



---

## CENTROIDAL VORONOI TESSELLATIONS

---

Max Gunzburger

*Department of Scientific Computing*

*Florida State University*

`gunzburg@fsu.edu`

Last updated in 2010

## *Collaborators*

Rebecca Brannon, John Burkardt, Qiang Du, Vance Faber, Lili Ju, Hyung-Chun Lee, Hoa Nguyen, Janet Peterson, Todd Ringler, Vincente Romero, Yuki Saka, Xiaoqiang Wang, Weidong Zhao

## CENTROIDAL VORONOI TESSELLATIONS

- Centroidal Voronoi tessellations (CVT's) are:
  - a way to select the location of points
  - a way to cluster data
- These two tasks are of substantial interest in lots of applications
- It turns out that for many of these applications, CVT's do a pretty good job

## APPLICATIONS OF CVT'S - partial list

- optimal quadrature rules
- covolume and finite difference methods for PDE's
- optimal representation, quantization, and clustering
- finite volume methods for PDE's
- optimal placement of sensors and actuators
- surrogate optimization
- particle methods
- stippling
- visualization of software metrics
- melodic structure improvement
- mosaic effects for images

- optimal distribution of resources
- cell division
- territorial behavior of animals
- data compression
- image segmentation and edge detection
- multichannel reconstruction
- grid generation
- point distributions and grid generation on surfaces
- meshfree methods
- reduced-order modeling
- hypercube point sampling

## TESSELLATIONS

- A **tessellation** of a set  $S$  is a division of the set into a **nonoverlapping, covering** collection of subsets  $\{S_1, S_2, \dots, S_K\}$
- Given a set  $S$  and an integer  $K \geq 2$ , one divides  $S$  into  $K$  subsets  $S_1, S_2, \dots, S_K$  such that
  - no member of a set  $S_i$  is a member of another set  $S_j$
  - every member of  $S$  belongs to one of the sets  $\overline{S}_i$
- $S_i \cap S_j = \emptyset, \quad j \neq i \quad \text{and} \quad \overline{S} = \bigcup_{i=1}^K \overline{S}_i$

## VORONOI TESSELLATIONS

- Given
  - a set  $S$  and an integer  $K \geq 2$
  - a set  $Z = \{z_i\}_{i=1}^K$  whose elements may or may not belong to  $S$
  - a distance function  $d(\cdot, \cdot)$  defined for  $S$  and  $Z$
- Then, the **Voronoi subset** or **Voronoi cell** or **Voronoi region**  $V_j \subset S$  is the set of all elements belonging to  $S$  that are closer to  $z_j$  than to any of the other elements  $z_i \in Z$ , that is,
$$V_j = \left\{ w \in S \mid d(w, z_j) < d(w, z_i), i = 1, \dots, K, i \neq j \right\}$$
- We call the set of Voronoi subsets  $\{V_1, V_2, \dots, V_K\}$  a **Voronoi tessellation** of  $S$  or a **Voronoi diagram** of  $S$  with respect to the given set of generators  $Z$

- Voronoi diagrams have been rediscovered many times and are thus known by many other names, depending on the application

Voronoi sets =

Dirichlet regions =

Meijering cells =

S-mosaics =

Thiessen polygons =

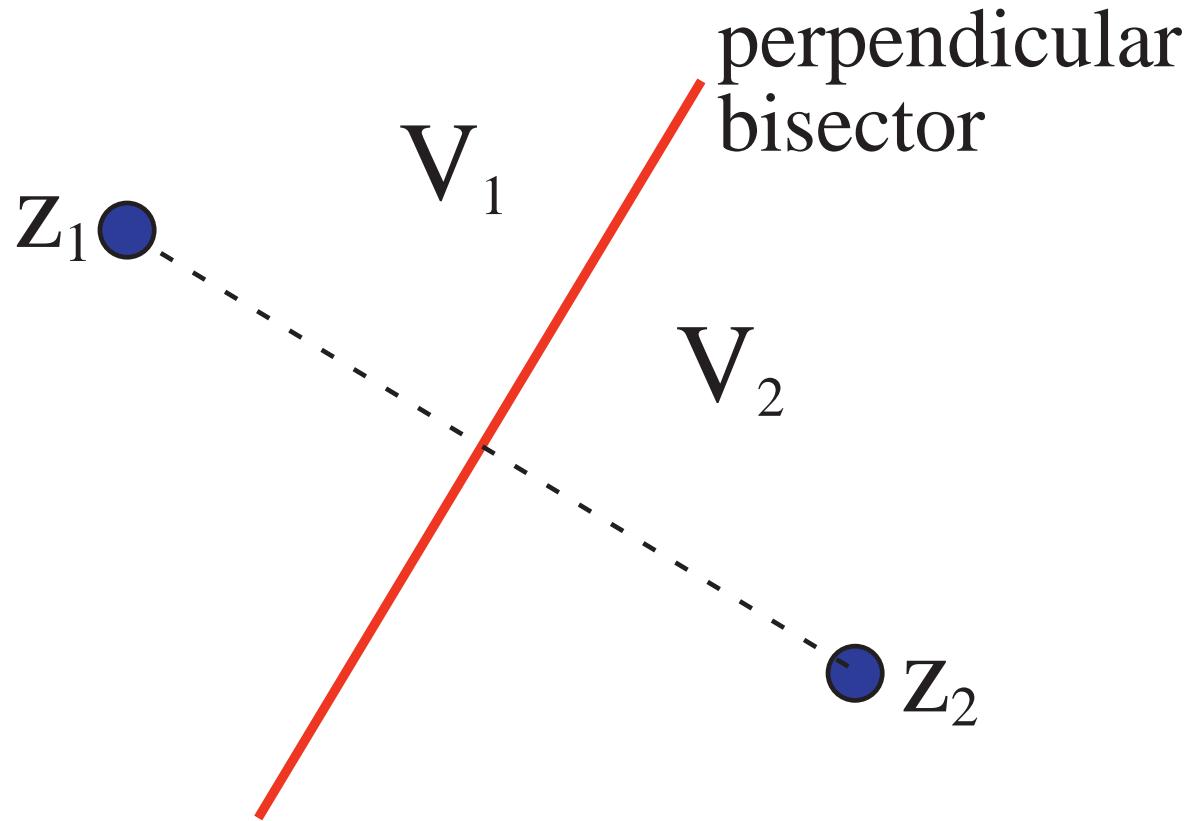
area of influence polygons =

etc., etc., etc.

