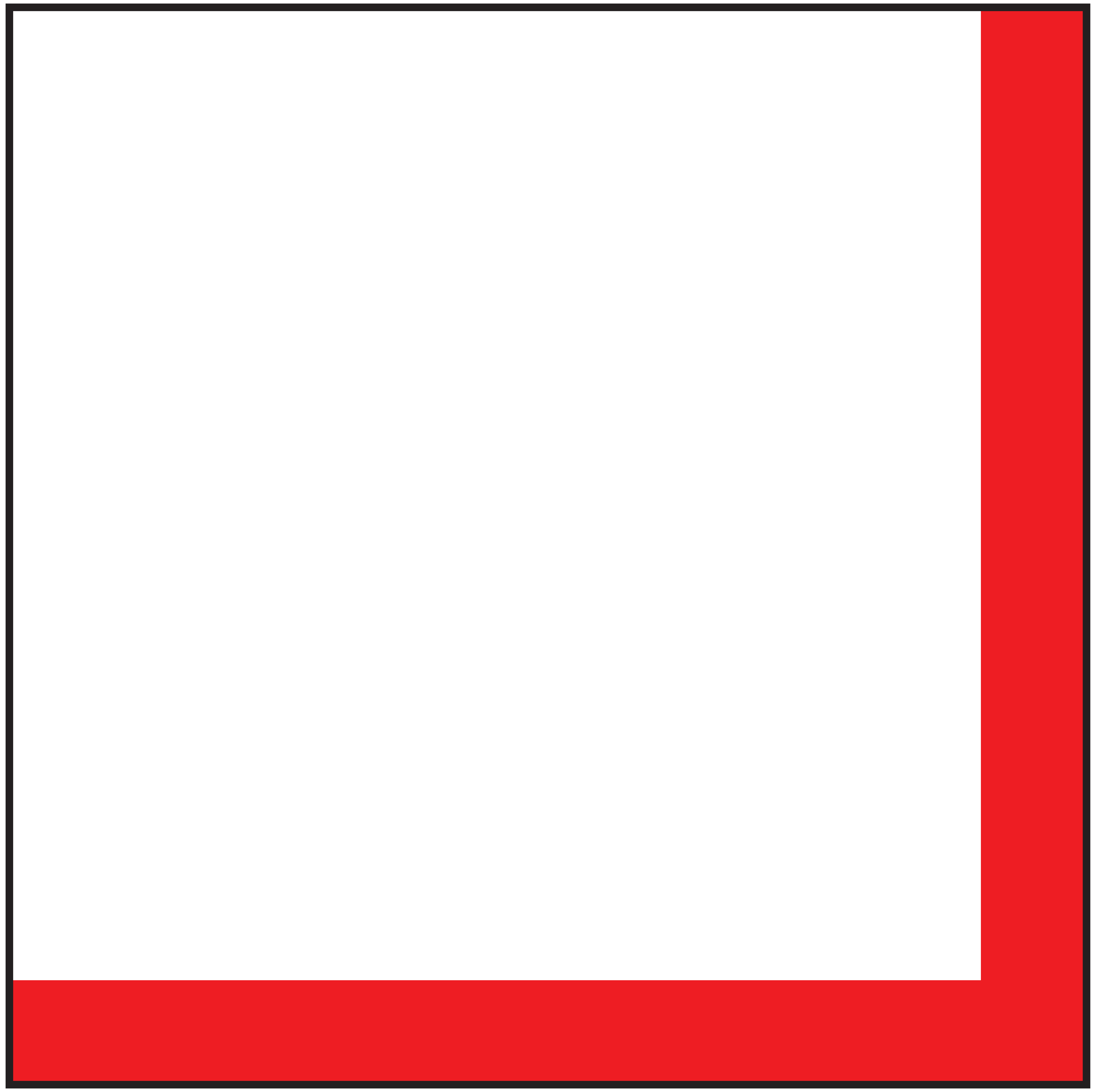
Ordered Laplacian

Nested dissection reordering

Input mesh

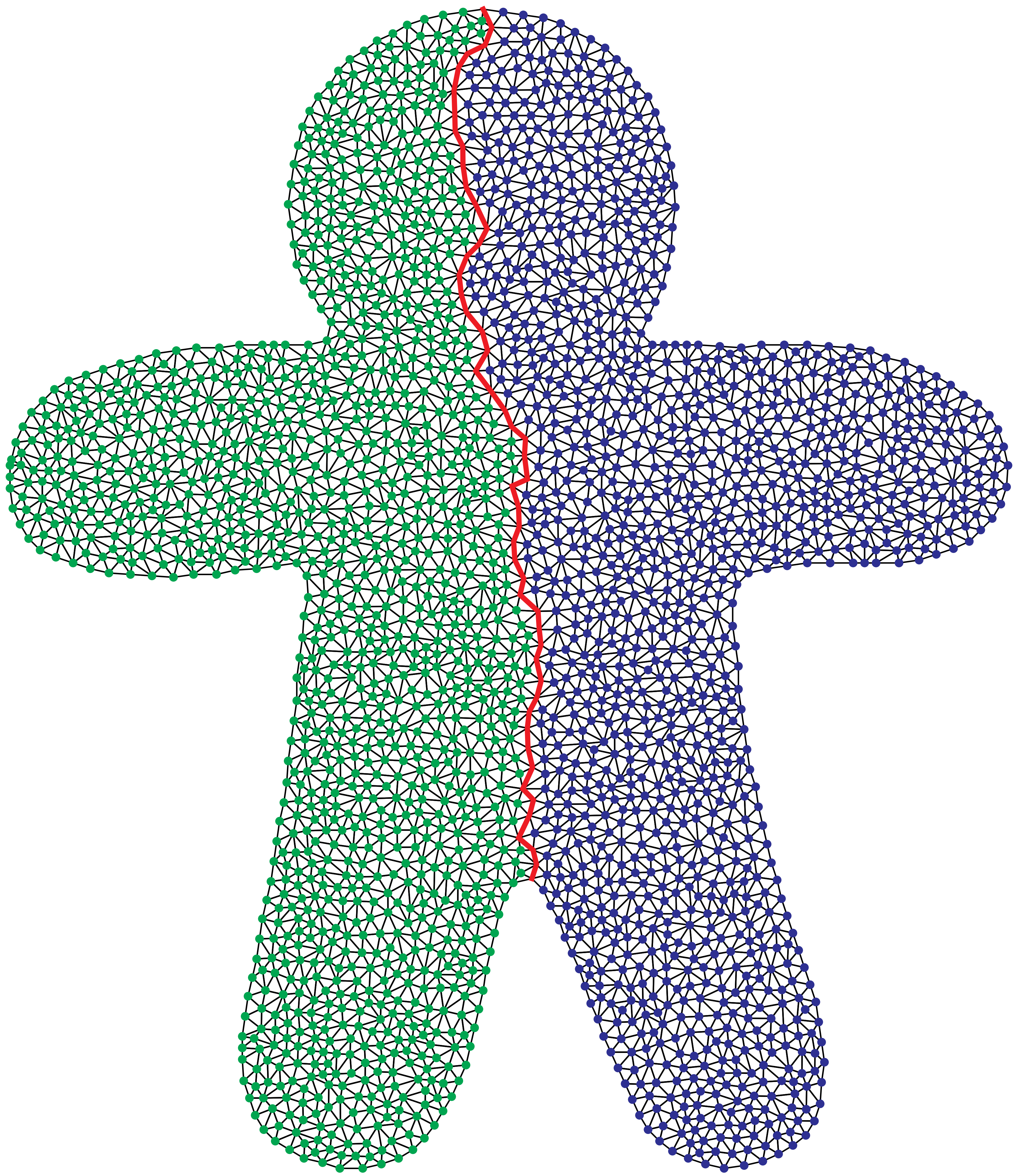


Ordered Laplacian

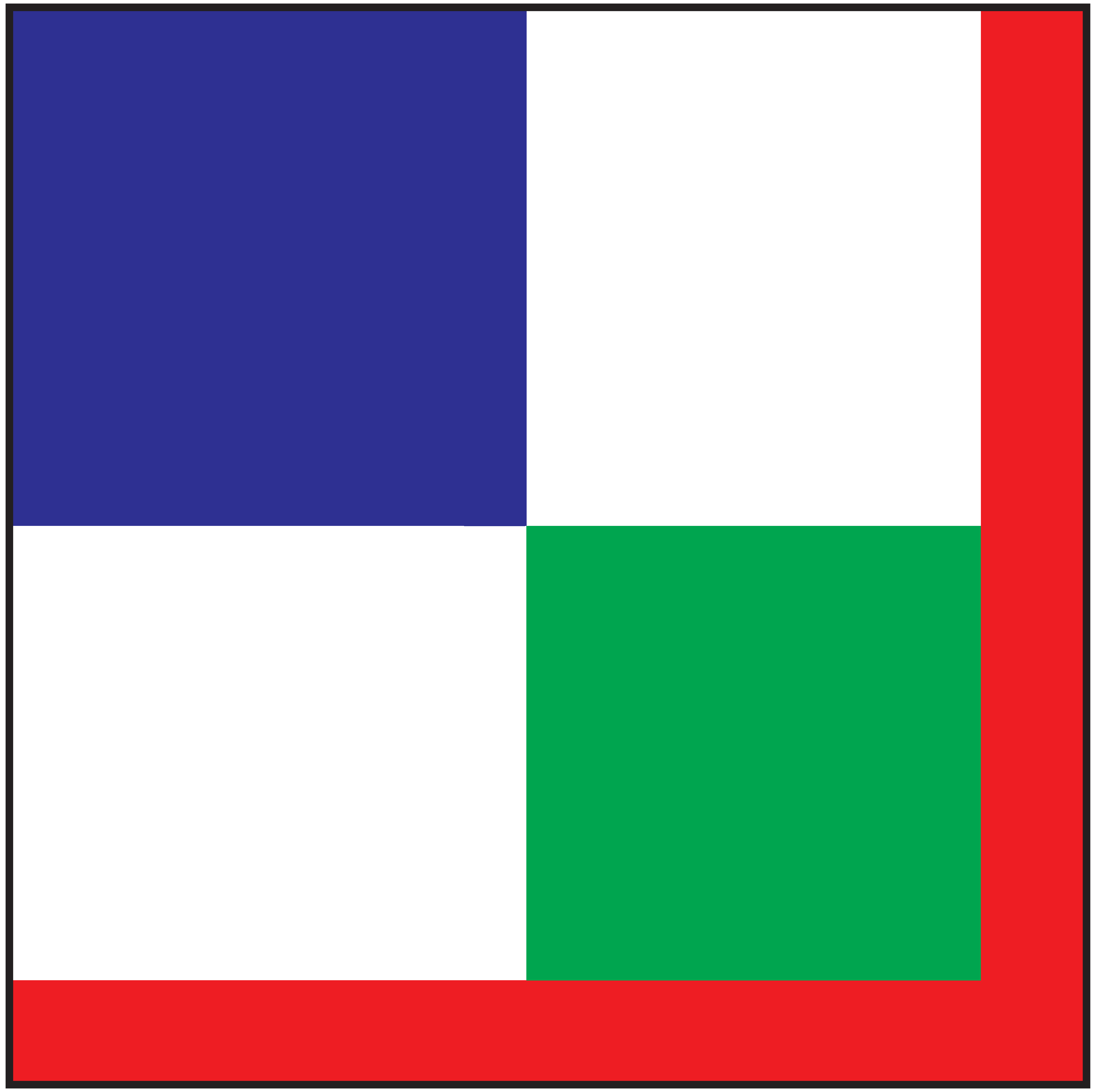


Nested dissection reordering

Input mesh

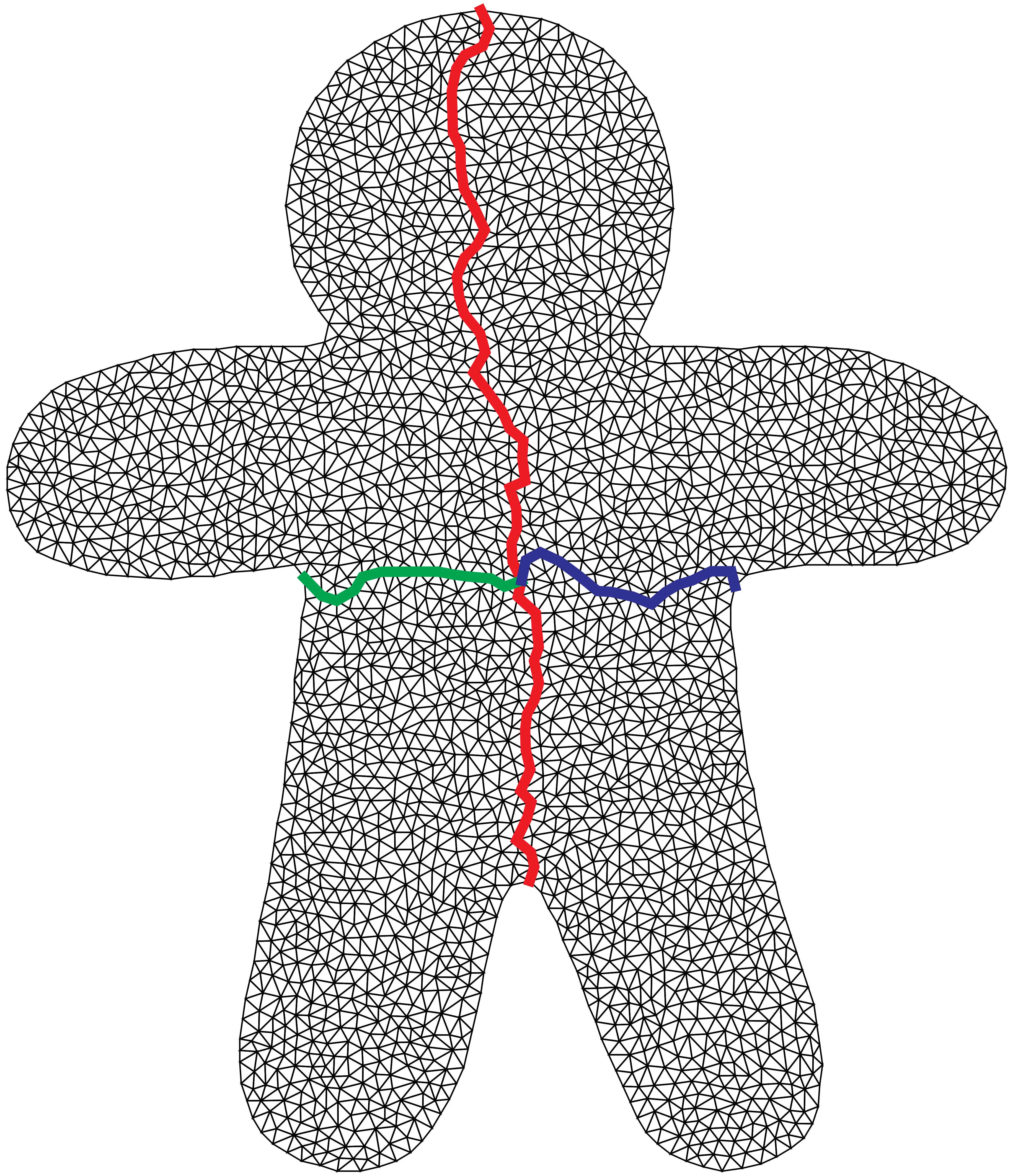


Ordered Laplacian

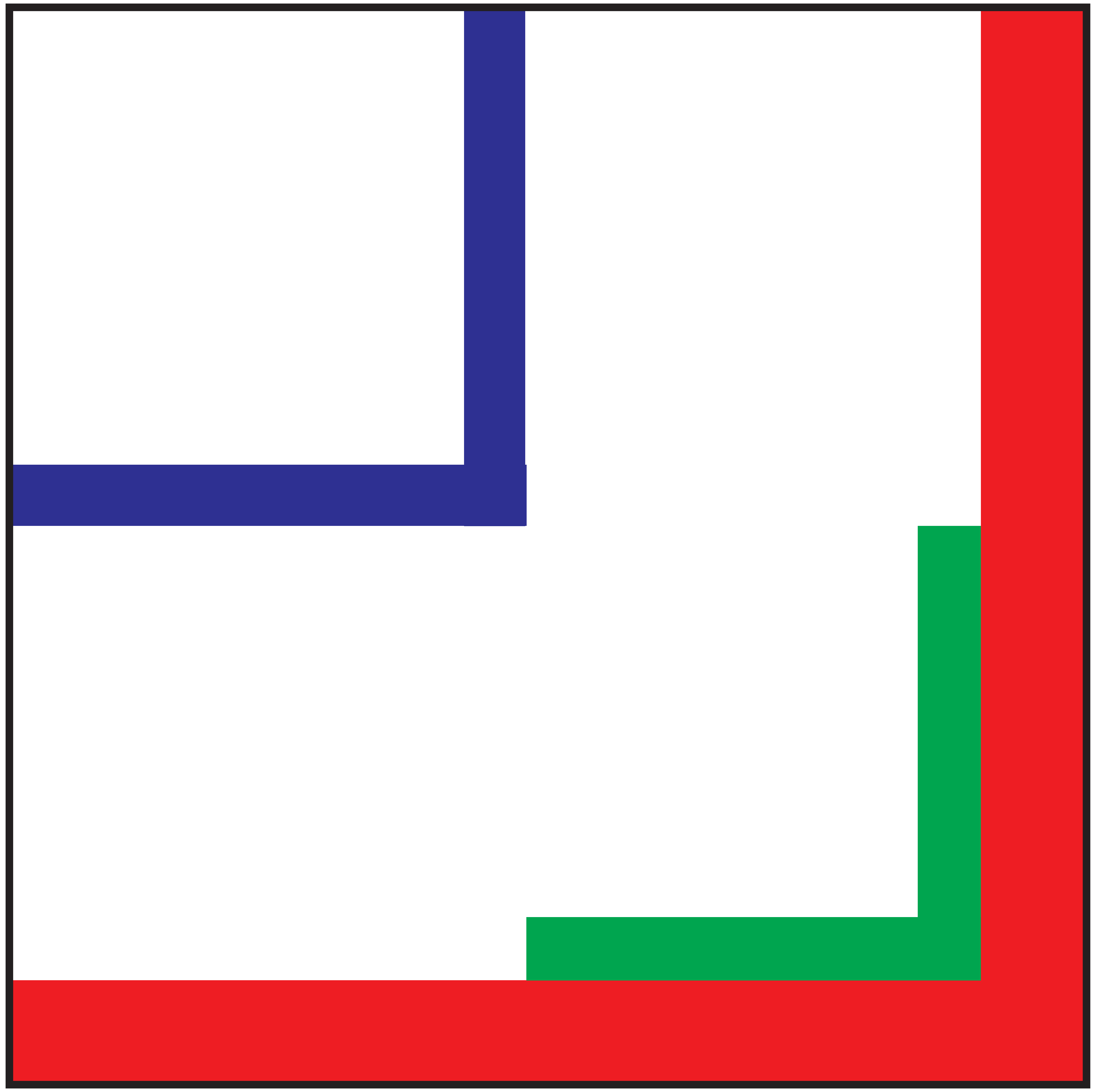


