# Optimizing Voronoi Diagrams for Polygonal Finite Element Computations

Daniel Sieger & Mario Botsch
Bielefeld University

Pierre Alliez
INRIA Sophia-Antipolis

# Outline

# Outline

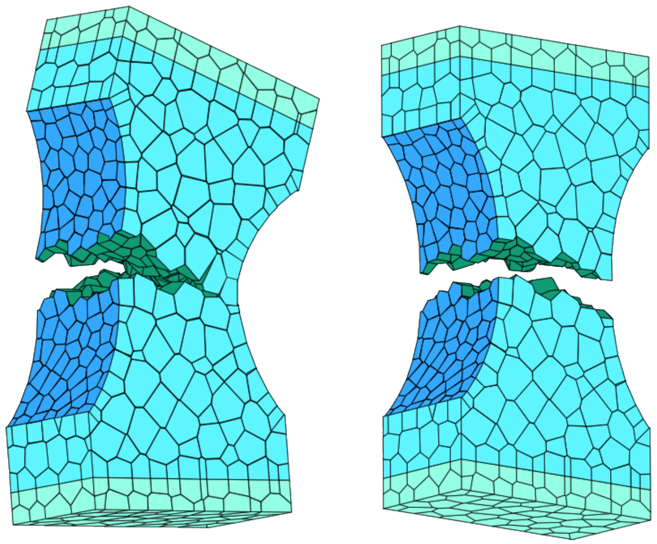
- Optimize
  - Centroidal Voronoi tessellations
  - Numerical robustness
  - Polygonal FEM

# Outline

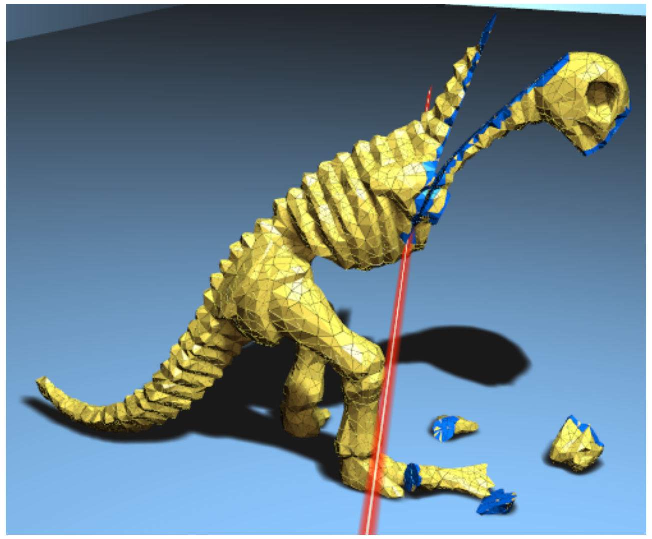
- Optimize
  - Centroidal Voronoi tessellations
  - Numerical robustness
  - Polygonal FEM
- Numerical robustness:
  - Problem dependence: 2D Poisson problem
  - Element shape
- Mesh optimization

# Polygonal Finite Elements

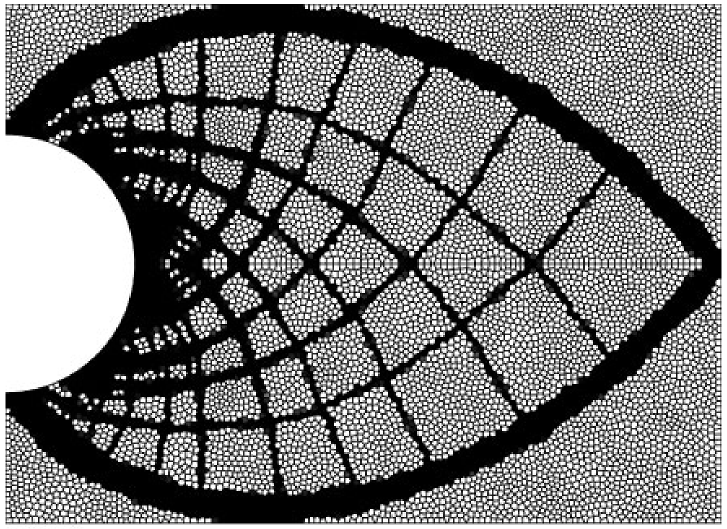
- FEM on polygonal or polyhedral meshes
- Advantages:
  - Flexible modeling
  - Fracture, cutting
  - Topology optimization



[Sukumar & Bolander, 2009]



[Martin et al, 2008]



[Talischi et al, 2010]

# 2D Poisson Problem

- Find $u : \mathbb{R}^2 \to \mathbb{R}$ such that

$$\begin{aligned} -\Delta u &= f \quad \text{in } \Omega \subset \mathbb{R}^2 \\ u &= \bar{u} \quad \text{on } \partial\Omega \end{aligned}$$

- $\bar{u}$ is the know solution

- Approximate $u$ by basis functions $N_i$ interpolating nodal DoFs $u_i$

$$u(\mathbf{x}) \approx \sum_{i=1}^{n} u_i N_i(\mathbf{x})$$

# Linear System

$$\mathbf{K}\mathbf{u} \;=\; \mathbf{f} \quad \text{with}\, \mathbf{K}_{ij} = \int_{\Omega} \nabla N_i \cdot \nabla N_j \, \mathrm{d}\Omega$$

# Linear System

- Inserting the approximation into the weak form leads to linear system:

$$\mathbf{K}\mathbf{u} = \mathbf{f} \quad \text{with } \mathbf{K}_{ij} = \int_\Omega \nabla N_i \cdot \nabla N_j \, \mathrm{d}\Omega$$

# Linear System

- Inserting the approximation into the weak form leads to linear system:

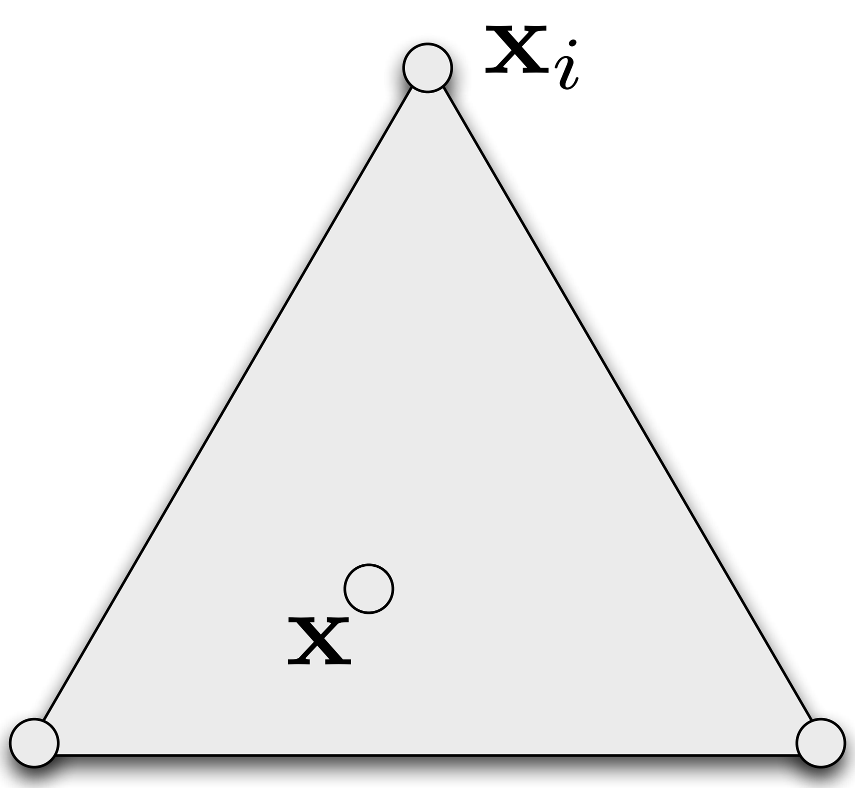
$$\mathbf{K}\mathbf{u} \;=\; \mathbf{f} \quad \text{with } \mathbf{K}_{ij} = \int_\Omega \nabla N_i \cdot \nabla N_j \, \mathrm{d}\Omega$$

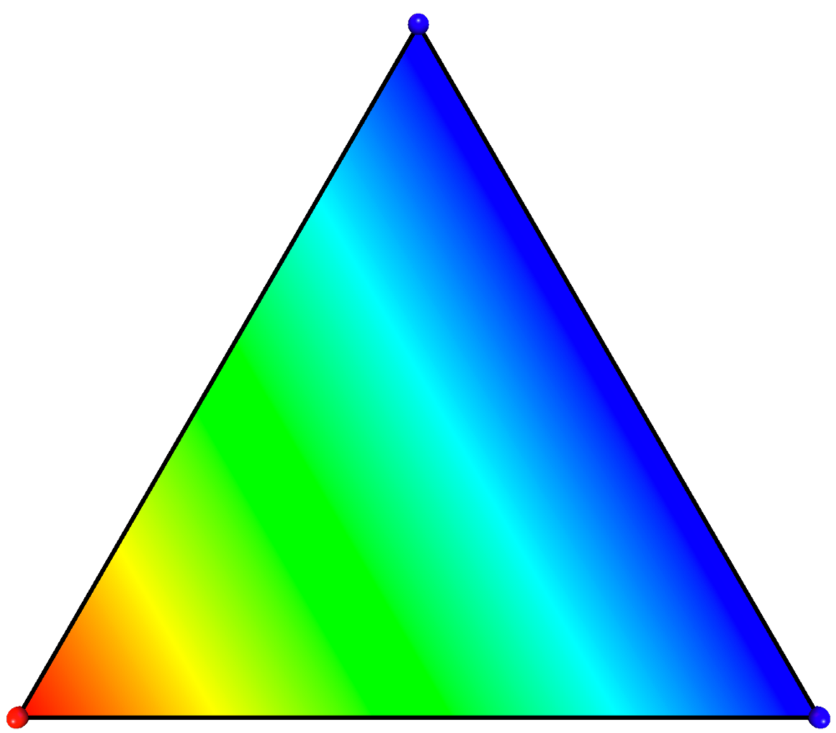
- Condition number of $\mathbf{K}$:

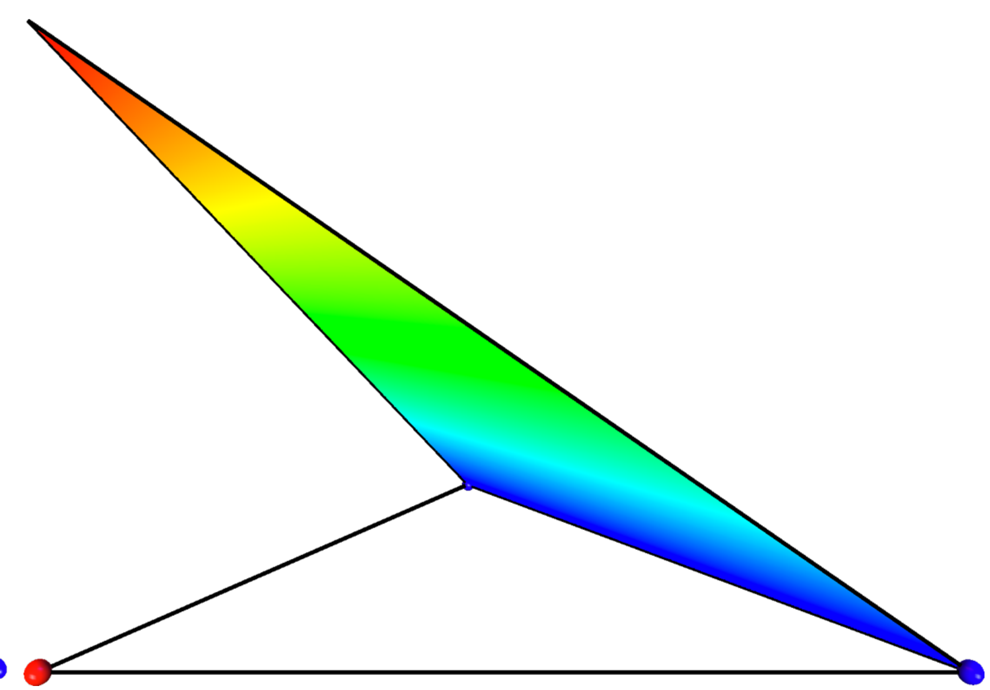
$$\kappa = \lambda_{\mathtt{max}} / \lambda_{\mathtt{min}}$$

# Triangular Basis Functions

- Barycentric coordinates
  - Partition of unity $\sum_{i=1}^{3} N_i(\mathbf{x}) = 1$
  - Linear precision $\sum_{i=1}^{3} N_i \mathbf{x}_i = \mathbf{x}$
  - Lagrange property $N_i(\mathbf{x}_j) = \delta_{i,j}$

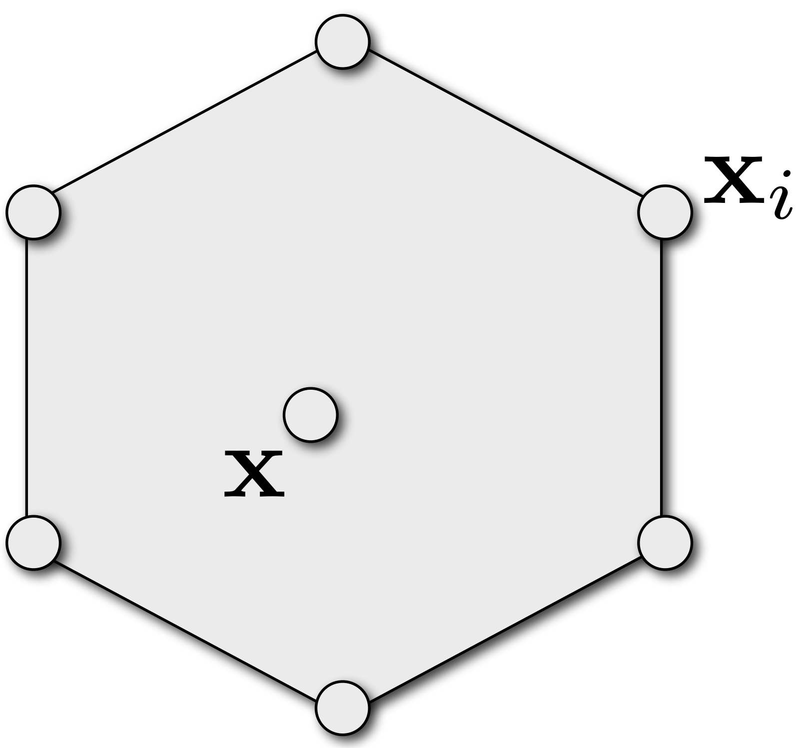


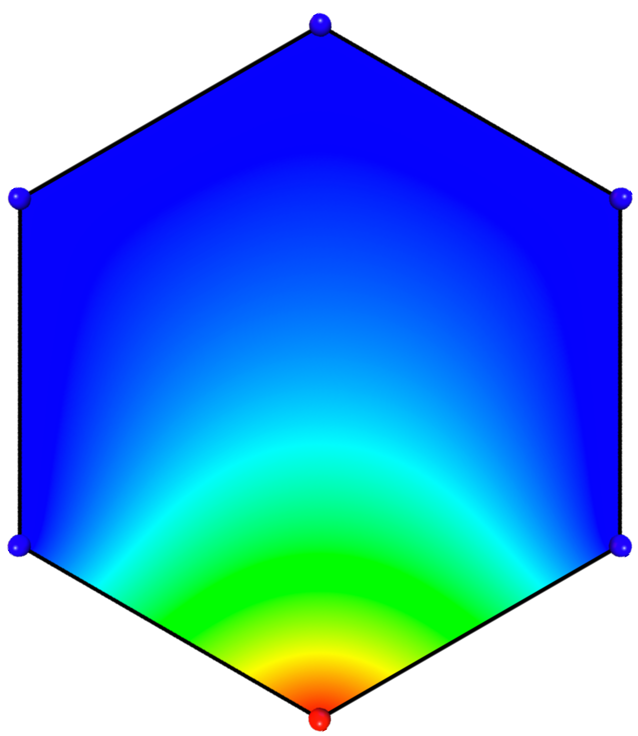



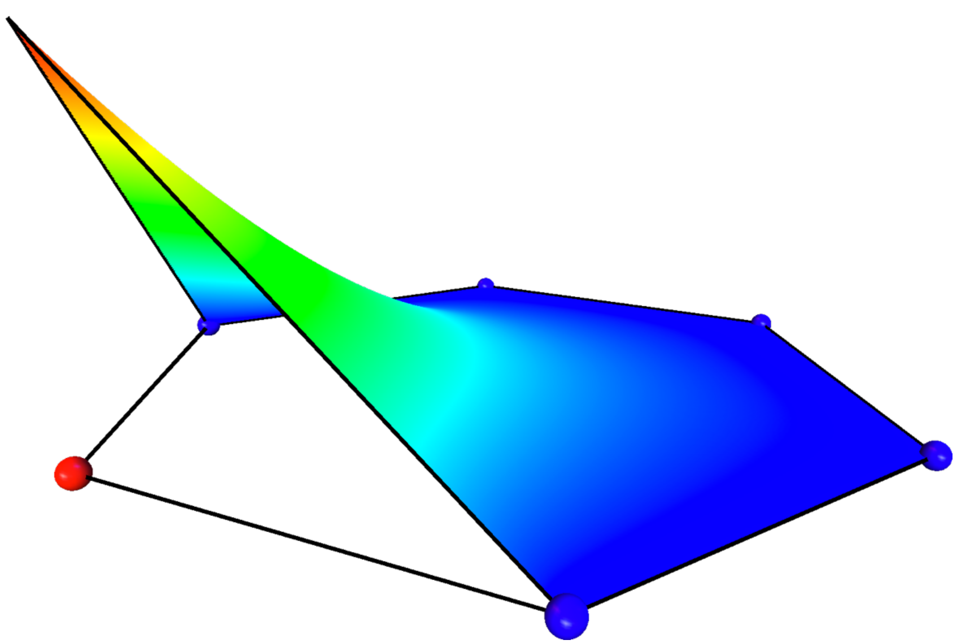


# Polygonal Basis Functions

- Generalized barycentric coordinates
  - Partition of unity $\sum_{i=1}^{n} N_i(\mathbf{x}) = 1$
  - Linear precision $\sum_{i=1}^{n} N_i \mathbf{x}_i = \mathbf{x}$
  - Lagrange property $N_i(\mathbf{x}_j) = \delta_{i,j}$