- Voronoi diagrams were known to Descartes

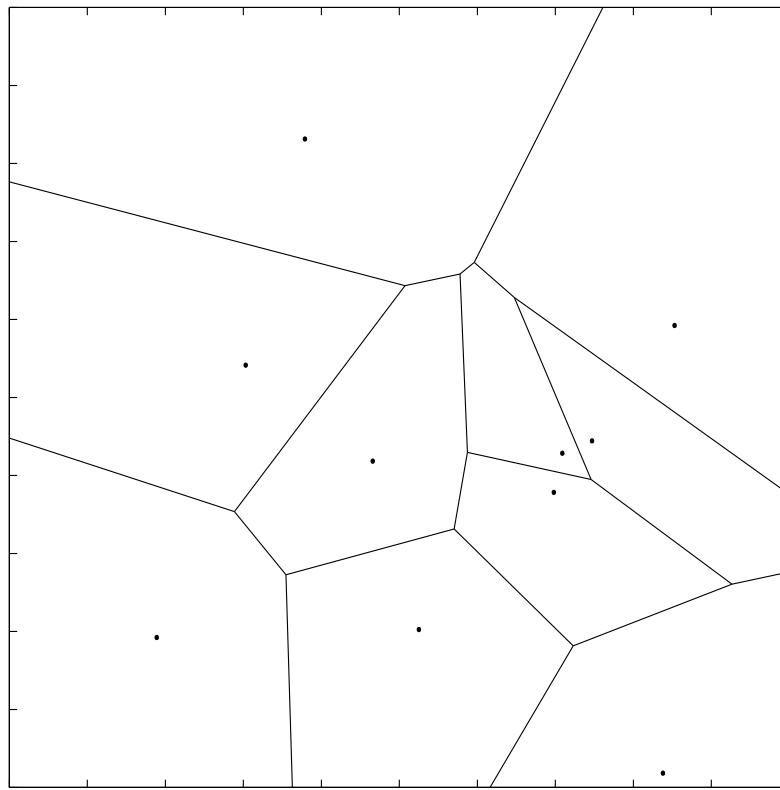


Georgi Voronoi



$S = \text{the plane}; \quad d(z, w) = \text{Euclidean distance}; \quad K = 2$

*The Voronoi regions for two points  $z_1$  and  $z_2$  in the plane are the two regions on either side of the perpendicular bisector of the line segment joining  $z_1$  and  $z_2$*



$S$  = a square;  $d(z, w)$  = Euclidean distance;  $K = 10$

*Voronoi tessellation for 10 randomly selected points in a square*

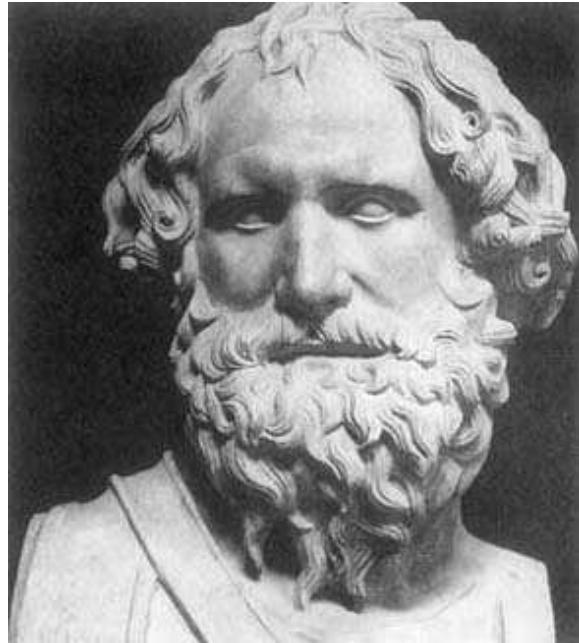
## CENTER OF MASS

- Given a **domain**  $\mathcal{D}$  in  $\mathbb{R}^n$  and a density function  $\rho(w)$  defined for  $w \in \mathcal{D}$ , the **center of mass** or **centroid**  $z^*$  of  $\mathcal{D}$  is given by

$$z^* = \frac{\int_{\mathcal{D}} w \rho(w) dw}{\int_{\mathcal{D}} \rho(w) dw}$$

- Or, given a **discrete** set of points  $W = \{w_j\}_{j=1}^M$  in  $\mathbb{R}^n$  and a density function  $\rho(w_j)$ ,  $j = 1, \dots, M$ , the **center of mass**  $z^*$  of  $W$  is given by

$$z^* = \frac{\sum_{j=1}^M w_j \rho(w_j)}{\sum_{j=1}^M \rho(w_j)}$$



Archimedes

- The notion of a center of mass can be generalized in many ways, e.g,

$$\int_{\mathcal{D}} \rho(w) f(d(z^*, w)) dw = \inf_{z \in \mathcal{D}^*} \int_{\mathcal{D}^*} \rho(w) f(d(z, w)) dw$$

where  $f(d(z^* - w))$  is convex in  $w$

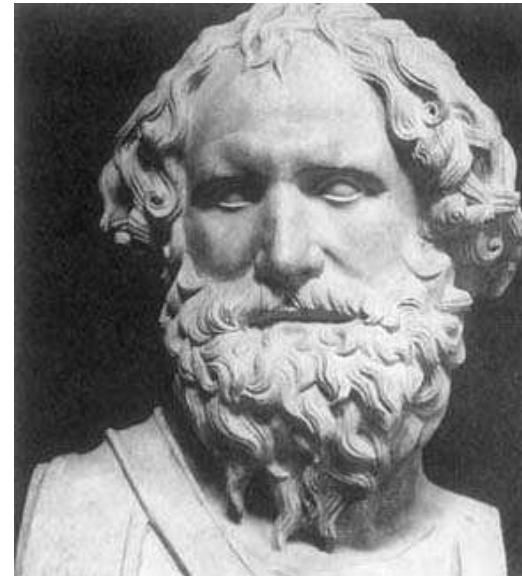
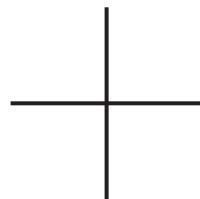
- We have now defined two different notions:

Voronoi tessellations

and

centers of mass

- Let's bring the two notions together



- Given
  - a set  $S$
  - a density function  $\rho(\cdot)$  defined over  $S$
  - an integer  $K \geq 2$
  - a set of generators  $Z = \{z_i\}_{i=1}^K$
  - a distance function  $d(\cdot, \cdot)$  defined between elements in  $S$  and  $Z$
- We can define the associated **Voronoi sets**
- Once having determined the Voronoi sets, we can define the associated **centroids**