Nested dissection reordering

Input mesh

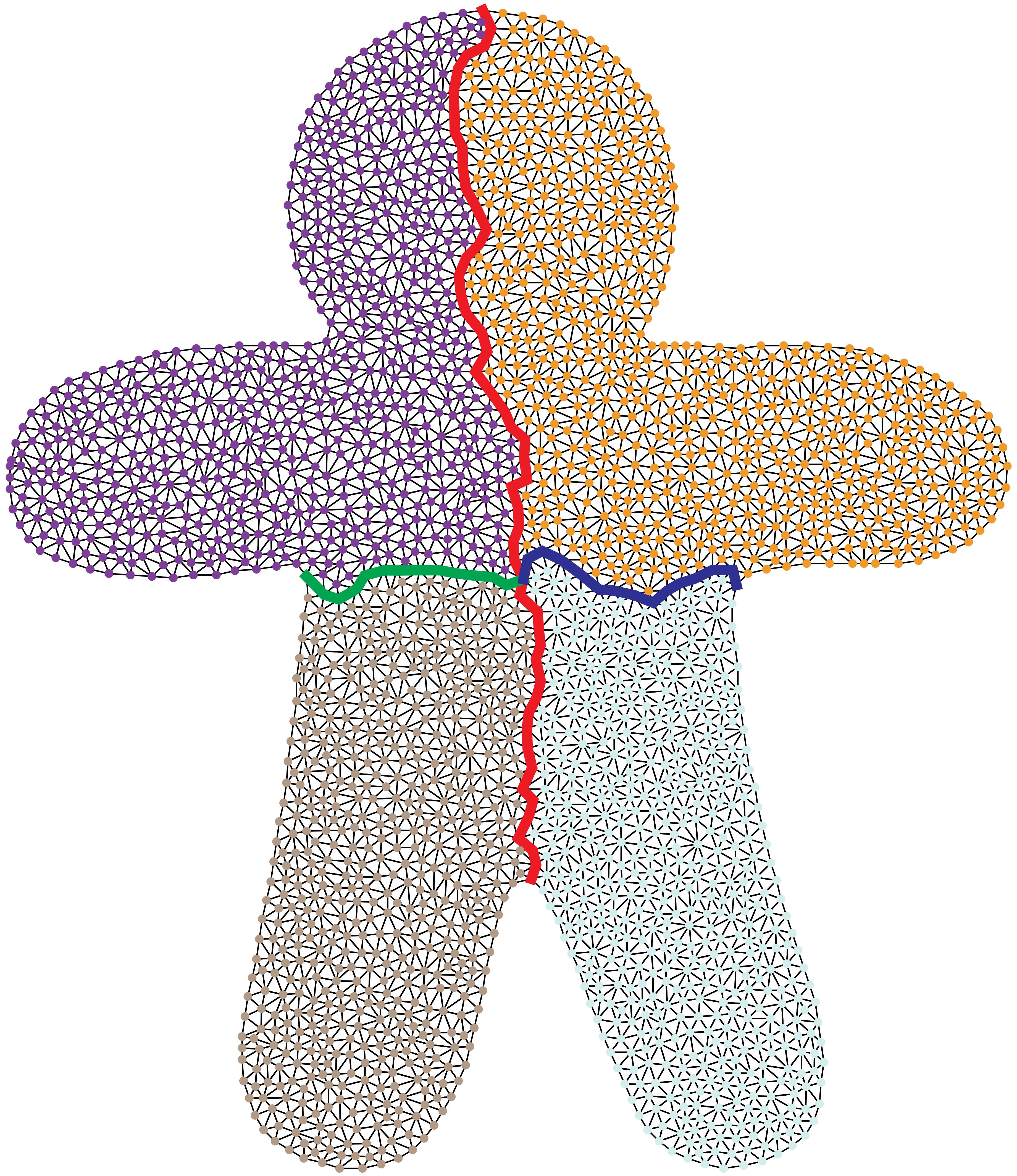


Ordered Laplacian

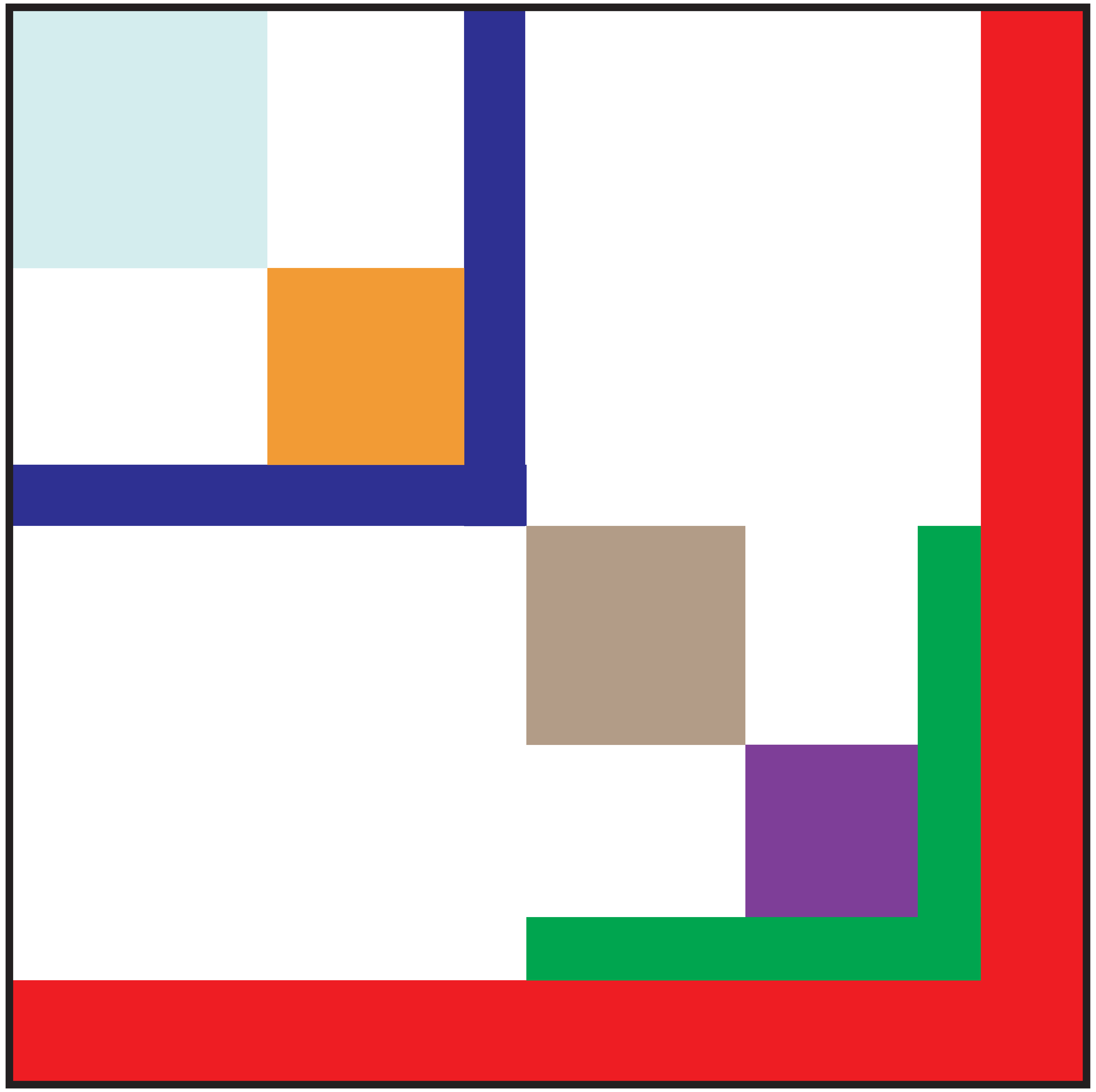


Nested dissection reordering

Input mesh

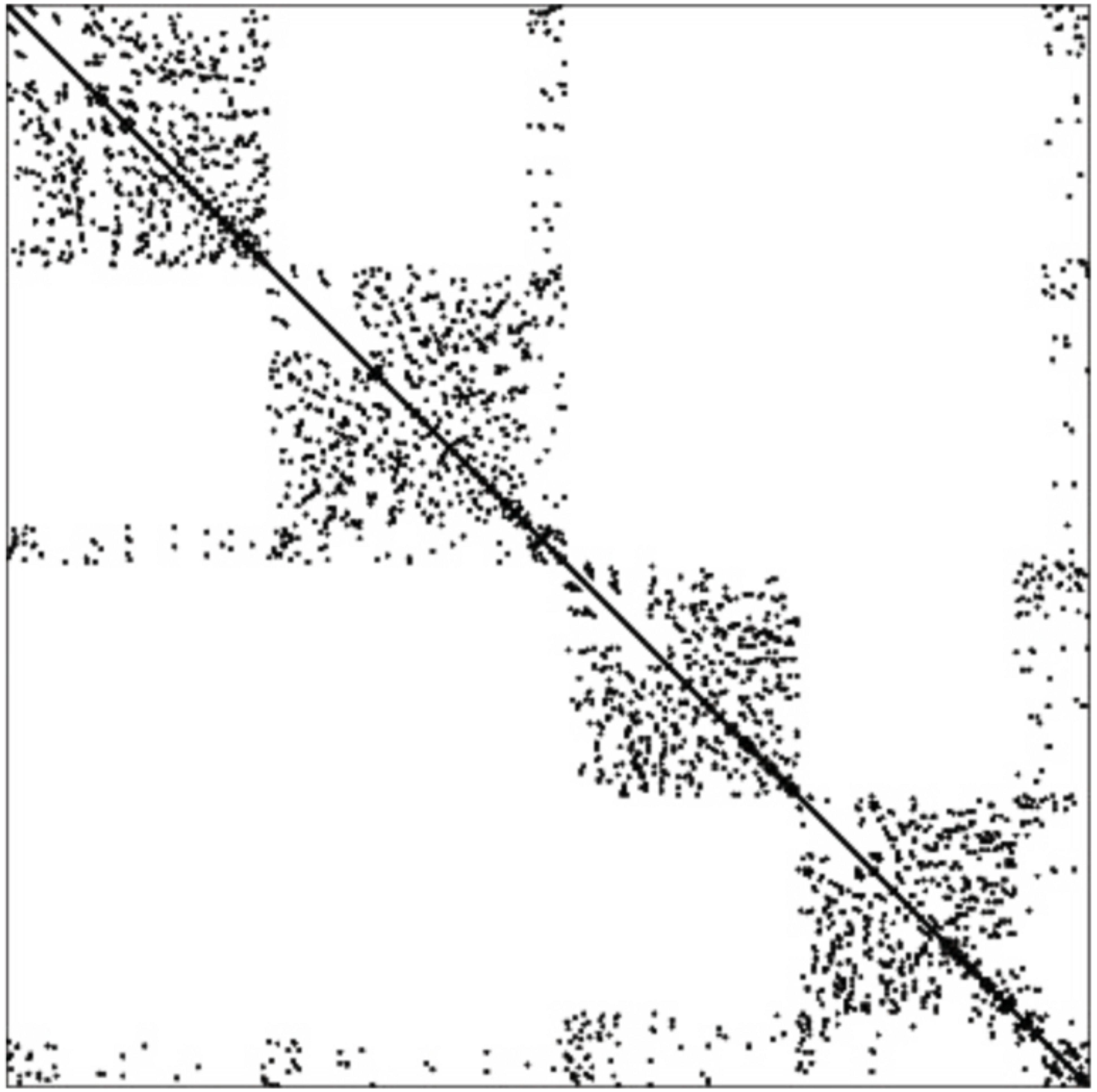


Ordered Laplacian

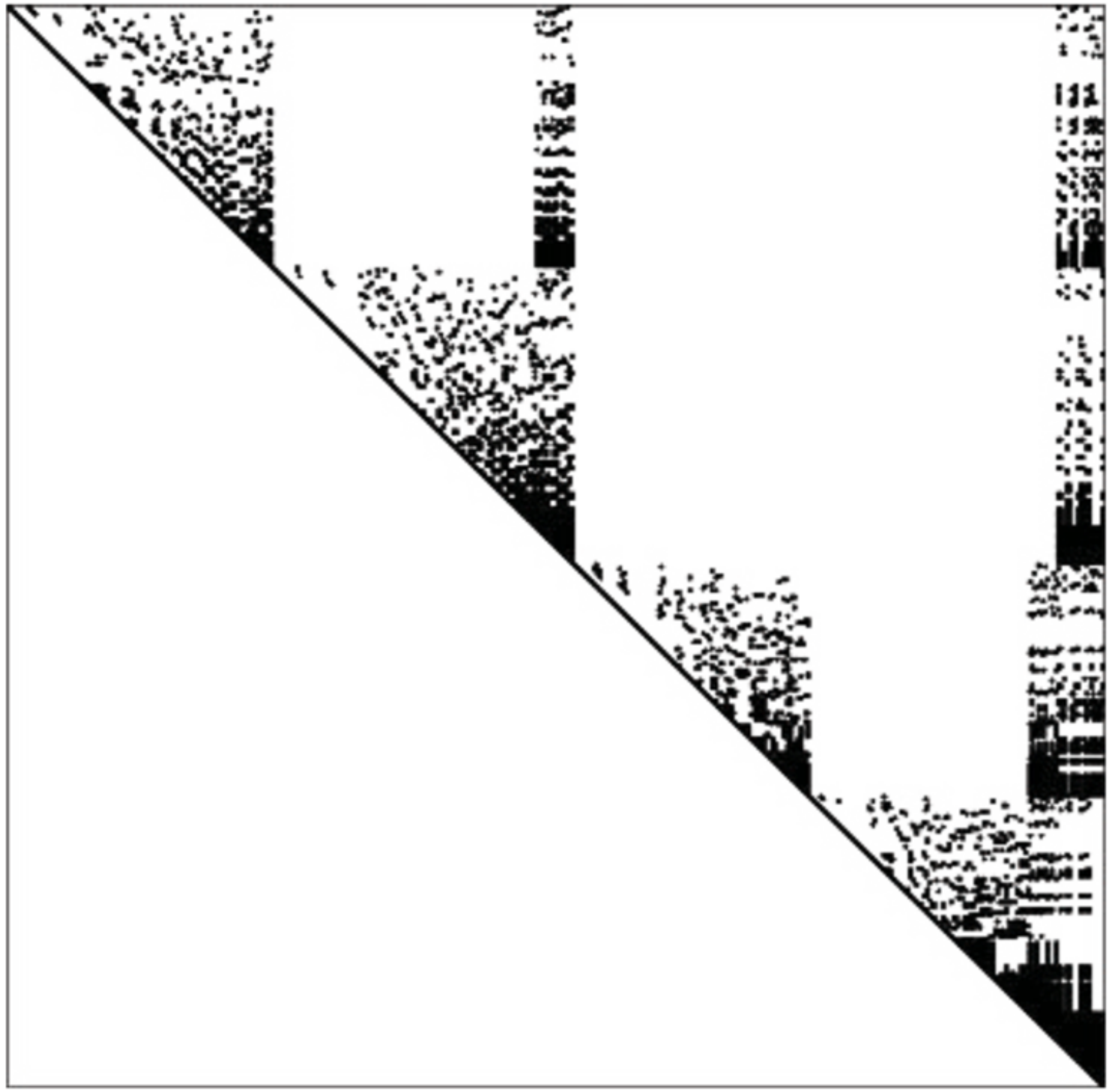


Nested dissection reordering

Input matrix