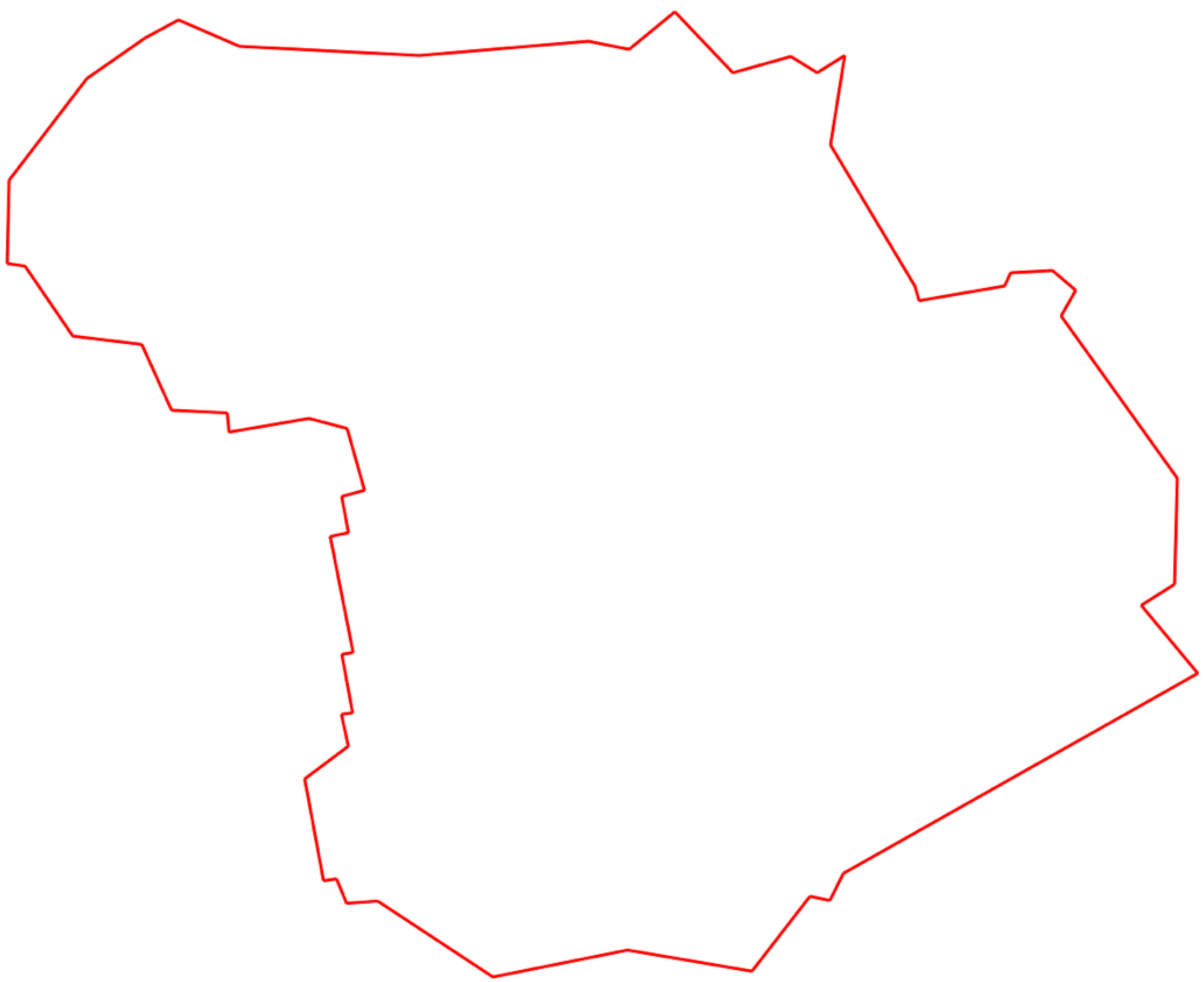


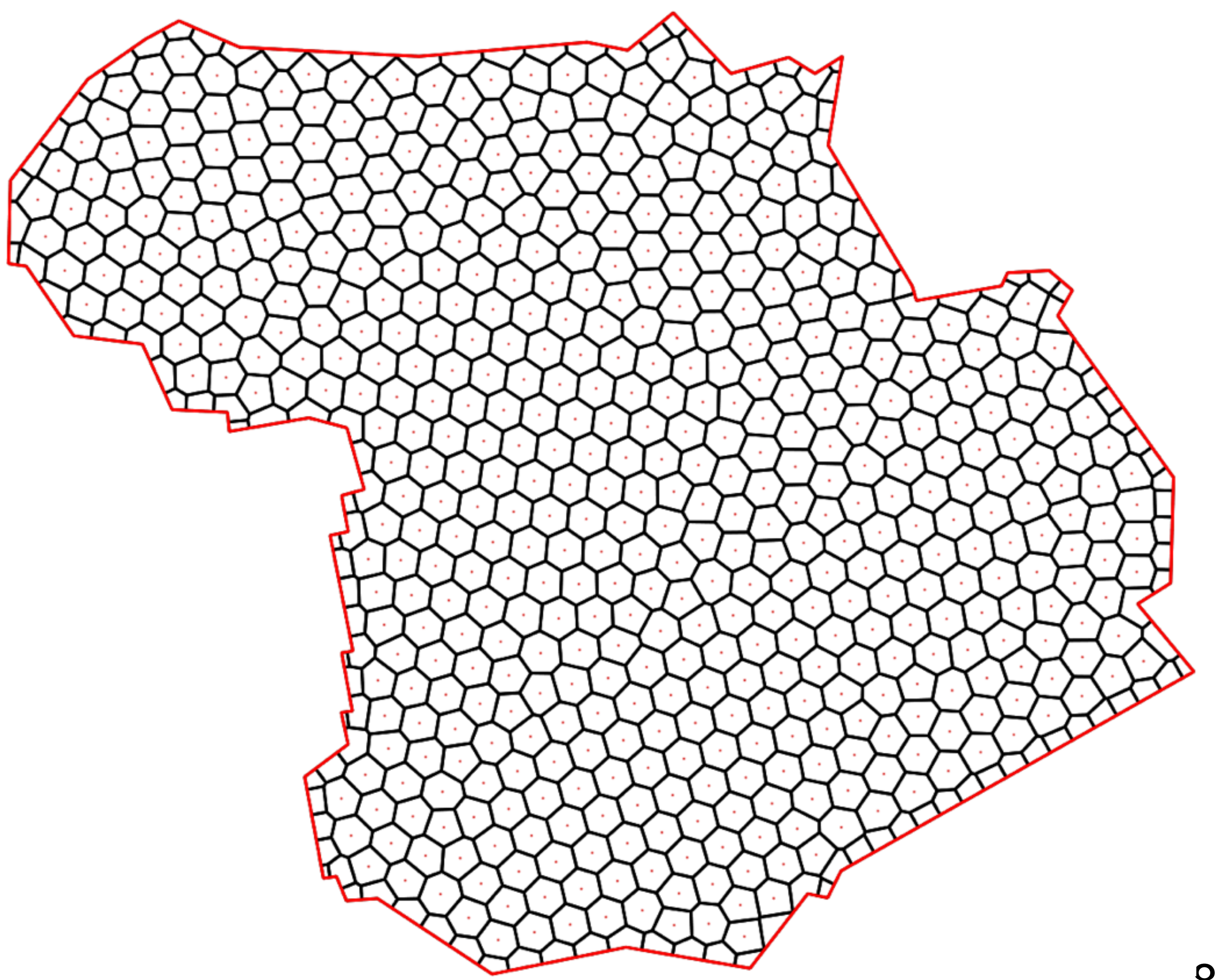





# Centroidal Voronoi Tessellations

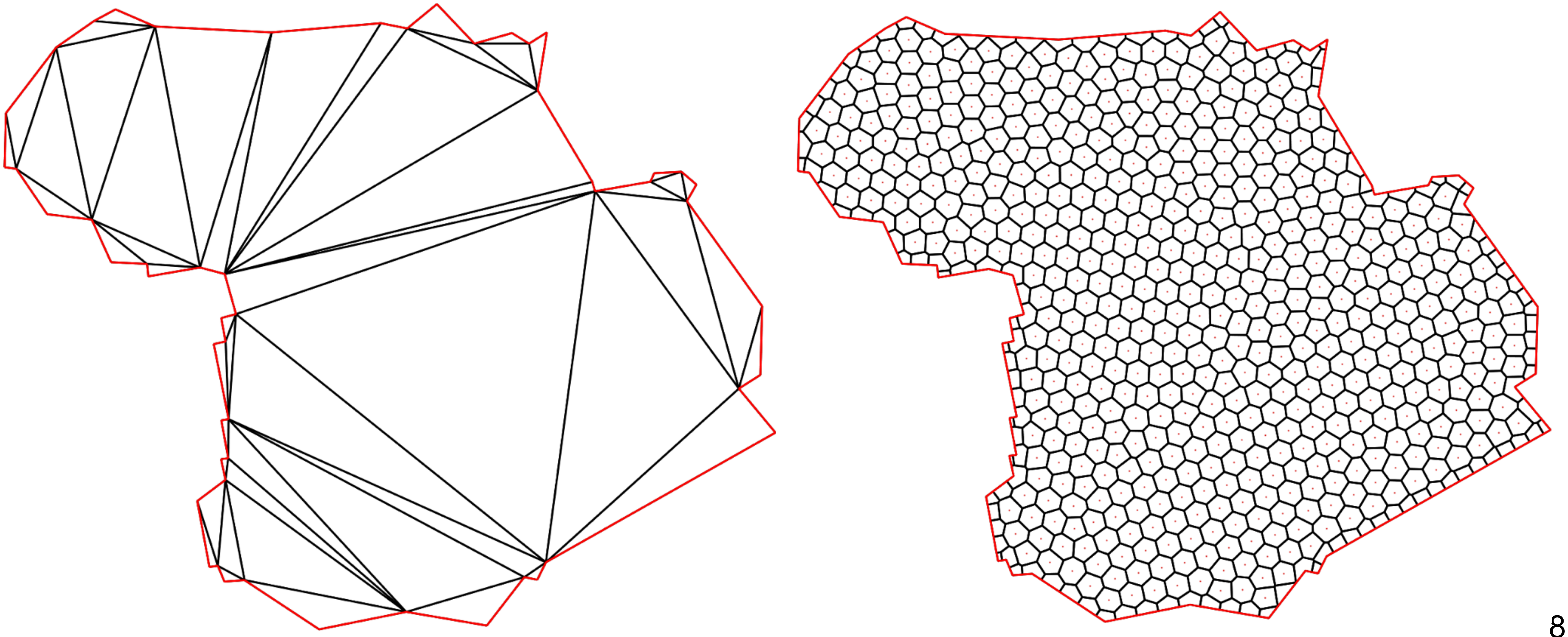
- Centroidal Voronoi tessellations (CVTs)
  - Convex & well-shaped elements
- Interleaved refinement & Lloyd relaxation





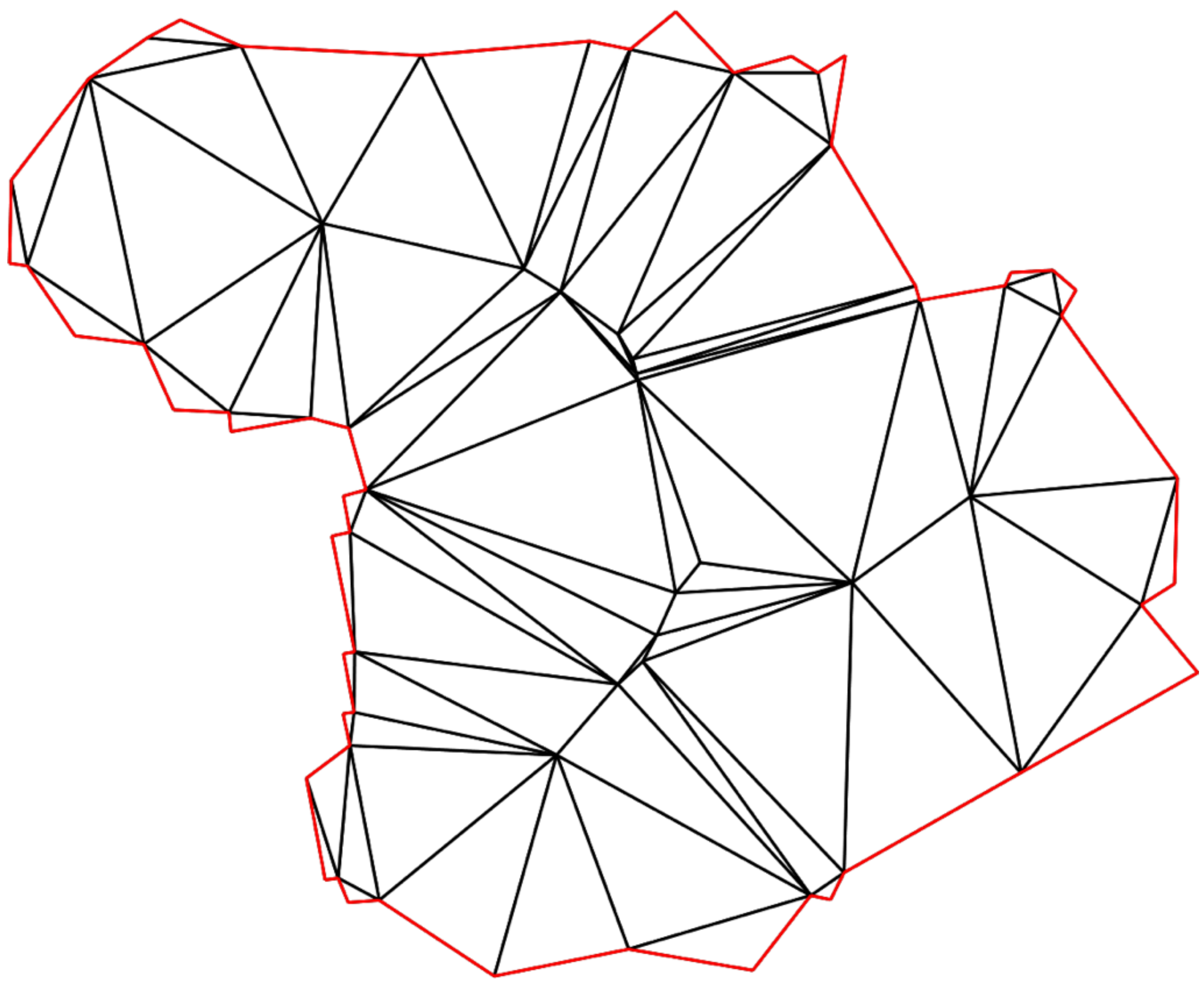
# Centroidal Voronoi Tessellations

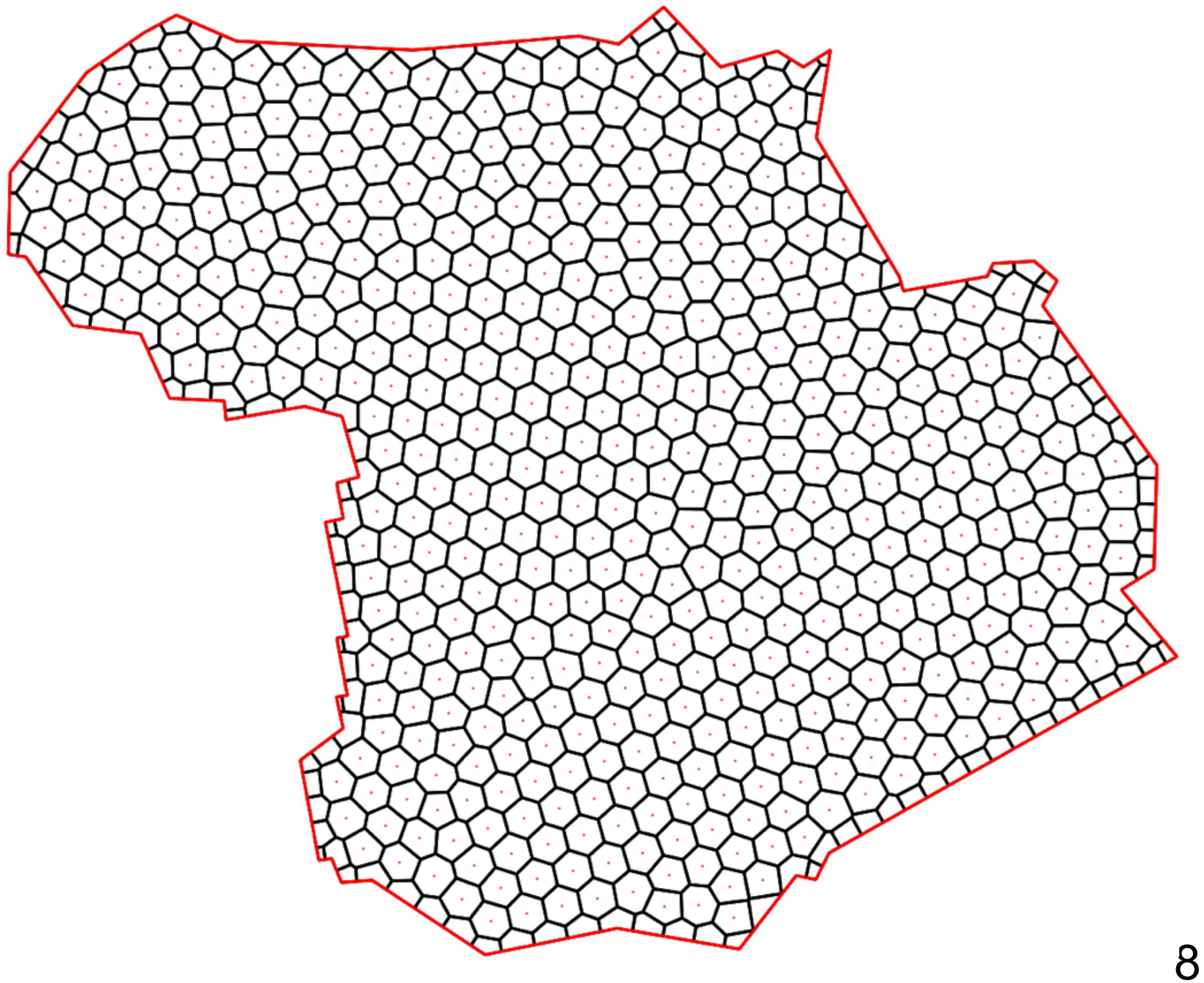
- Centroidal Voronoi tessellations (CVTs)
  - Convex & well-shaped elements
- Interleaved refinement & Lloyd relaxation



# Centroidal Voronoi Tessellations

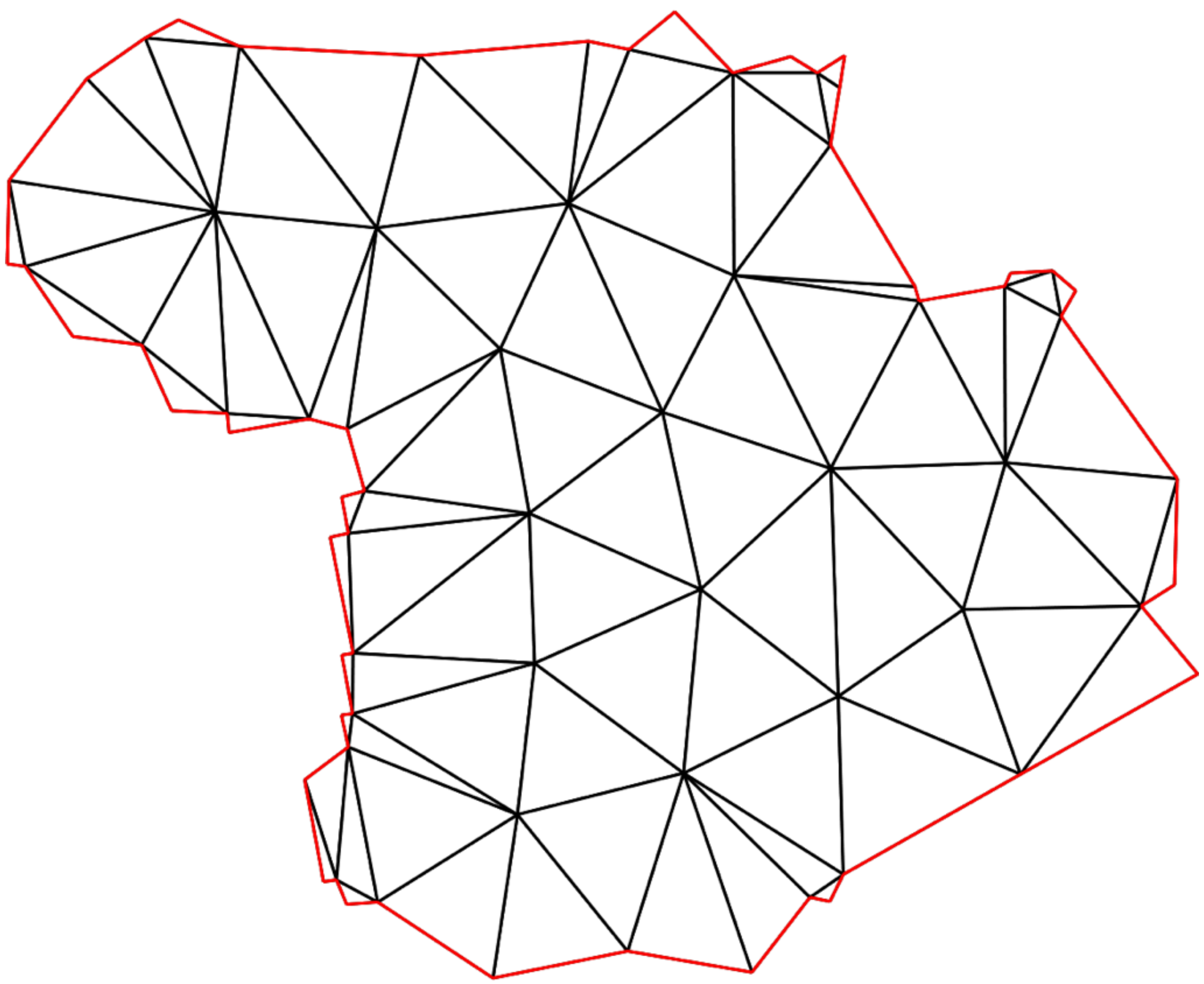
- Centroidal Voronoi tessellations (CVTs)
  - Convex & well-shaped elements
- Interleaved refinement & Lloyd relaxation

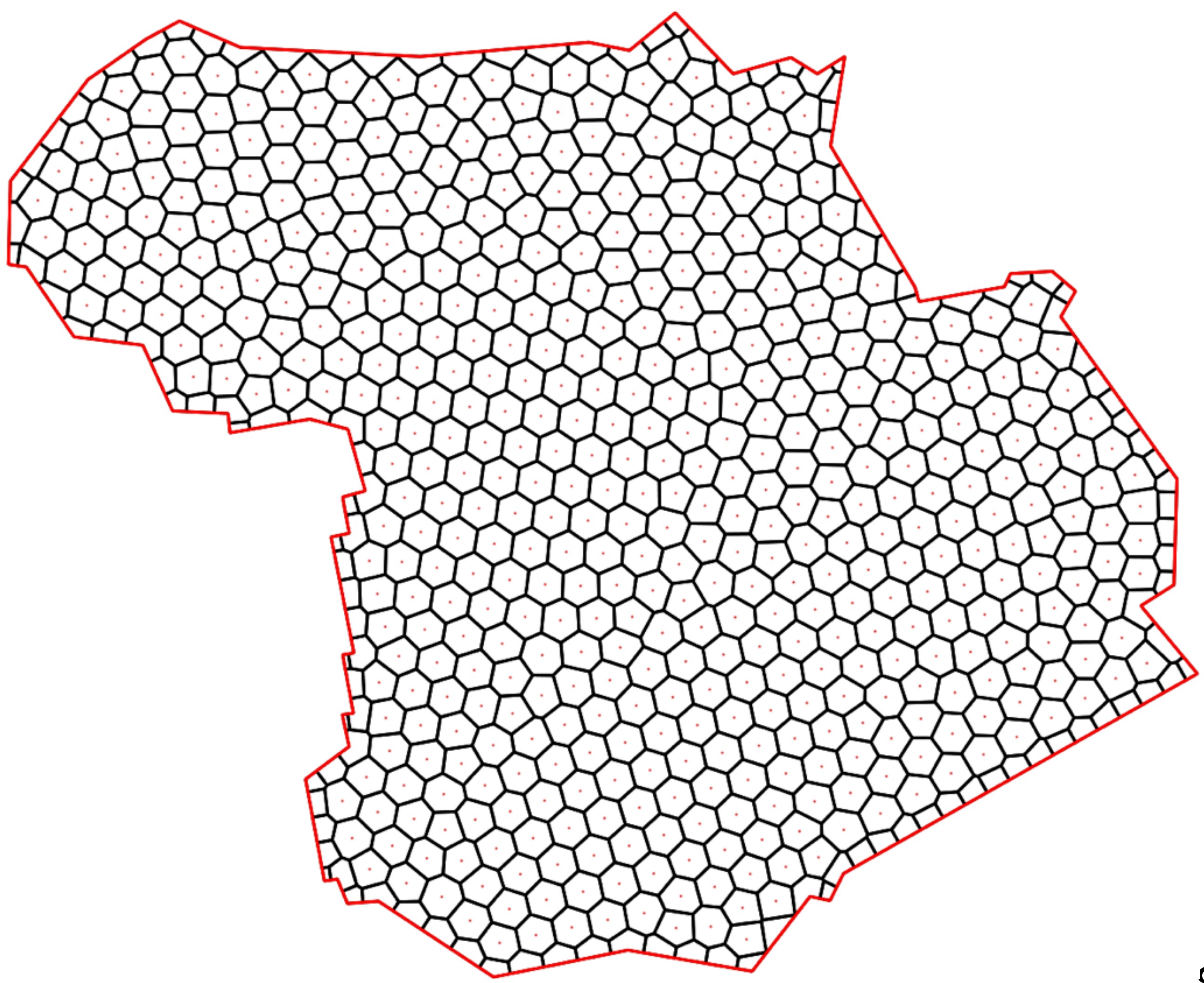




# Centroidal Voronoi Tessellations

- Centroidal Voronoi tessellations (CVTs)
  - Convex & well-shaped elements
- Interleaved refinement & Lloyd relaxation