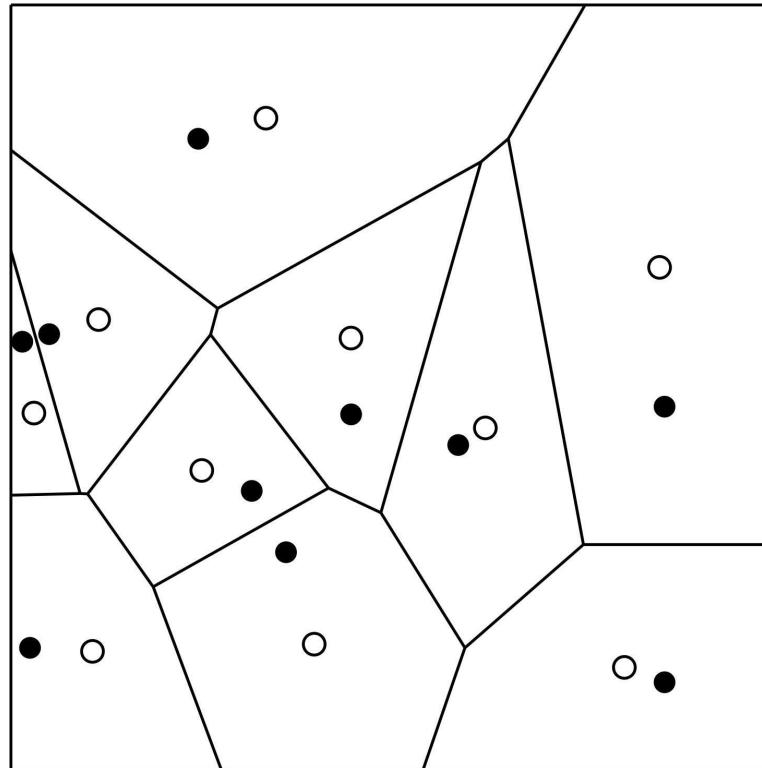
$$V_i, \quad i = 1, \dots, K$$

$$z_i^*, \quad i = 1, \dots, K$$

of each of the Voronoi sets

- In general, the centers of mass of the Voronoi sets do not coincide with the generators of the Voronoi sets

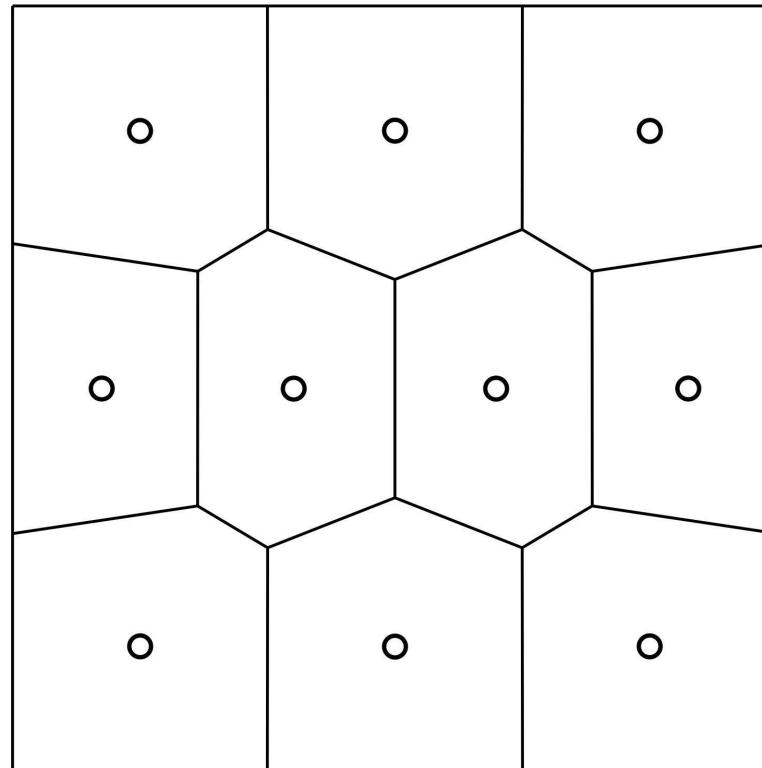
$$z_i \neq z_i^*, \quad i = 1, \dots, K$$



*The Voronoi regions and their centers of mass (with respect to a uniform density) for 10 randomly selected points in a square; the generators (the dots) and the centers of mass (the circles) do not coincide*

- We are interested in the very special cases for which  
the generators and centers of mass do coincide

$$z_i = z_i^*, \quad i = 1, \dots, K$$



*The 10 dots are simultaneously the generators of the Voronoi tessellation and the centers of mass (with respect to a uniform density) of the Voronoi cells*

- We call a **tessellation** with the property

$$\textcolor{red}{z}_i = \textcolor{red}{z}_i^*, \quad i = 1, \dots, K$$

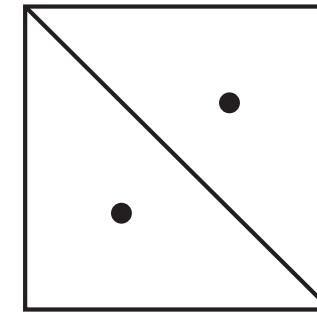
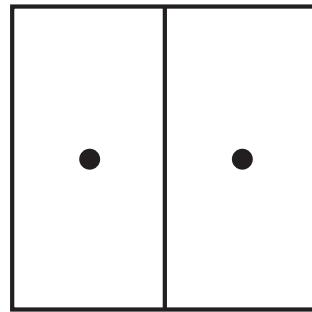
a **centroidal Voronoi tessellation** or **CVT**

- We call the **generators** of a centroidal Voronoi tessellation **CVT generators** or **CVT points**
- Centroidal Voronoi tessellations are very special Voronoi tessellations
  - they **must be constructed**

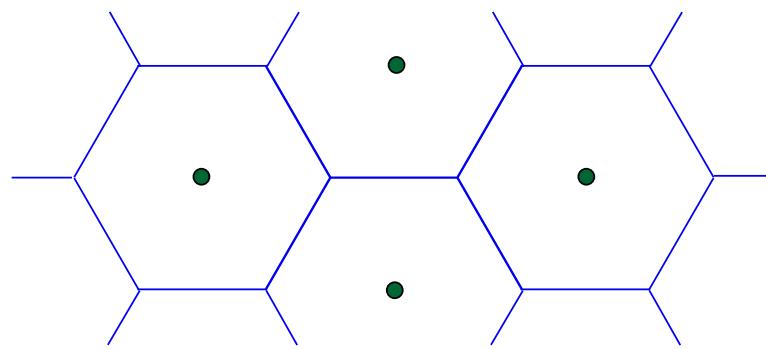
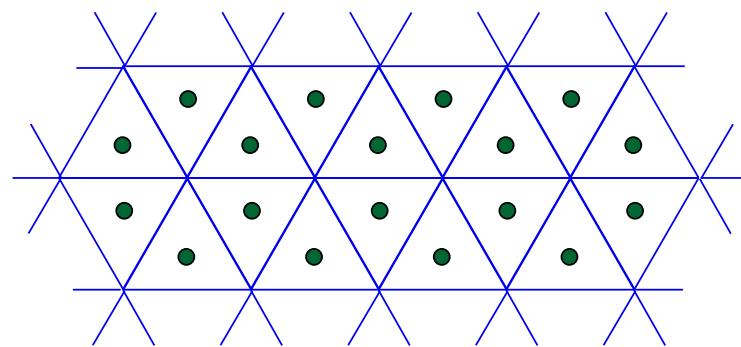
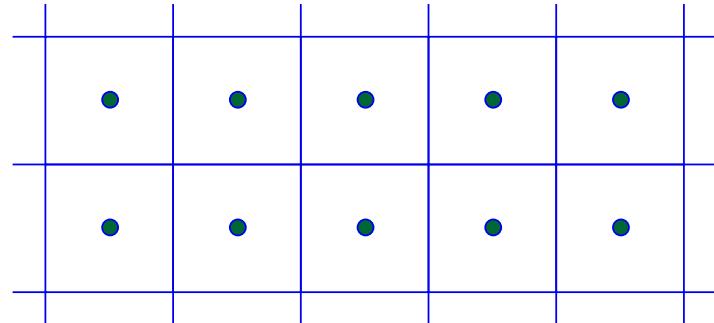
## THE CONSTRUCTION PROBLEM FOR CVT'S