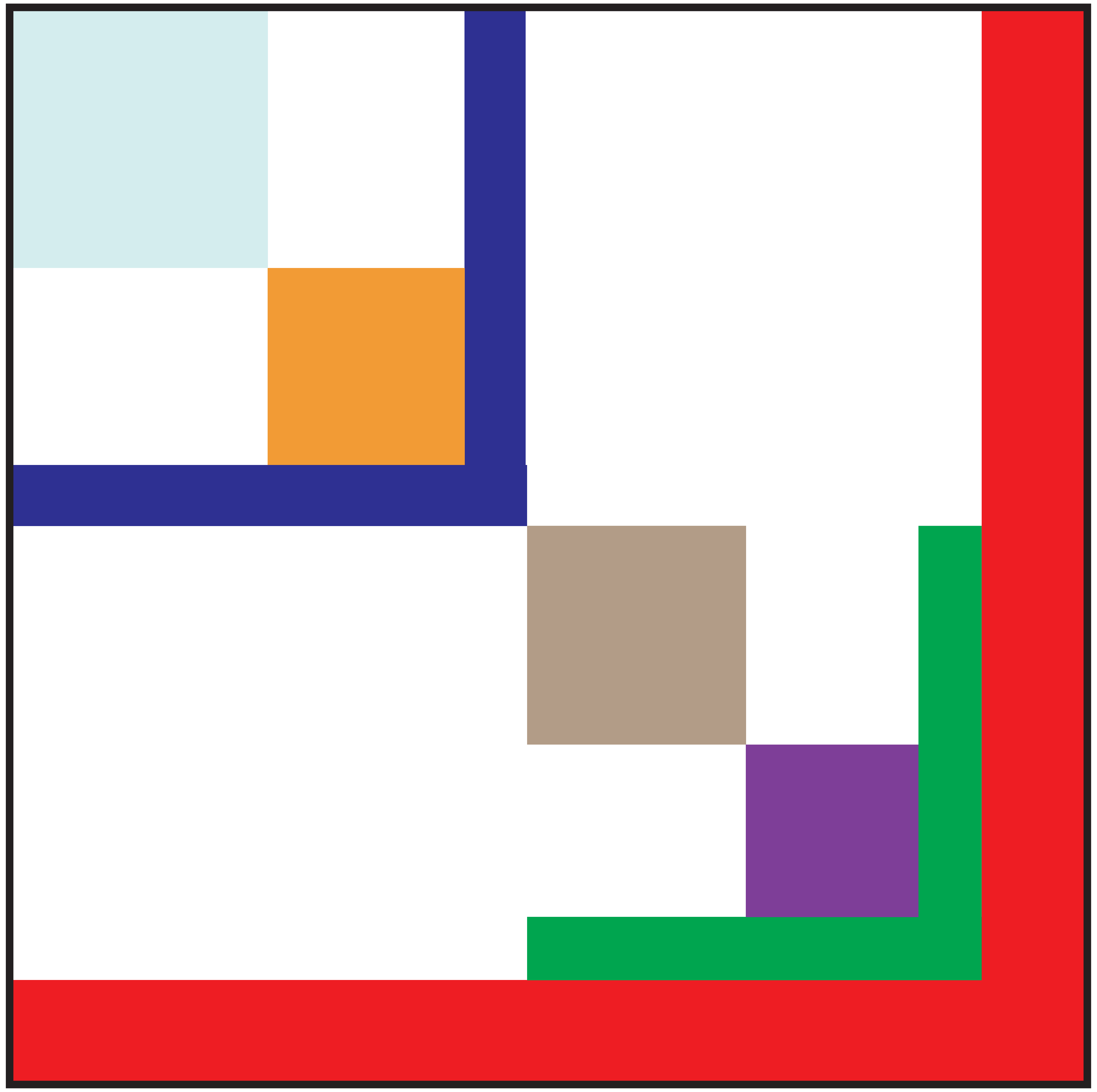


Cholesky factor $\mathbf{L}^T$

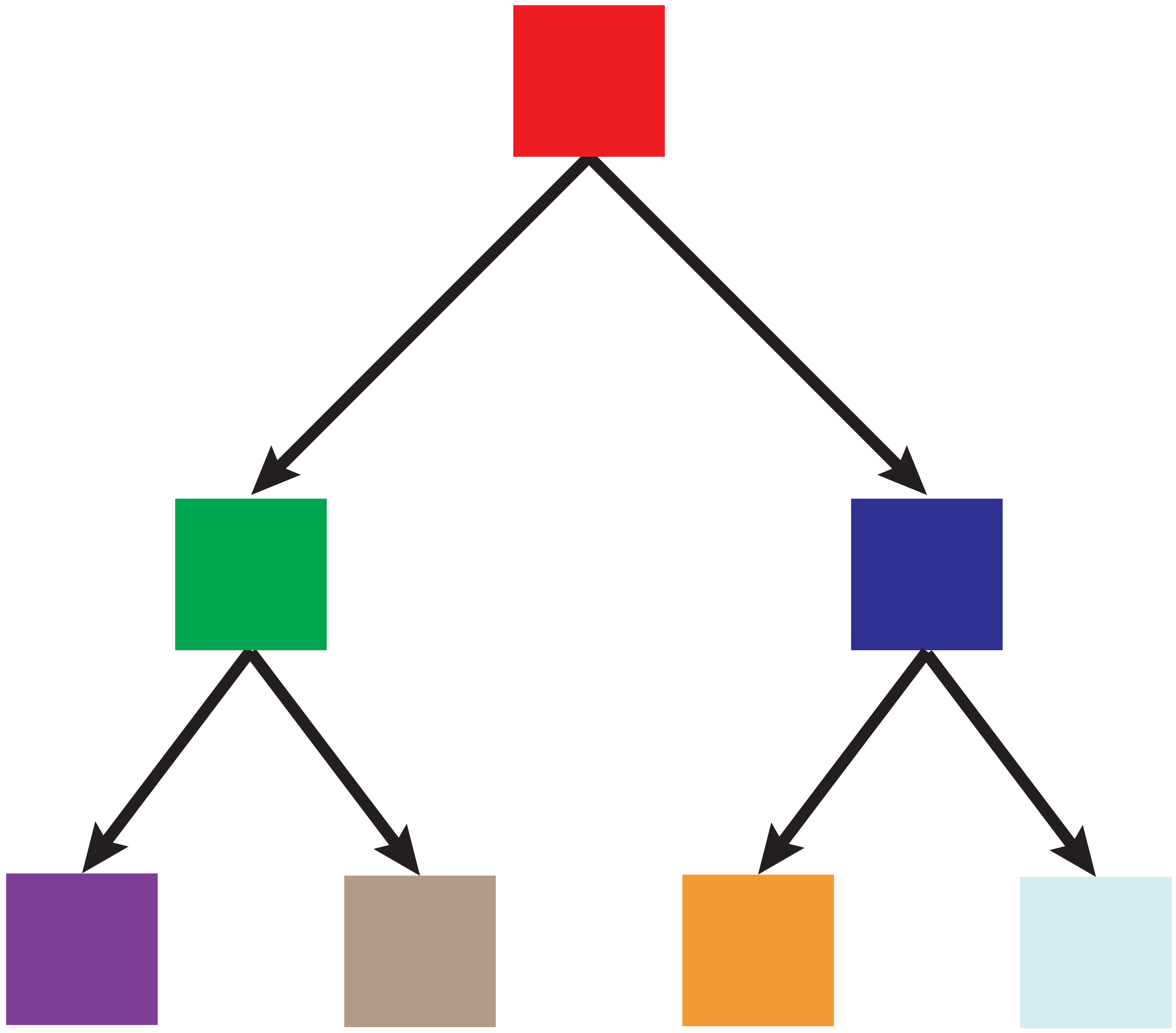


Nested dissection reordering

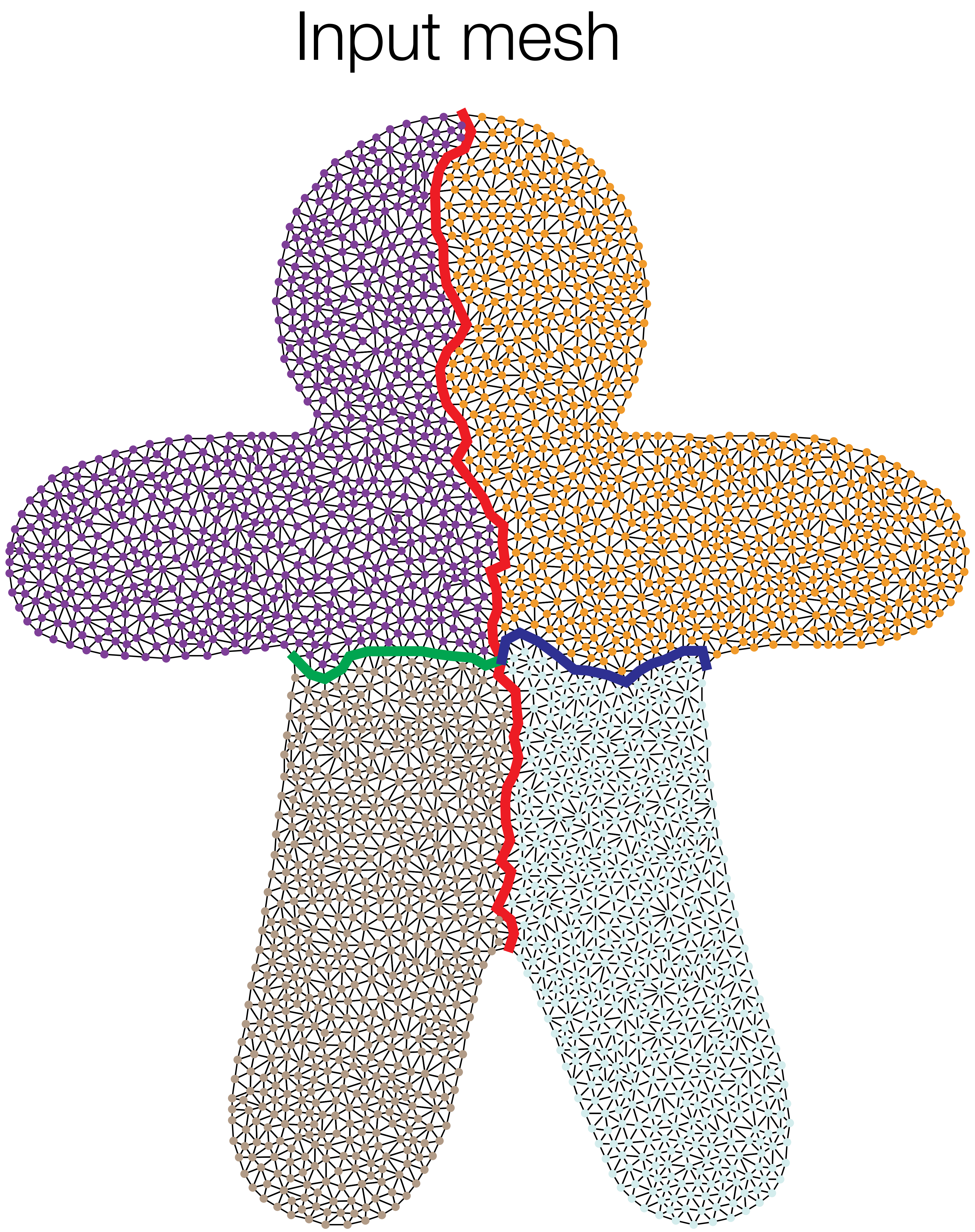
Cholesky factor

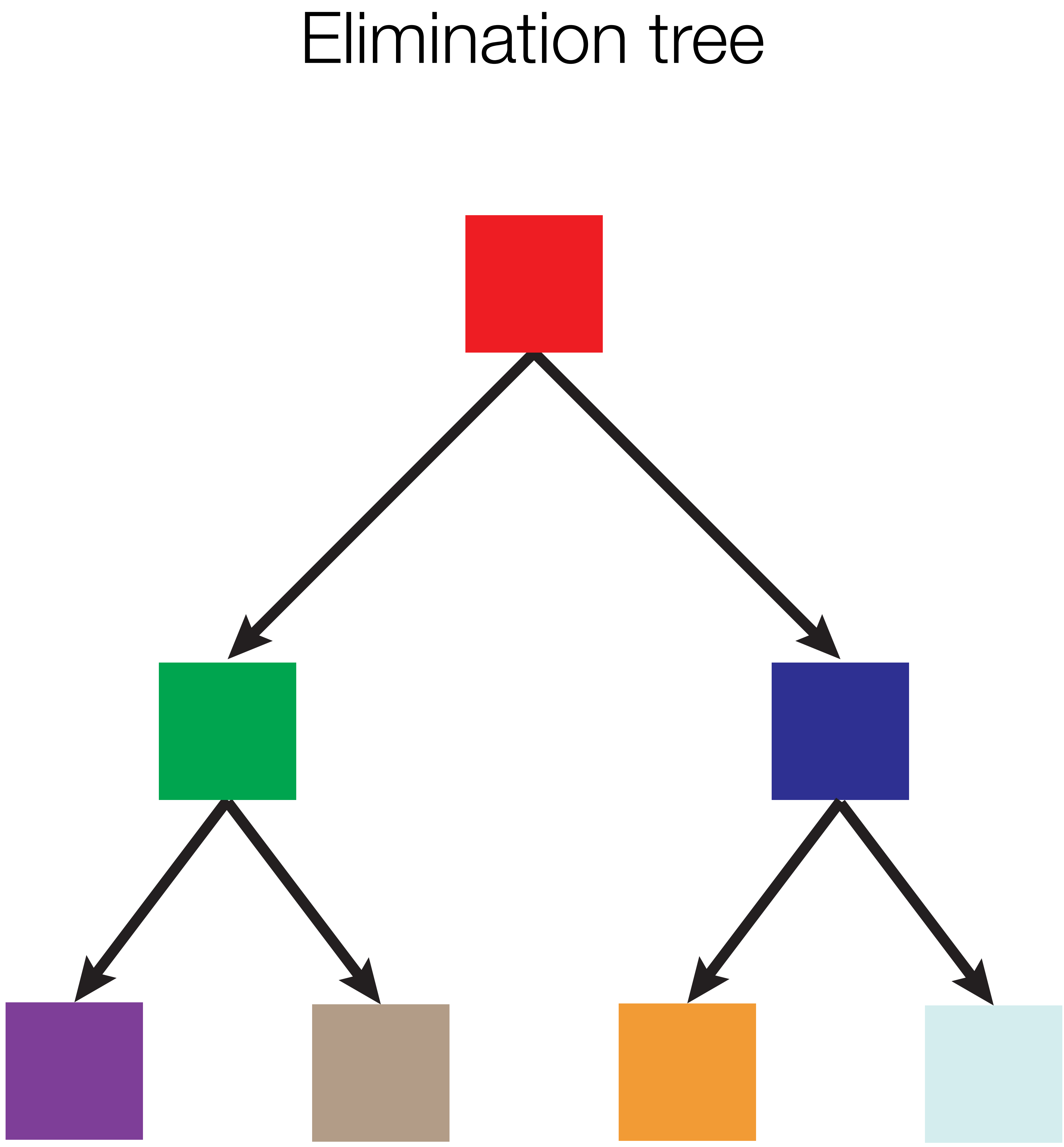


Elimination tree



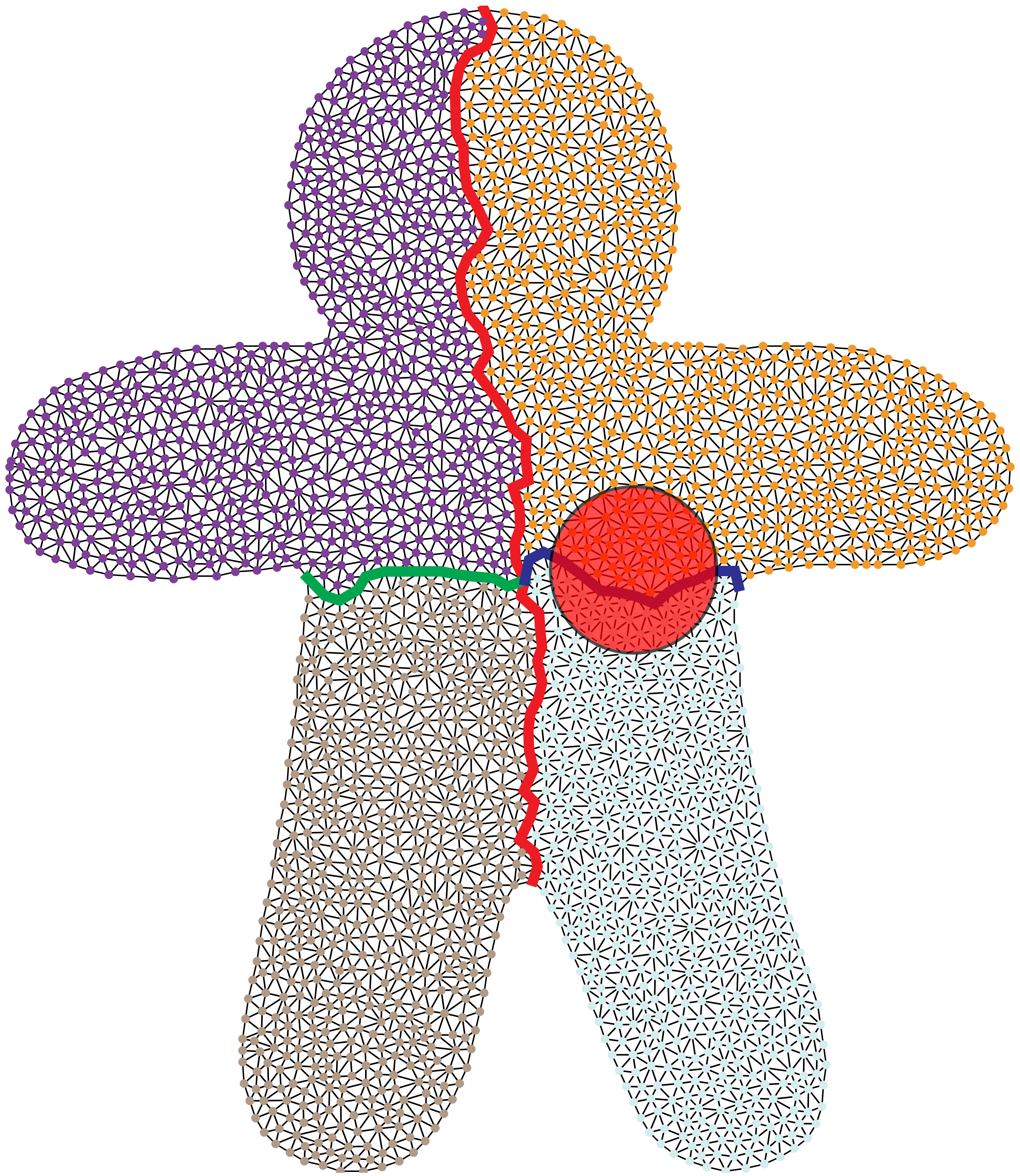
Nested dissection reordering