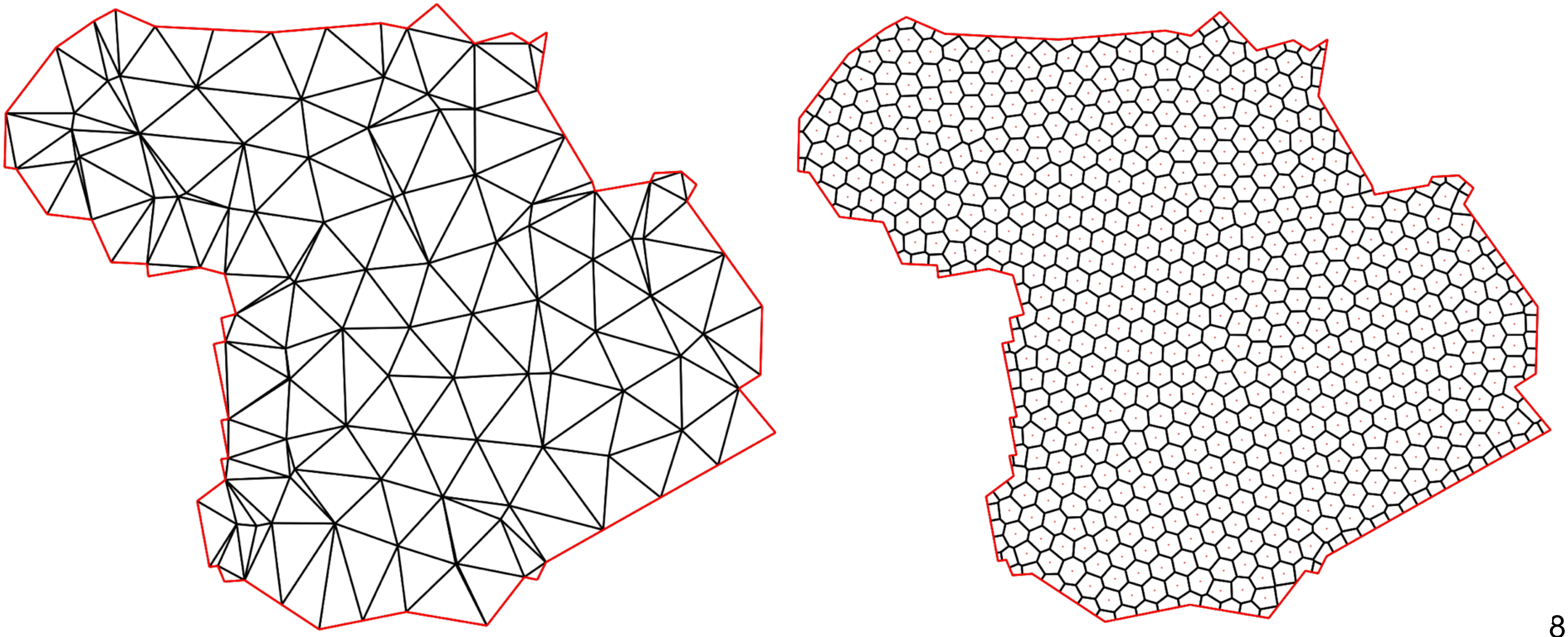




8

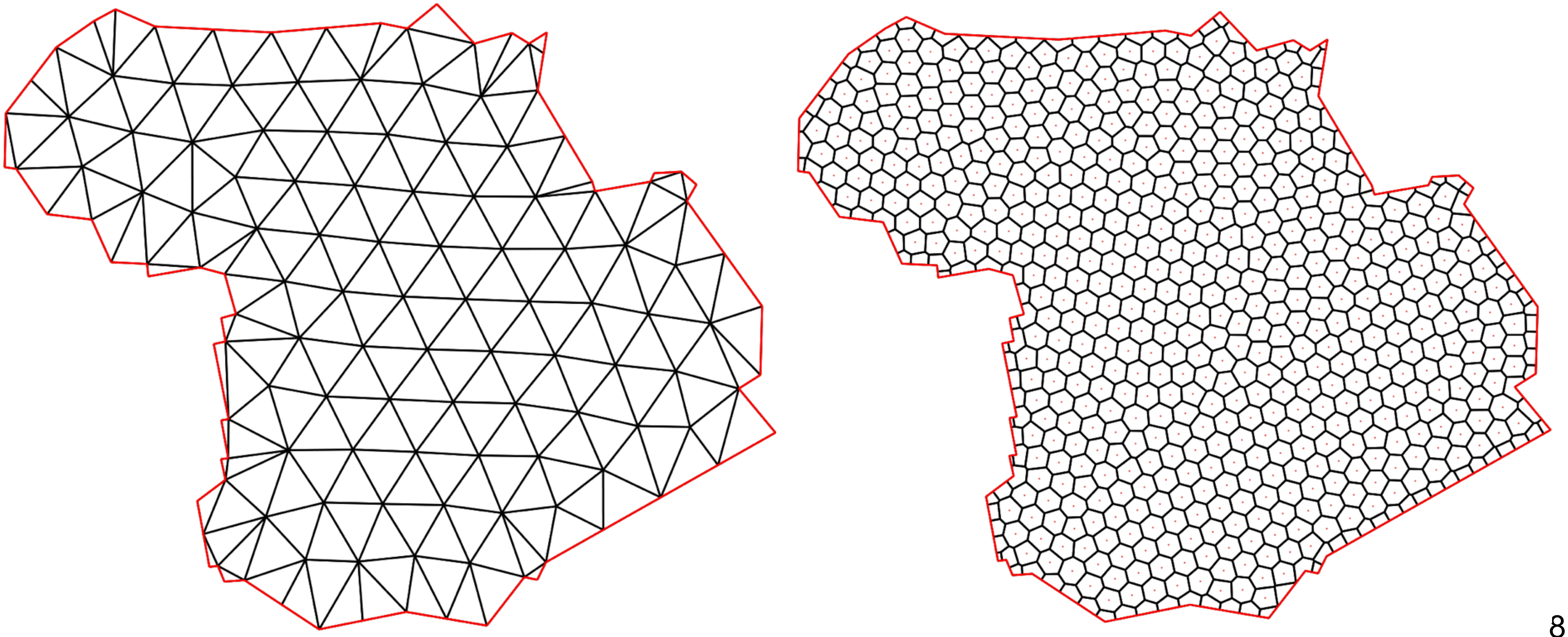
# Centroidal Voronoi Tessellations

- Centroidal Voronoi tessellations (CVTs)
  - Convex & well-shaped elements
- Interleaved refinement & Lloyd relaxation



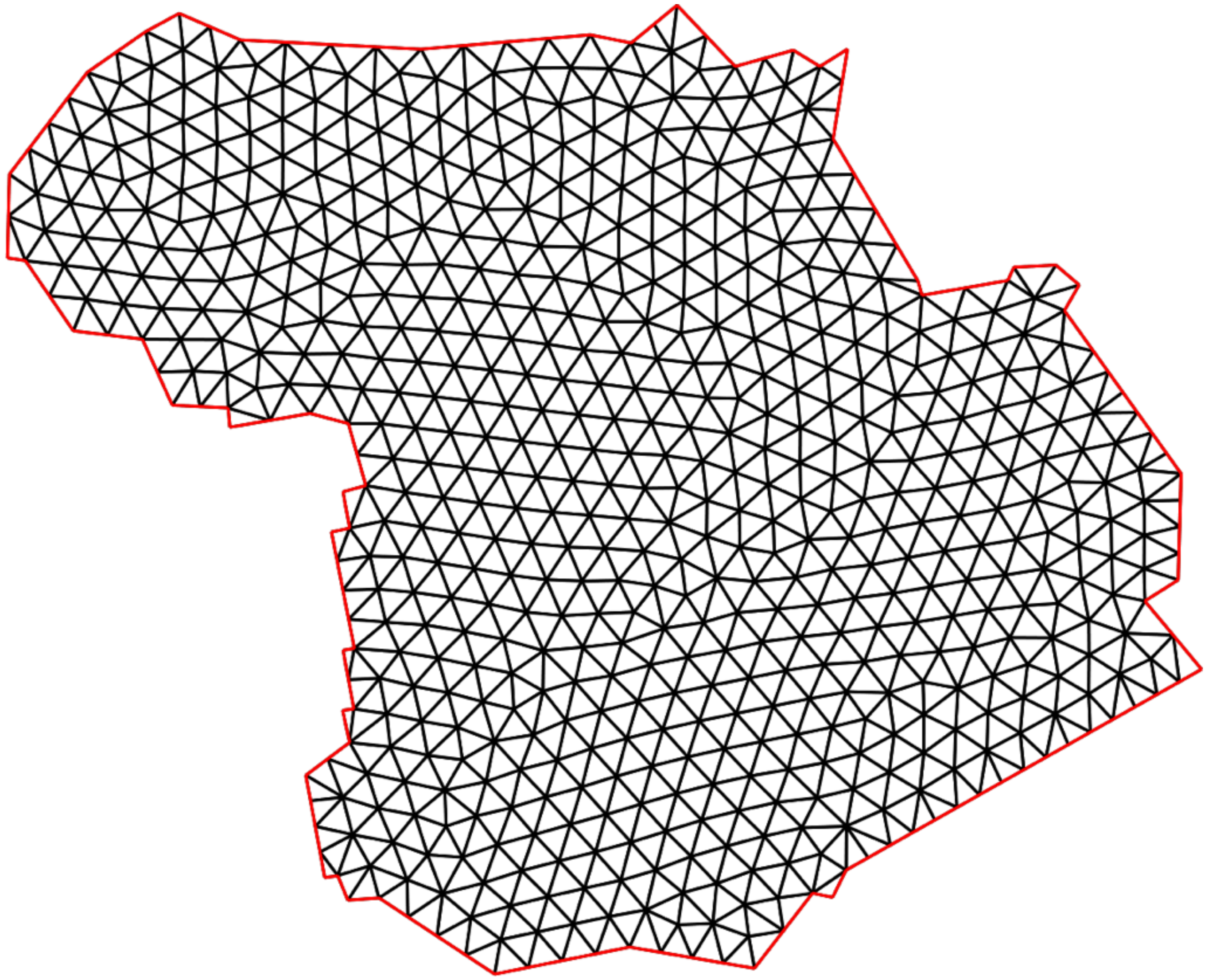
# Centroidal Voronoi Tessellations

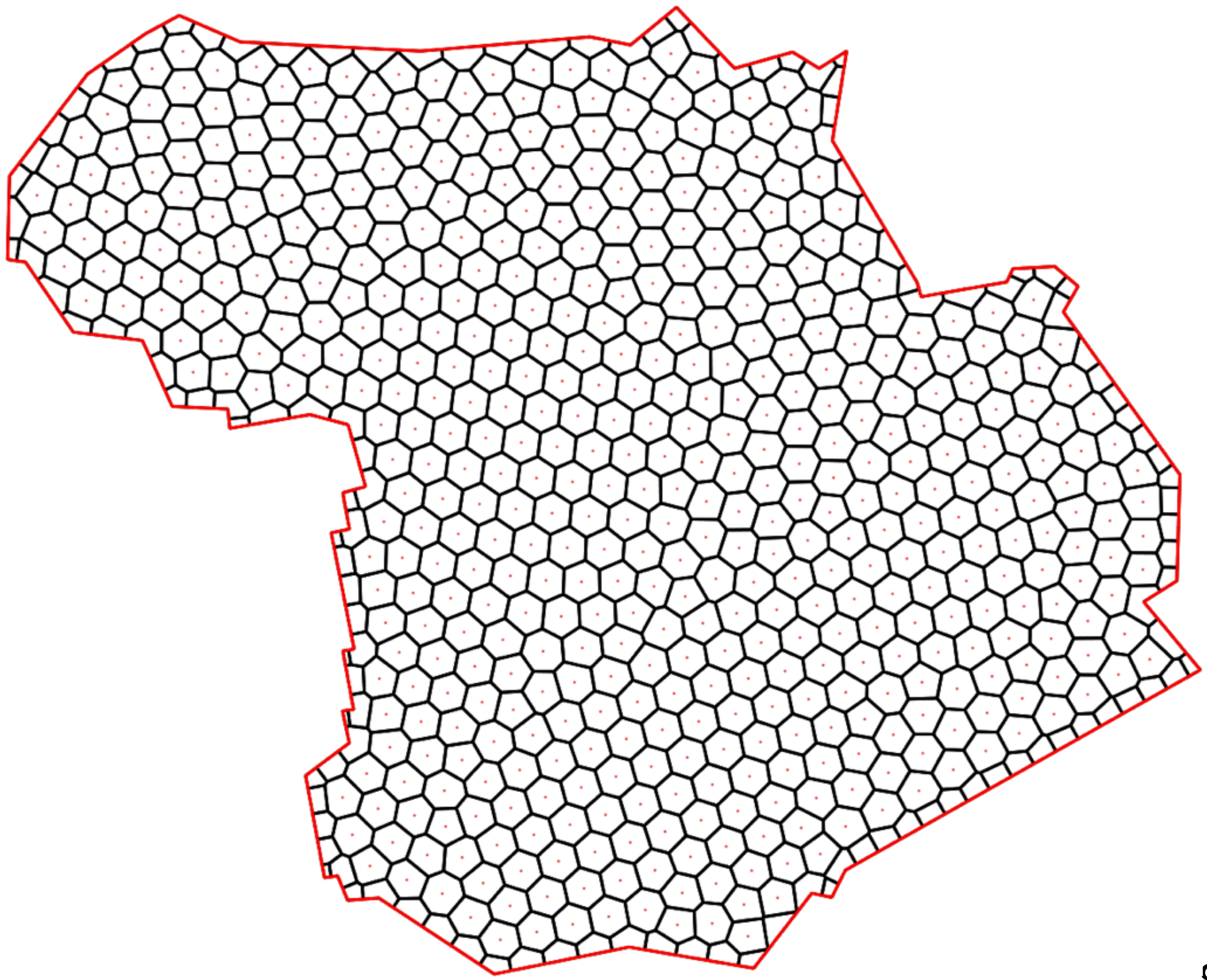
- Centroidal Voronoi tessellations (CVTs)
  - Convex & well-shaped elements
- Interleaved refinement & Lloyd relaxation



# Centroidal Voronoi Tessellations

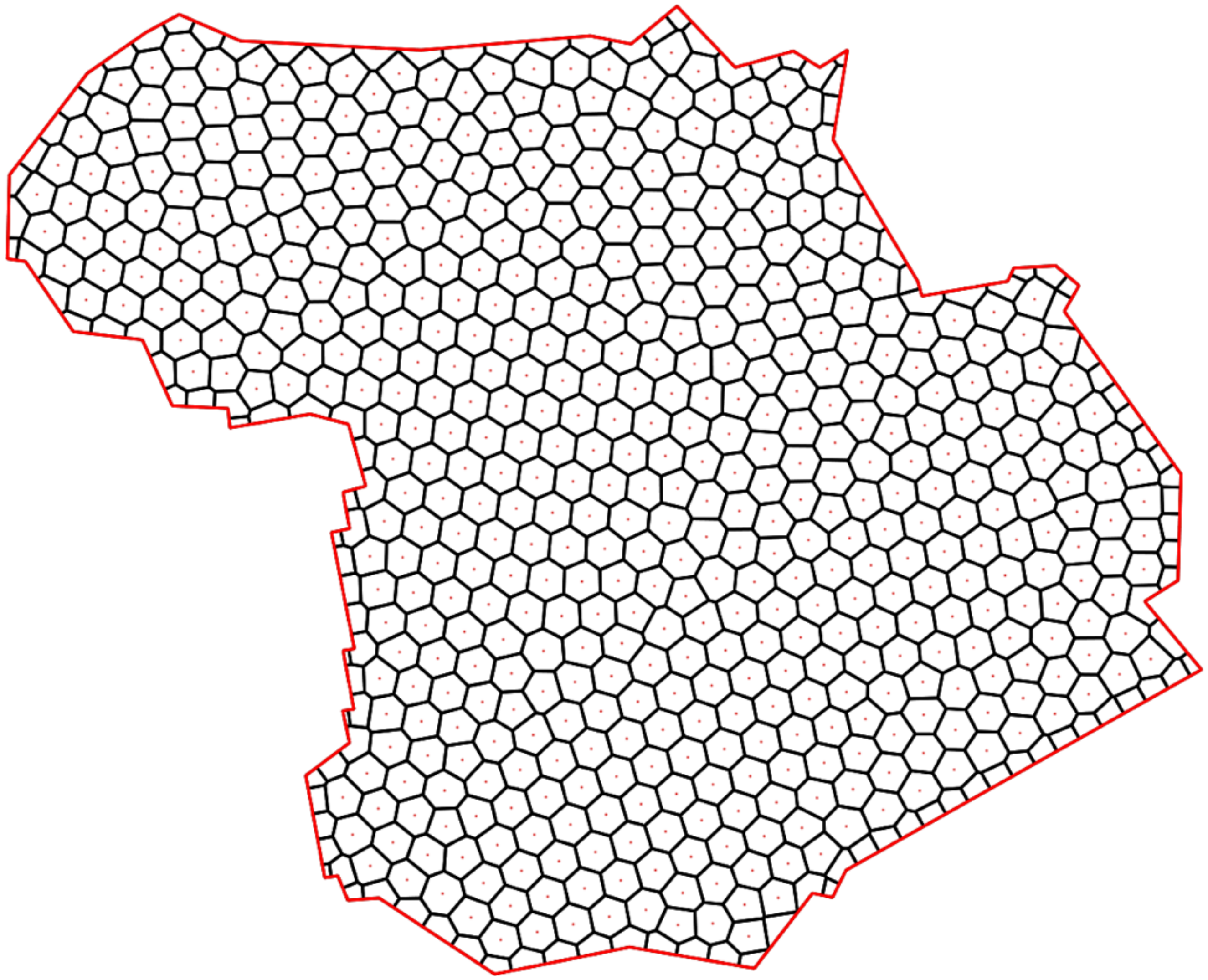
- Centroidal Voronoi tessellations (CVTs)
  - Convex & well-shaped elements
- Interleaved refinement & Lloyd relaxation