- Given
  - a set  $S$ , an integer  $K \geq 2$ ,
  - a density function  $\rho(\cdot)$ , a distance function  $d(\cdot, \cdot)$
- we are interested in finding
  - $K$  elements  $\{z_i\}_{i=1}^K$  and  $K$  sets  $\{V_i\}_{i=1}^K$
- such that
  - $\{V_i\}_{i=1}^K$  tessellates  $S$
- and simultaneously
  - the regions  $\{V_i\}_{i=1}^K$  are the **Voronoi regions** for the generators  $\{z_i\}_{i=1}^K$
  - the elements  $\{z_i\}_{i=1}^K$  are the **centers of mass** of the regions  $\{V_i\}_{i=1}^K$

- Note that, in general, one **does not have uniqueness**



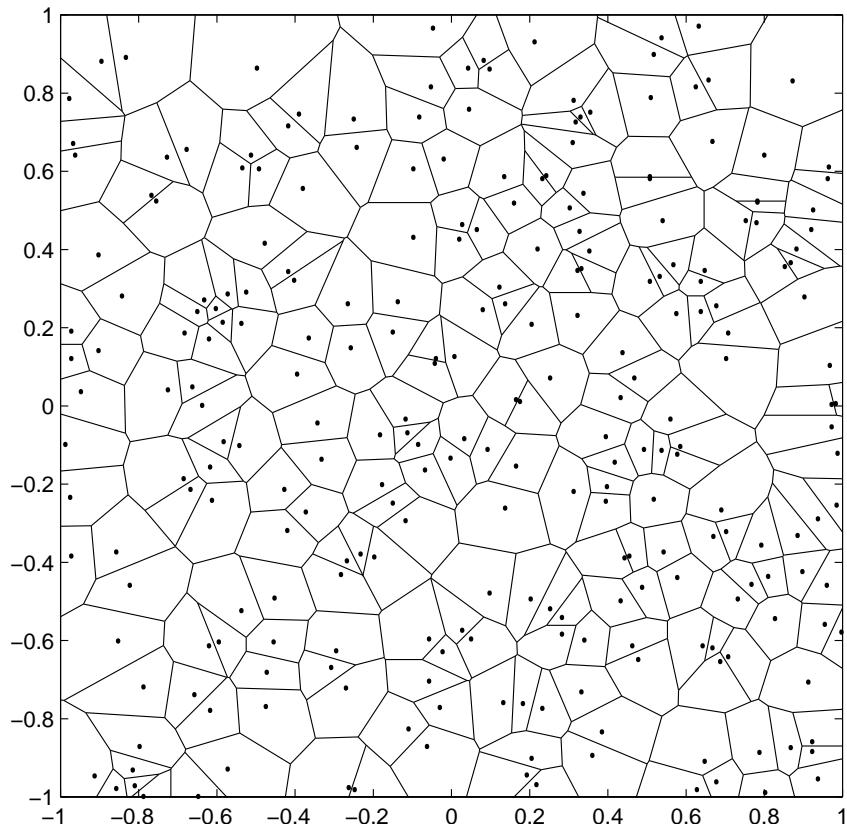
*Two two-point centroidal Voronoi tessellations of a square*