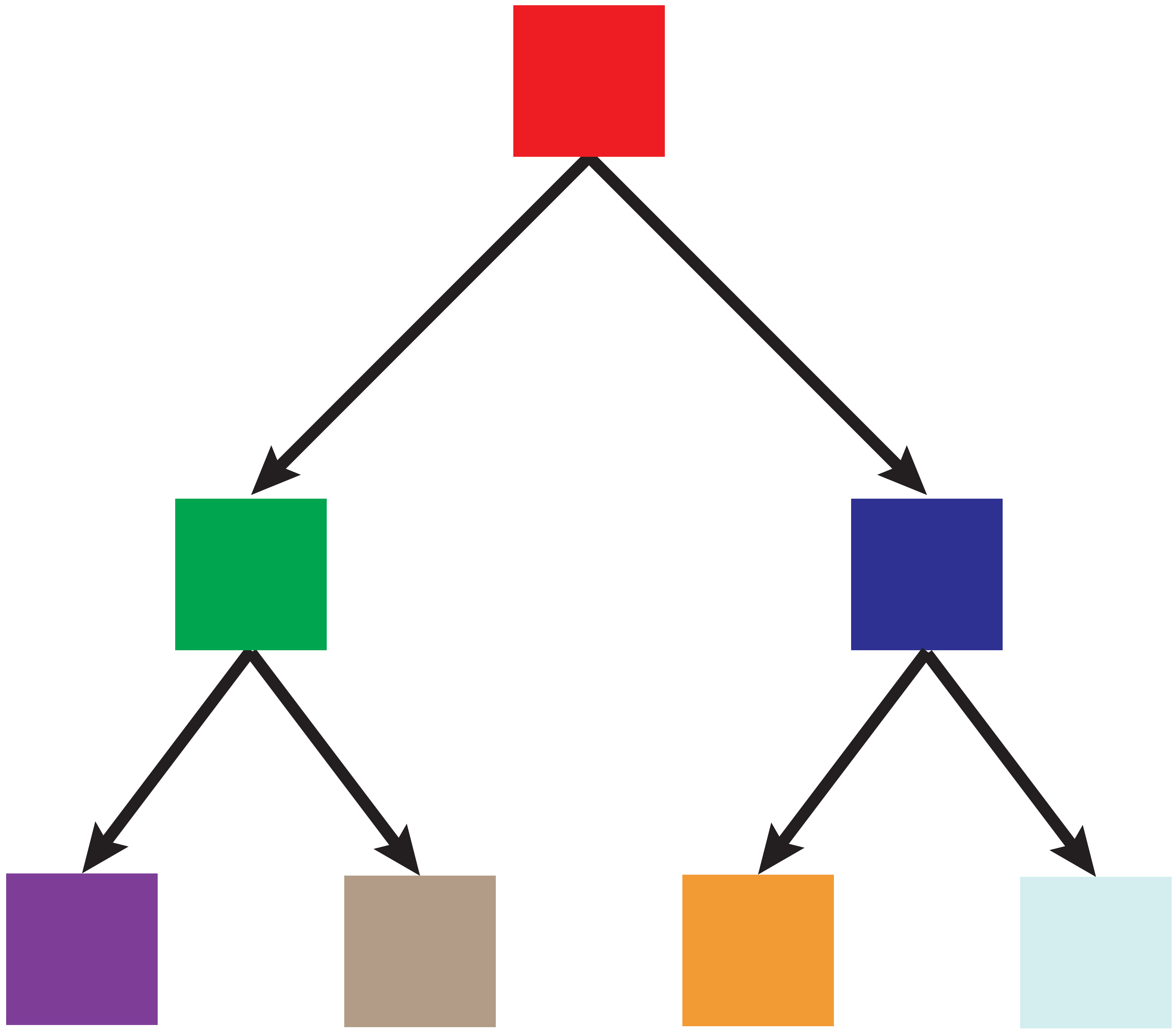


Nested dissection reordering

Input mesh

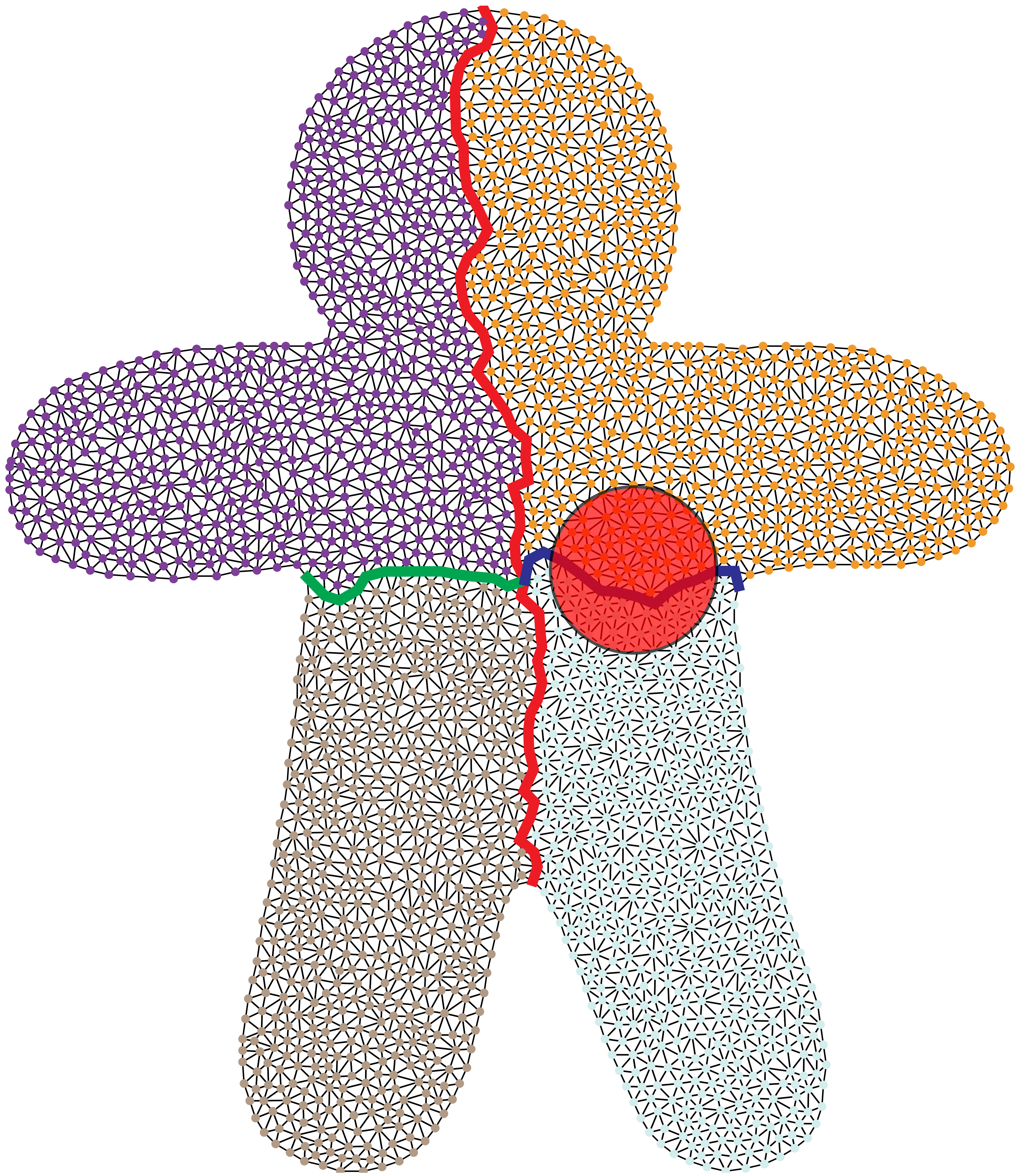


Elimination tree

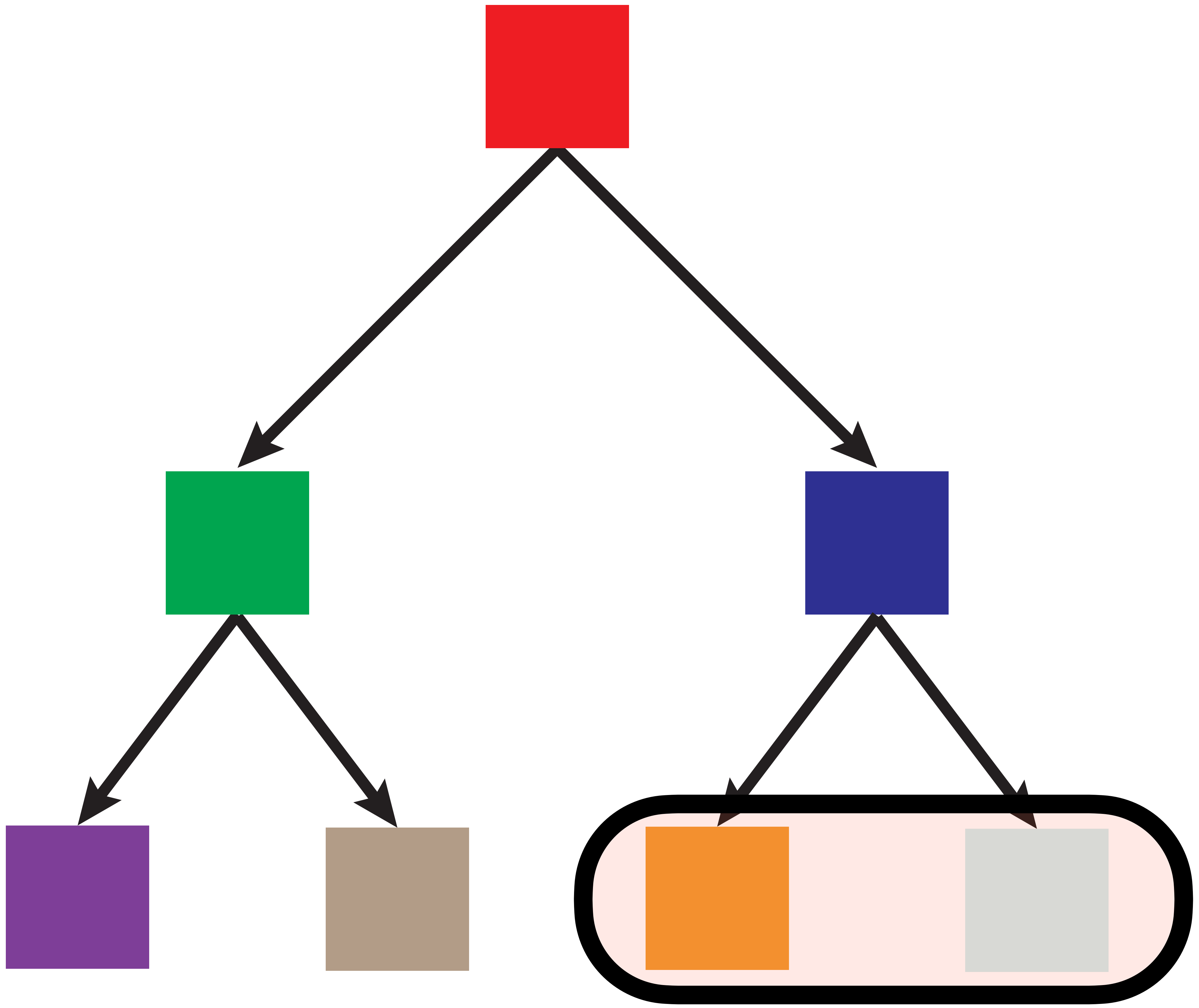


Nested dissection reordering

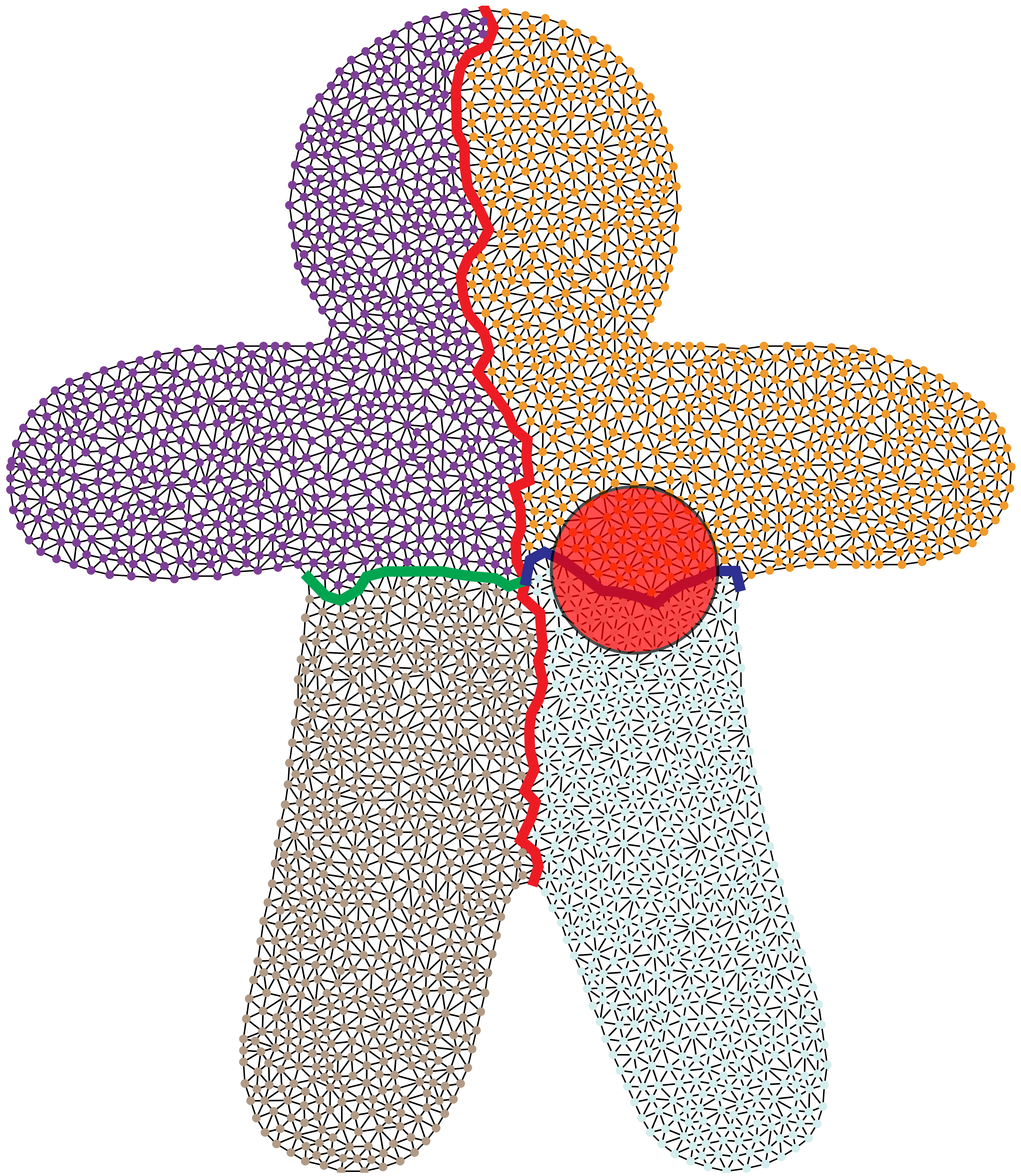
Input mesh

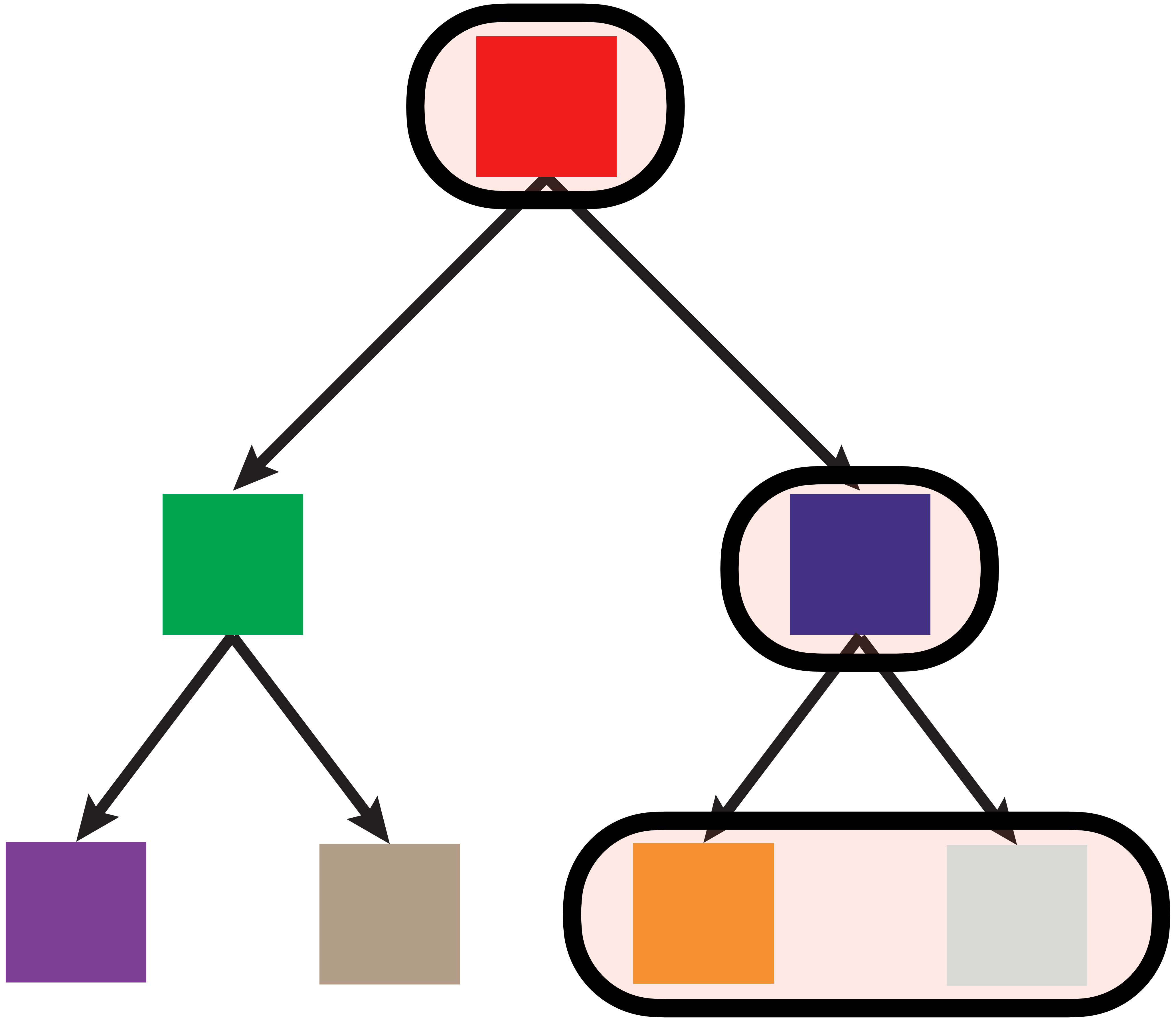


Elimination tree



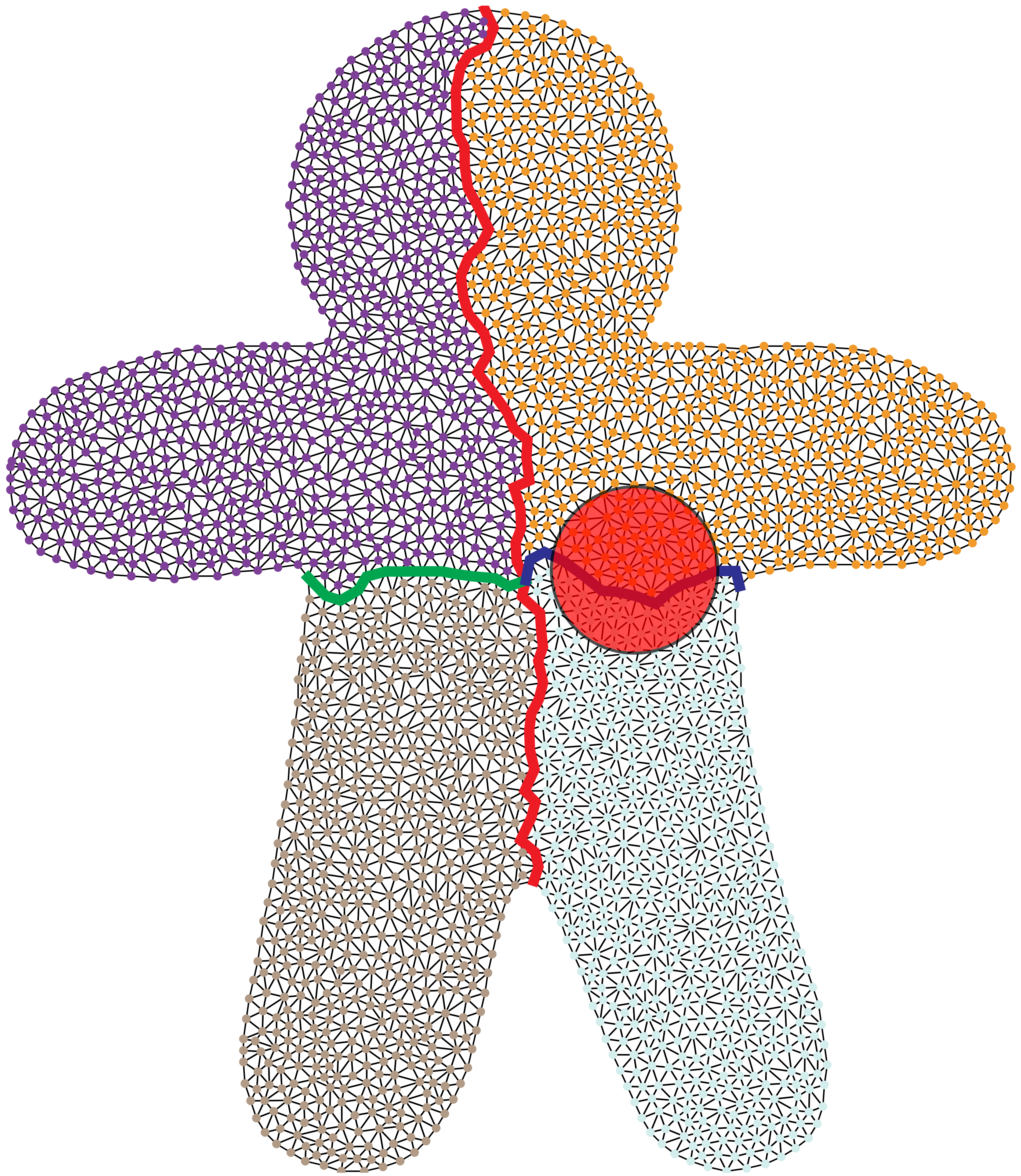