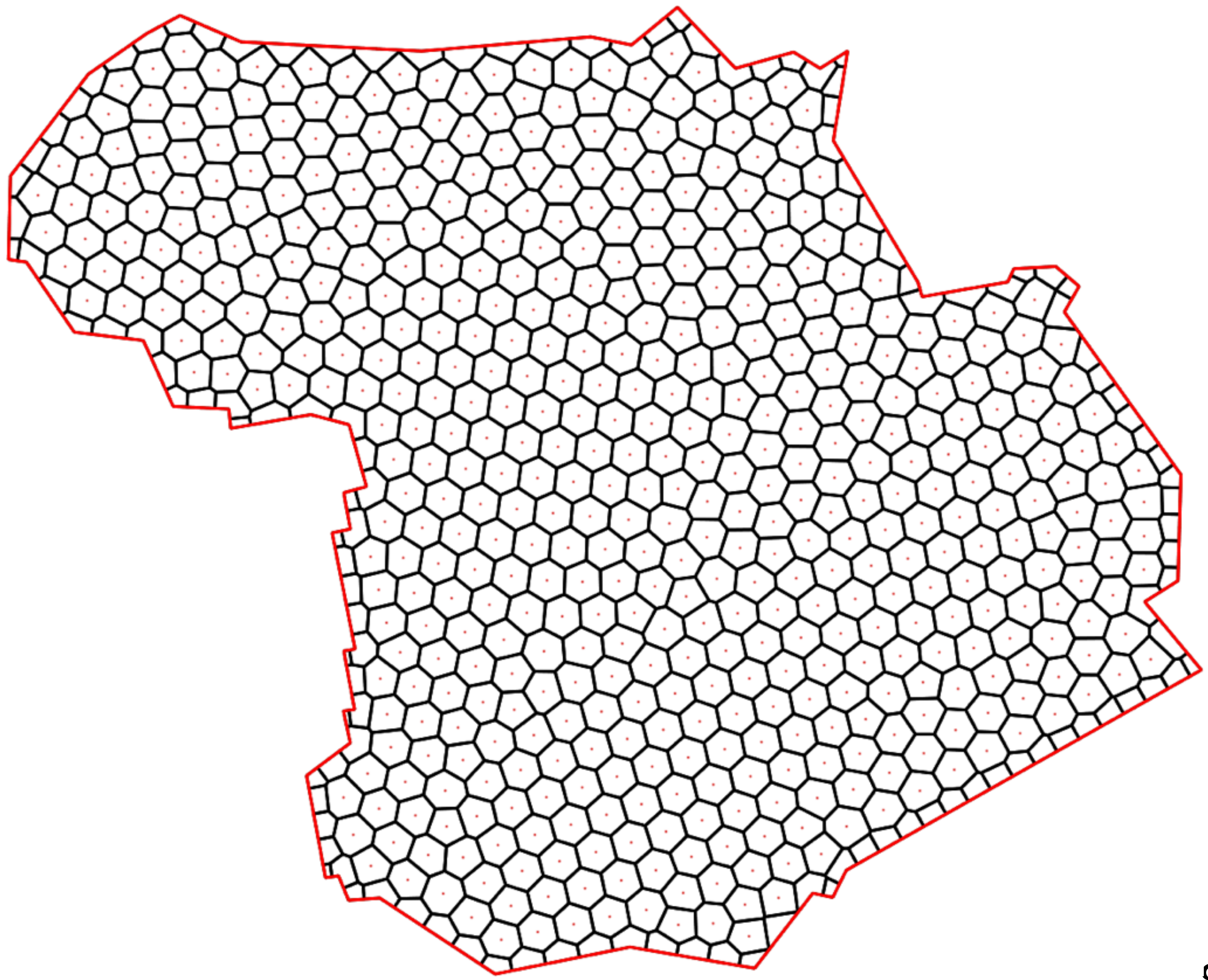




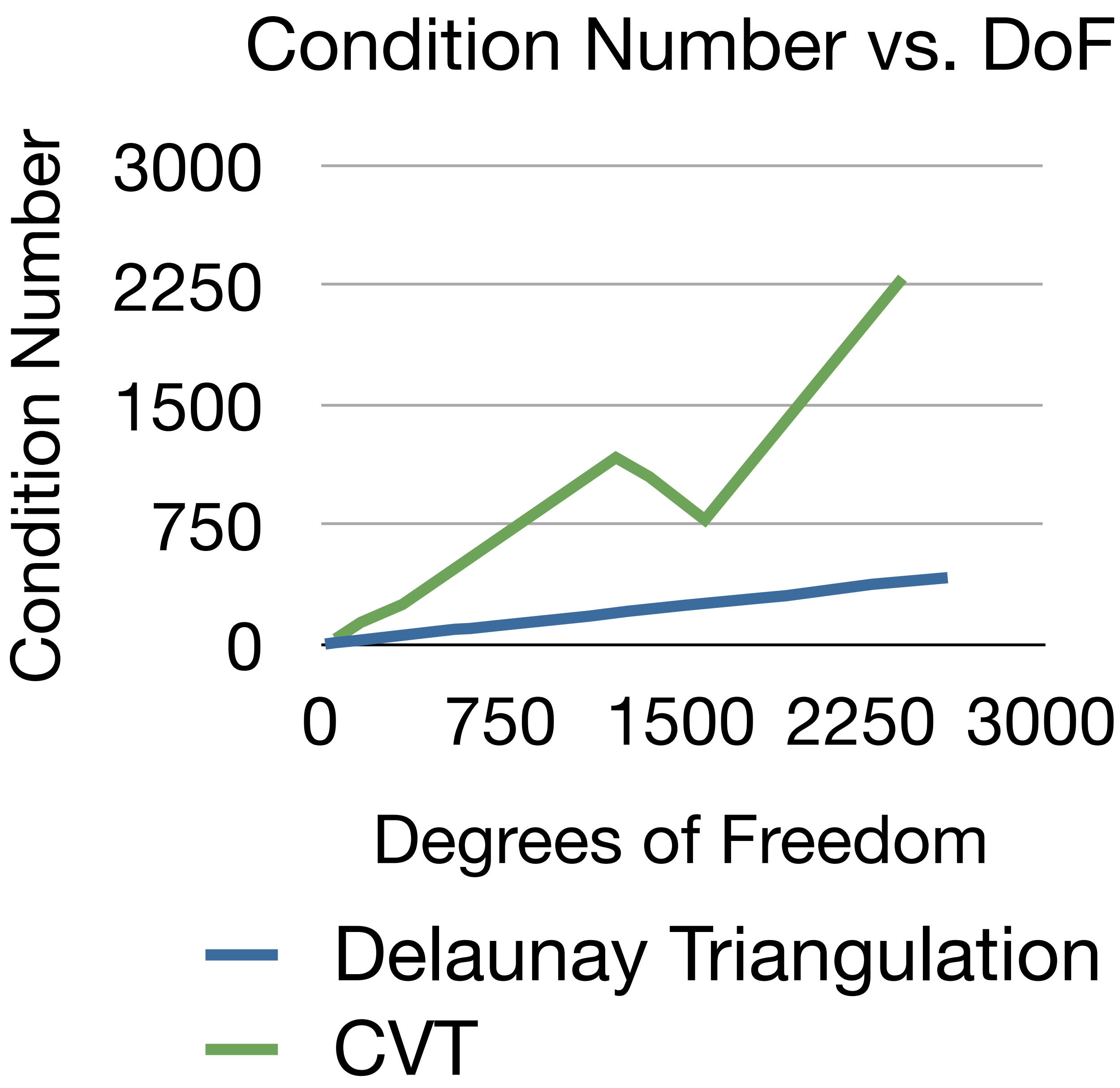
# Centroidal Voronoi Tessellations

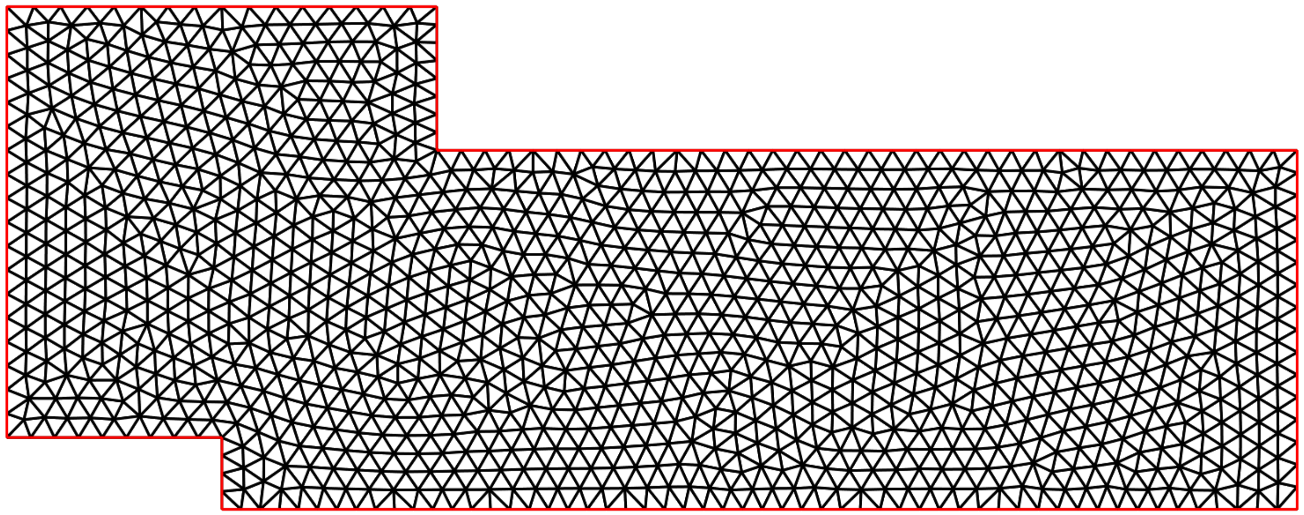
- Centroidal Voronoi tessellations (CVTs)
  - Convex & well-shaped elements
- Interleaved refinement & Lloyd relaxation

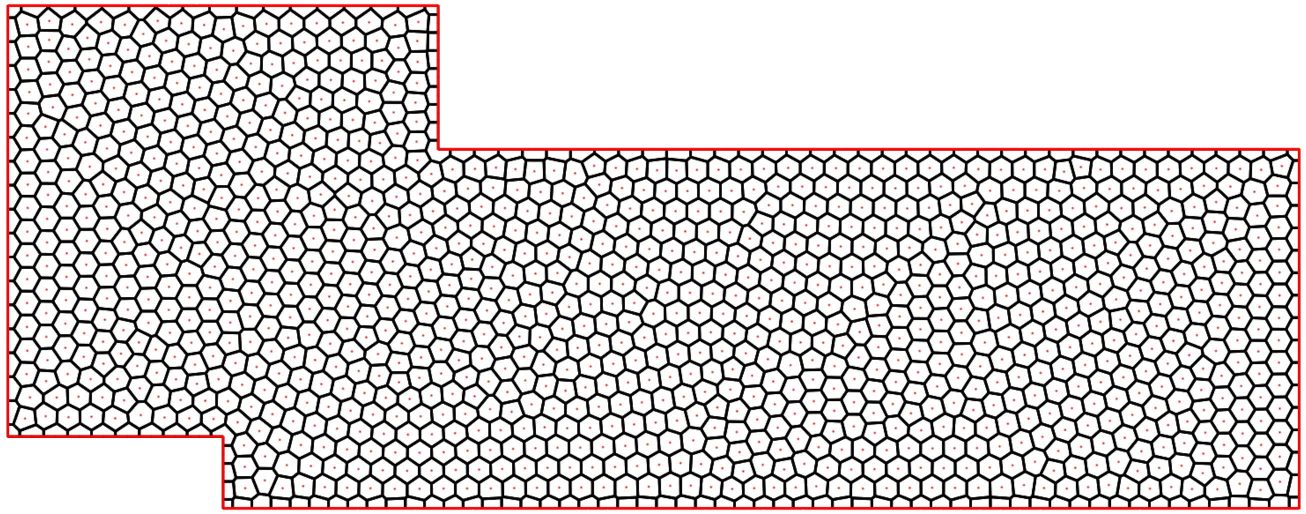




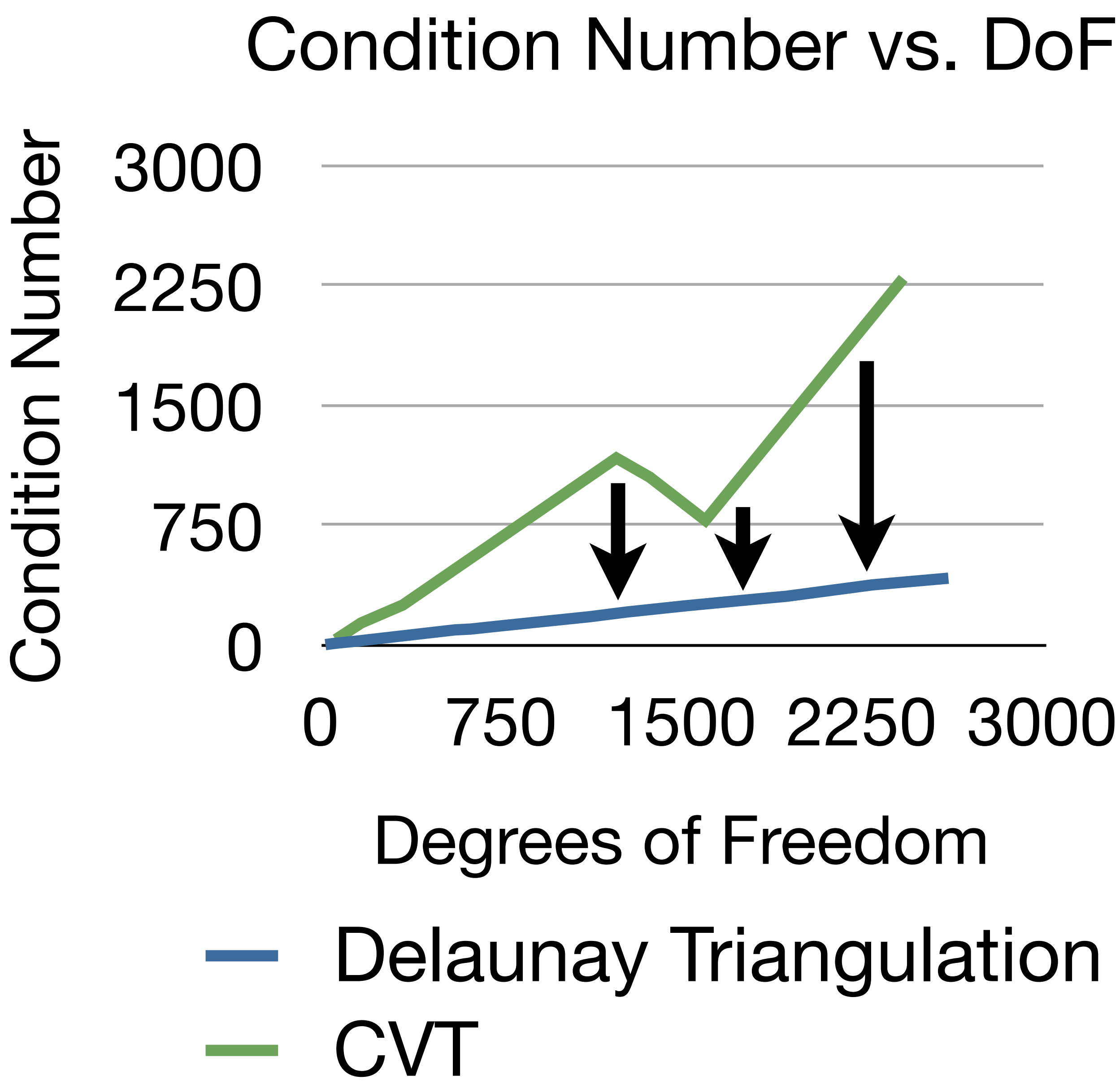
# Experimental Comparison

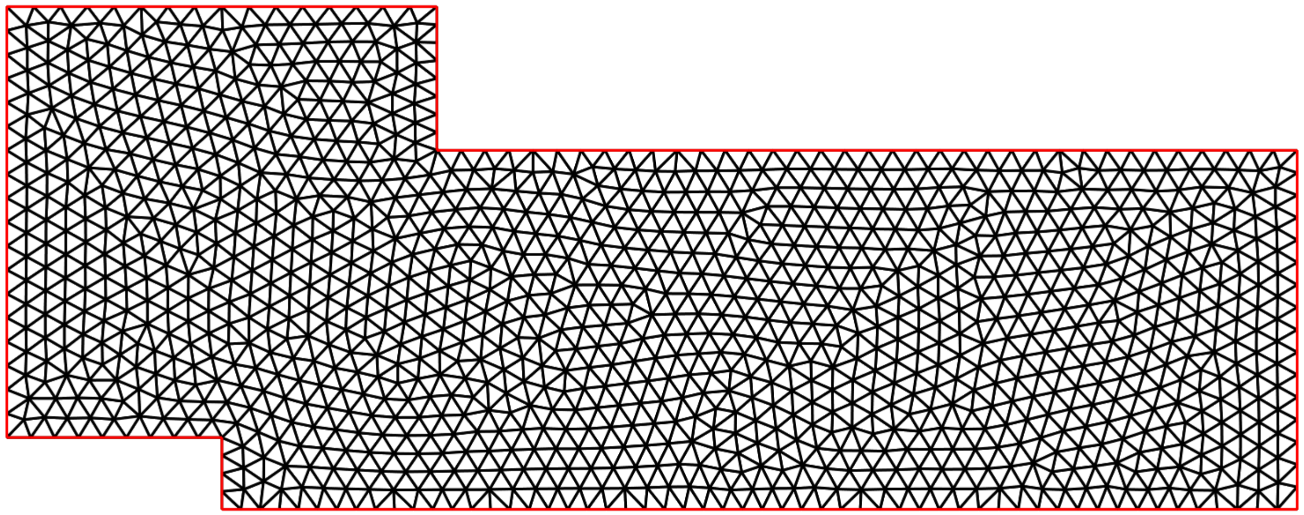


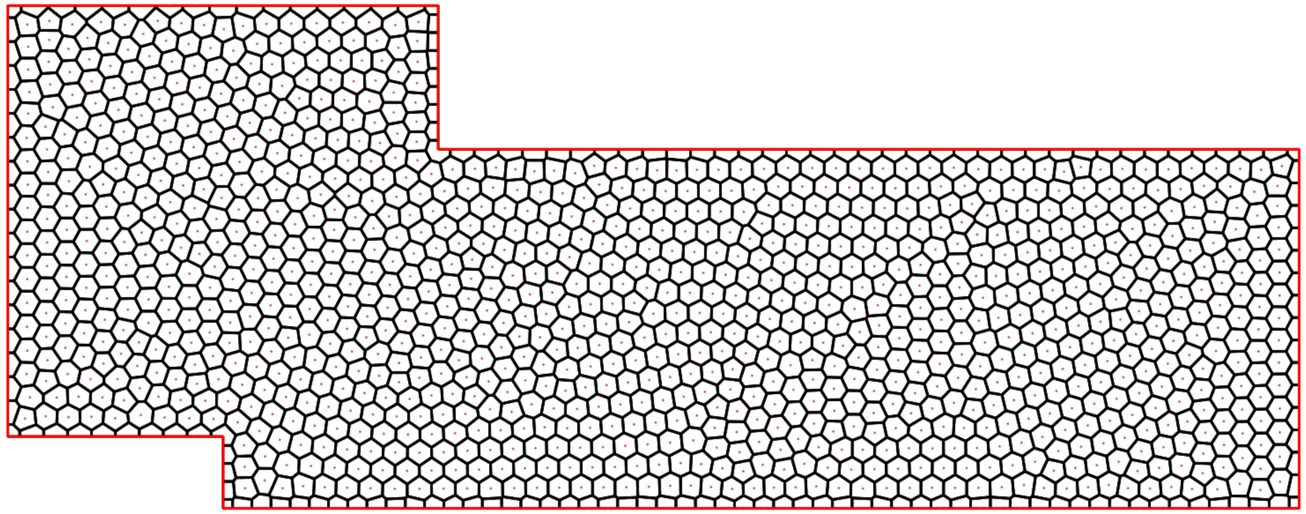





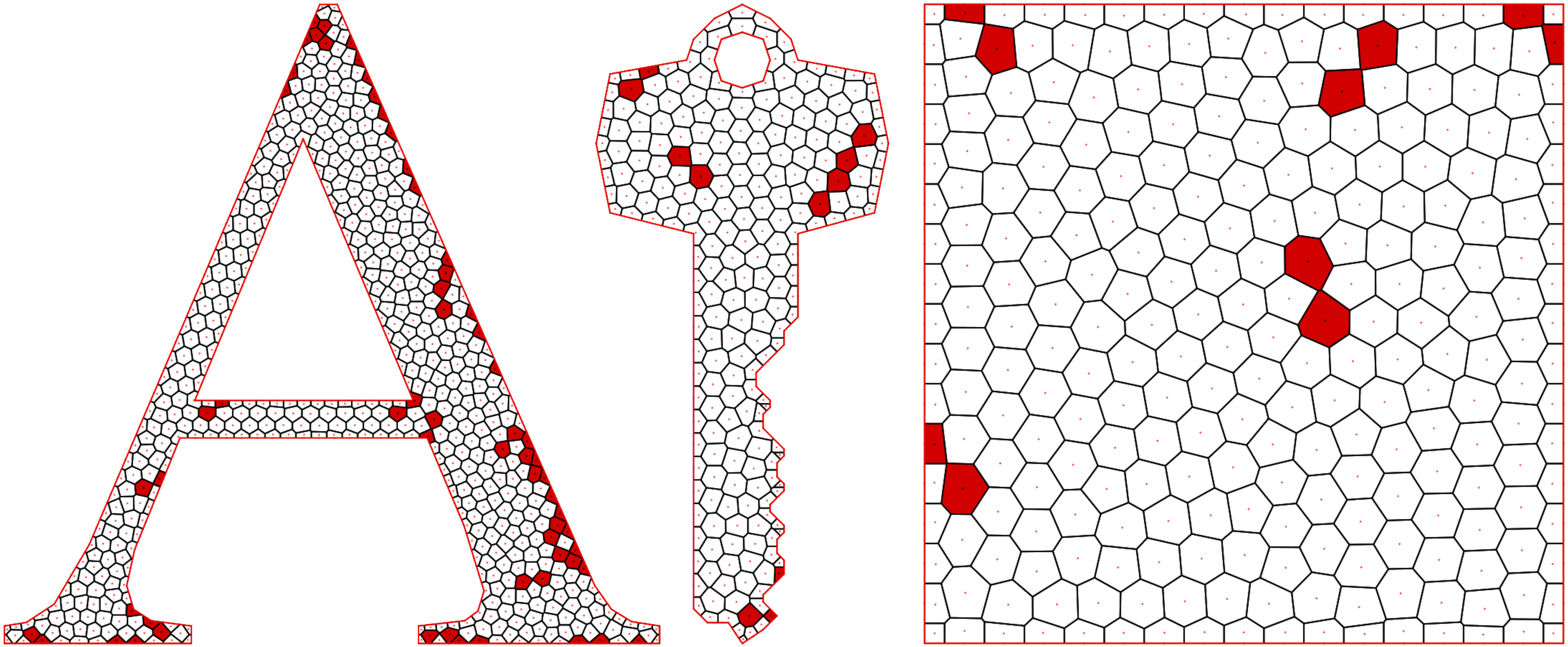
# Experimental Comparison





# Short Edges in CVTs

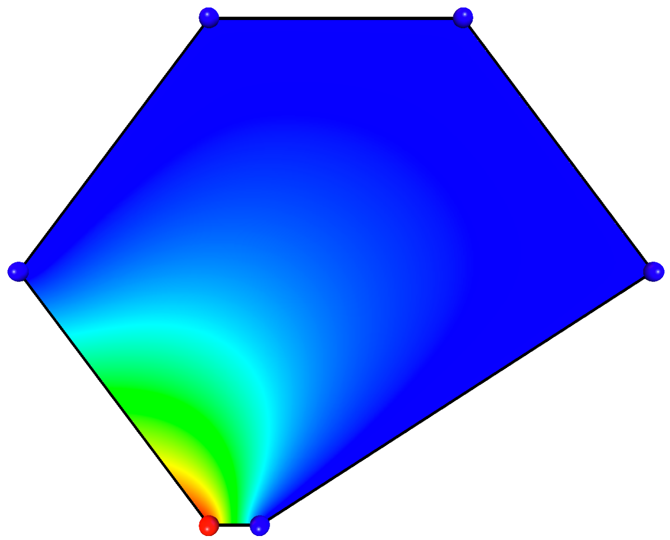


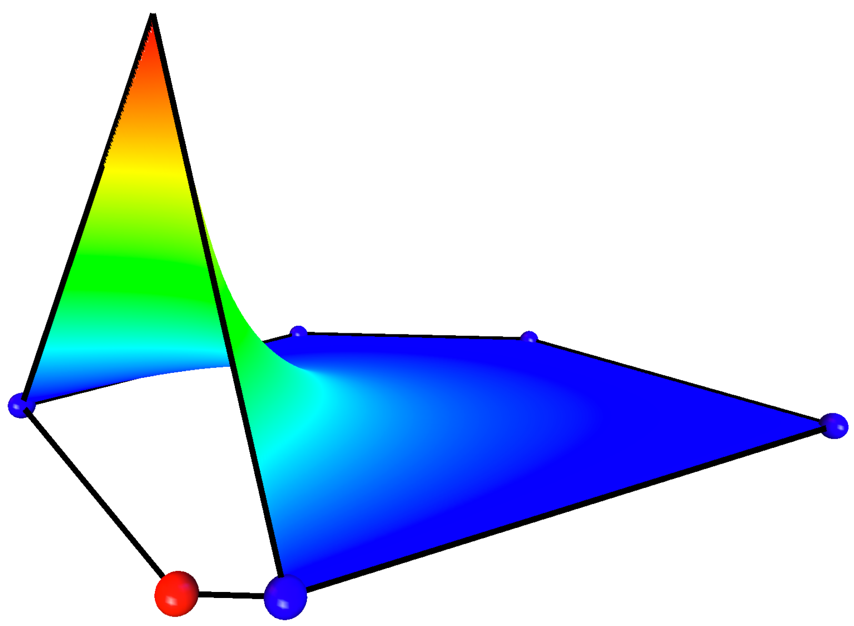
# Short Edges & Conditioning

- Linear interpolation on the edges
- Short edge: Large gradient
- Gradients enter stiffness matrix