*Three regular tessellations of the plane*

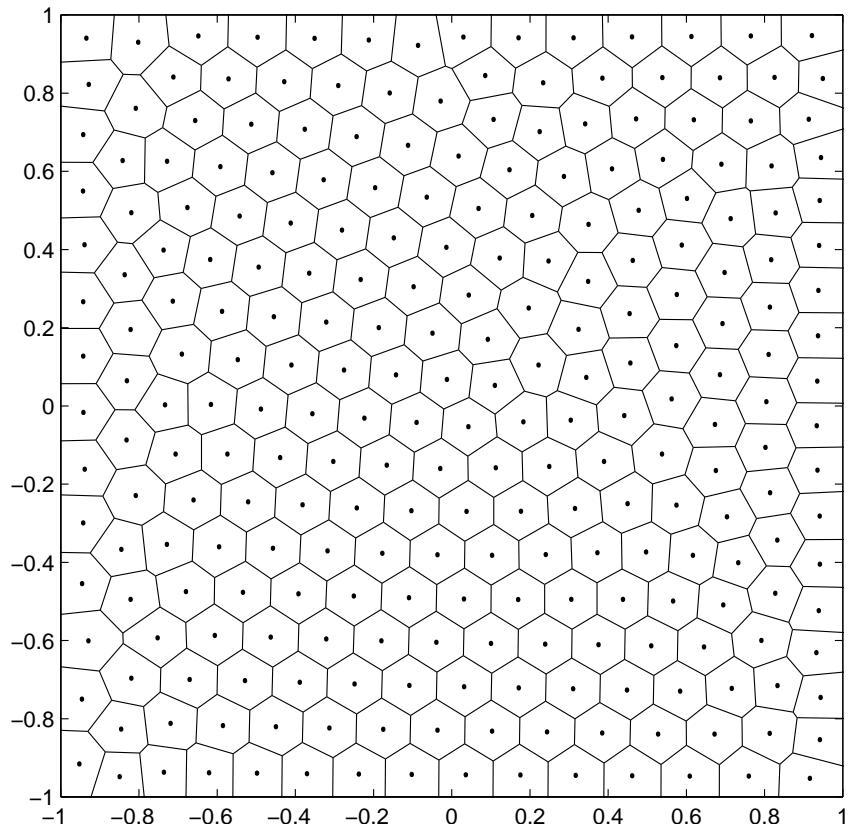
## A CVT OF THE SQUARE WITH MORE POINTS

uniform density



*Random sampling*

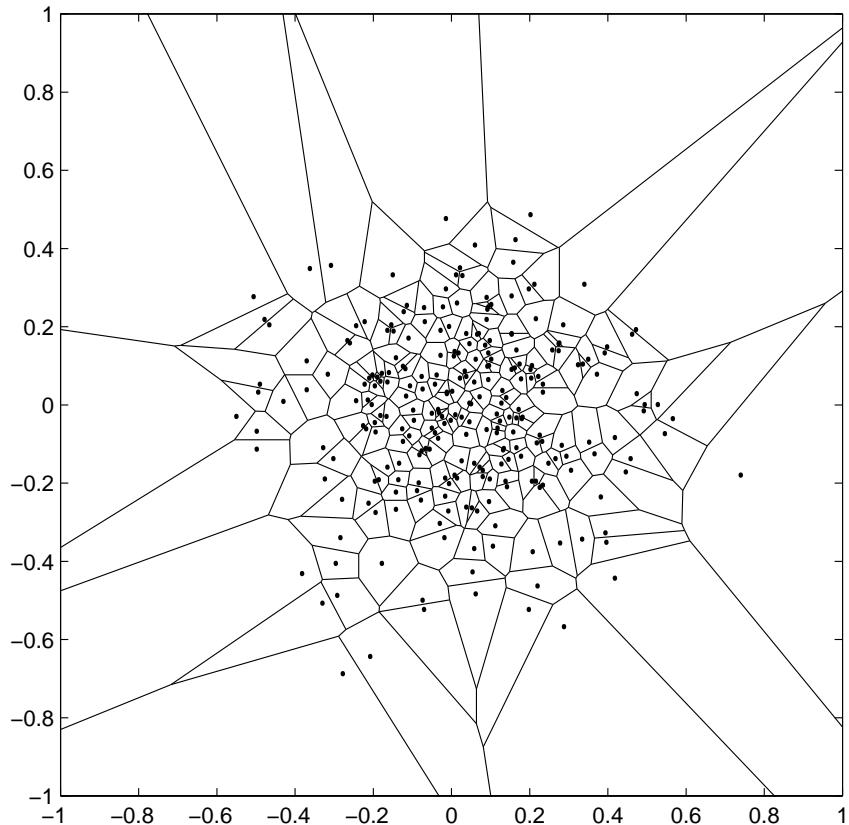
256 generators



*Centroidal Voronoi*

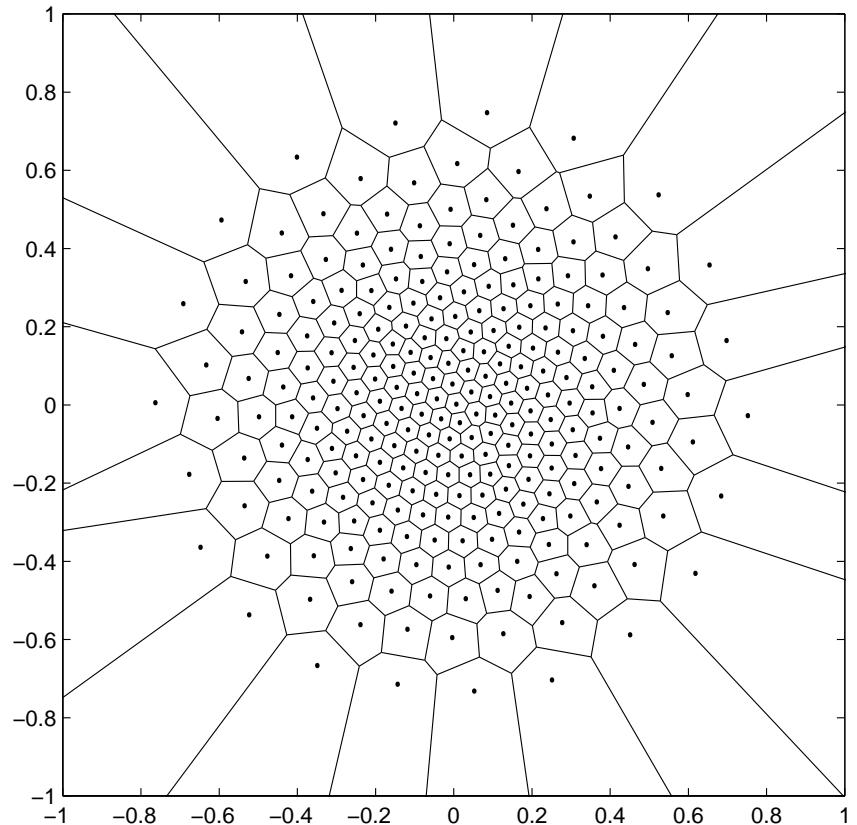
# CVT'S OF THE SQUARE HAVING NONUNIFORM DENSITY

density with peak in middle



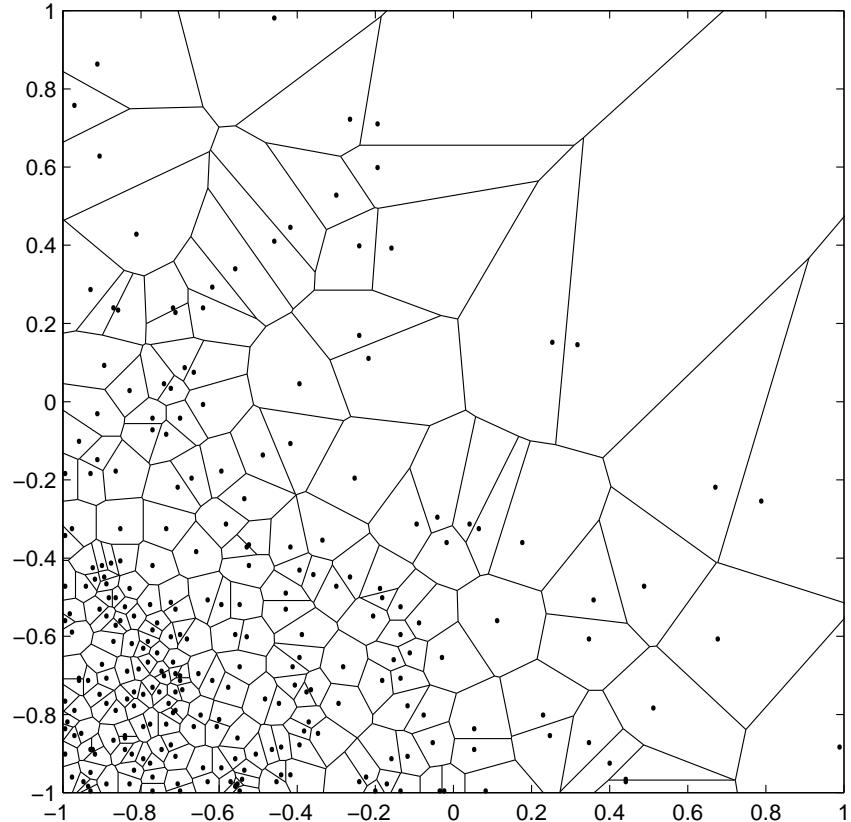
Random sampling

256 generators



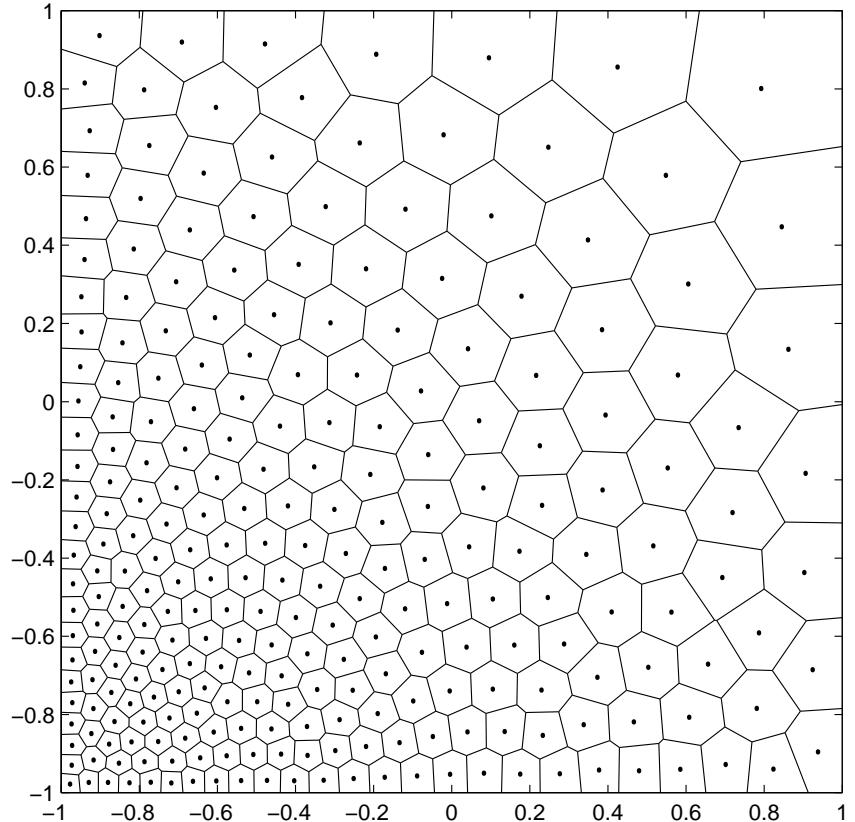