Nested dissection reordering



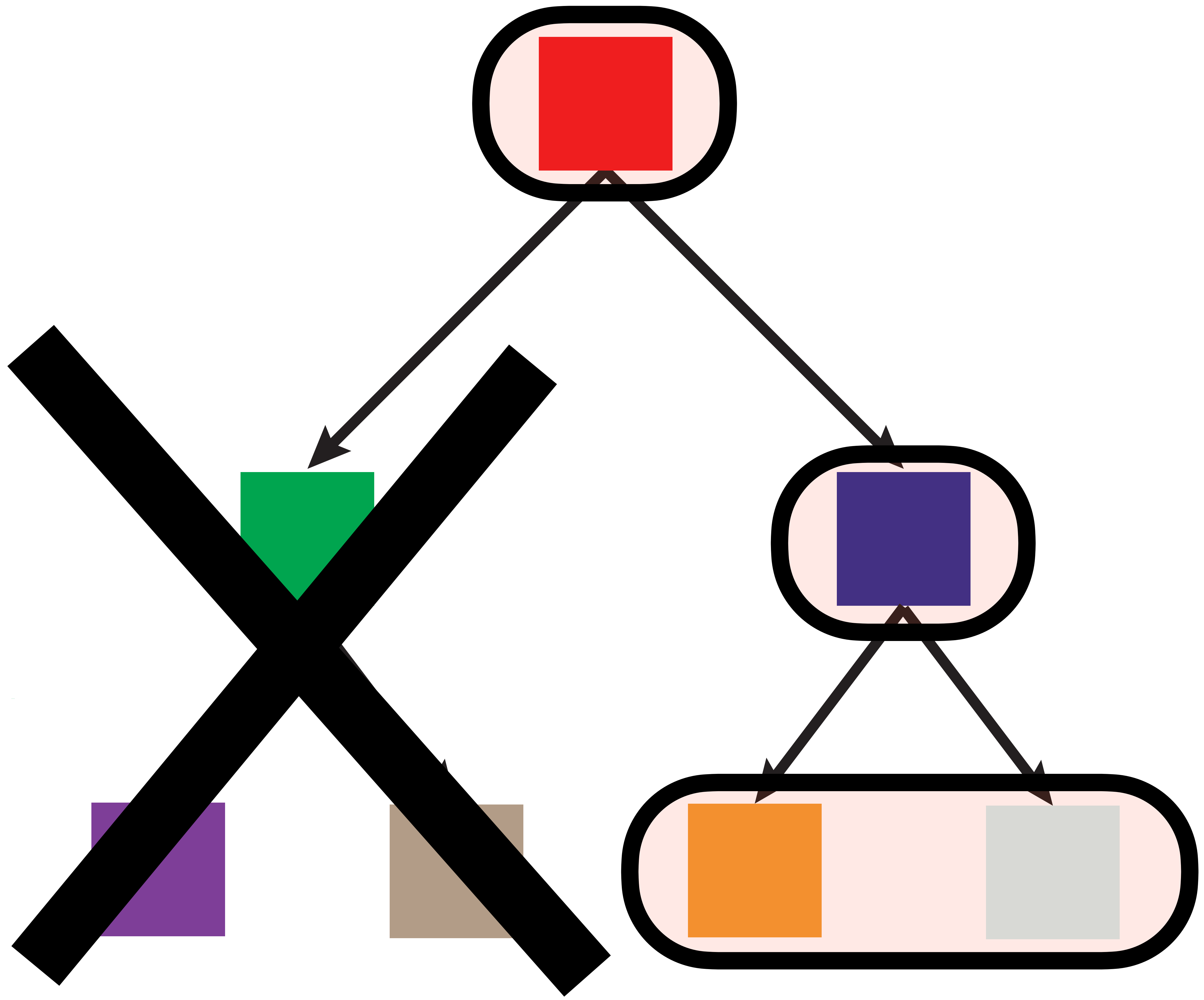


Nested dissection reordering

Input mesh

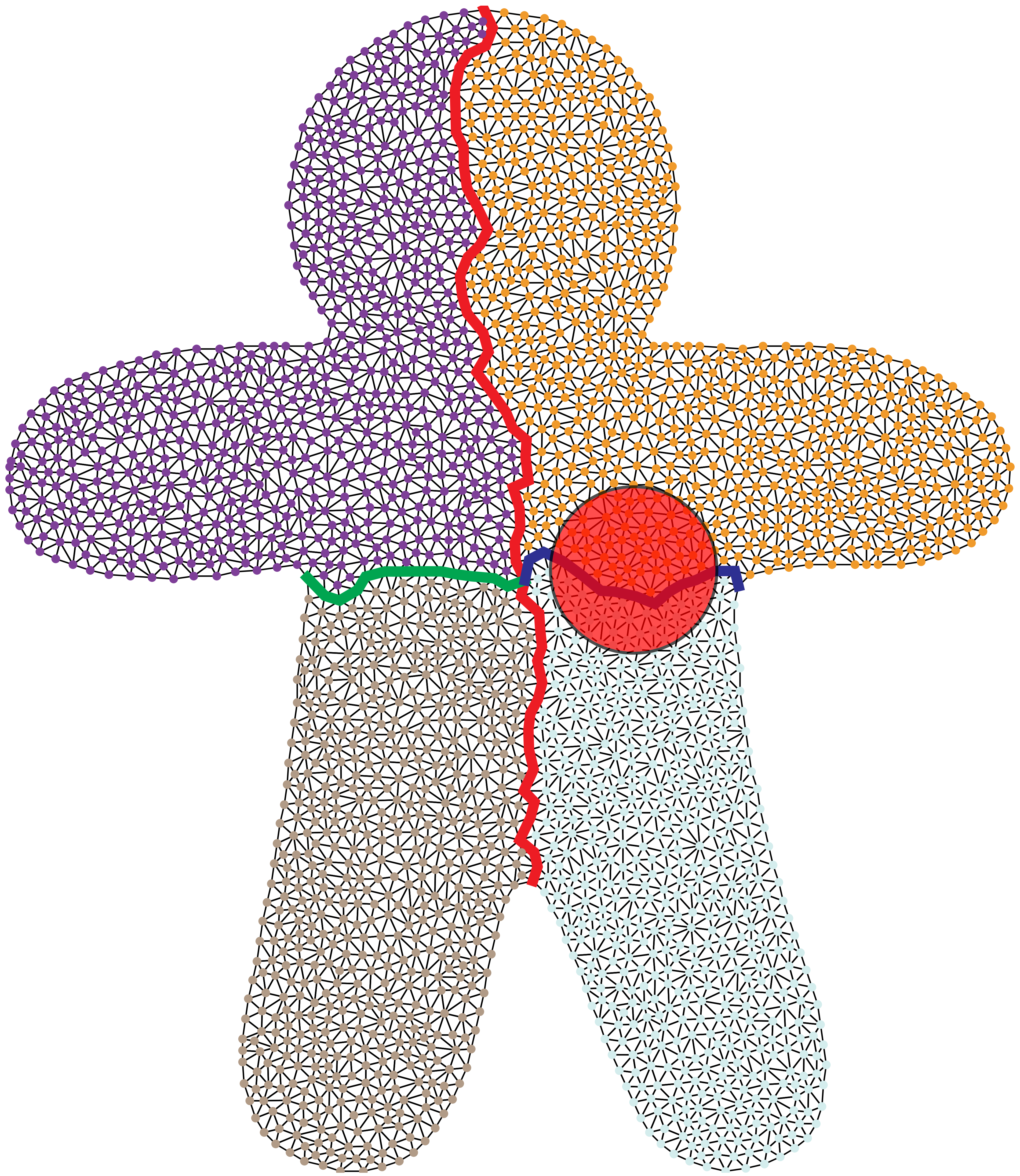


Elimination tree

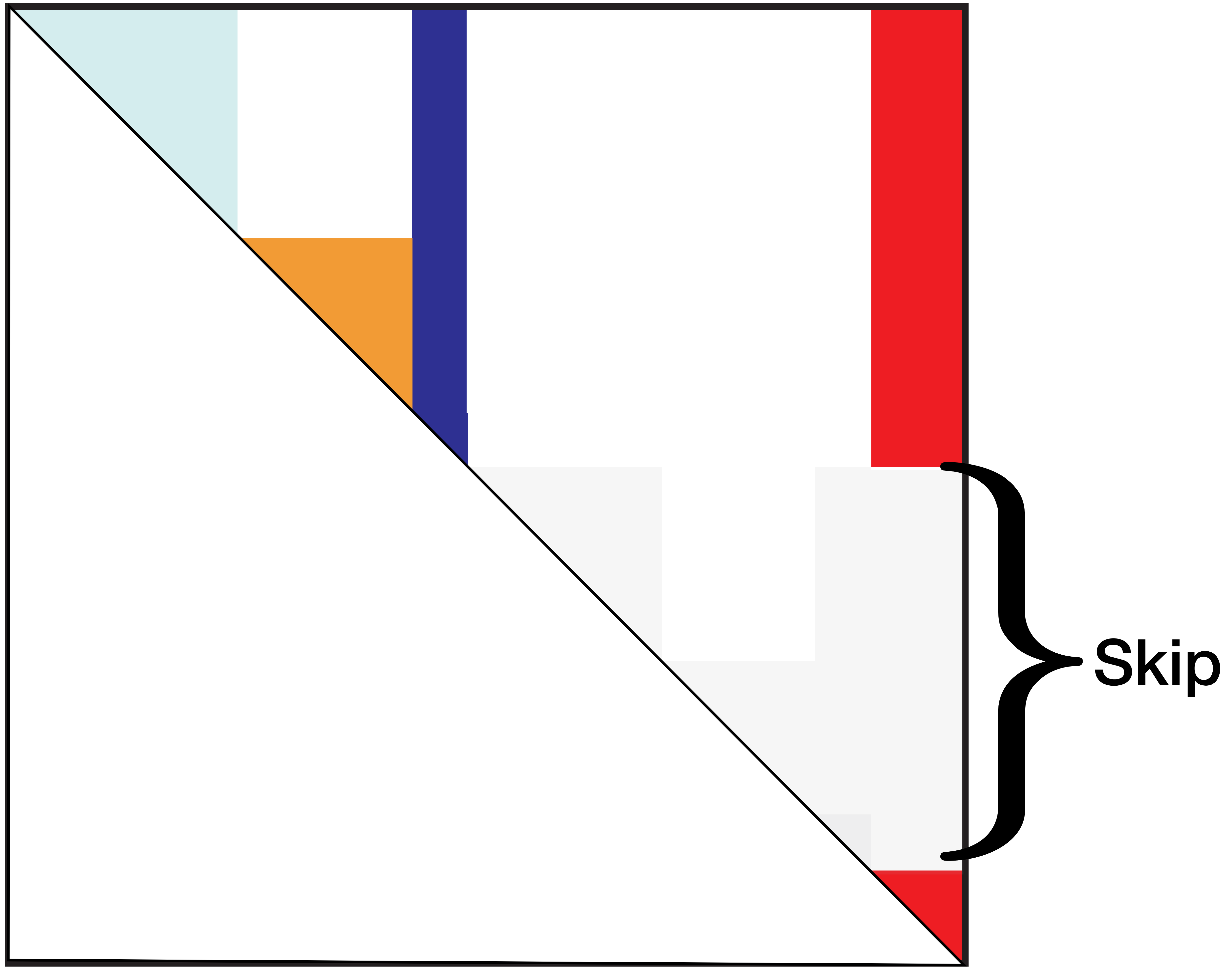


Nested dissection reordering

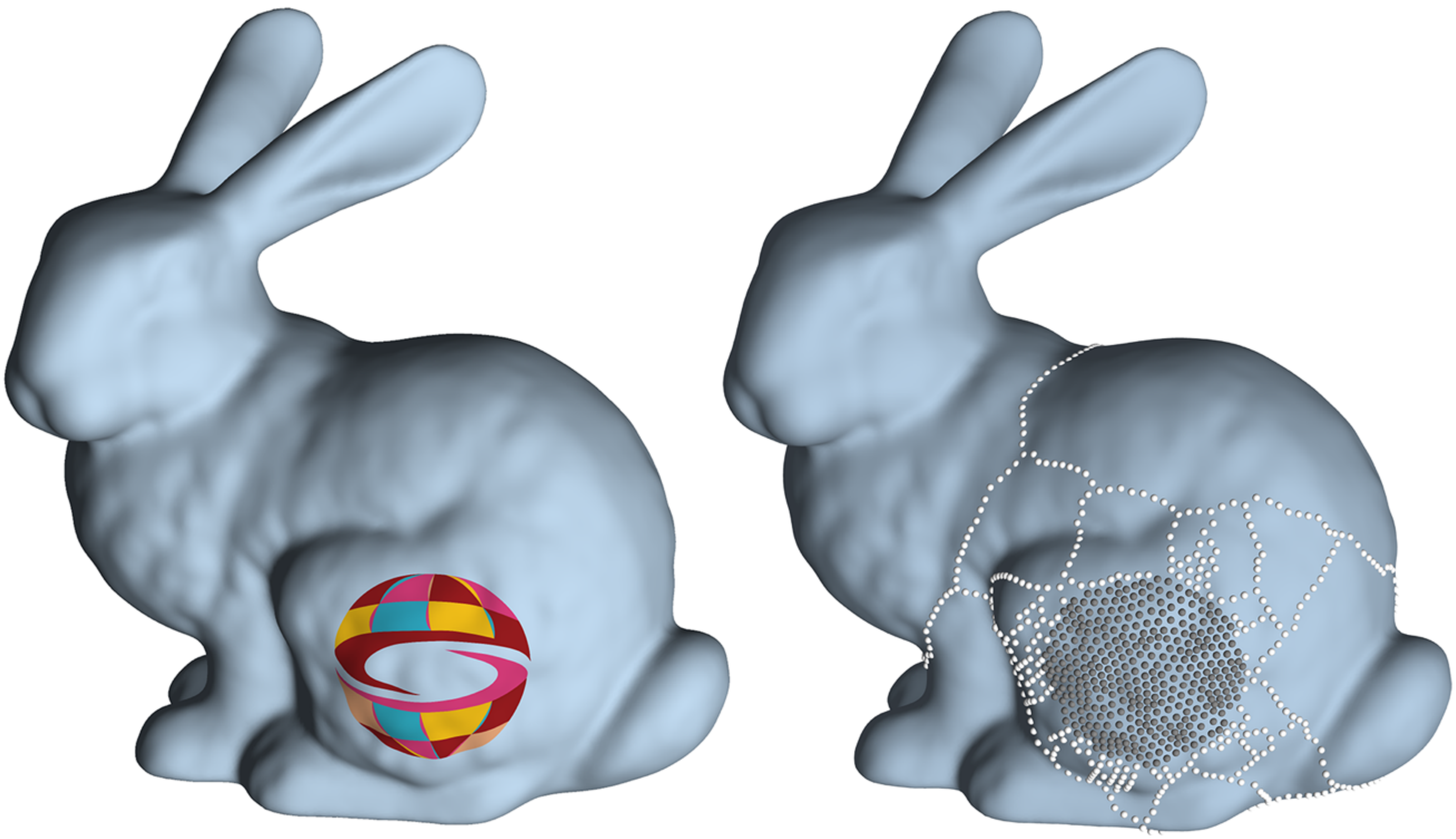
Input mesh



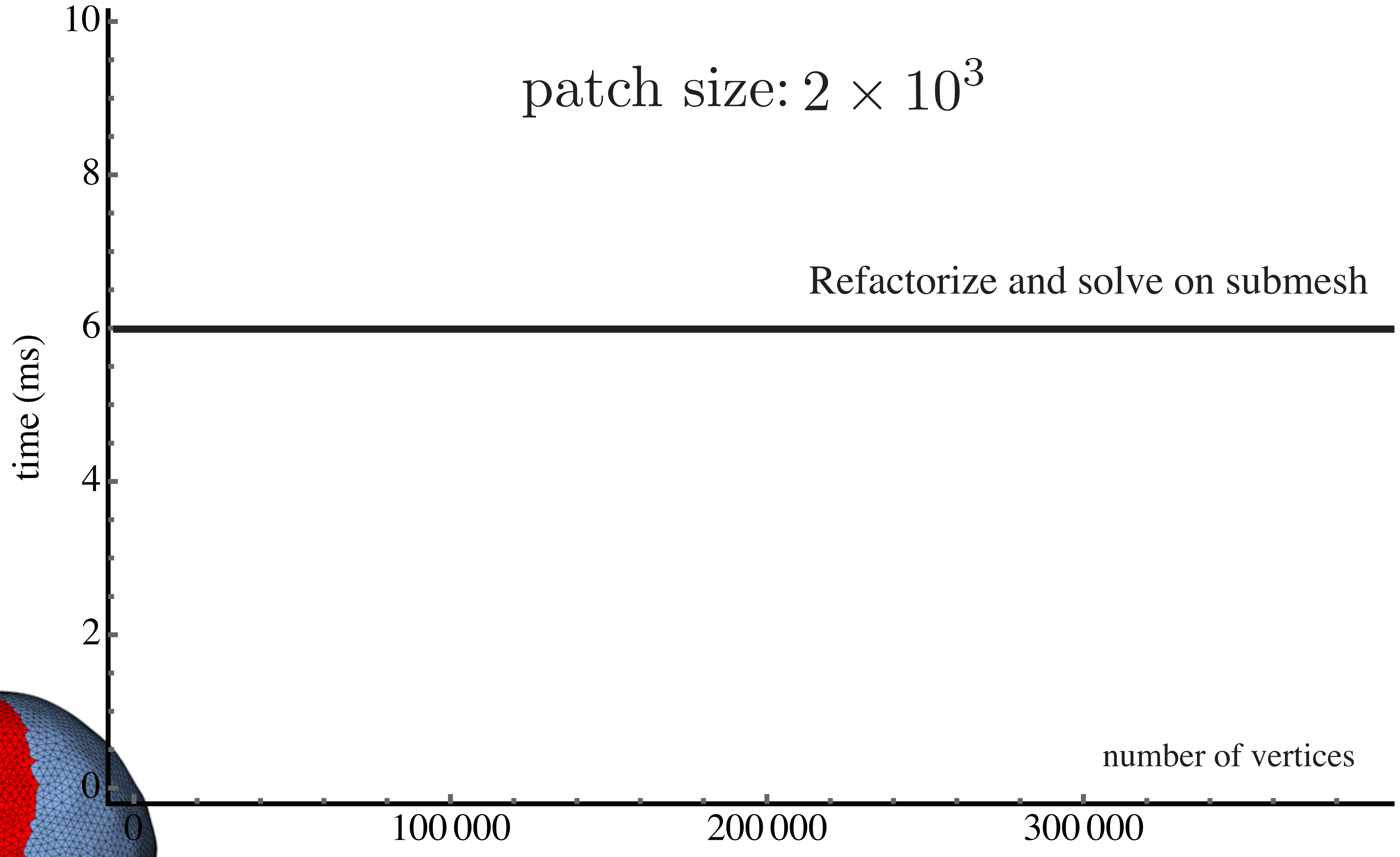