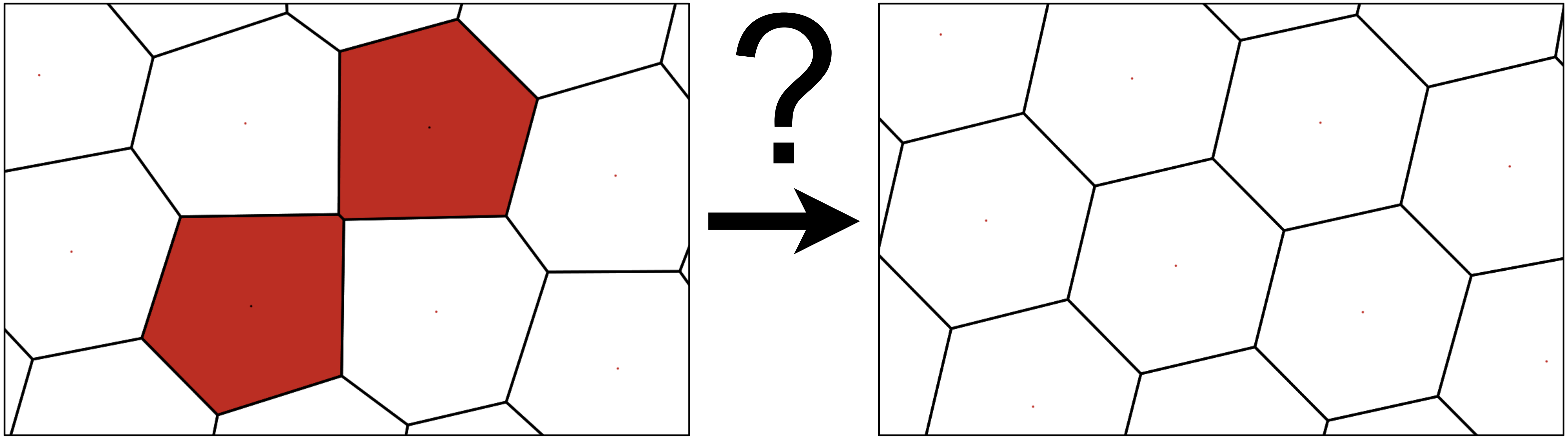
$$\mathbf{K}_{ij} = \int \nabla N_i \cdot \nabla N_j$$

- Large condition number

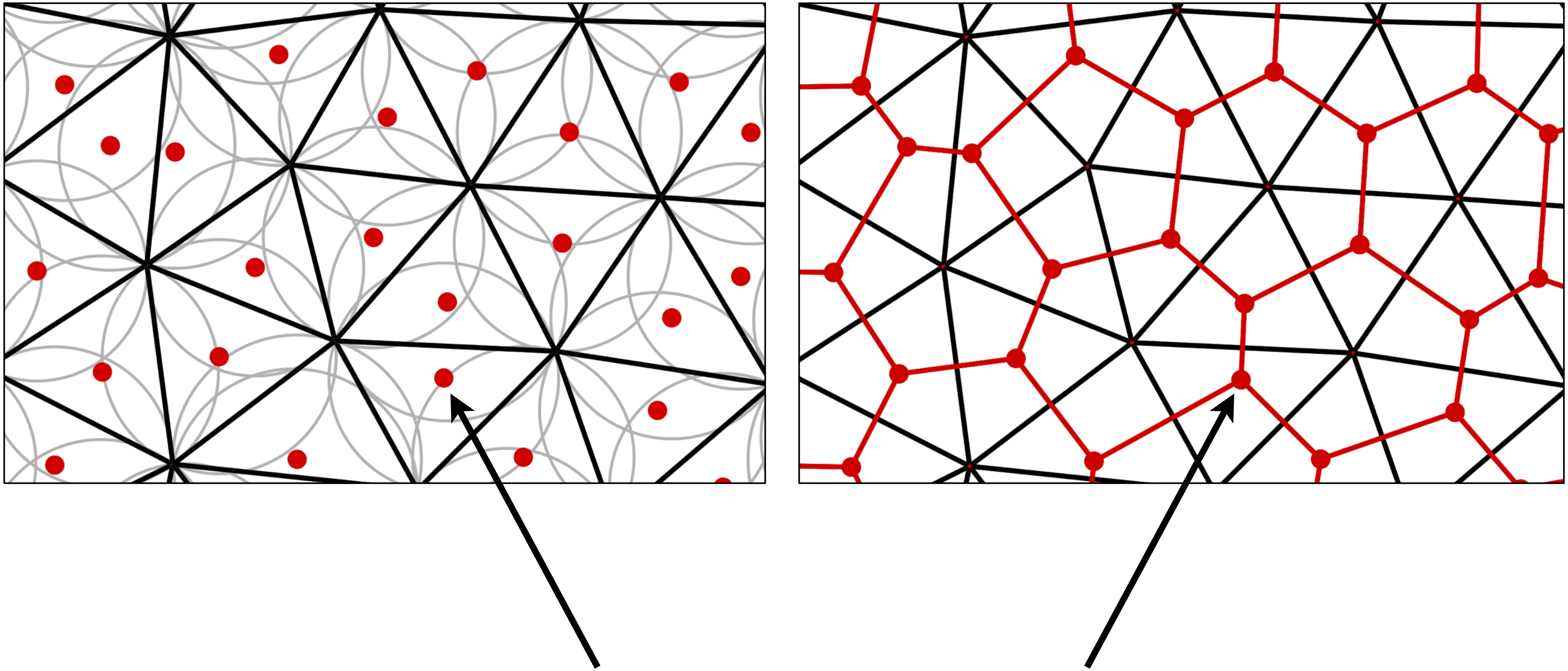




# Mesh Optimization



# Delaunay Triangulation & Voronoi Diagram

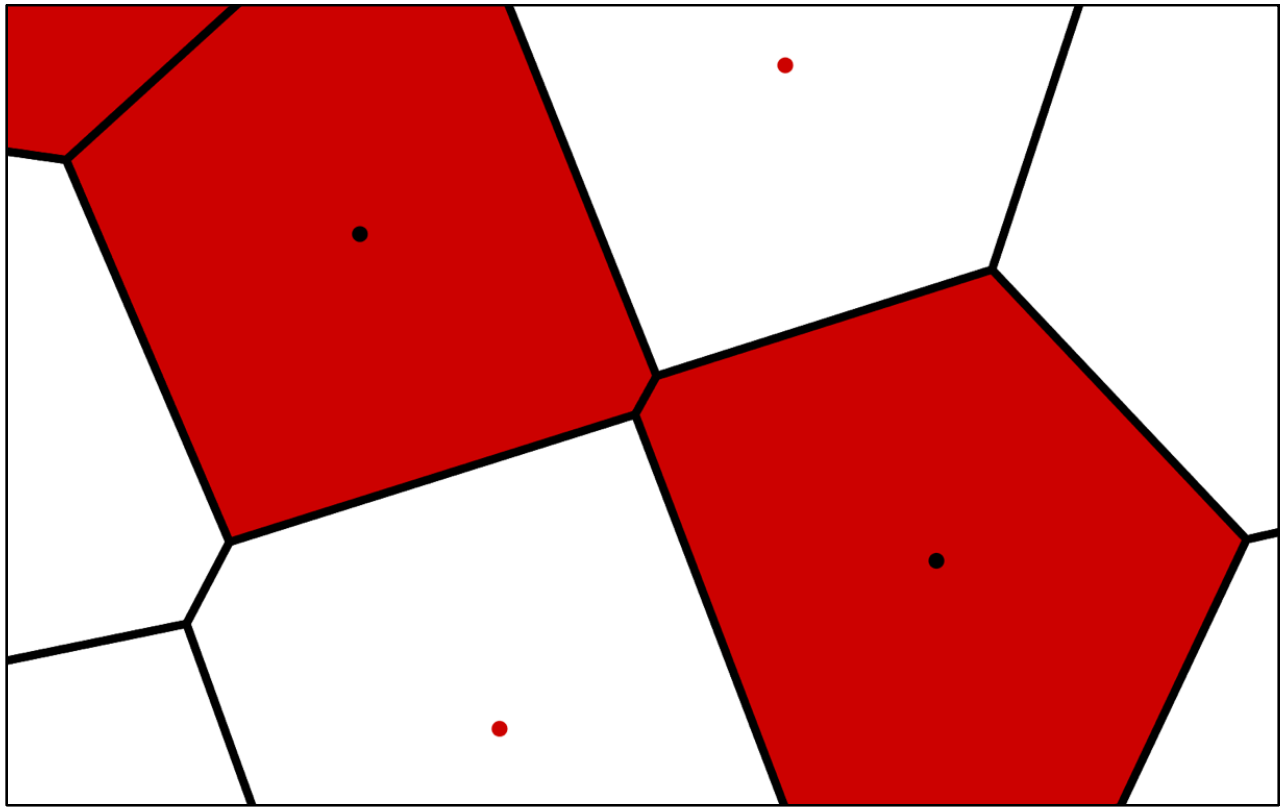


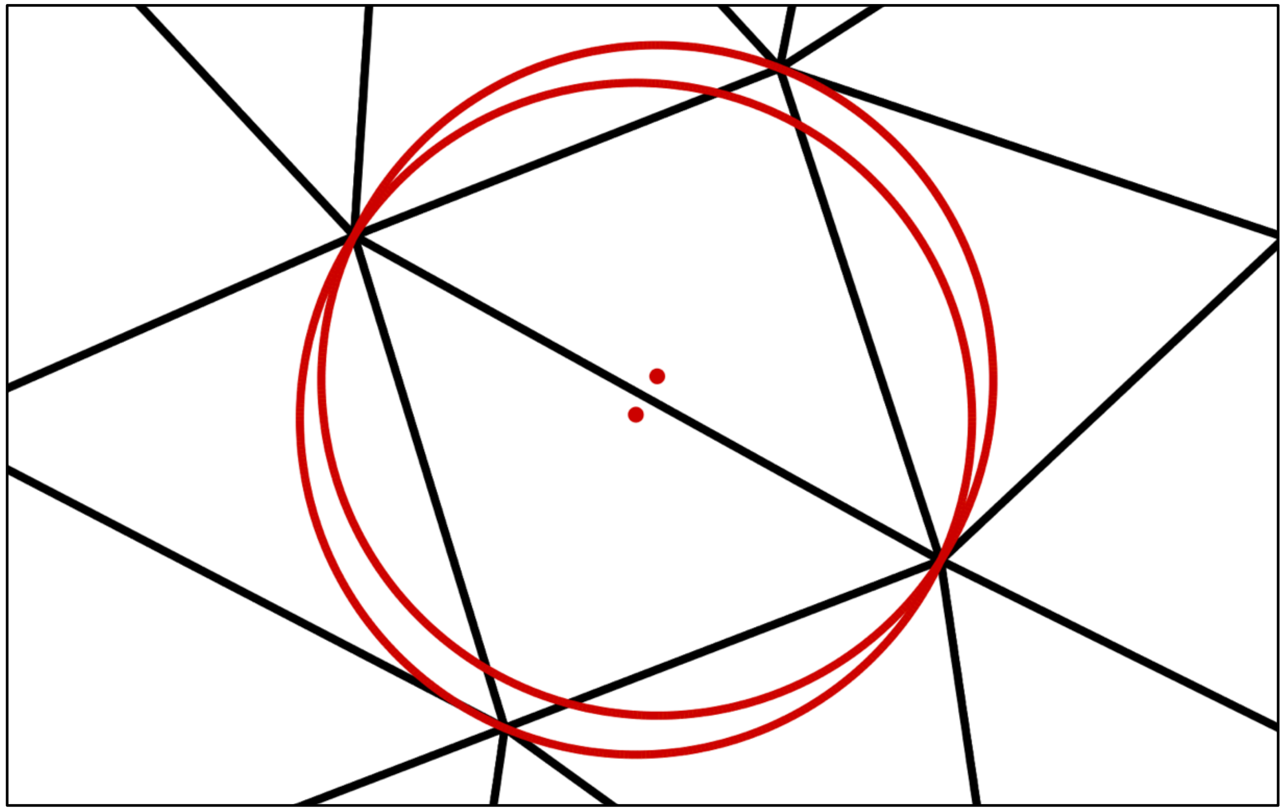
Circumcenter = Voronoi vertex



13

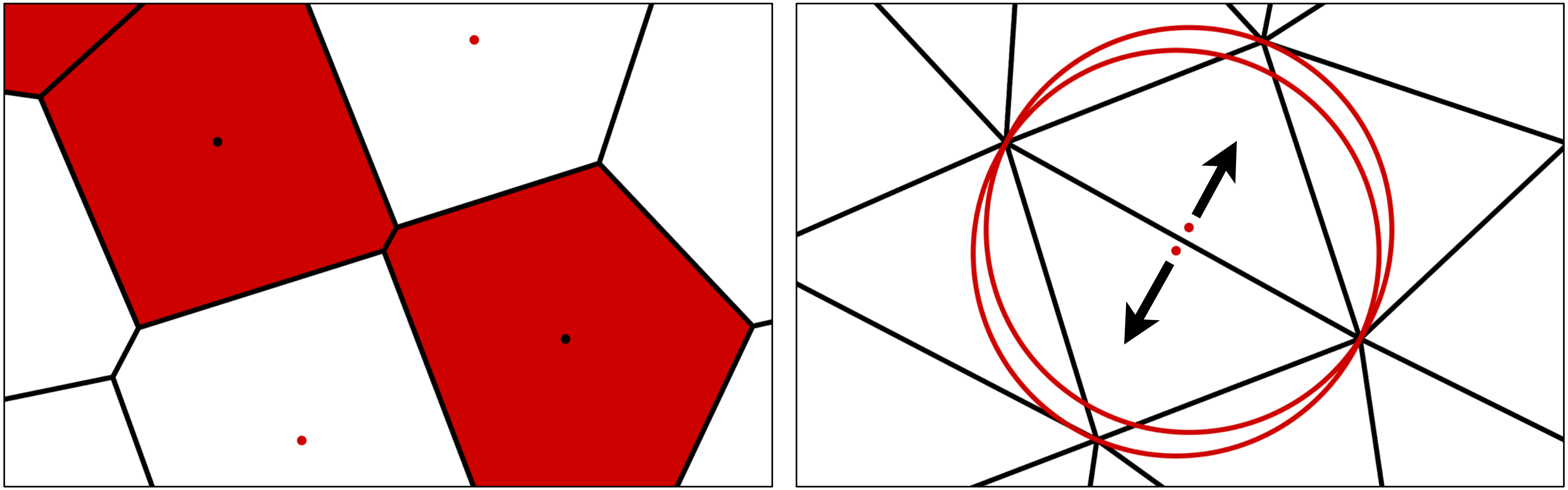
# Short Edges in CVTs





Short edge = spatially close circumcenters

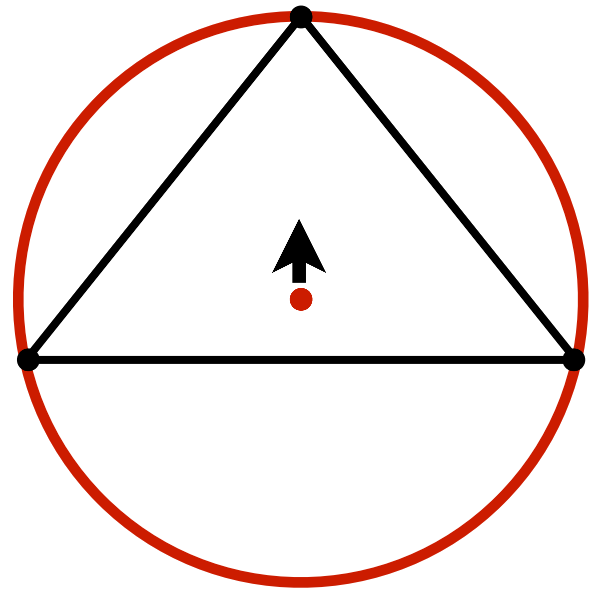
# Short Edges in CVTs



Short edge = spatially close circumcenters

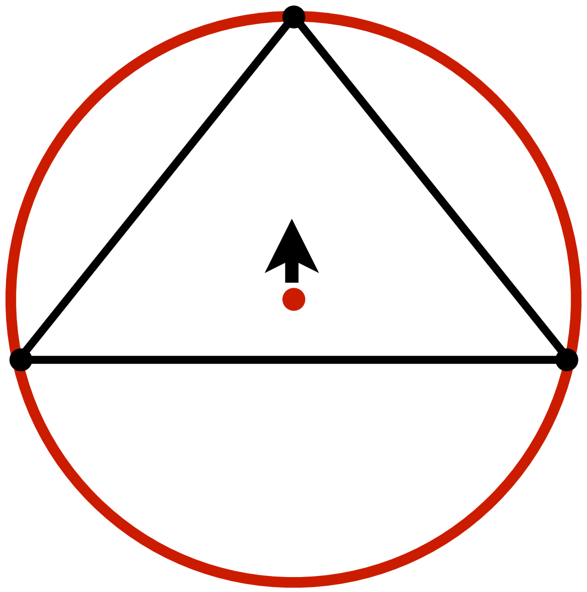
# Key Idea

- Push circumcenter as far into the triangle as possible



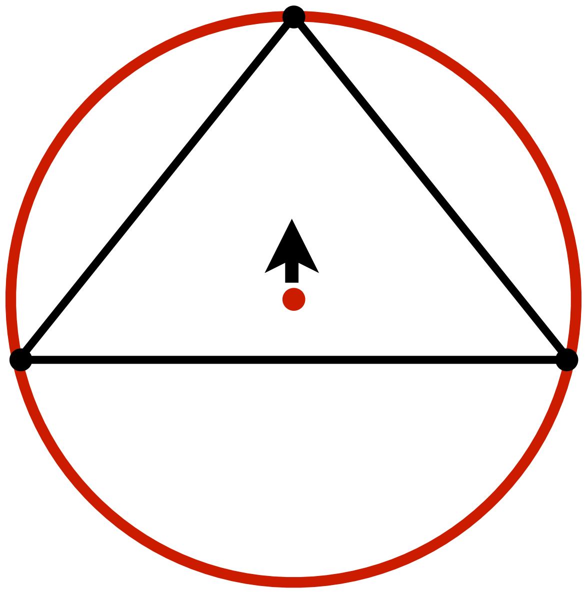
# Key Idea

- Push circumcenter as far into the triangle as possible

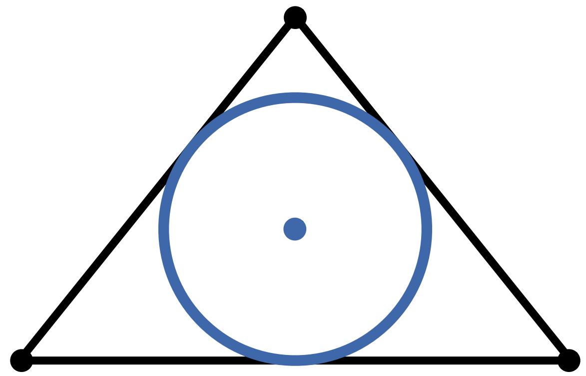


# Key Idea

- Push circumcenter as far into the triangle as possible

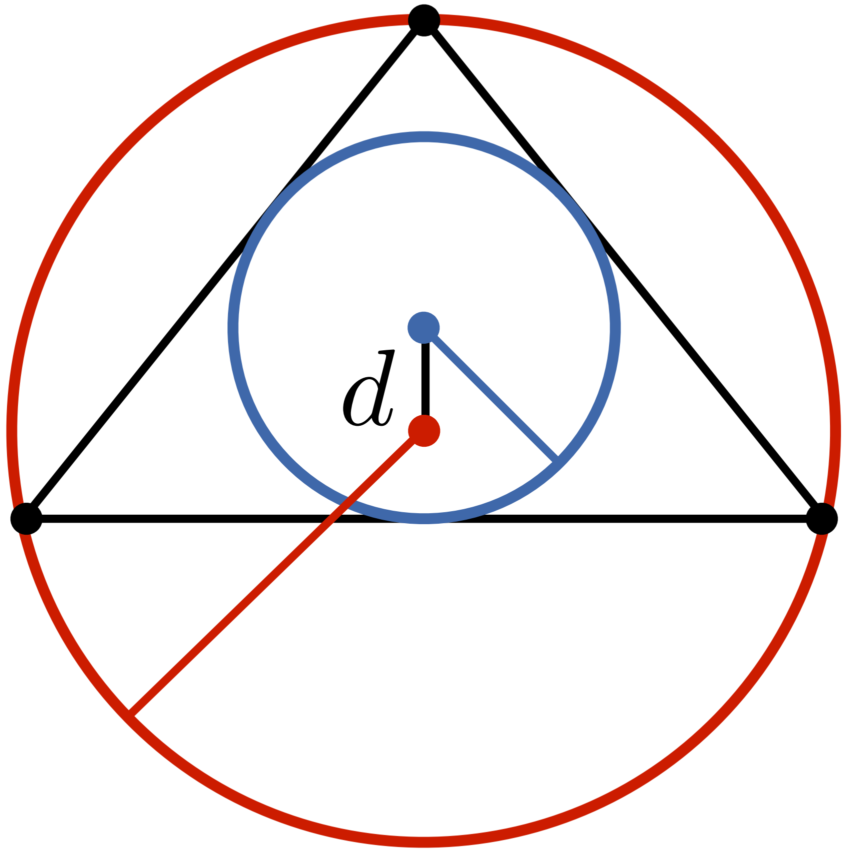


- “As interior as possible”: Incenter



# Energy Setup

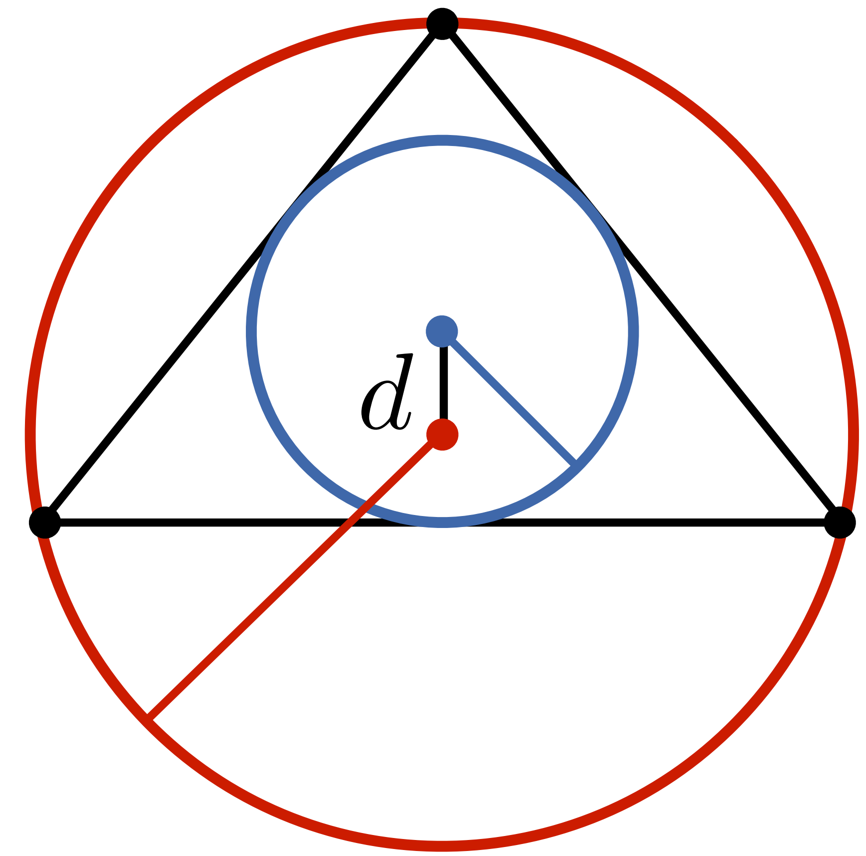
- Energy depending on Delaunay vertices $\mathbf{v}_i$:



# Energy Setup

- Energy depending on Delaunay vertices $\mathbf{v}_i$:

$$E(\mathbf{v}_1, \ldots, \mathbf{v}_m) \;=\; \frac{1}{2} \sum_{t \in \mathcal{T}} d_t^2$$



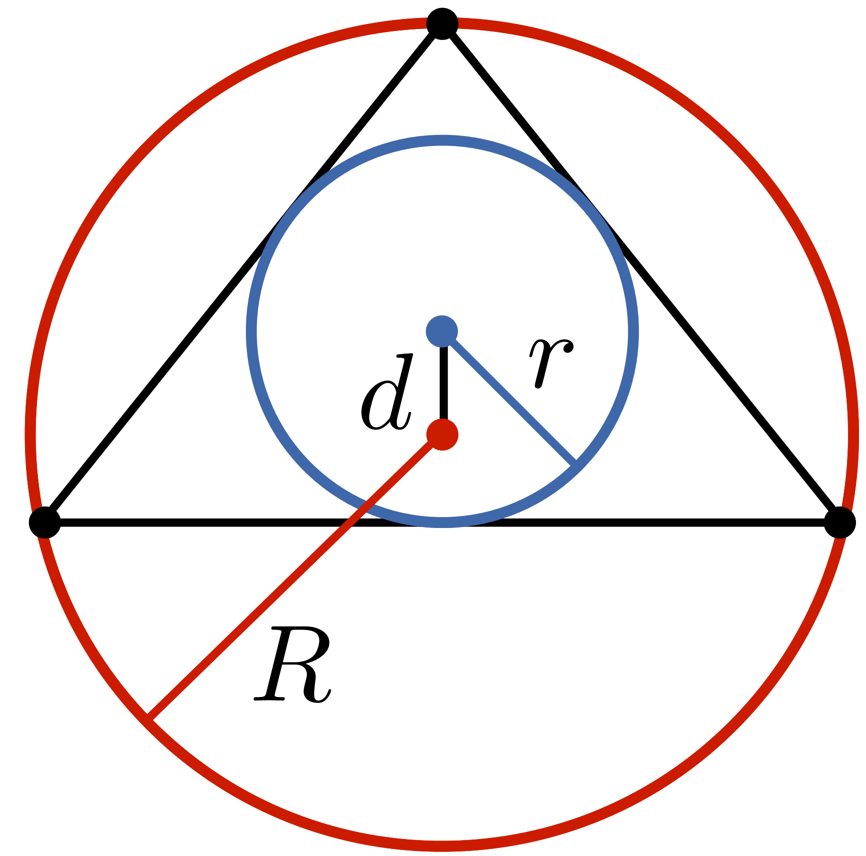
# Energy Setup

- Energy depending on Delaunay vertices $\mathbf{v}_i$:

$$E(\mathbf{v}_1, \ldots, \mathbf{v}_m) = \frac{1}{2} \sum_{t \in \mathcal{T}} d_t^2$$

- Euler's triangle formula:

$$d^2 = R(R - 2r)$$



# Energy Setup

- Energy depending on Delaunay vertices $\mathbf{v}_i$:

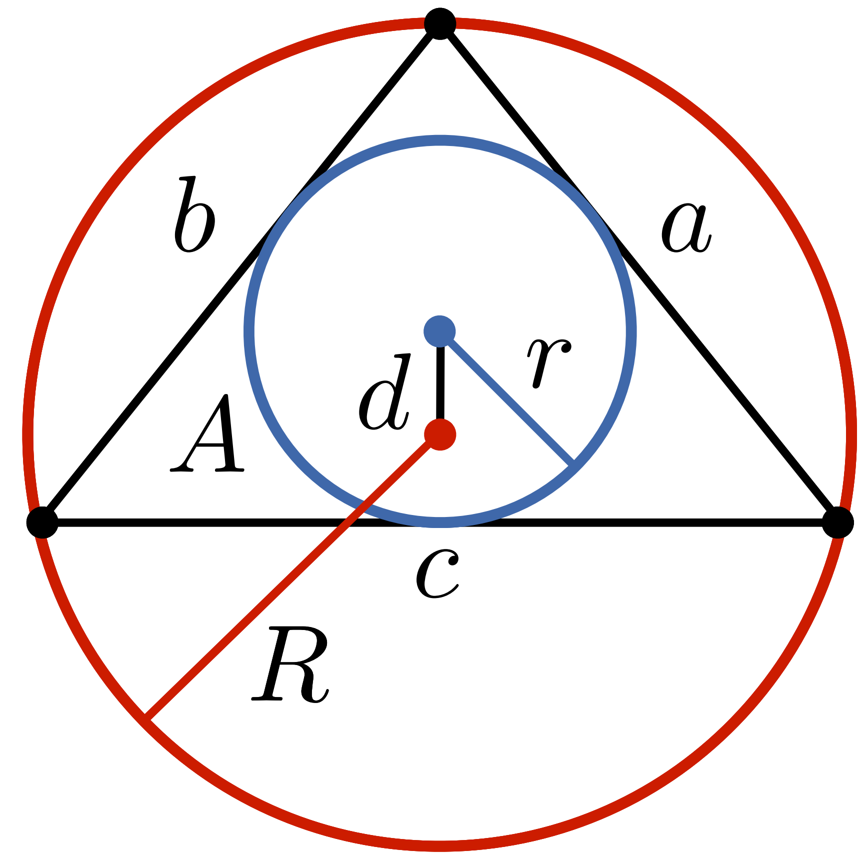
$$E(\mathbf{v}_1, \ldots, \mathbf{v}_m) \;=\; \frac{1}{2} \sum_{t \in \mathcal{T}} d_t^2$$

- Euler's triangle formula:

$$d^2 = R(R - 2r)$$

- Circumcircle & incircle radius:

$$R \;=\; \frac{abc}{4A} \quad \text{and} \quad r \;=\; \frac{2A}{a+b+c}$$



# Energy Minimization

- Global optimization
- DoF: Vector $\mathbf{V}$ of Delaunay vertex positions
- Iterative gradient descent:

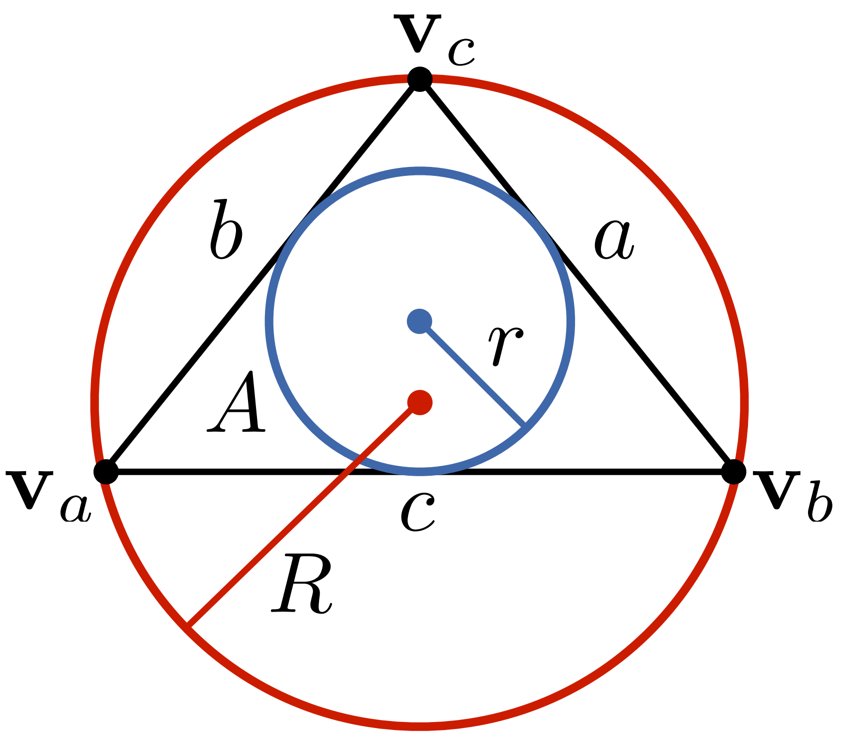
$$\mathbf{V}^{(k+1)} \leftarrow \mathbf{V}^{(k)} - h \nabla E(\mathbf{V}^{(k)})$$

- Determine step-size $h$ using bisection


17

# Gradient Computation

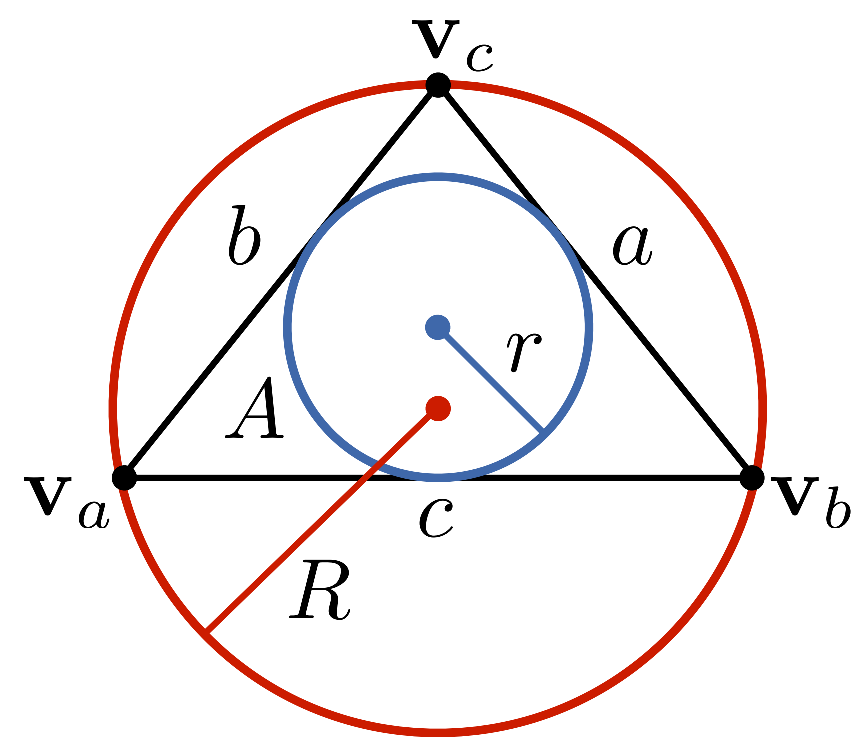
- Gradient of $E$:



# Gradient Computation

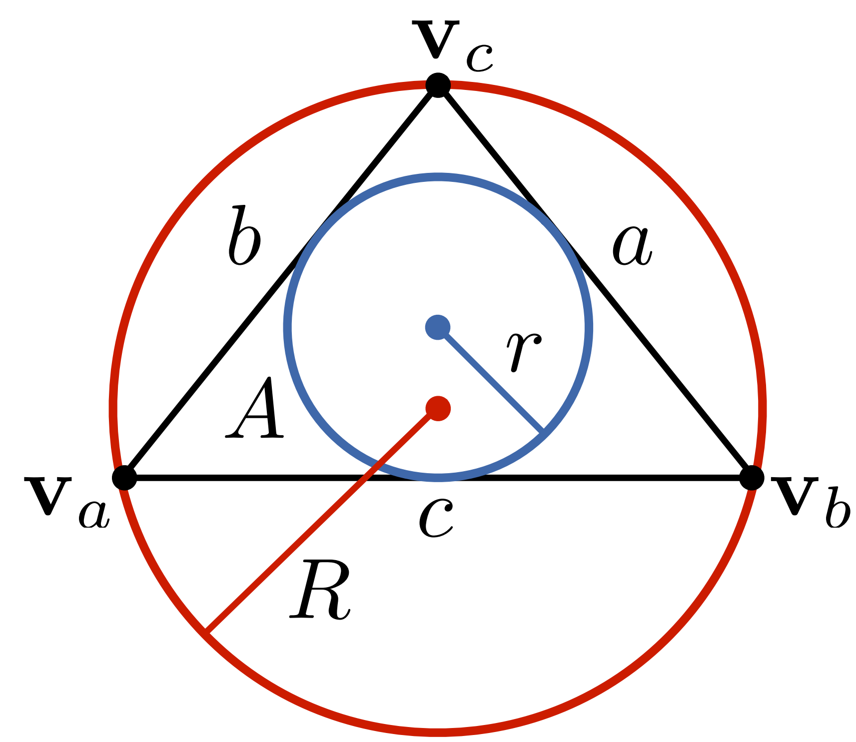
- Gradient of $E$:

$$\frac{\partial E}{\partial \mathbf{v}_a} = \sum_{t \in \mathcal{T}} \left[ (R_t - r_t) \frac{\partial R_t}{\partial \mathbf{v}_a} - R_t \frac{\partial r_t}{\partial \mathbf{v}_a} \right]$$



# Gradient Computation

- Gradient of $E$:




$$\frac{\partial E}{\partial \mathbf{v}_a} = \sum_{t \in \mathcal{T}} \left[ (R_t - r_t) \frac{\partial R_t}{\partial \mathbf{v}_a} - R_t \frac{\partial r_t}{\partial \mathbf{v}_a} \right]$$

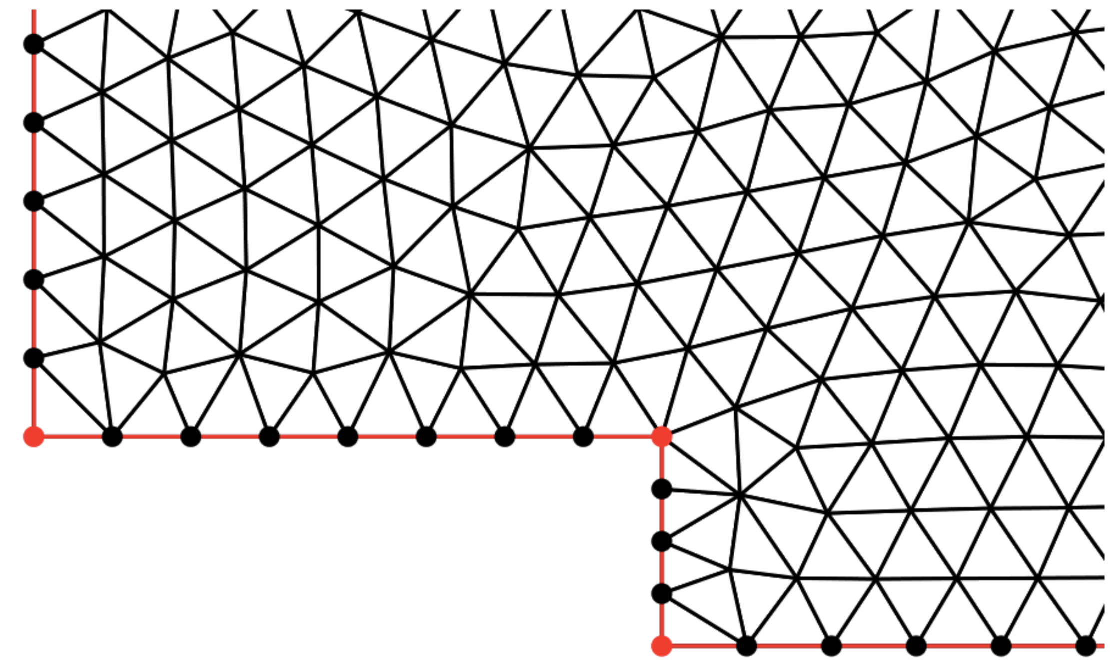
- Circumcircle & incircle radius gradients:

$$\frac{\partial R}{\partial \mathbf{v}_a} = R \left[ \frac{1}{c^2} (\mathbf{v}_a - \mathbf{v}_b) + \frac{1}{b^2} (\mathbf{v}_a - \mathbf{v}_c) - \frac{1}{2A} (\mathbf{v}_c - \mathbf{v}_b)^{\perp} \right]$$

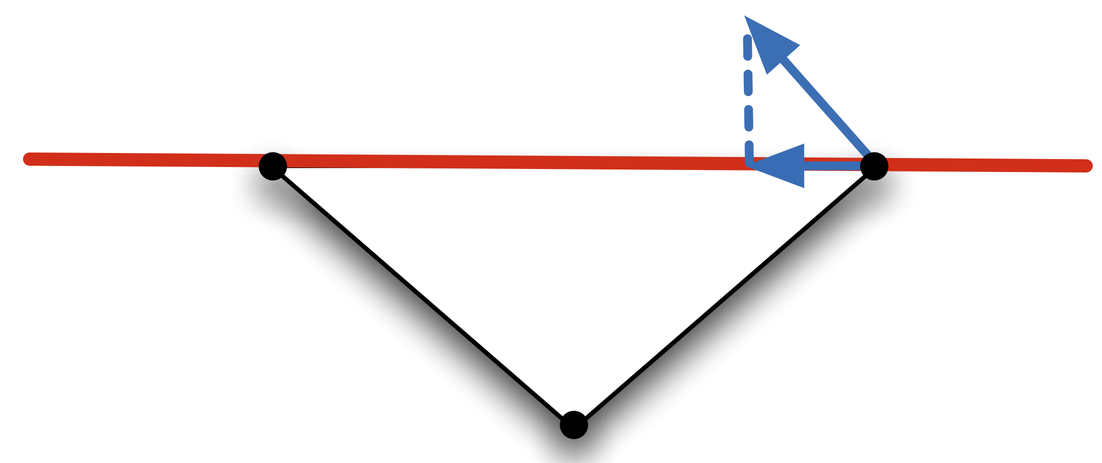
$$\frac{\partial r}{\partial \mathbf{v}_a} = \frac{-2}{(a+b+c)^2} \left[ \frac{A}{c} (\mathbf{v}_a - \mathbf{v}_b) + \frac{A}{b} (\mathbf{v}_a - \mathbf{v}_c) - \frac{a+b+c}{2} (\mathbf{v}_c - \mathbf{v}_b)^{\perp} \right]$$

# Boundary Treatment I

- Boundary:
  - Keep original input vertices fixed
  - Optimize refined vertices



- Project gradients to straight line segment

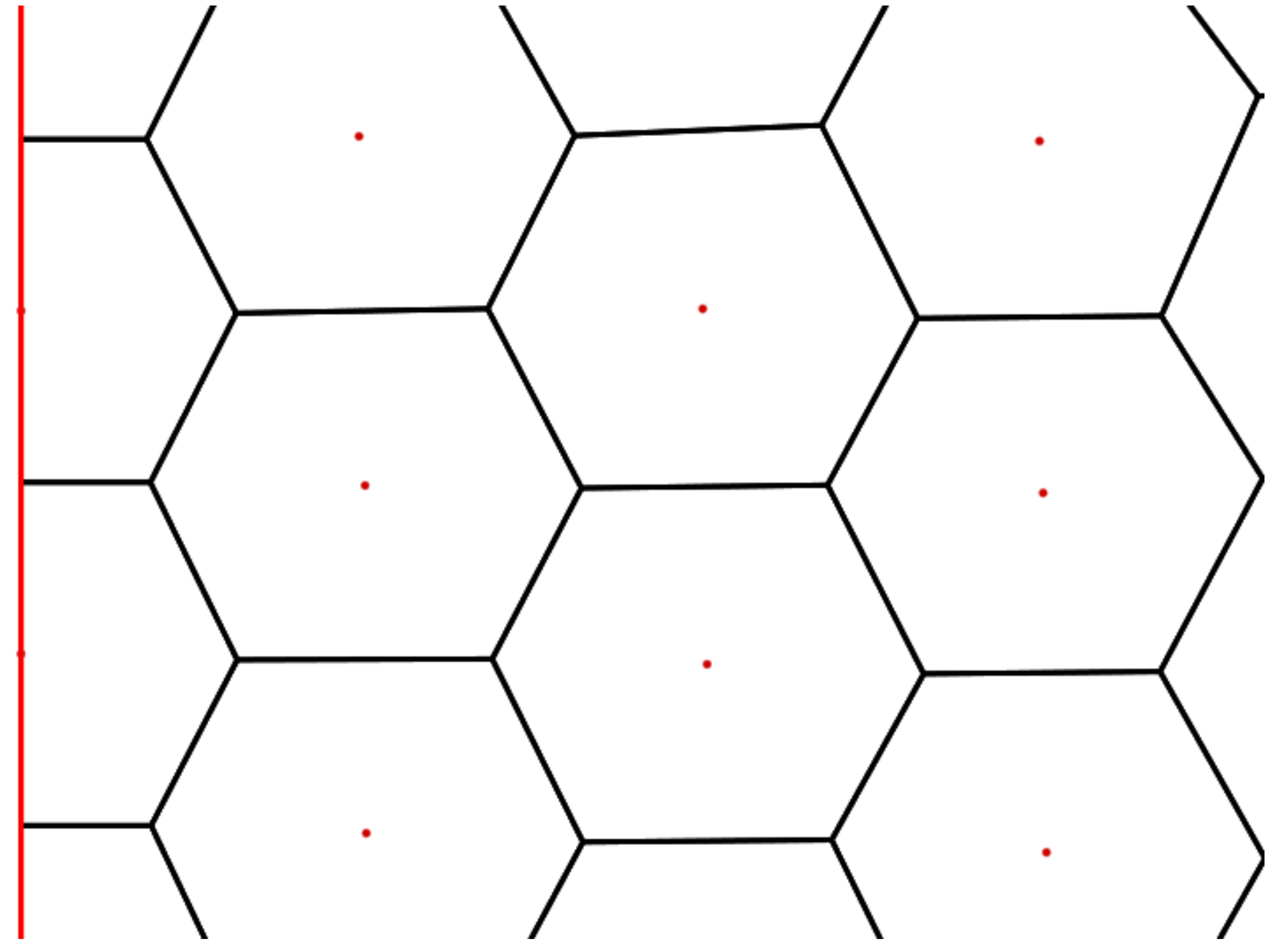


# Boundary Treatment II

- Twice the weight for boundary triangles:
  - Short edges due to truncation
  - Conforming Gabriel property