Centroidal Voronoi

density with peak at a corner



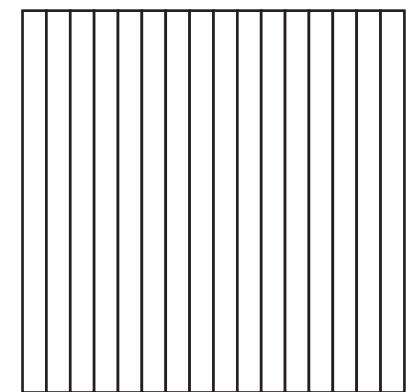
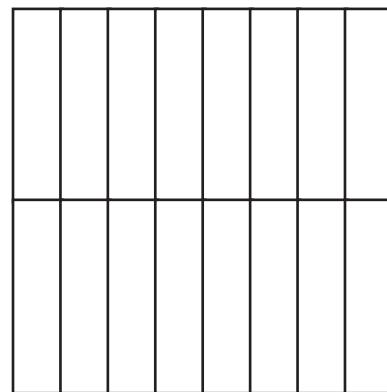
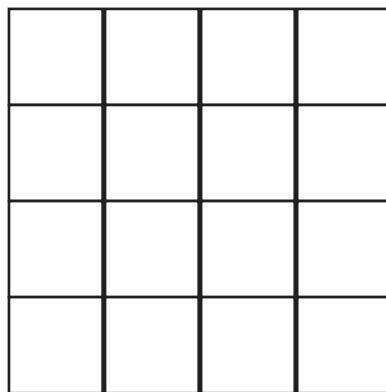
*Random sampling*

256 generators



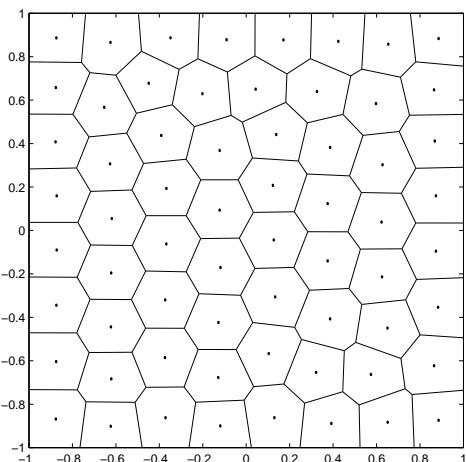
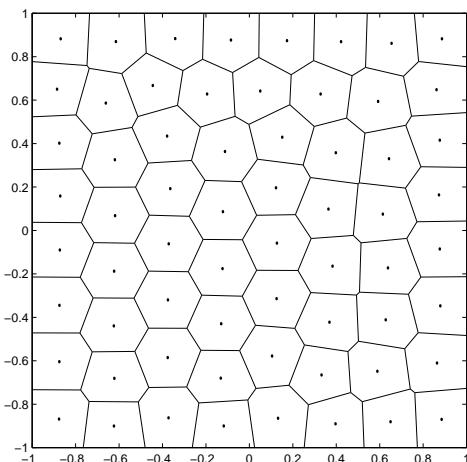
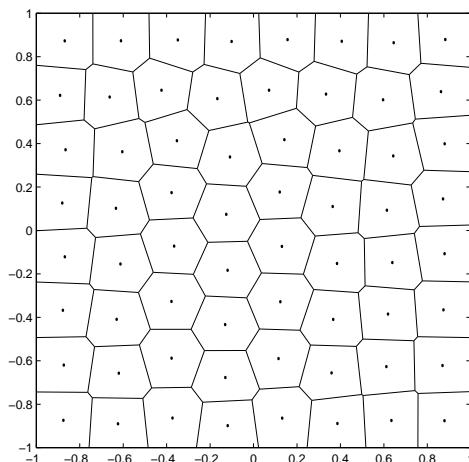
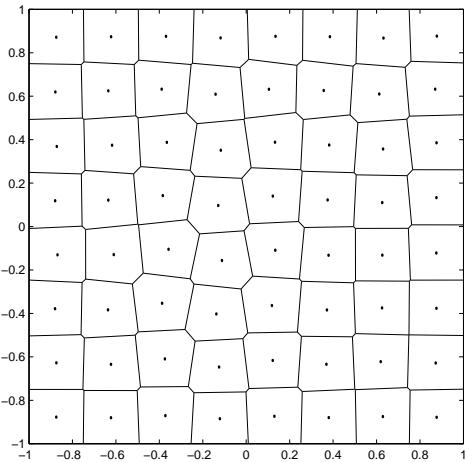
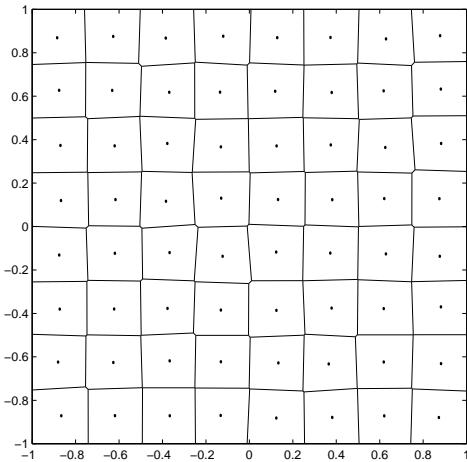
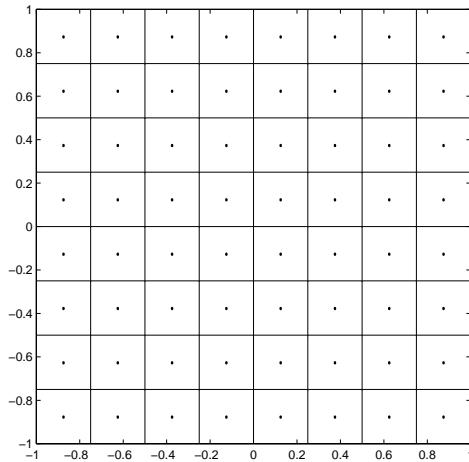
*Centroidal Voronoi*