Cholesky factor

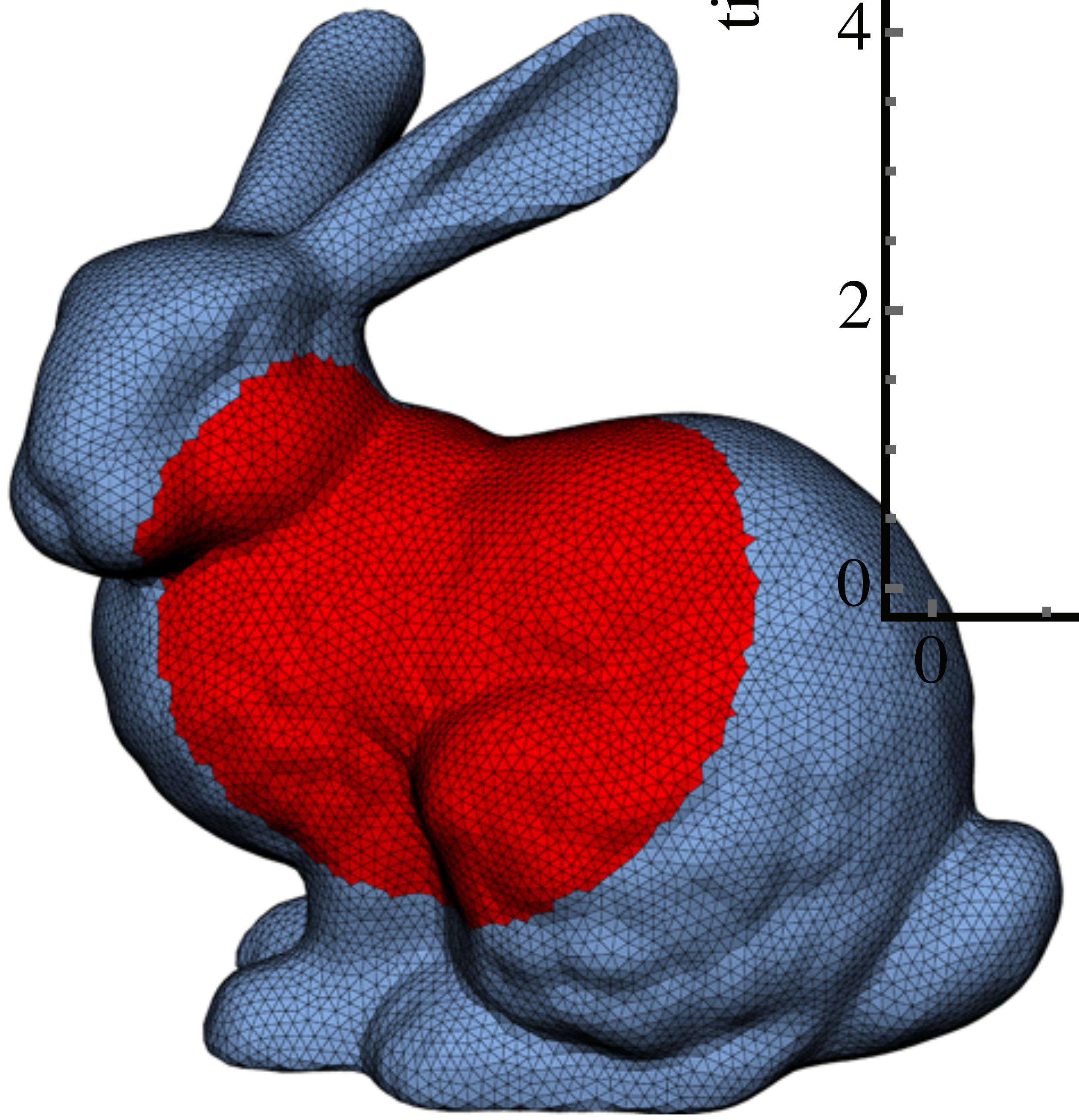


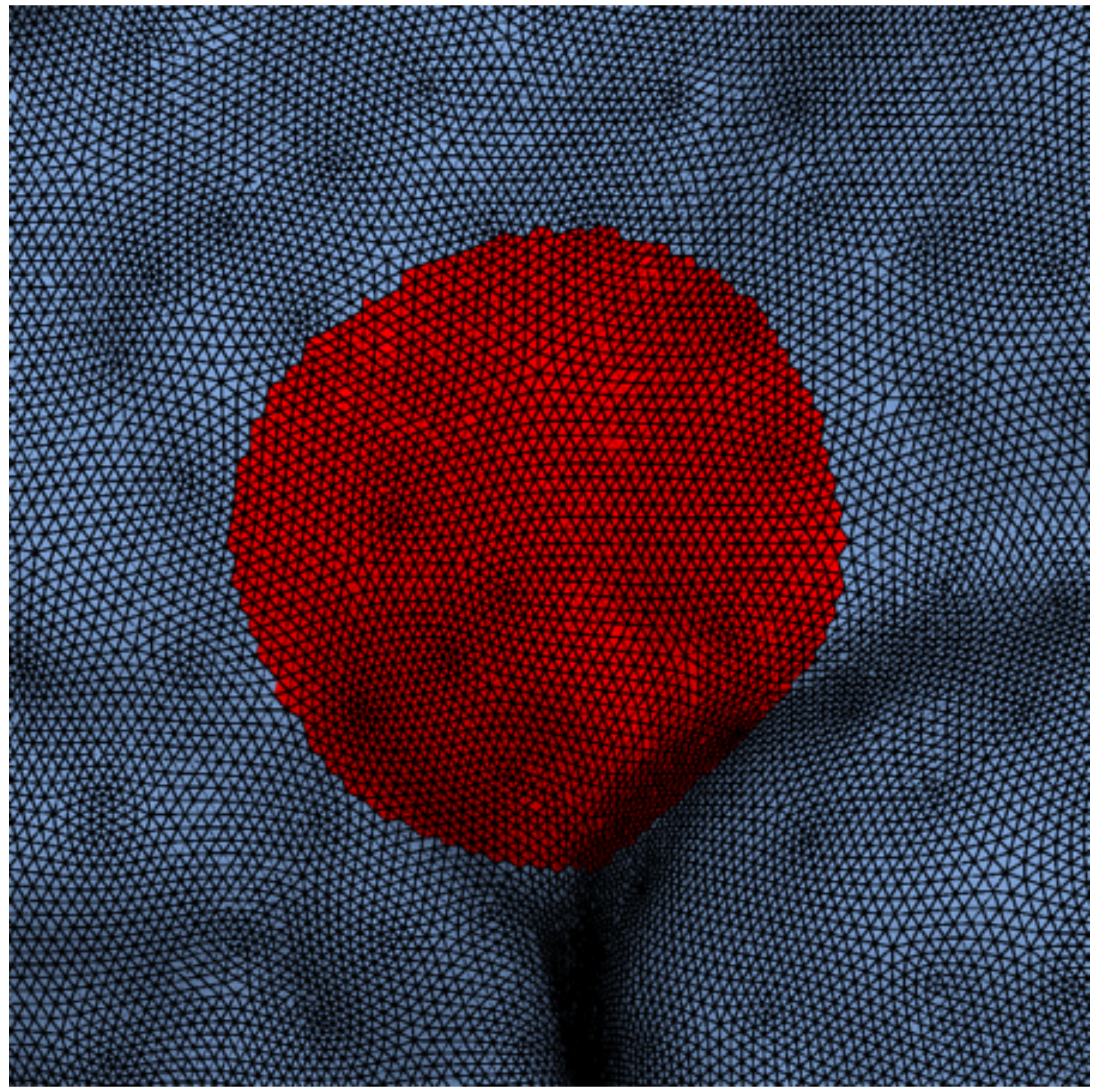
Applications: Parameterization



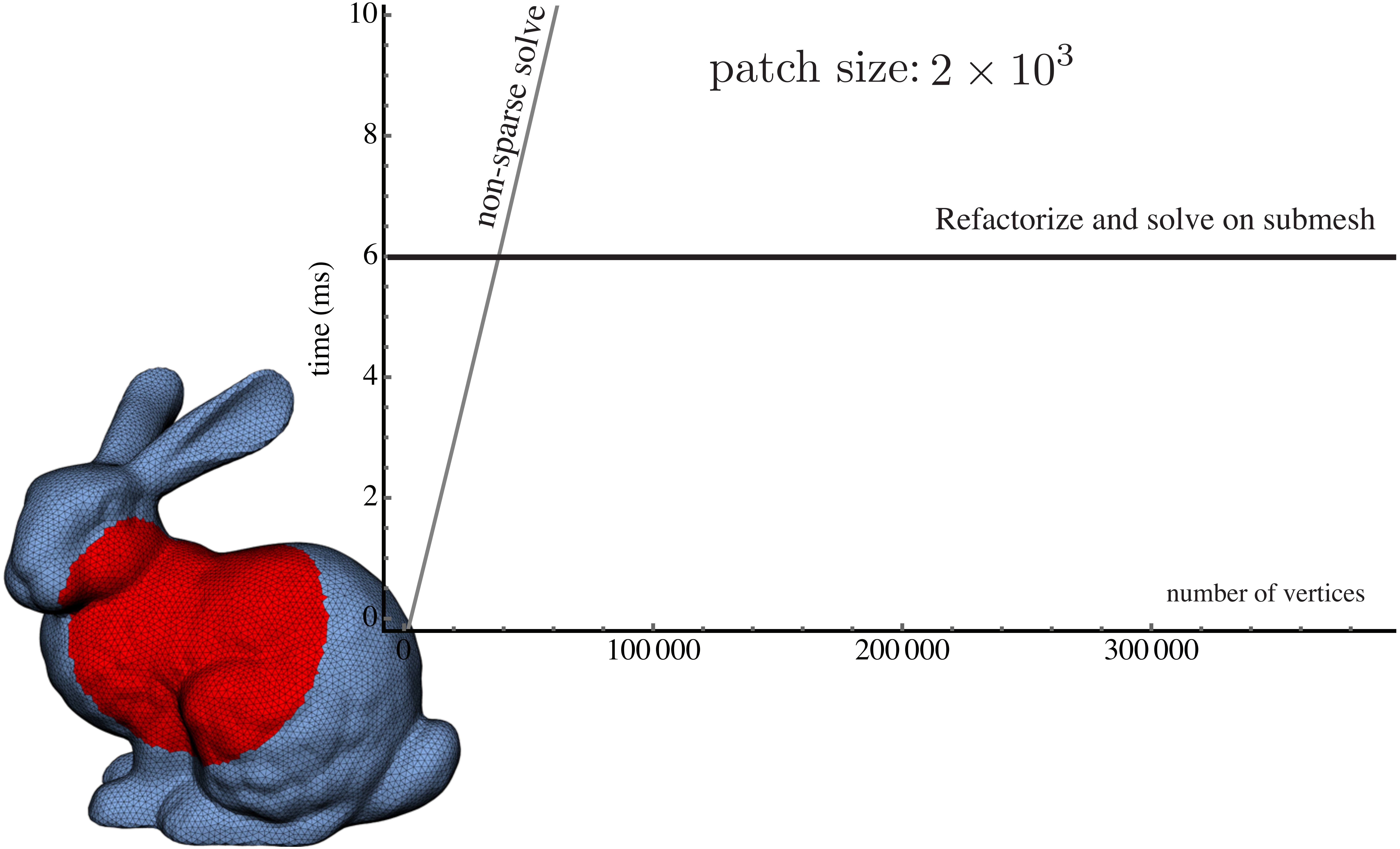
Is it worth it?

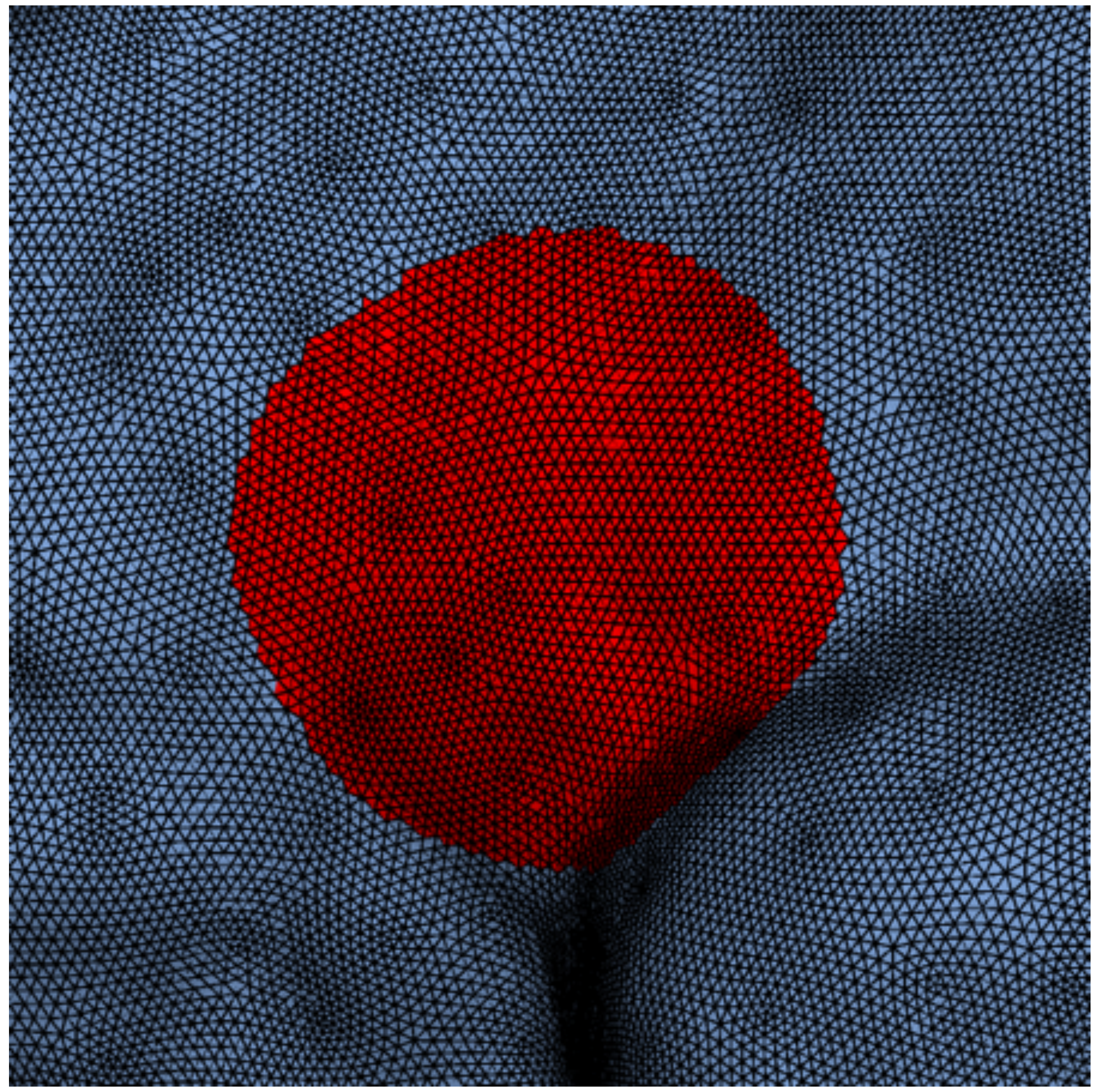




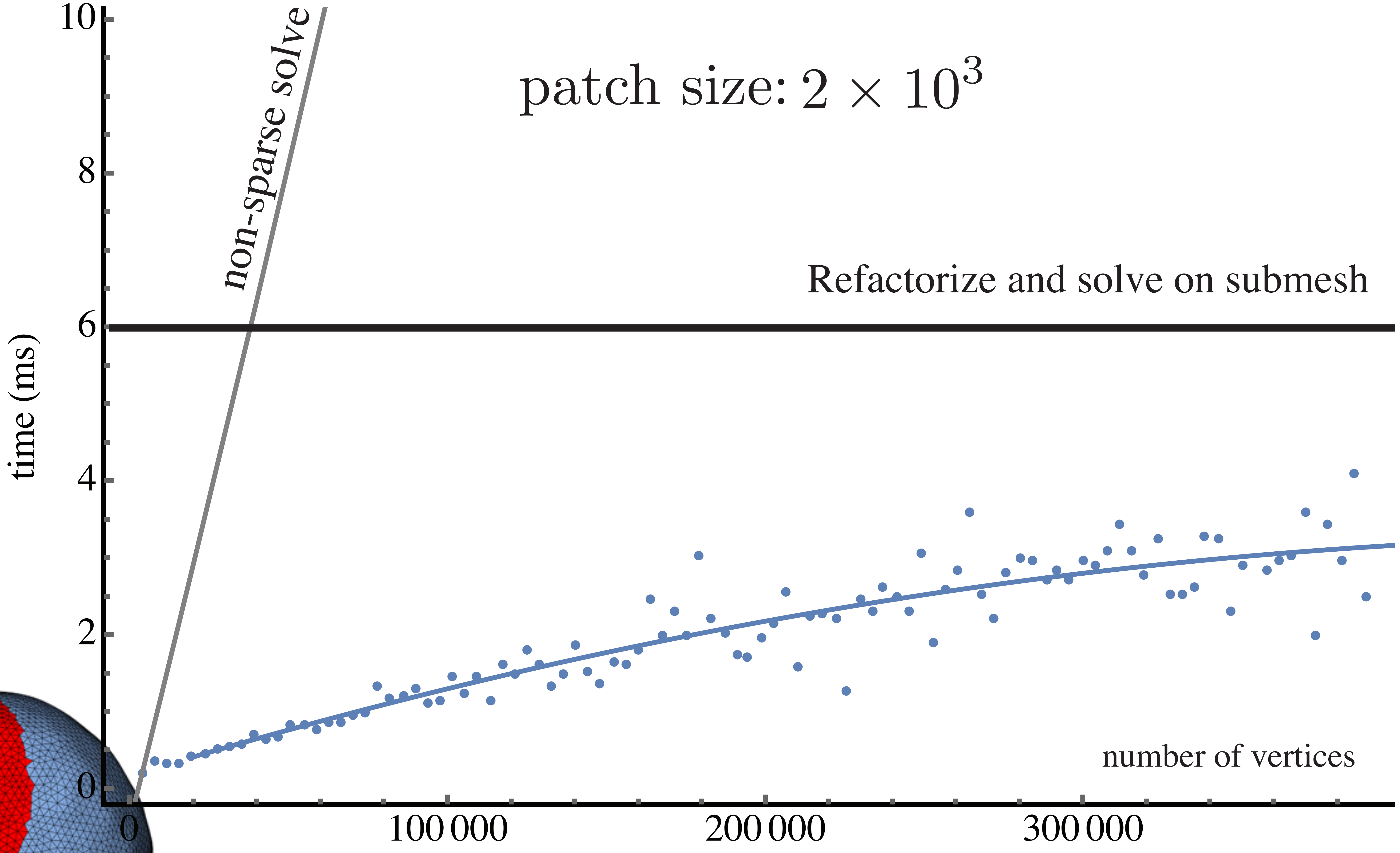


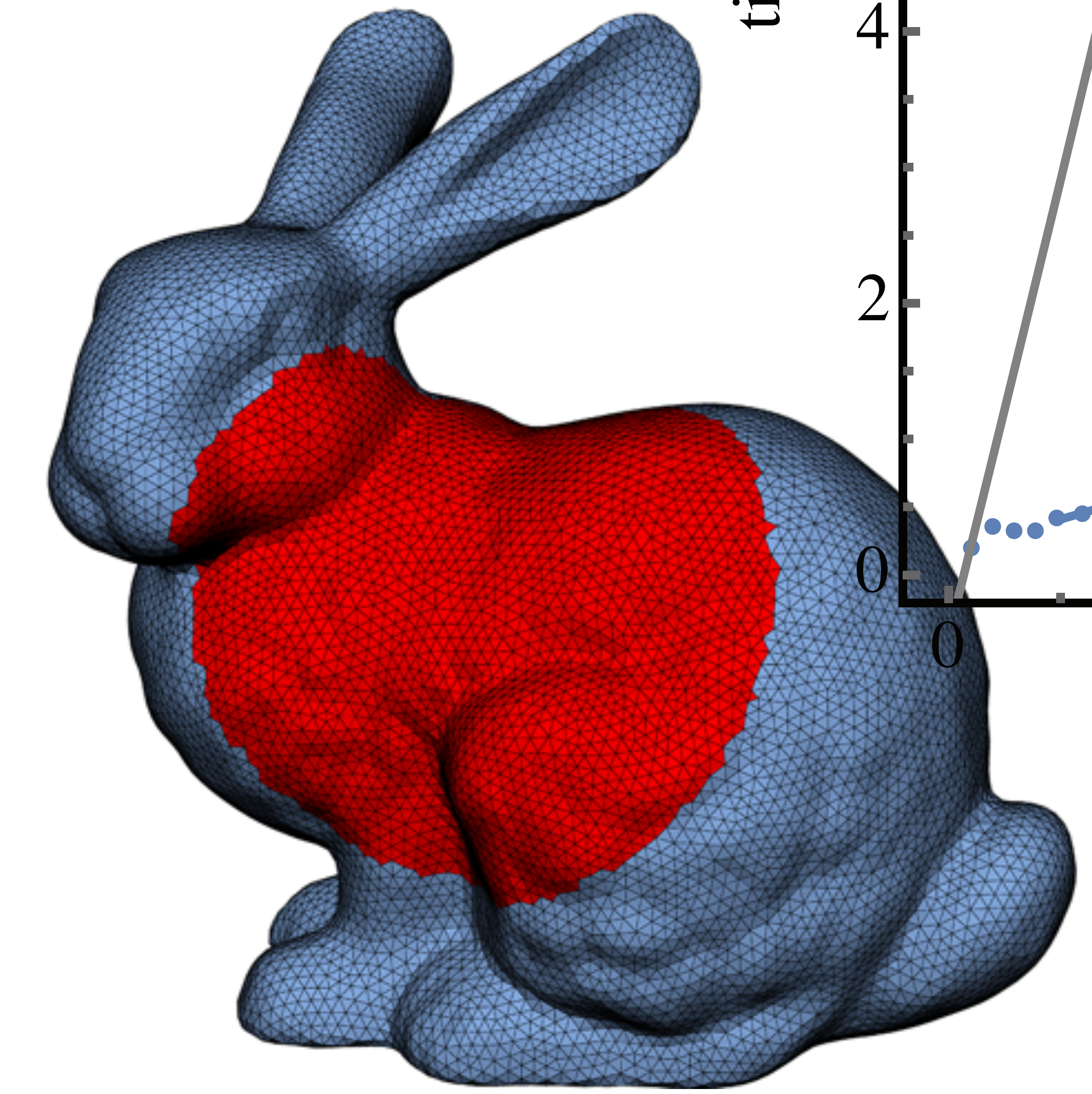
Is it worth it?