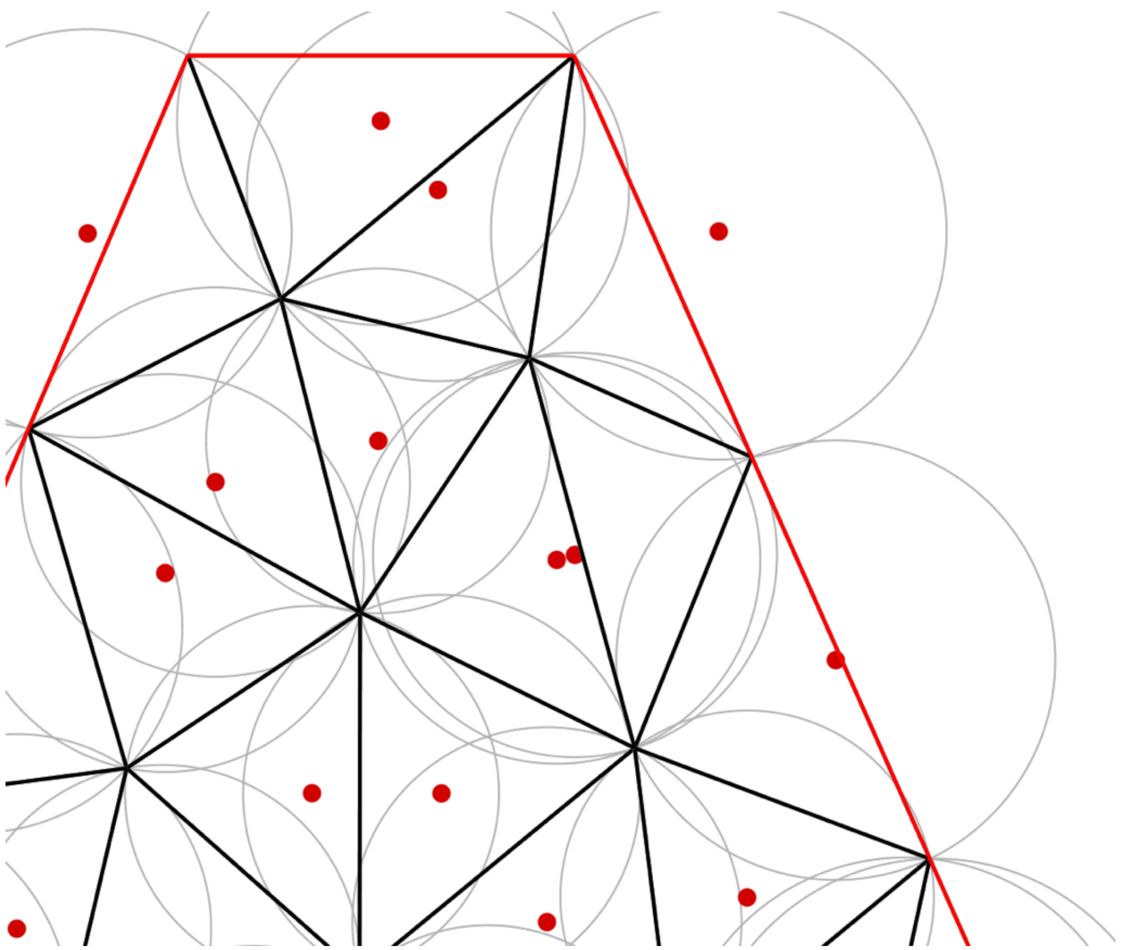
# Boundary Treatment II

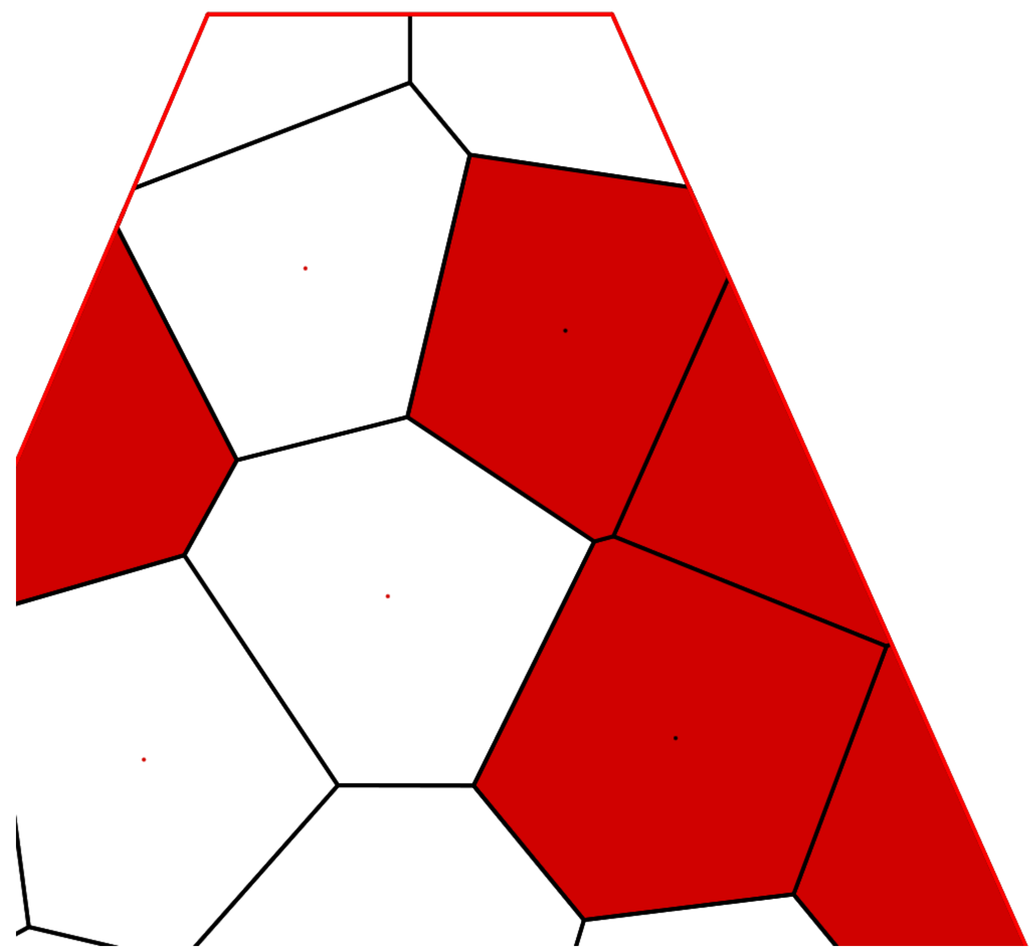
- Twice the weight for boundary triangles:
  - Short edges due to truncation
  - Conforming Gabriel property



# Boundary Treatment II

- Twice the weight for boundary triangles:
  - Short edges due to truncation
  - Conforming Gabriel property



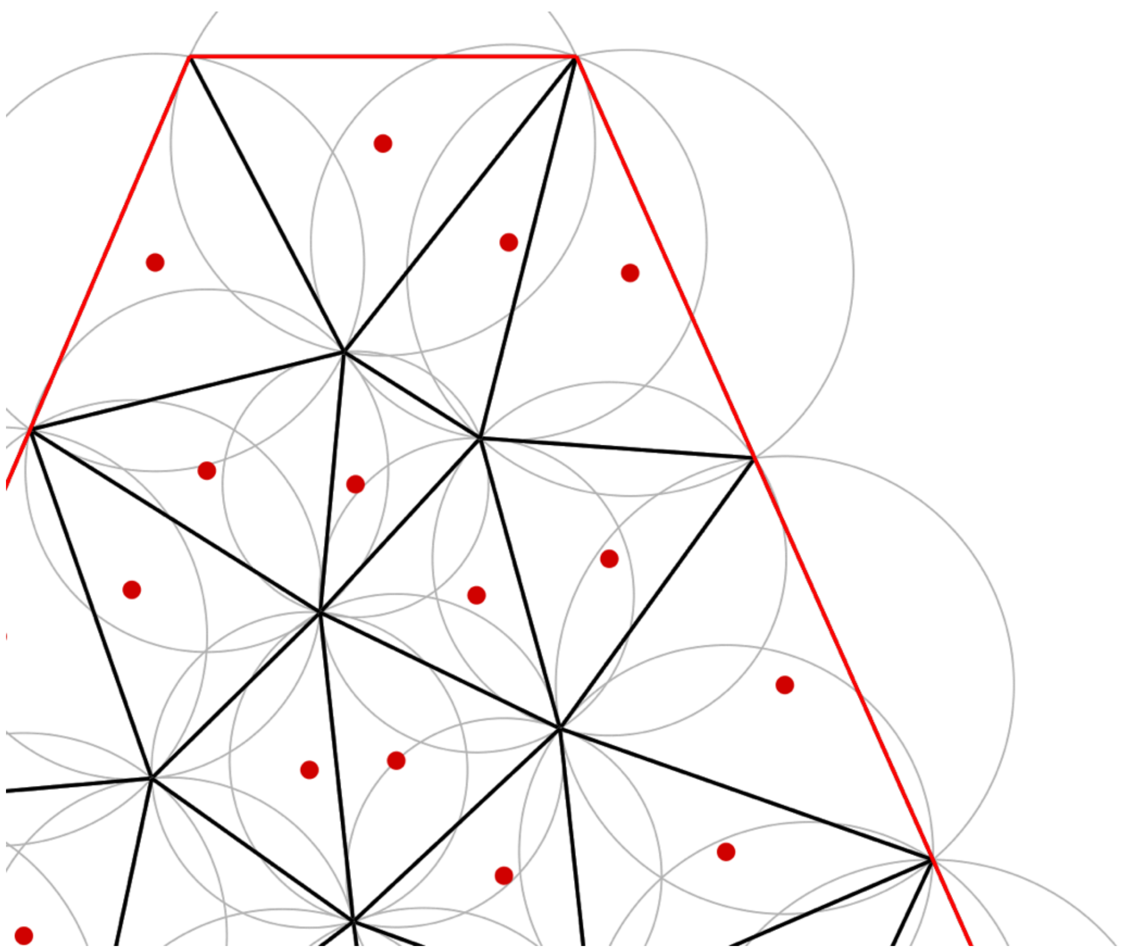


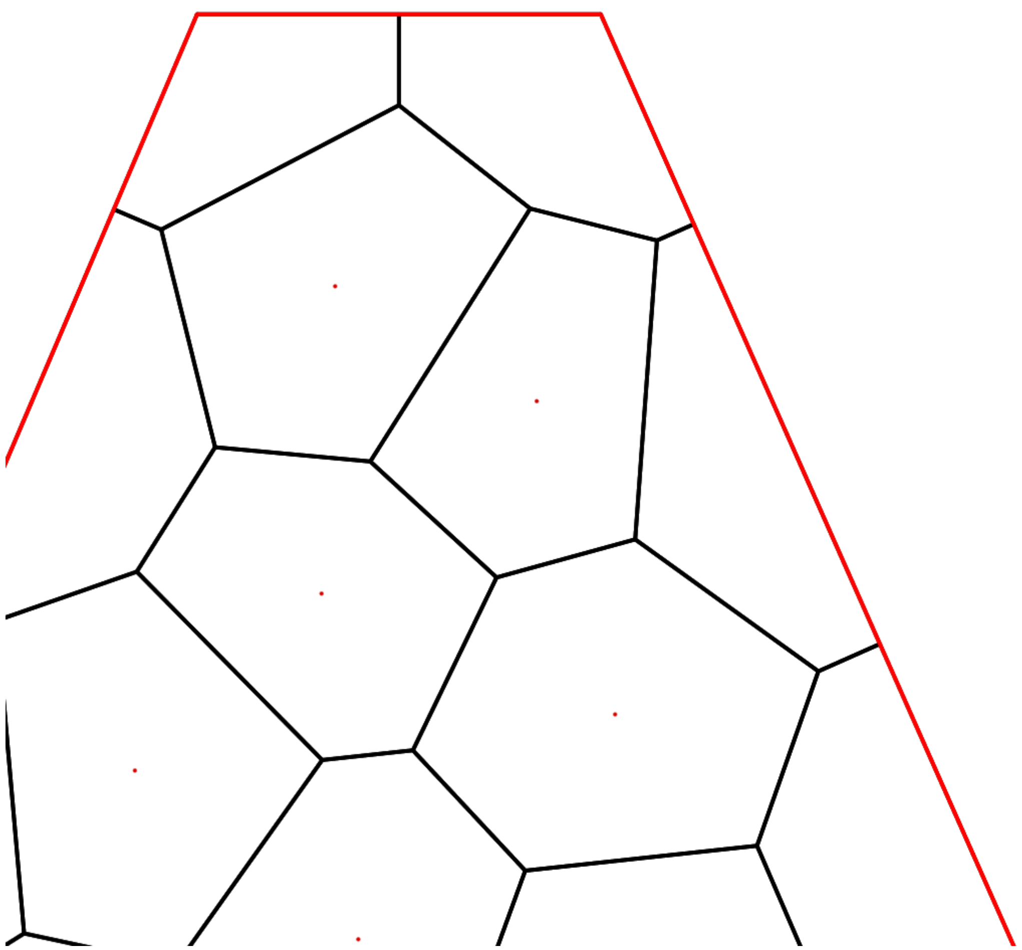


20

# Boundary Treatment II

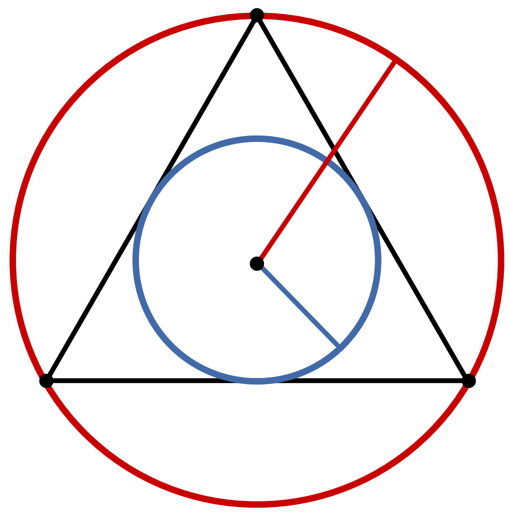
- Twice the weight for boundary triangles:
  - Short edges due to truncation
  - Conforming Gabriel property

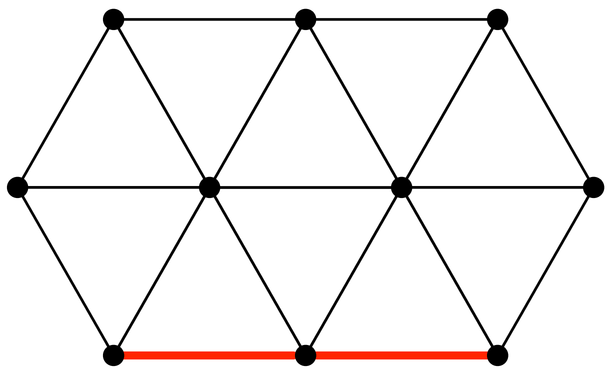


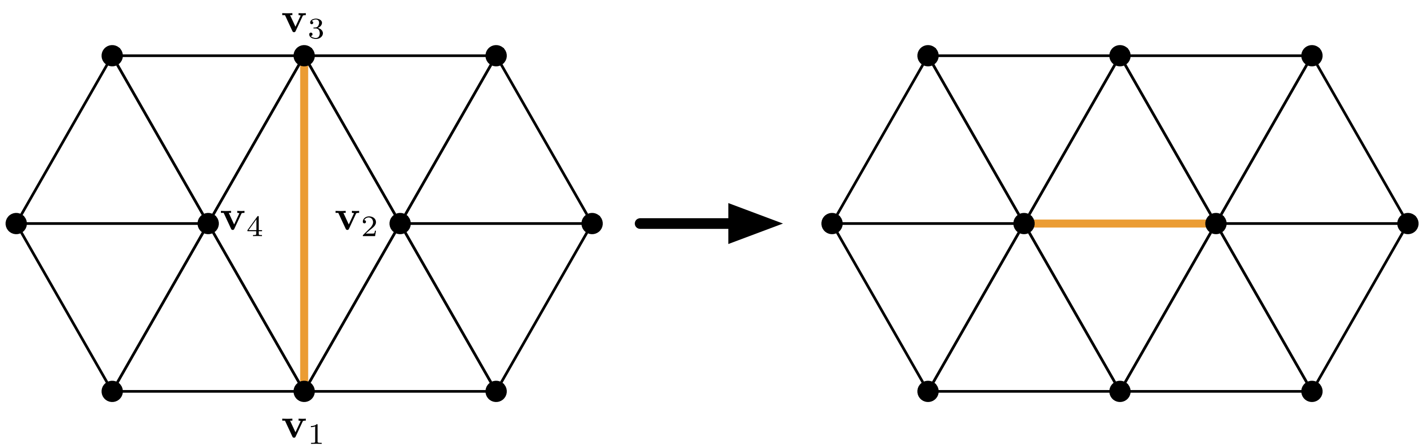


# Preprocessing

- Equilateral triangles are the minima of the energy
- Only possible around vertices with valence 6 (4 on the boundary)
- Improve valences towards the optimum by edge flips







# Algorithm Overview

1. Flip edges to improve vertex valence
2. Iteratively minimize energy:
   a) Compute gradient
   b) Determine step size
   c) Update vertex positions
3. Re-establish Delaunay property

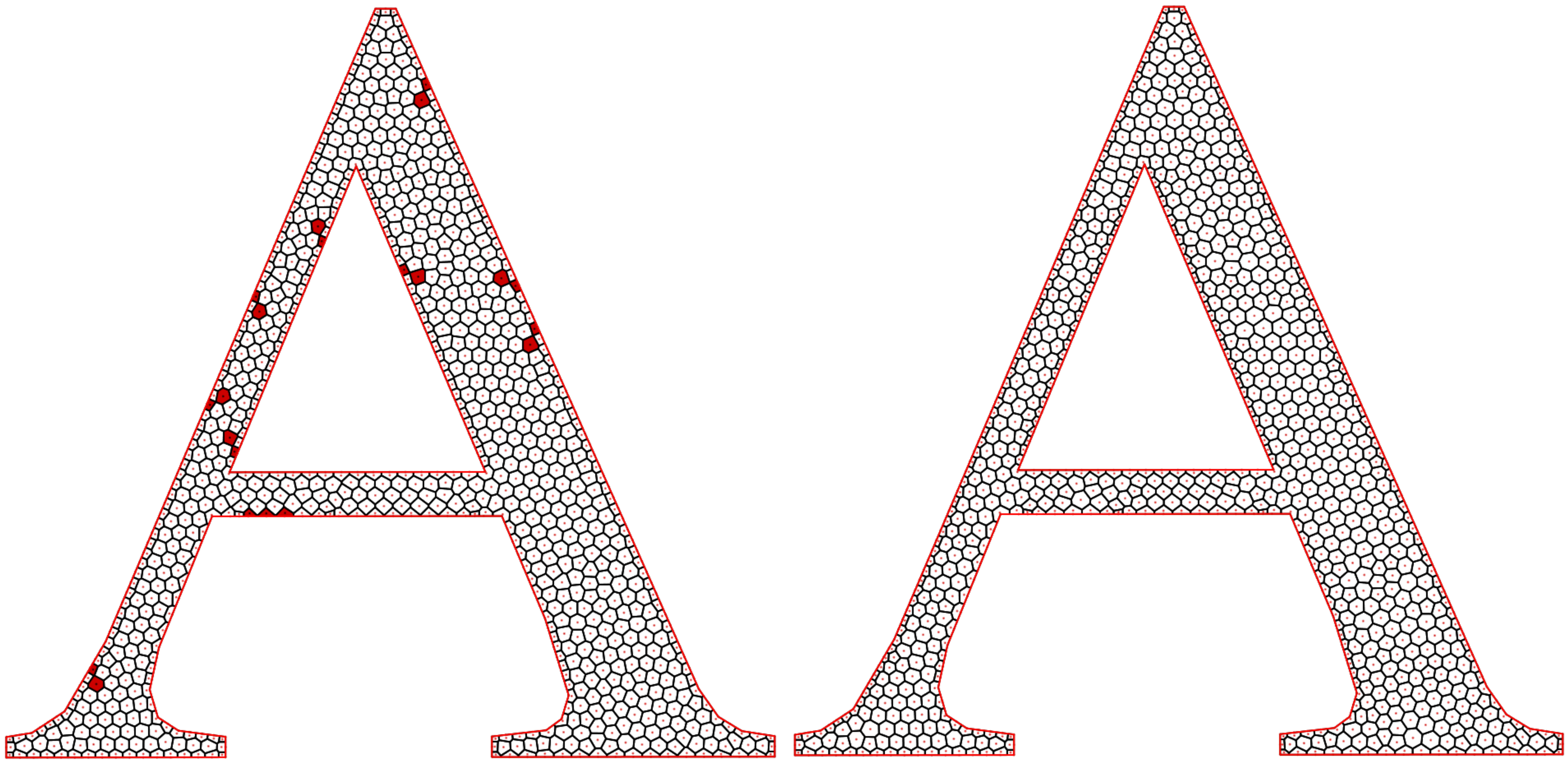
# Results

- Condition number before and after optimization
- Red elements: Edge < 5% of sizing