- Clearly, even for a uniform density and the Euclidean metric, there are many possible CVT's for a square
- For example, here are three 16-point CVT's of the square



- Which CVT will a computer give you?

- Starting from any CVT, you will end up with a CVT that is as **hexagonal** as possible



- Why hexagons? Because they possess a certain optimality property (more on this later)

## CVT'S FOR DISCRETE SETS

- For discrete data sets, we can view CVT's as a **clustering algorithm**,
  - i.e., a method for subdividing a set into subsets, each of which contains elements with a closely related attribute
- In its simplest form, CVT clustering reduces to the very well-known **k-means** clustering algorithm (also known as **vector quantization**)
  - thus, CVT can be viewed as a generalization of k-means clustering
- There are, of course, many applications where clustering is important; we will discuss some of these later

## SOME PROPERTIES OF CVT'S

### Centroidal Voronoi tessellations as minimizers

Given

- $\Omega \subset \mathbb{R}^N$
- a positive integer  $K$
- a density function  $\rho(\cdot)$  defined on  $\overline{\Omega}$

Let

- $\{\mathbf{z}_i\}_{i=1}^K$  denote **any** set of  $K$  points belonging to  $\overline{\Omega}$
- $\{V_i\}_{i=1}^K$  denote **any** tessellation of  $\Omega$  into  $K$  regions

Let

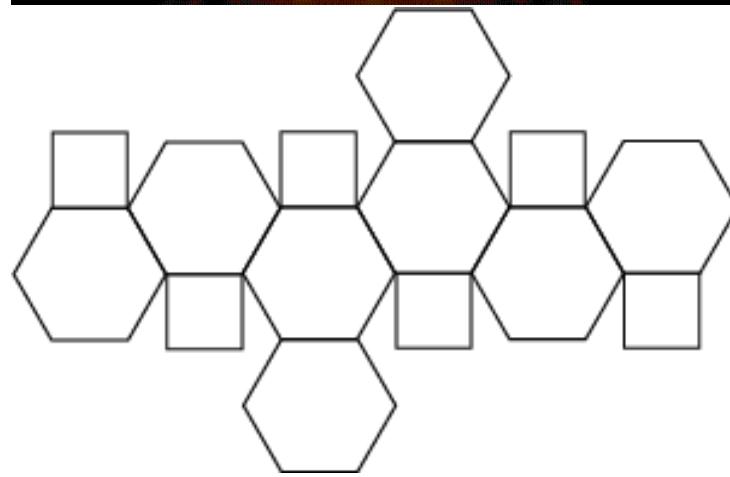
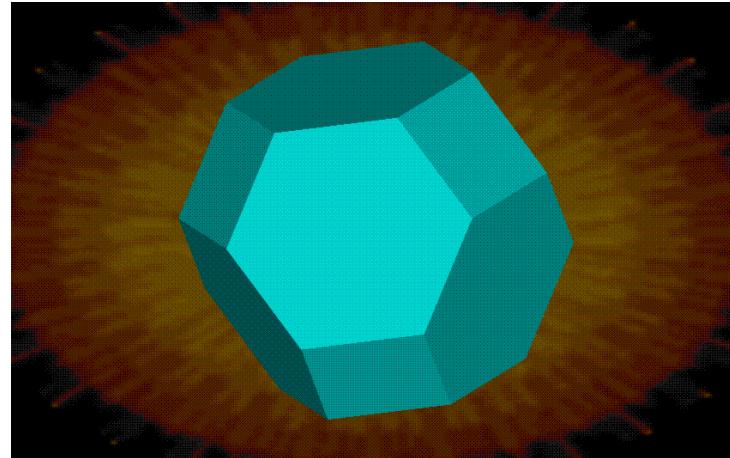
$$\mathcal{F}((\mathbf{z}_i, V_i), i = 1, \dots, K) = \sum_{i=1}^K \int_{\mathbf{y} \in V_i} \rho(\mathbf{y}) |\mathbf{y} - \mathbf{z}_i|^2 d\mathbf{y}$$

Then, a necessary condition for  $\mathcal{F}$  to be minimized is that  $\{V_i\}_{i=1}^K$  and  $\{\mathbf{z}_i\}_{i=1}^K$  form a centroidal Voronoi tessellation of  $\Omega$

- If  $\Omega \in \mathbb{R}^N$  is bounded, then  $\mathcal{F}$  has a global minimizer
- Assume that  $\rho(\cdot)$  is positive except on a set of measure zero in  $\Omega$ 
  - then  $\mathbf{z}_i \neq \mathbf{z}_j$  for  $i \neq j$
- For general metrics, existence is provided by the compactness of the Voronoi regions; uniqueness can also be attained under some assumptions, e.g., convexity, on the Voronoi regions and the metric
- There are many additional results available for the discrete case
  - many of these are in the nature of limiting results as the sample size increases