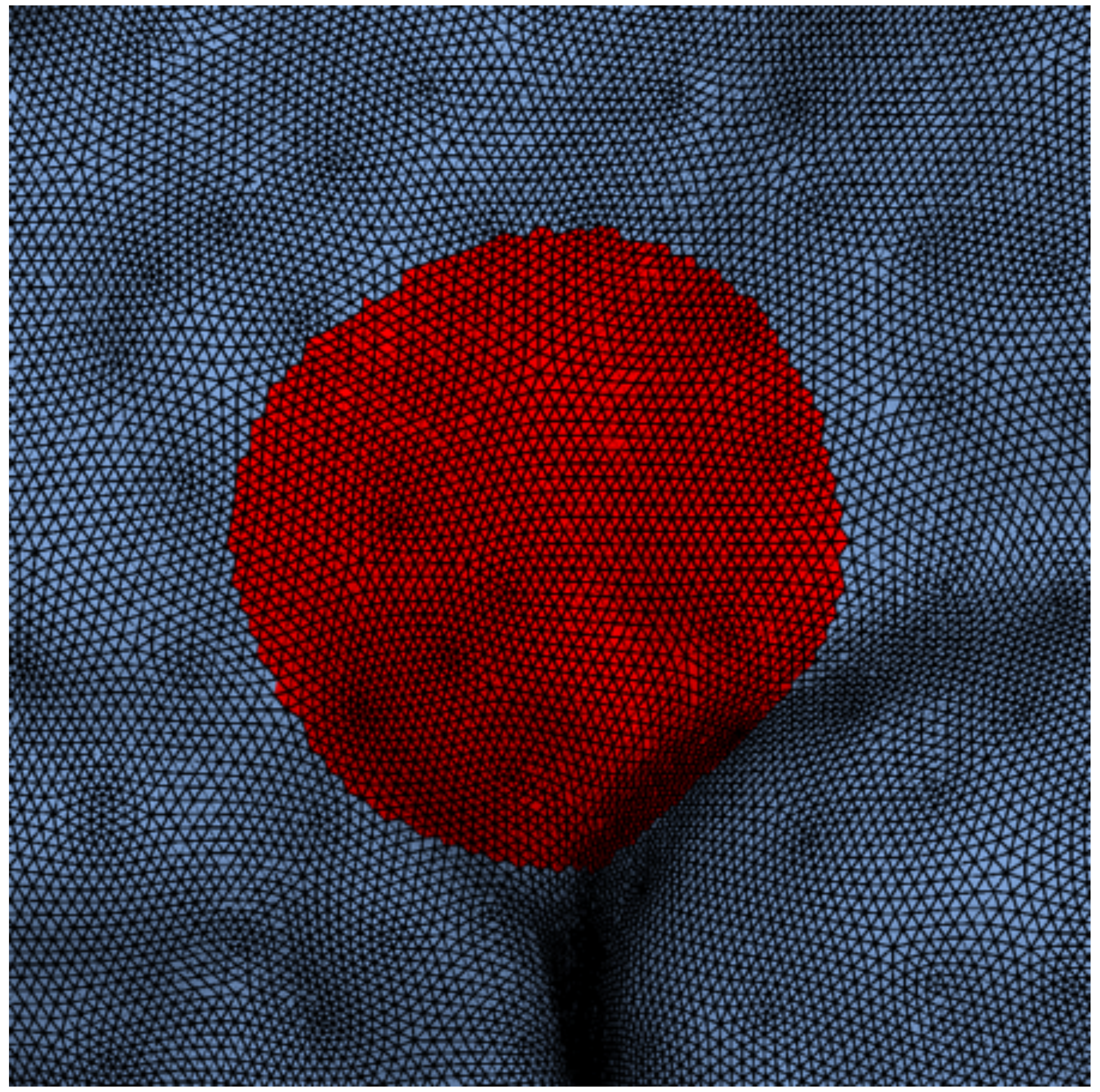




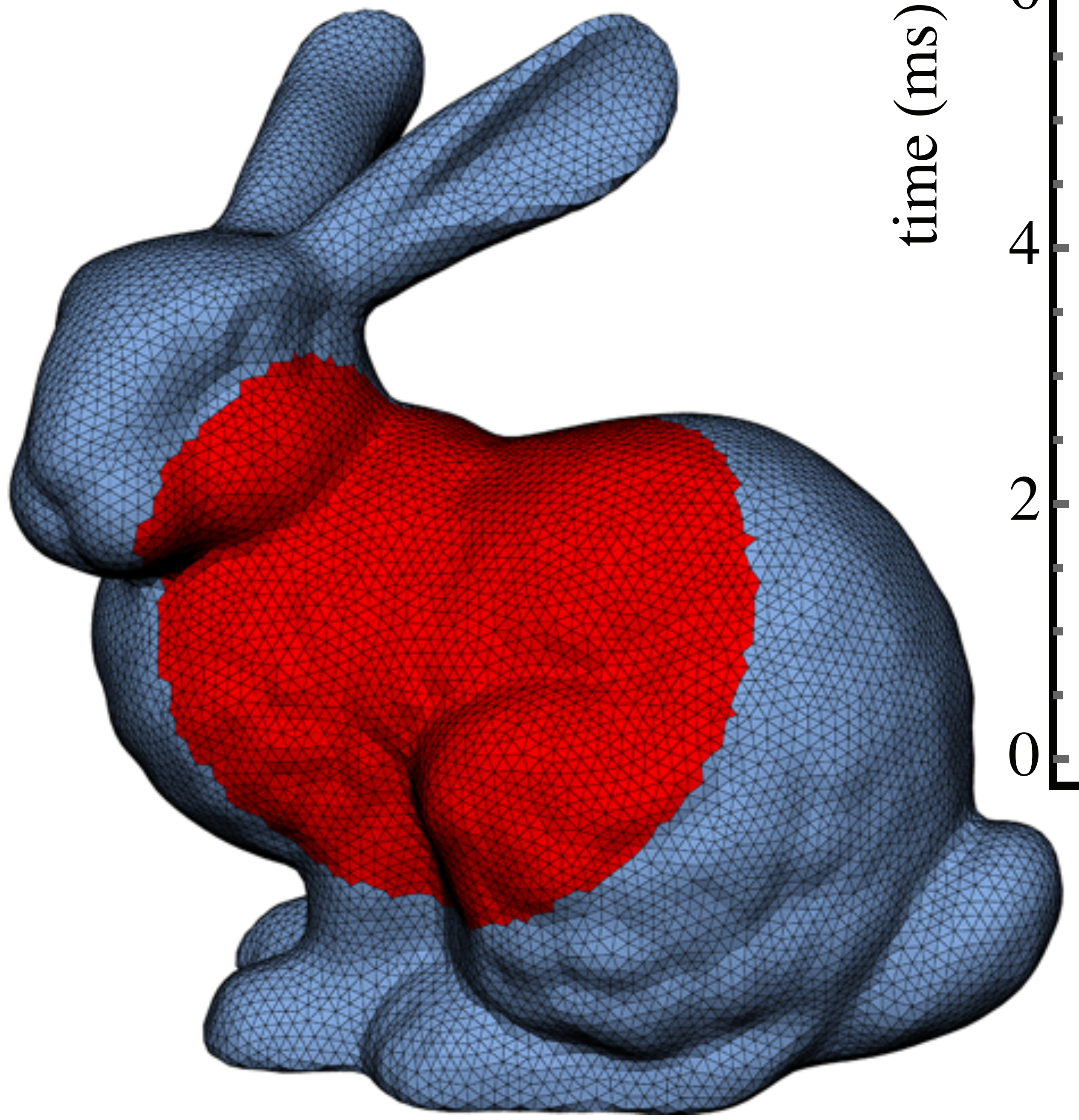
Is it worth it?

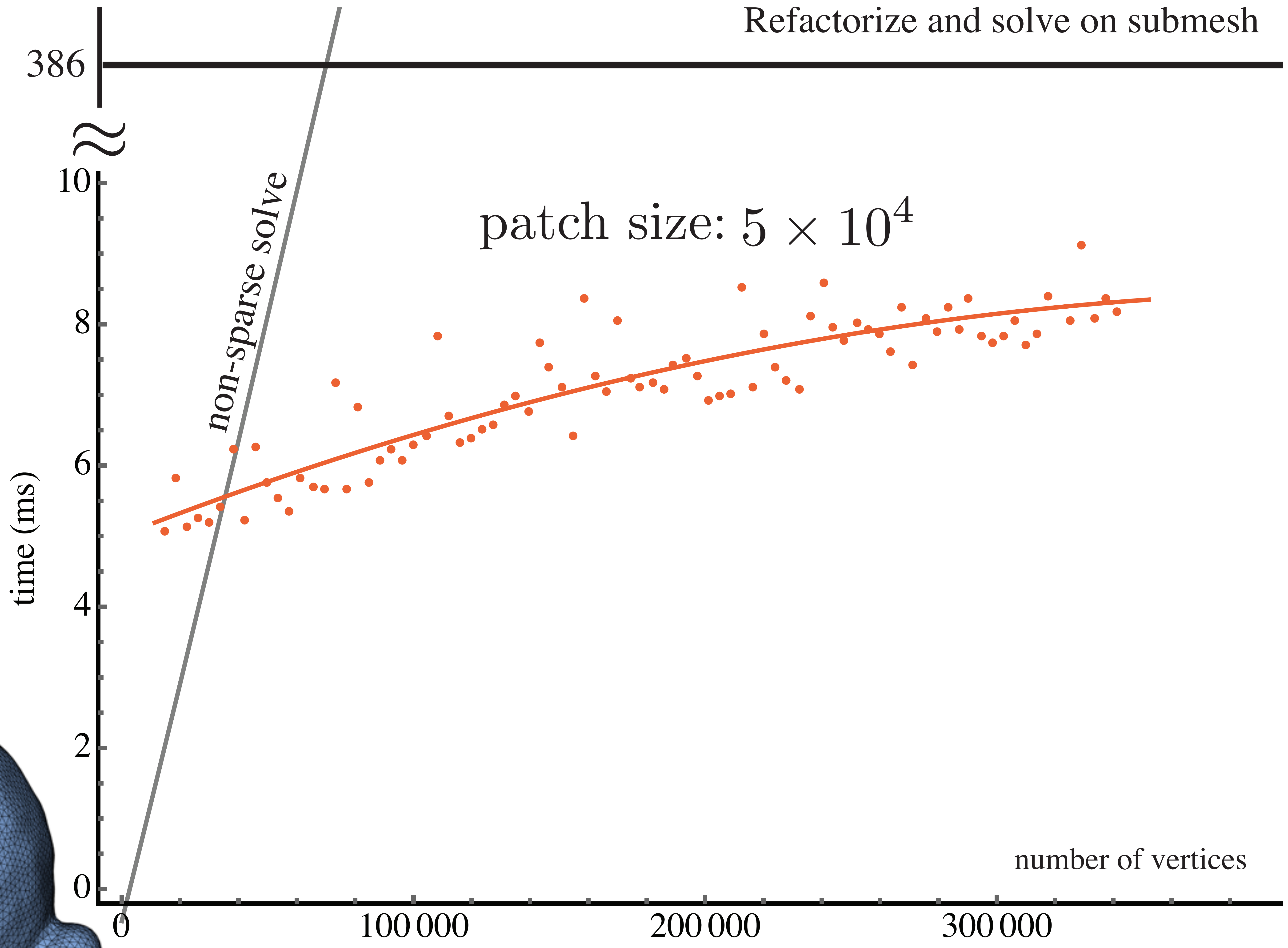


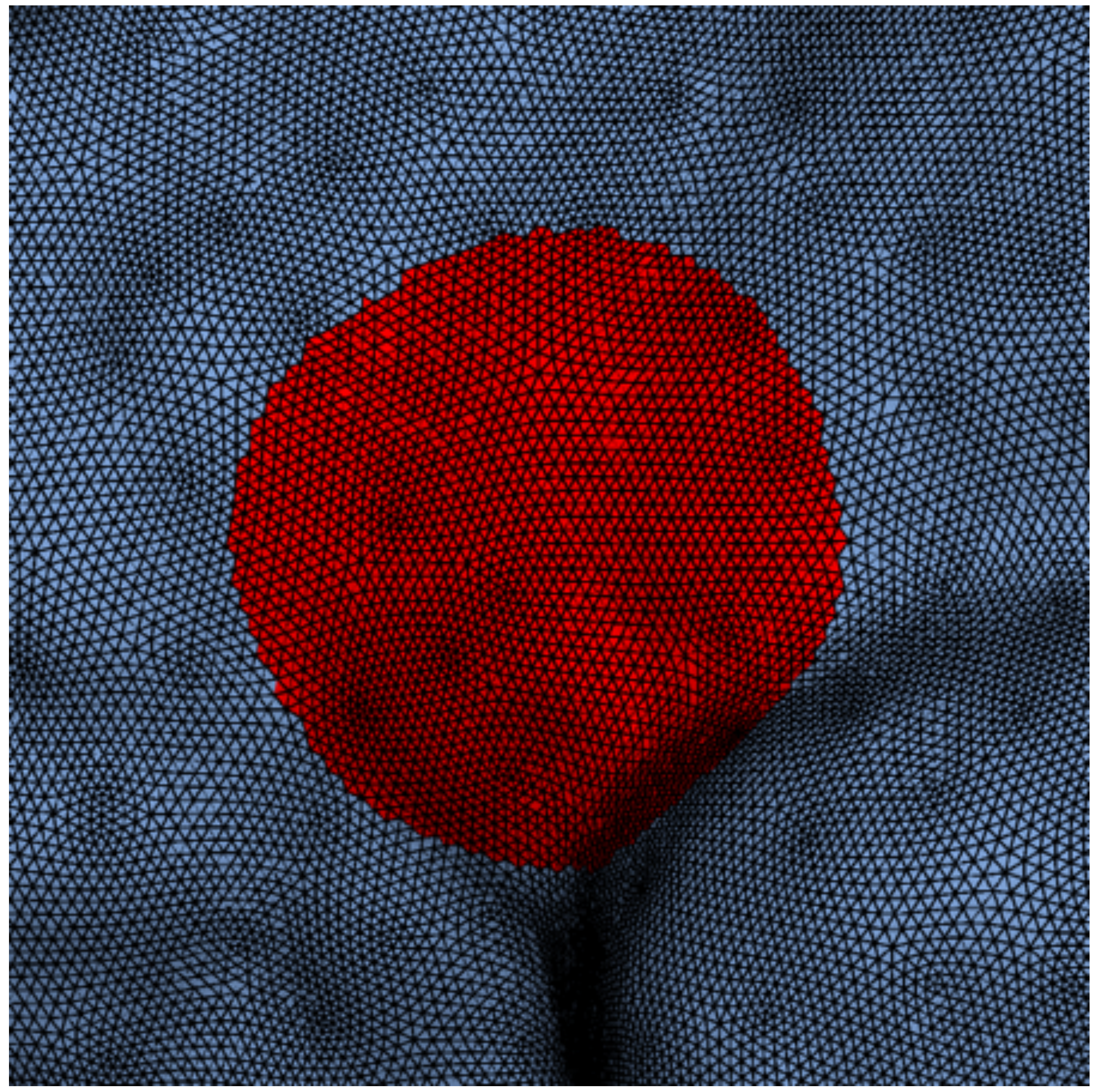




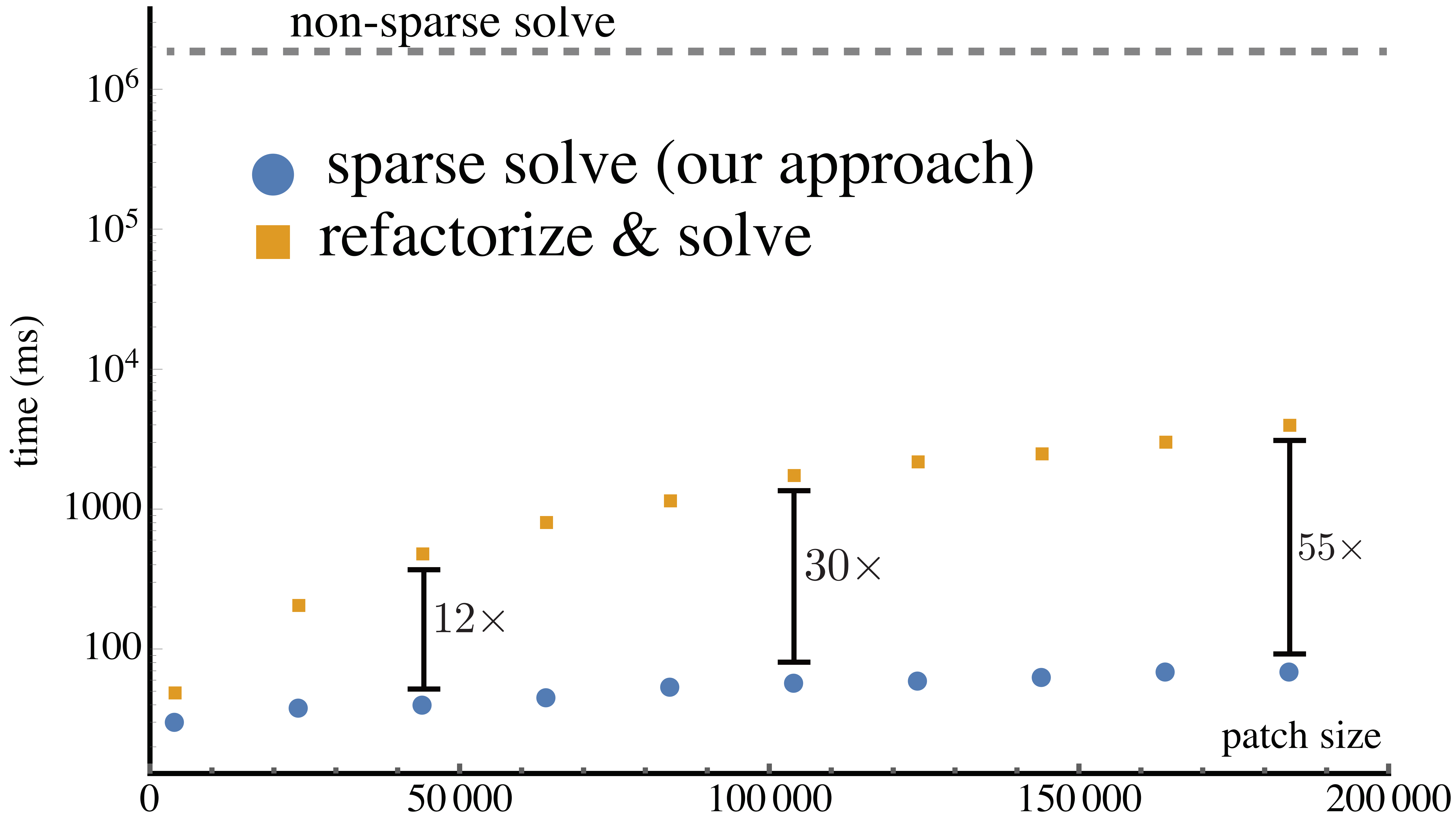
Is it worth it?







Is it worth it?