# Simple Mesh

- A-shape, 1799 triangles, time < 1 second

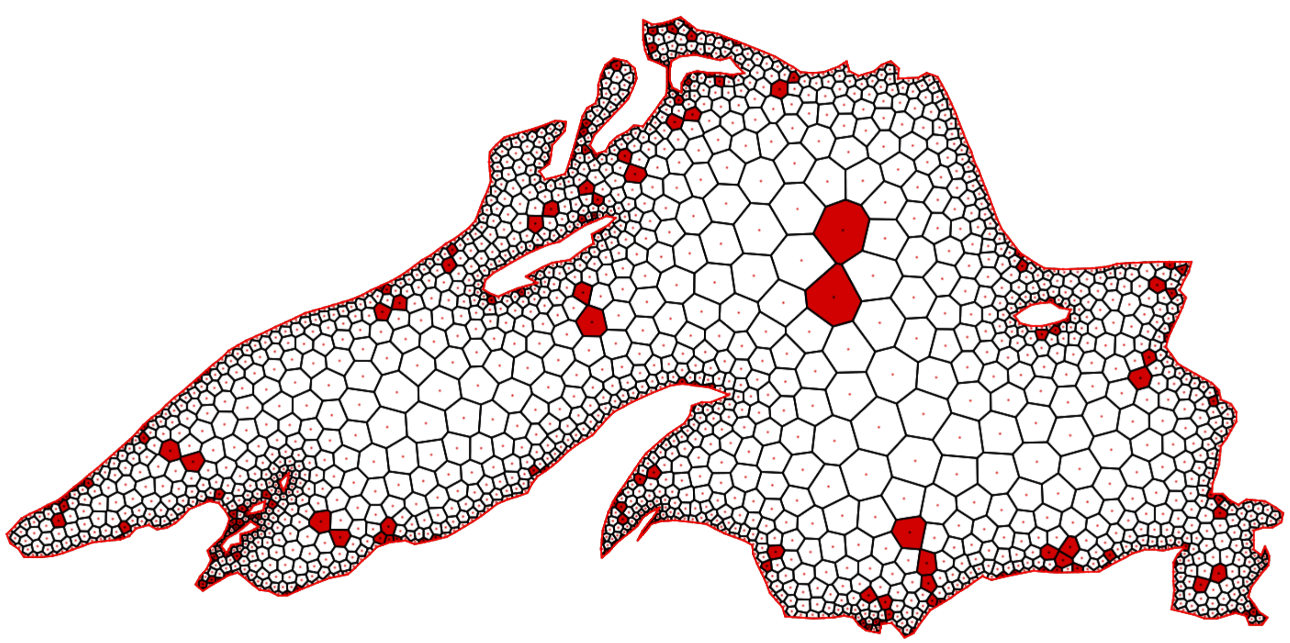
CVT: 421 Our Method: 75



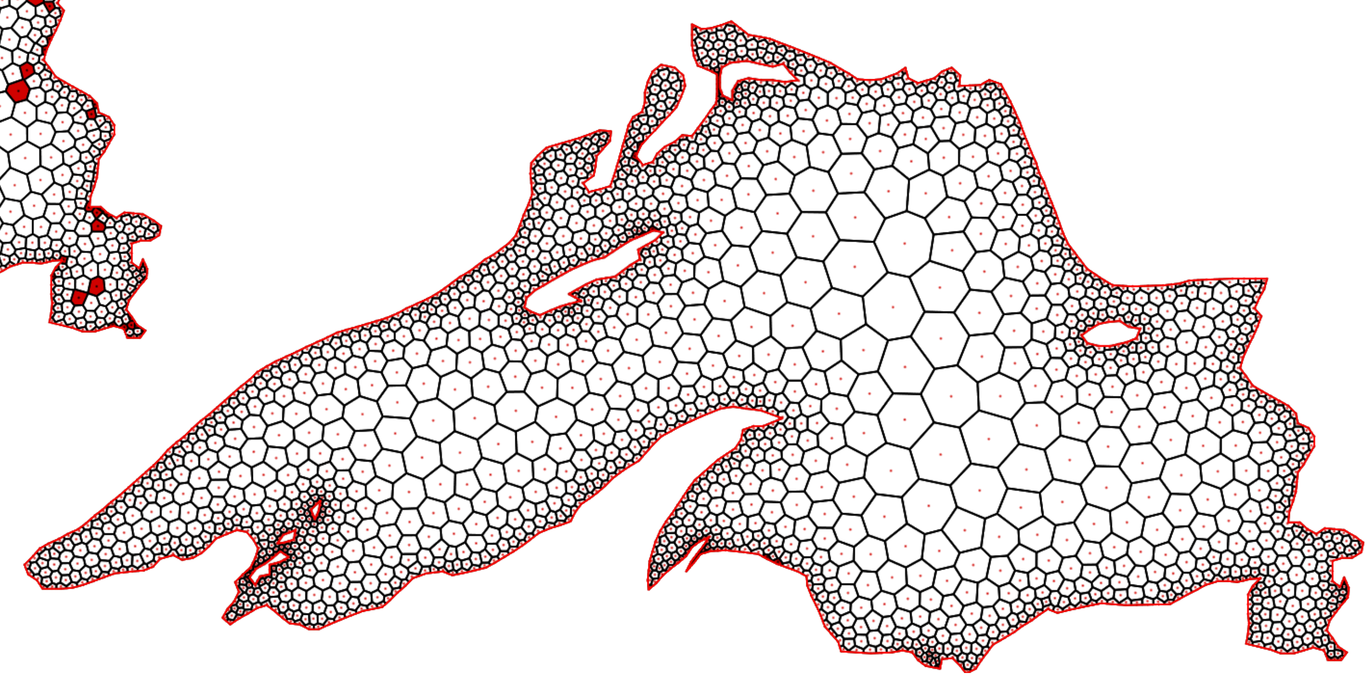
# Complex Domain, Graded Mesh

- Lake Superior, 4036 triangles, graded mesh, time < 1 second

CVT: 371877



Our Method: 190





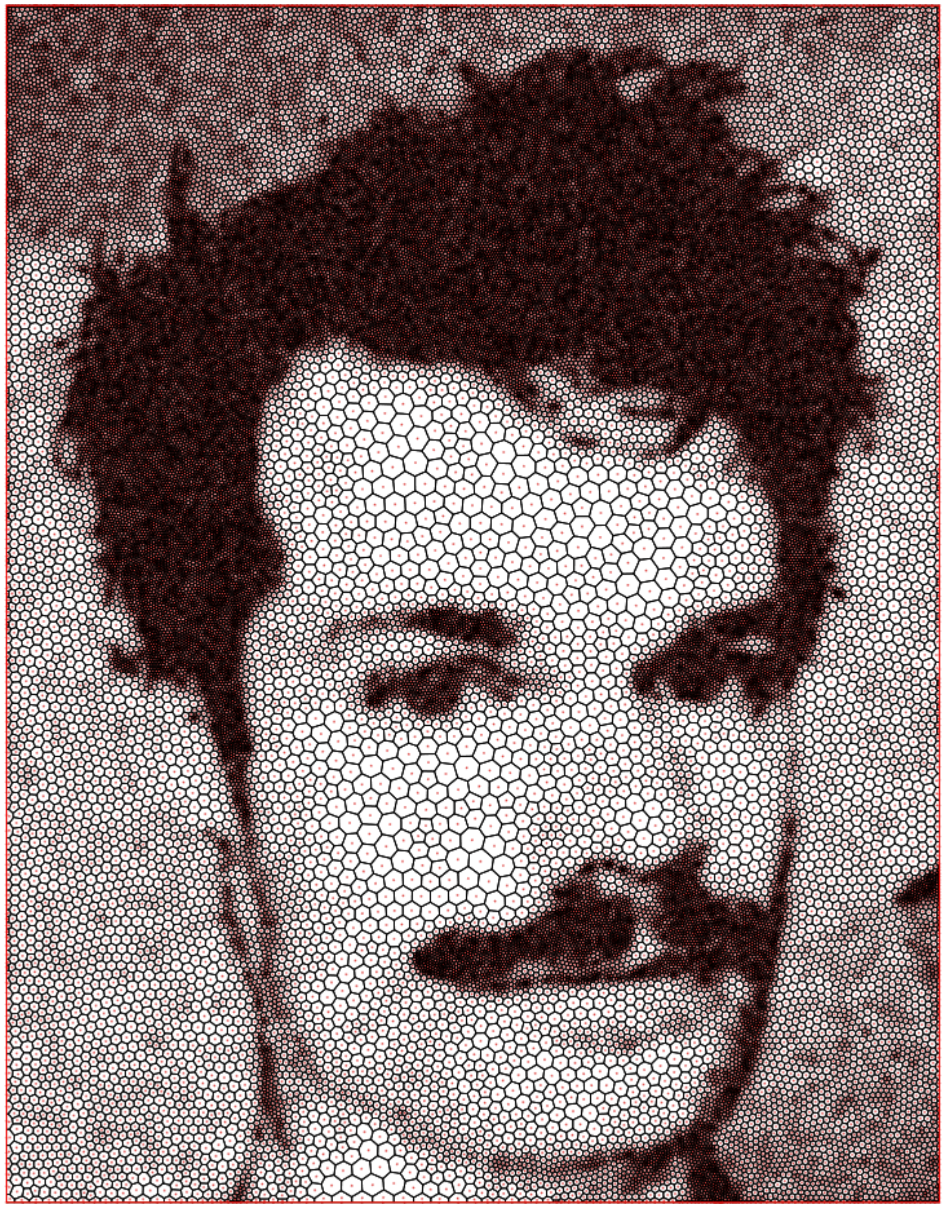
25

# Large & Heavily Graded Mesh

- Mesh generated from a photo, 113196 triangles, time < 1 minute

CVT: 354630

Our Method: 44208

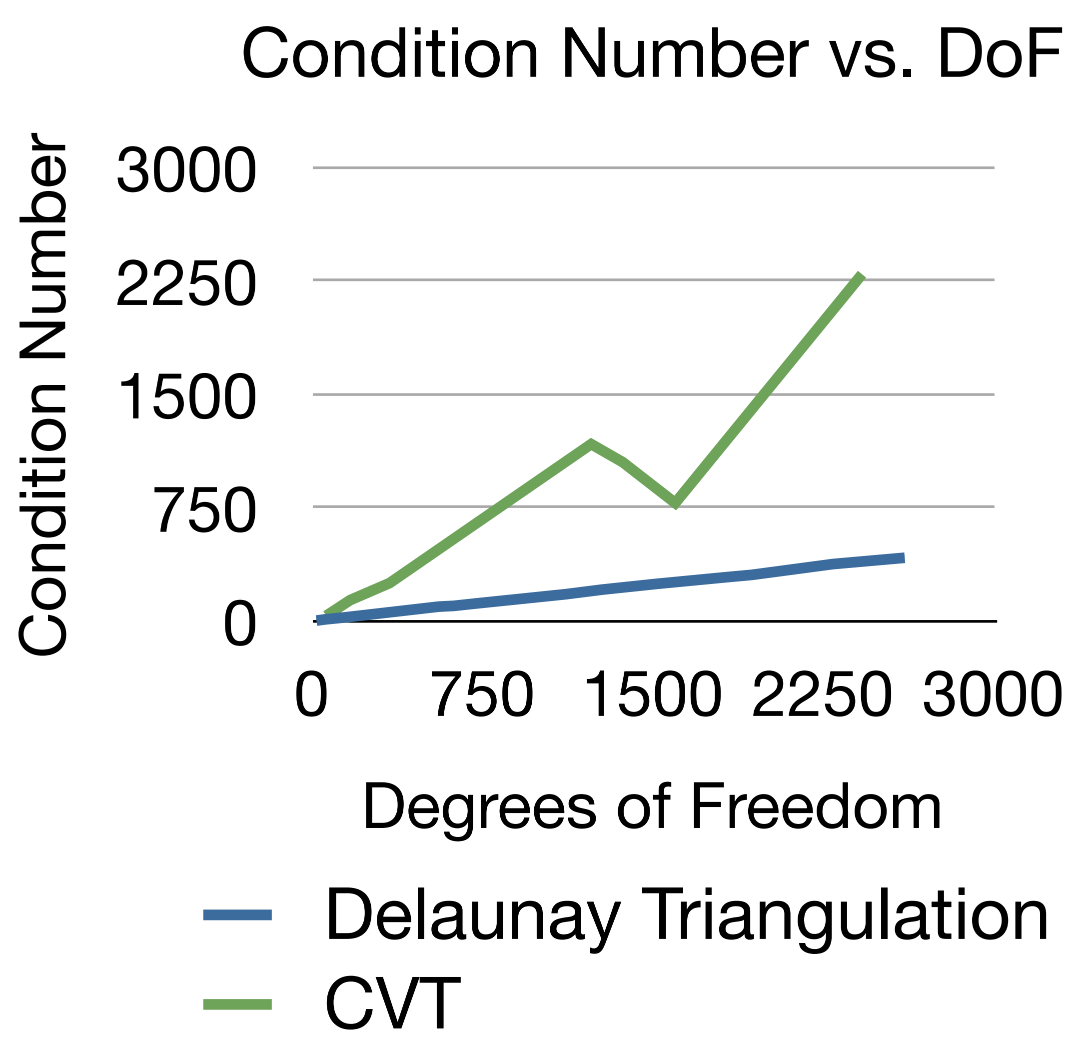




26

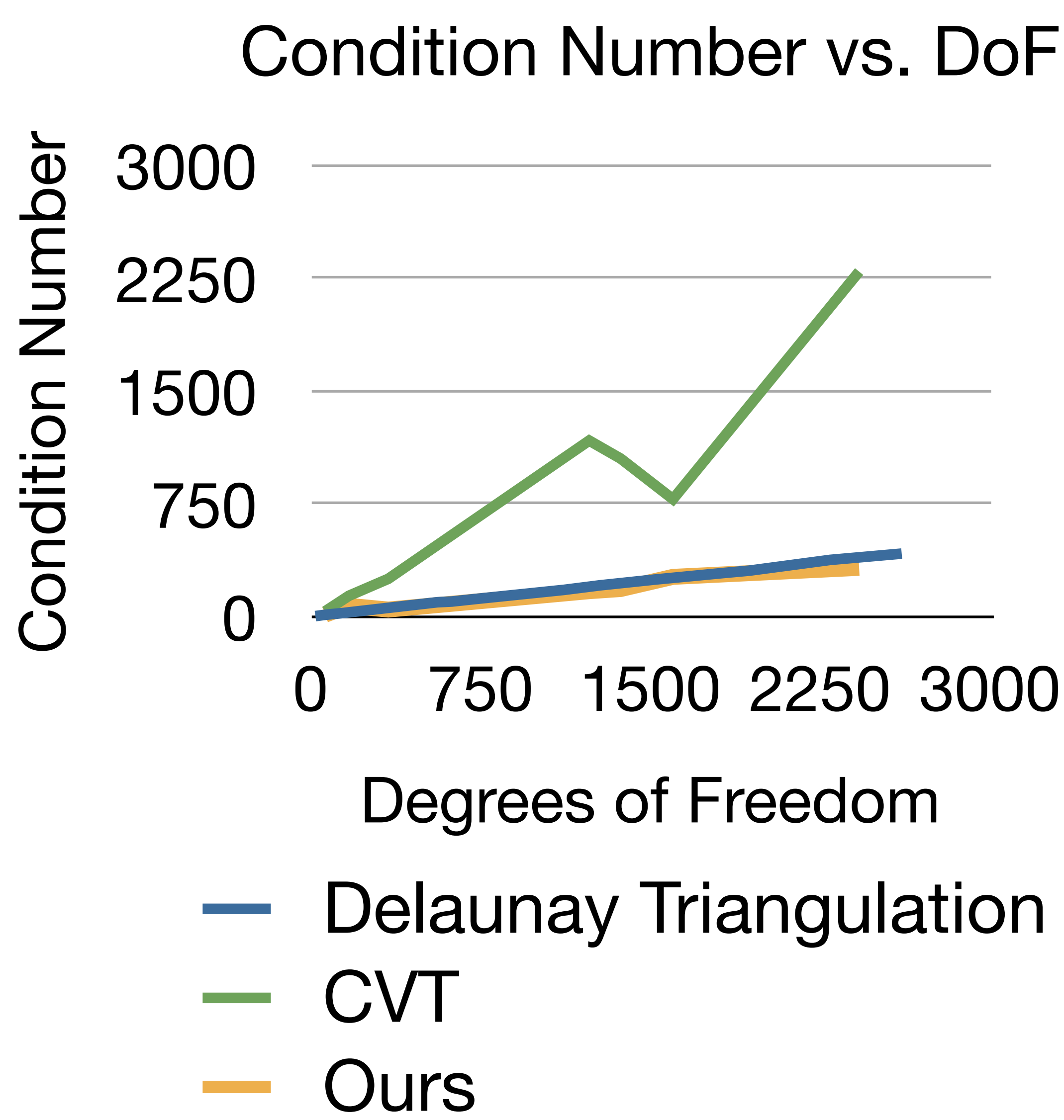
# Numerical Robustness

- Comparison with triangular meshes



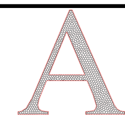
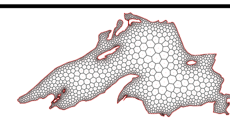
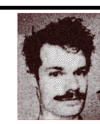
# Numerical Robustness

- Comparison with triangular meshes



# Comparison

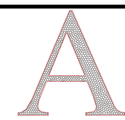
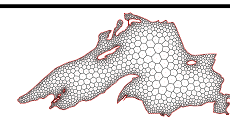
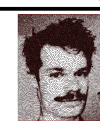
- Edge collapse (EC), Laplacian smoothing (LS)

| Shape | Ours | EC | LS |
|---|---|---|---|
|  | 75 | 66 | 43 |
|  | 190 | 200k | 1.2m |
|  | 44k | 350k | 40k |


28

# Comparison

- Edge collapse (EC), Laplacian smoothing (LS)

| Shape | Ours | EC | LS |
|---|---|---|---|
|  | 75 | 66 | 43 |
|  | 190 | 200k | 1.2m |
|  | 44k | 350k | 40k |

- Drawbacks of EC & LS:
  - Elements no longer as well-shaped
  - No more dual Delaunay triangulation
  - No valid Voronoi diagram


28

# Well-Centered Triangulations

- Each triangle contains its circumcenter

| | VanderZee et al., 2010 | Our Method |
|---|---|---|
| Energy | Distance of vertices to opposite edges of incident triangles, normalized by circumradii | Distance between circumcenter and incenter |

# Well-Centered Triangulations

- Each triangle contains its circumcenter

| | VanderZee et al., 2010 | Our Method |
|---|---|---|
| Energy | Distance of vertices to opposite edges of incident triangles, normalized by circumradii | Distance between circumcenter and incenter |
| Norm | $L^p$-Norm ($p$=4, 6, 8, 10) | Simple $L^2$-Norm |

# Well-Centered Triangulations

- Each triangle contains its circumcenter

| | VanderZee et al., 2010 | Our Method |
|---|---|---|
| Energy | Distance of vertices to opposite edges of incident triangles, normalized by circumradii | Distance between circumcenter and incenter |
| Norm | $L^p$-Norm ($p$=4, 6, 8, 10) | Simple $L^2$-Norm |
| Minimization | Conjugate gradients, sophisticated line search, approximated gradients | Gradient descent, bisection line search, analytical gradients |

# Well-Centered Triangulations

- Each triangle contains its circumcenter

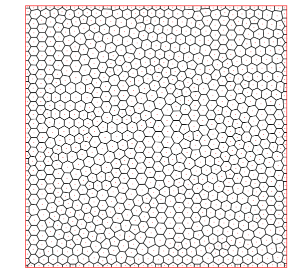
| | VanderZee et al., 2010 | Our Method |
|---|---|---|
| Energy | Distance of vertices to opposite edges of incident triangles, normalized by circumradii | Distance between circumcenter and incenter |
| Norm | $L^p$-Norm ($p=4, 6, 8, 10$) | Simple $L^2$-Norm |
| Minimization | Conjugate gradients, sophisticated line search, approximated gradients | Gradient descent, bisection line search, analytical gradients |
| Preprocess | Sophisticated multi-stage process | Edge flips only |

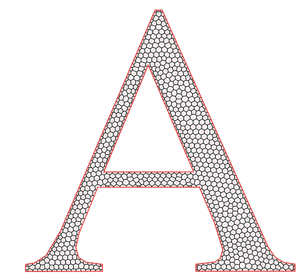
# Well-Centered Triangulations

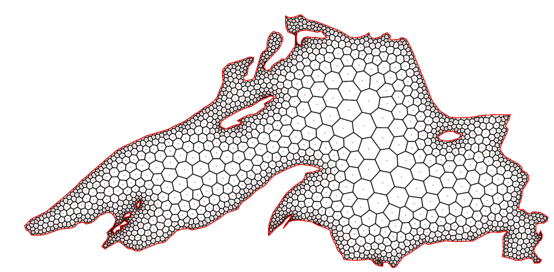
- Each triangle contains its circumcenter

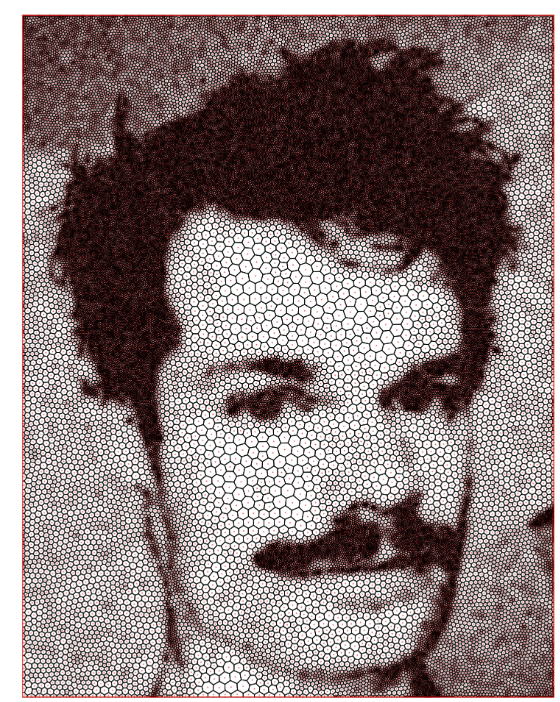
| | VanderZee et al., 2010 | Our Method |
|---|---|---|
| Energy | Distance of vertices to opposite edges of incident triangles, normalized by circumradii | Distance between circumcenter and incenter |
| Norm | $L^p$-Norm ($p$=*4, 6, 8, 10*) | Simple $L^2$-Norm |
| Minimization | Conjugate gradients, sophisticated line search, approximated gradients | Gradient descent, bisection line search, analytical gradients |
| Preprocess | Sophisticated multi-stage process | Edge flips only |
| Result | Completely well-centered triangulation | Well-centered except ~1% |

# Conclusion & Outlook









- Simple & efficient optimization to remove short edges from CVTs
- Significantly improves stiffness matrix conditioning
- Preserves element shape and grading
- Outlook:
  – Well-centered triangulations
  – Implementation in 3D


30