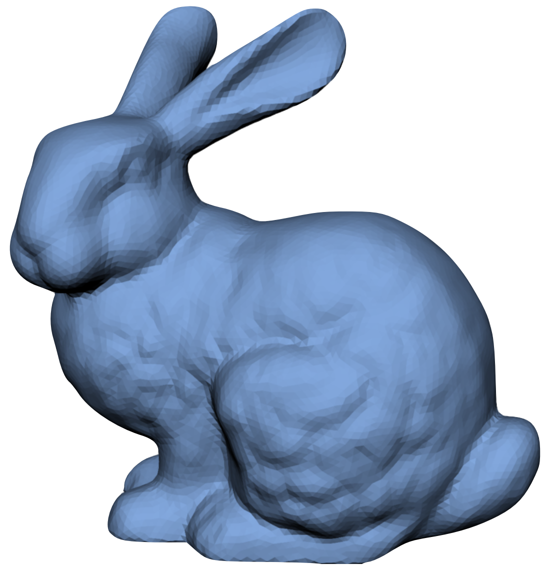


Amortization of factorization time

$n \times \Big($ $\mathbf{A}_1 = \mathbf{A}\big($  $\big)$

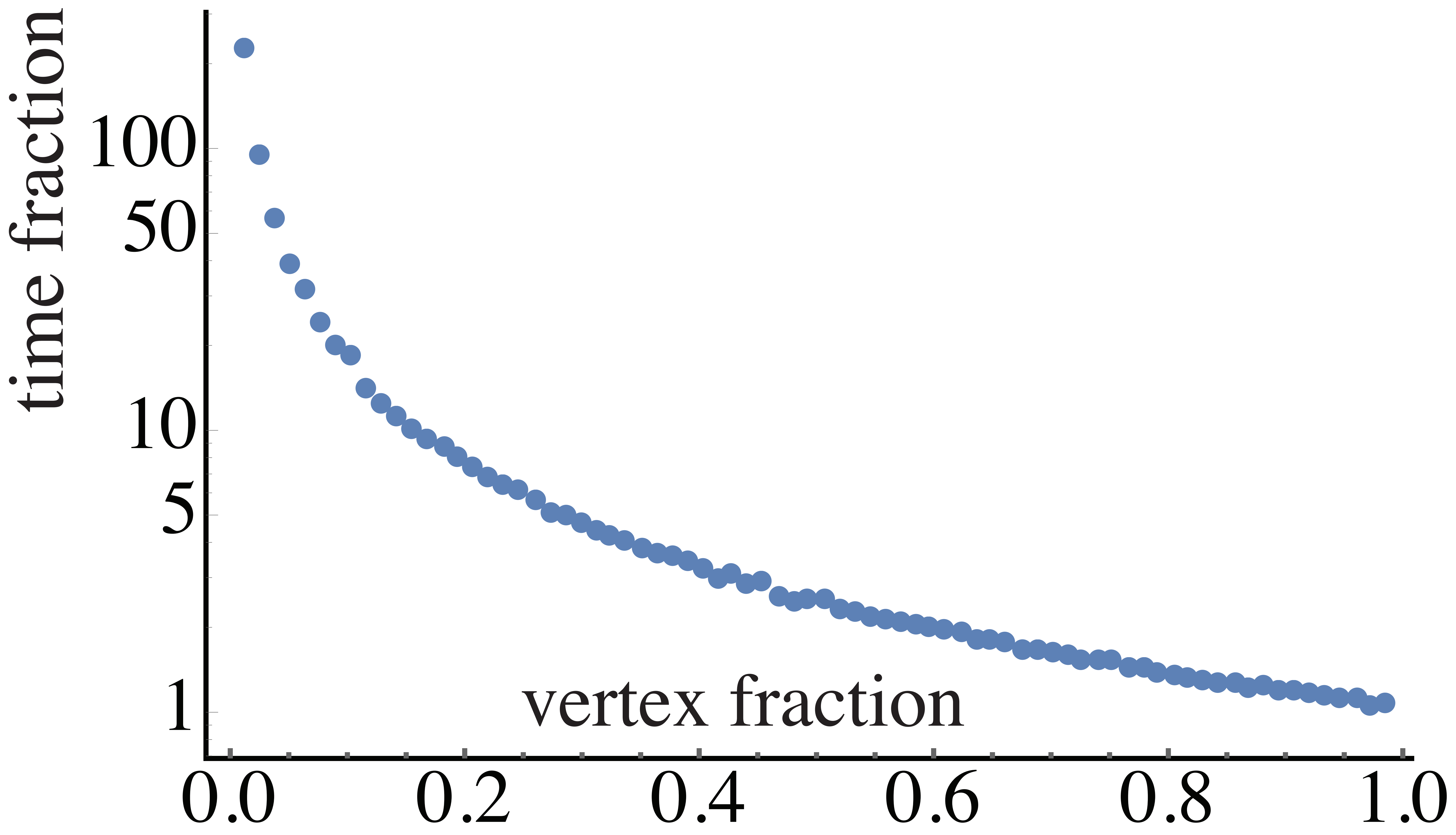
$\mathbf{A}_1 = \mathbf{LDL}^T$ $\Big)$

vs

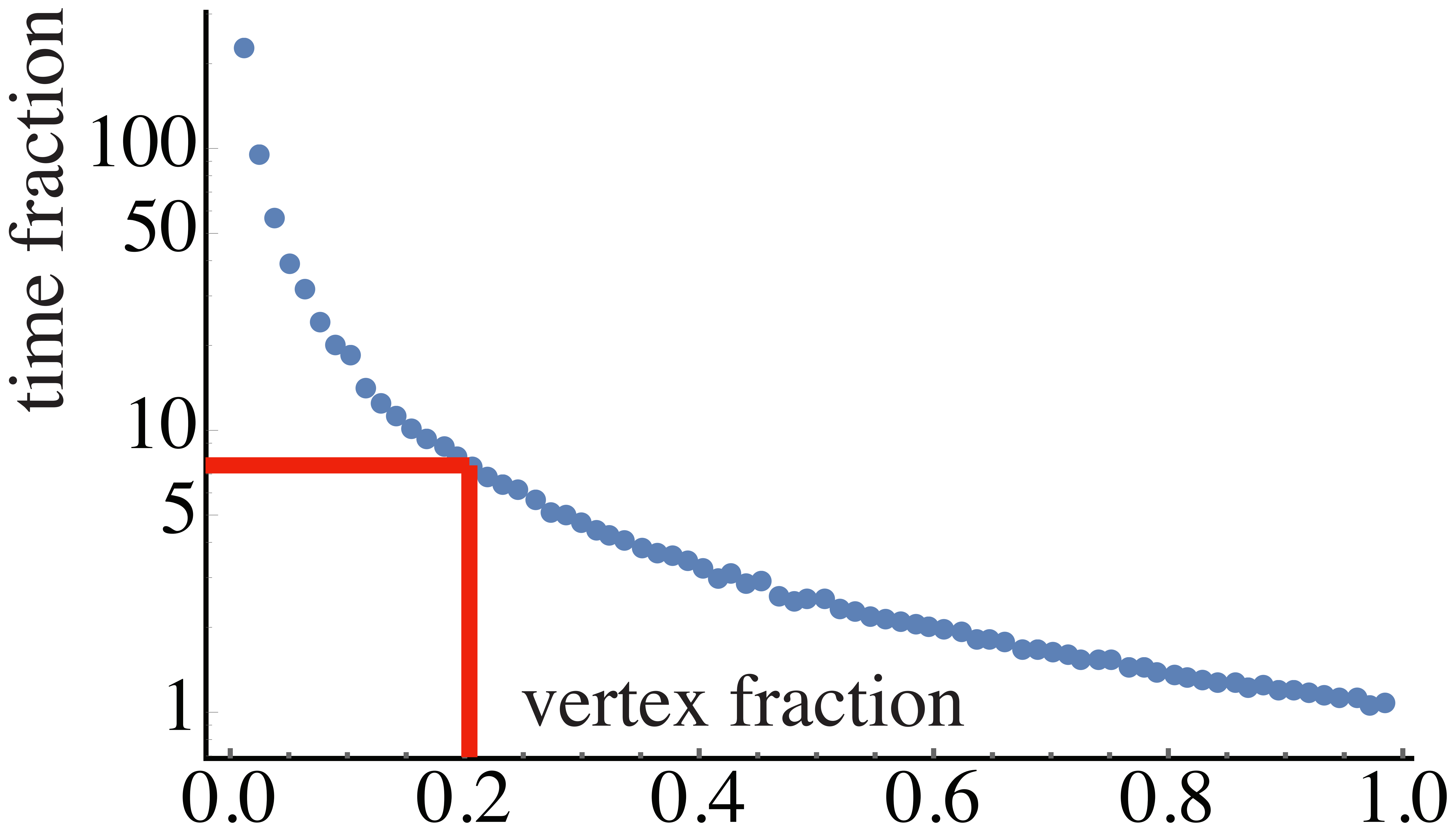
$\mathbf{A} = \mathbf{A}\big($  $\big)$

$\mathbf{A} = \mathbf{LDL}^T$

Amortization of factorization time



Amortization of factorization time

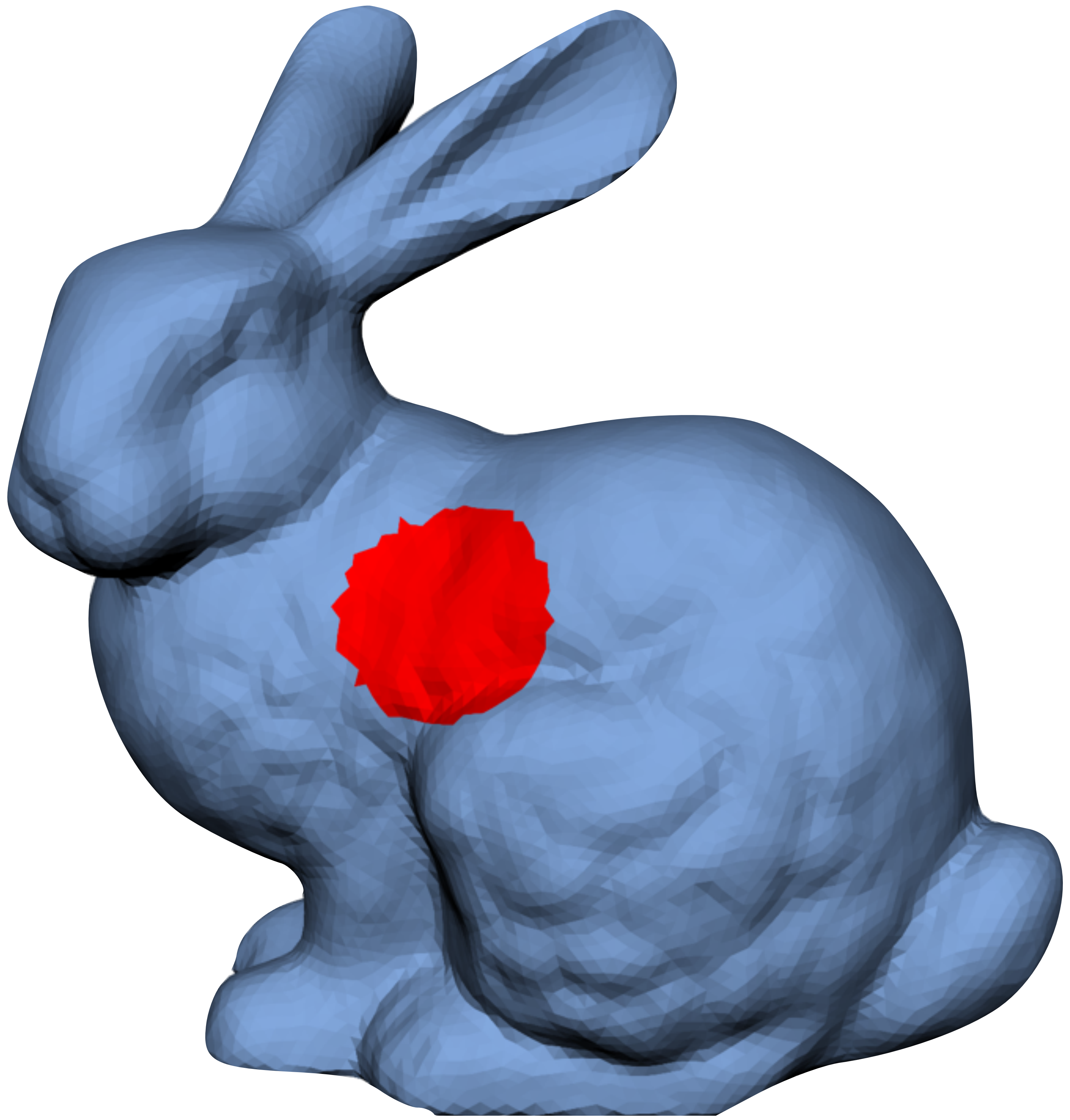


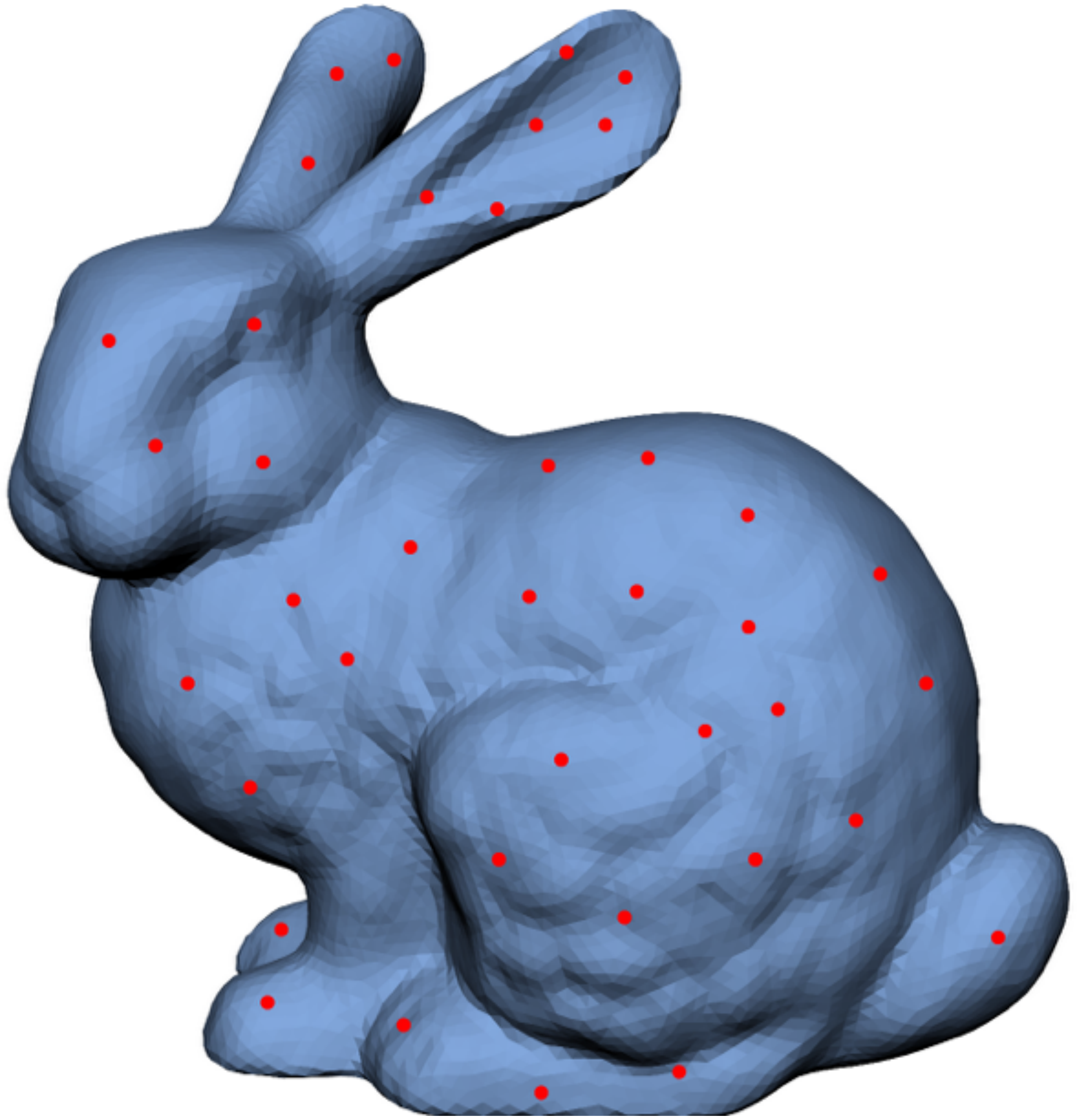
Limitations

- Only works if the requested values are local.

Limitations

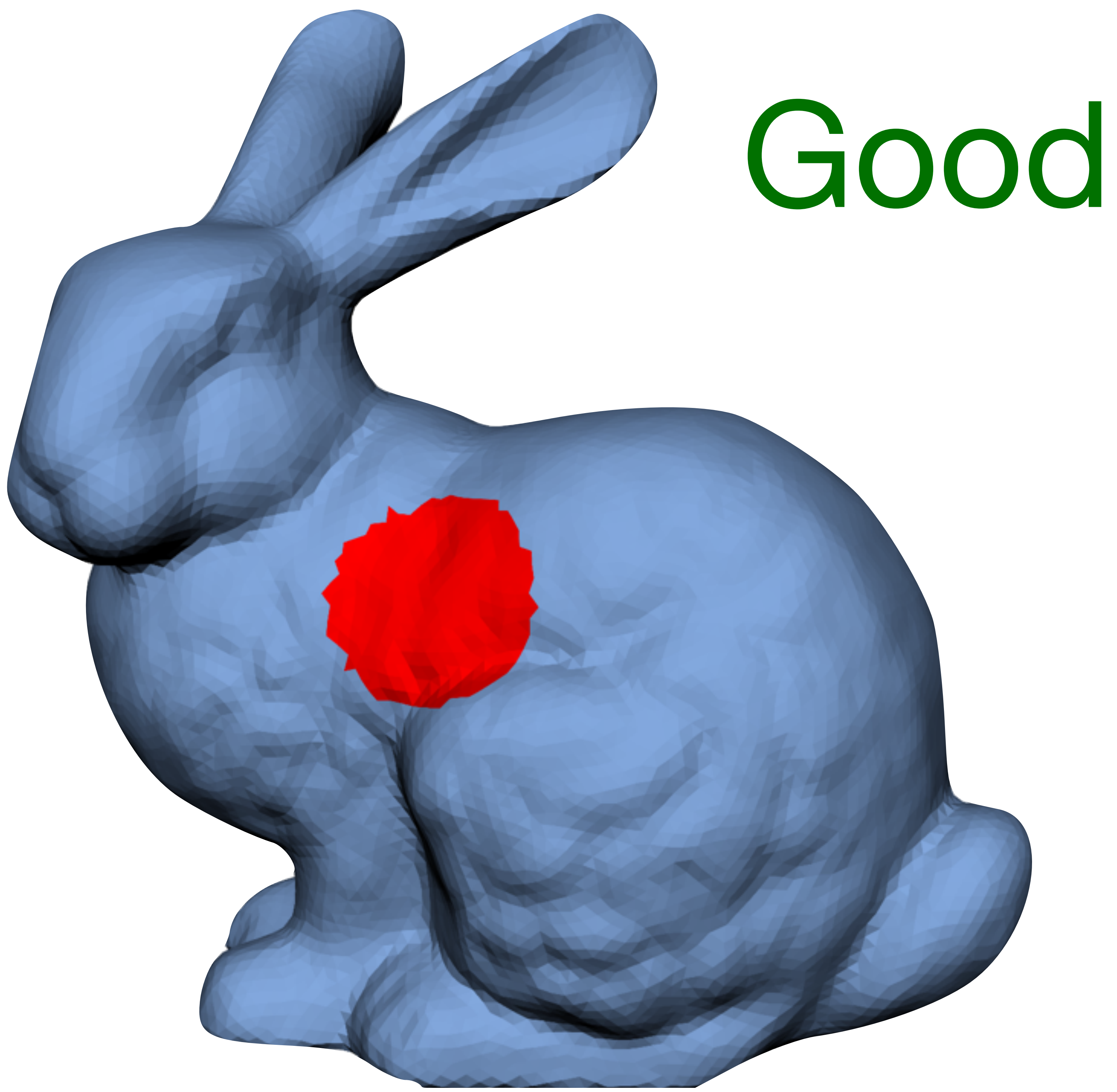
- Only works if the requested values are local.

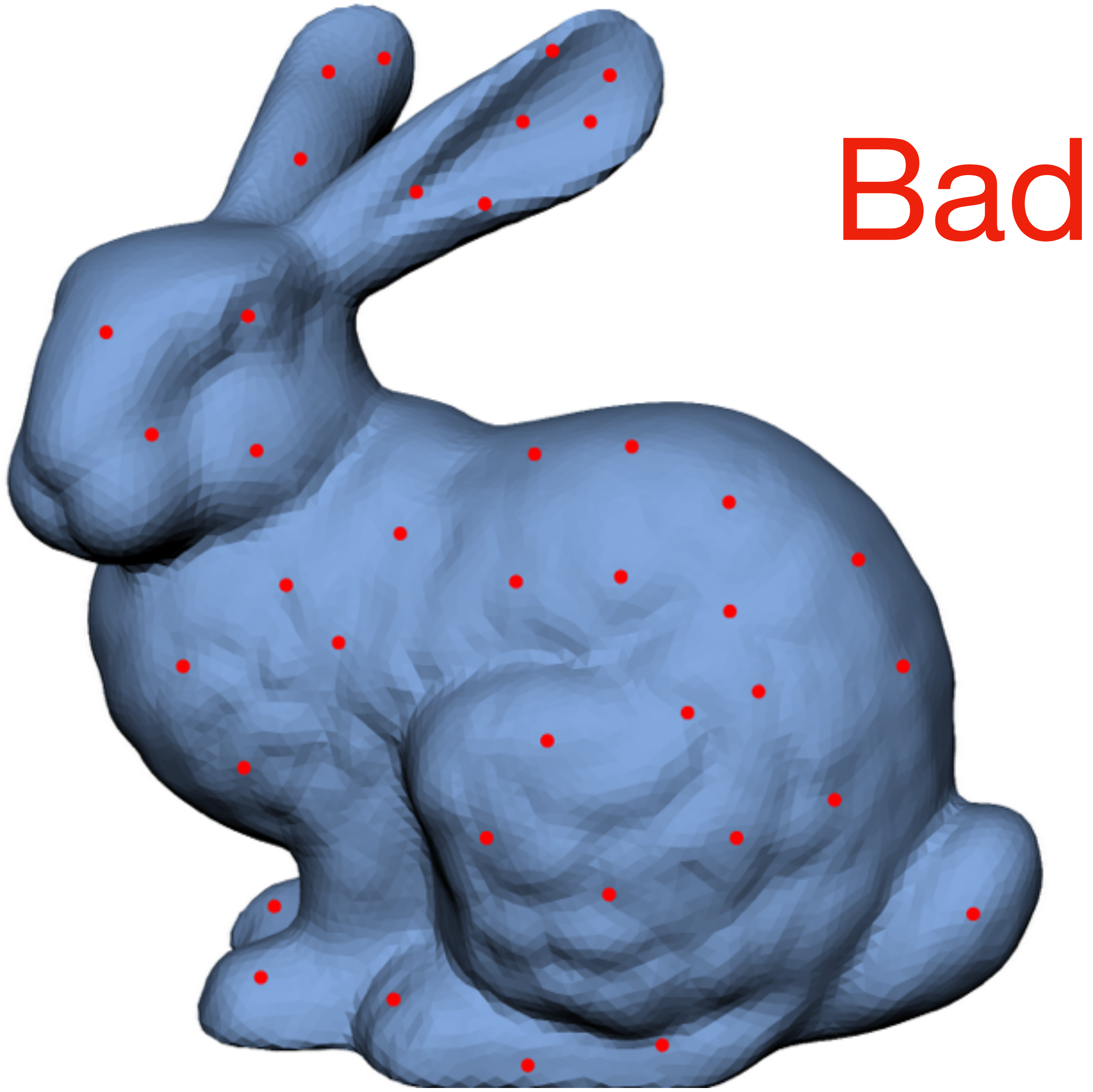




Limitations

- Only works if the requested values are local.





Limitations

- Only works if the requested values are local.
- The local problem needs to be formulated in terms of a global operator.

Conclusion

- Back-substitution can be significantly accelerated if only a few values are requested.
- If requested values belong to a localized patch on a mesh, performance benefits the most (because of nested dissection).
- Considering sparsity in the solution can improve performance by orders of magnitude.
- The method is trivial to implement.

Thank you for your attention!

Contact: philipp.herholz@tu-berlin.de

Applications: Auto-diffusion

- Compute heat diffusion from a single source vertex and evaluate the amount of heat staying at the source after some time.

- Only one value in $\mathbf{x}$ is required.