## Gersho's conjecture (proven in 2D under assumptions)

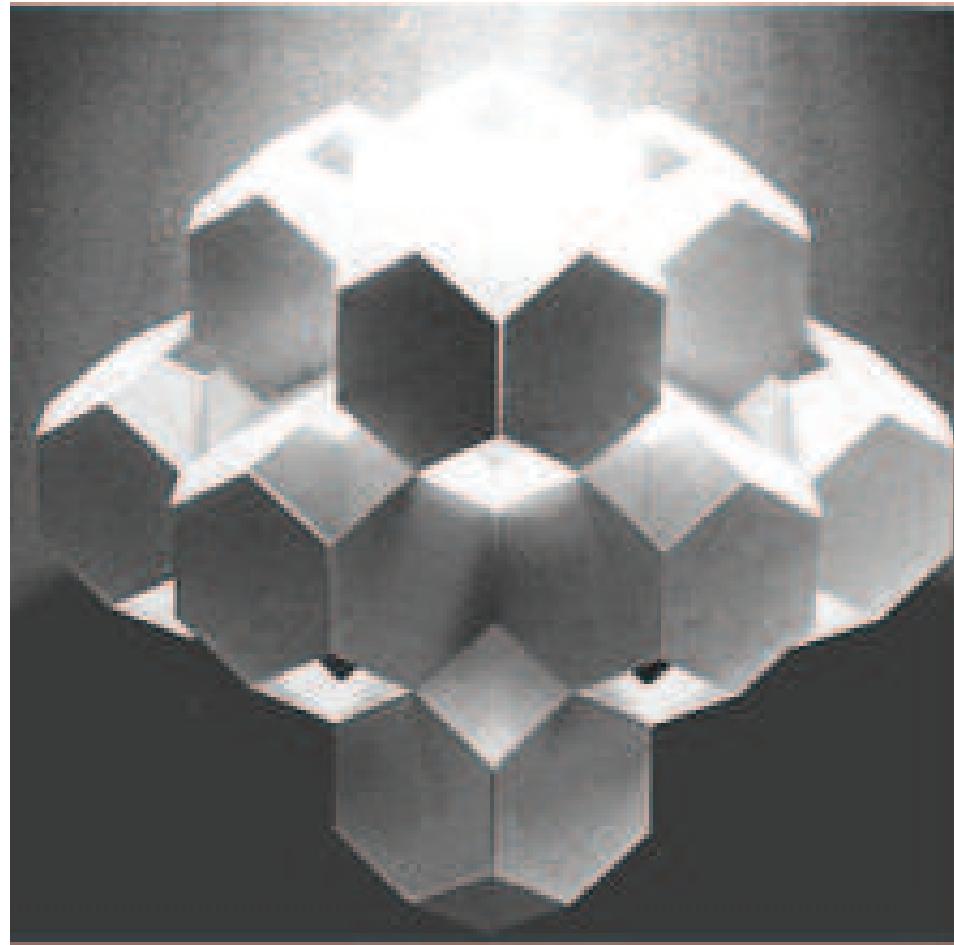
- For any density function, as the number of points increases, **the distribution of CVT points becomes locally uniform**
  - this means that if one looks at small enough patch and a large enough number of points, a CVT point distribution will look uniform, regardless of how nonuniform it is globally
- Gersho's conjecture is a key observation that helps explain the effectiveness of CVT's (more on this later)
- Locally in 2D, CVT Voronoi regions are always congruent **regular hexagons**
  - that is, locally, CVT points are always vertices of congruent equilateral triangles
  - locally, dual Delaunay CVT grids are always congruent equilateral triangles
- In higher dimensions, the basic cell of a CVT is not known
  - in 3D, computational studies indicate that it is a **truncated octahedron**



*The truncated octahedron and the paper cutout for its construction*

From: <http://www.ul.ie/~cahird/polyhedronmode/truncate1.htm>

<http://mathworld.wolfram.com/TruncatedOctahedron.html>

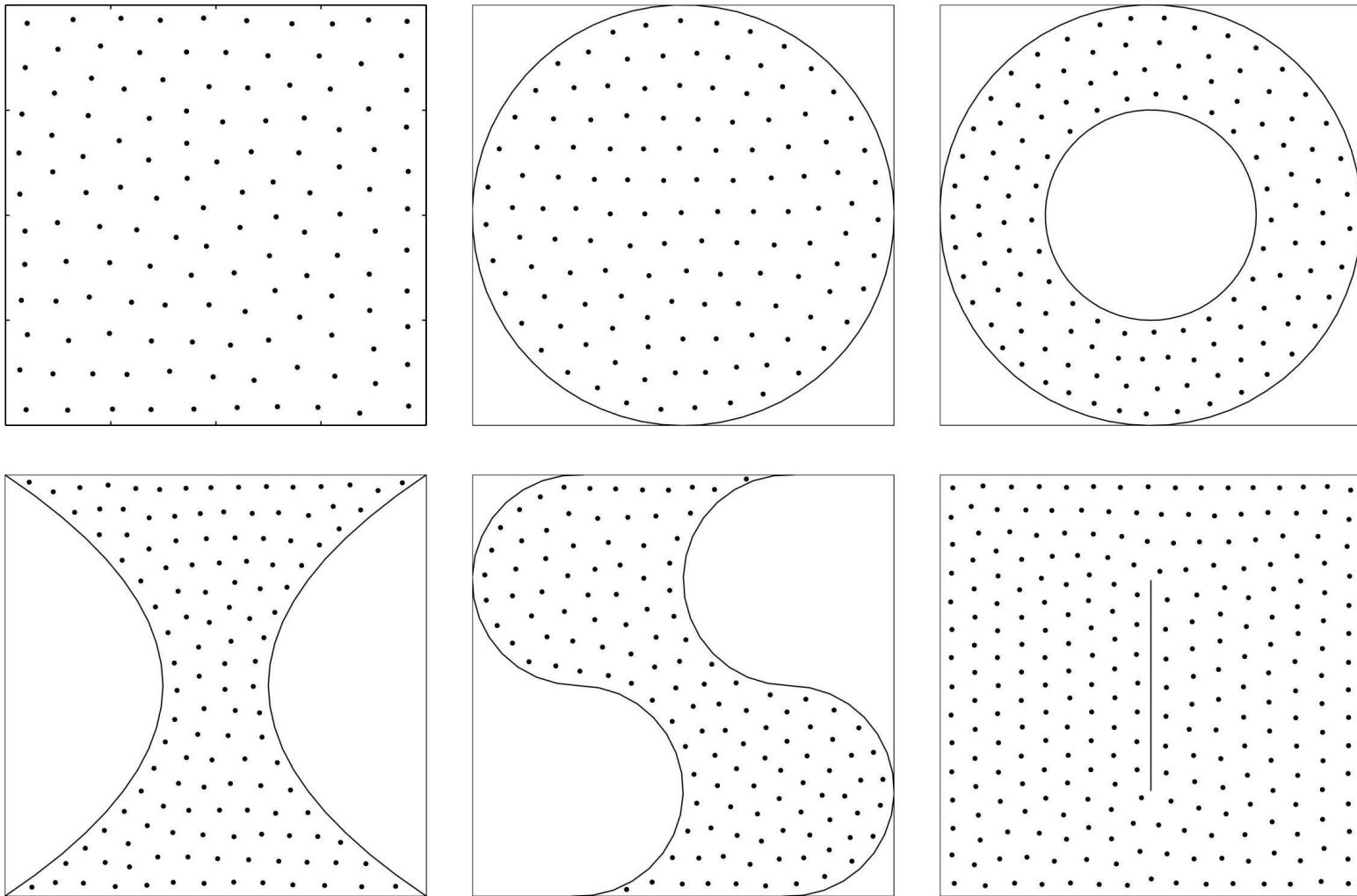


*Truncated octahedra tessellating  $\mathbb{R}^3$*

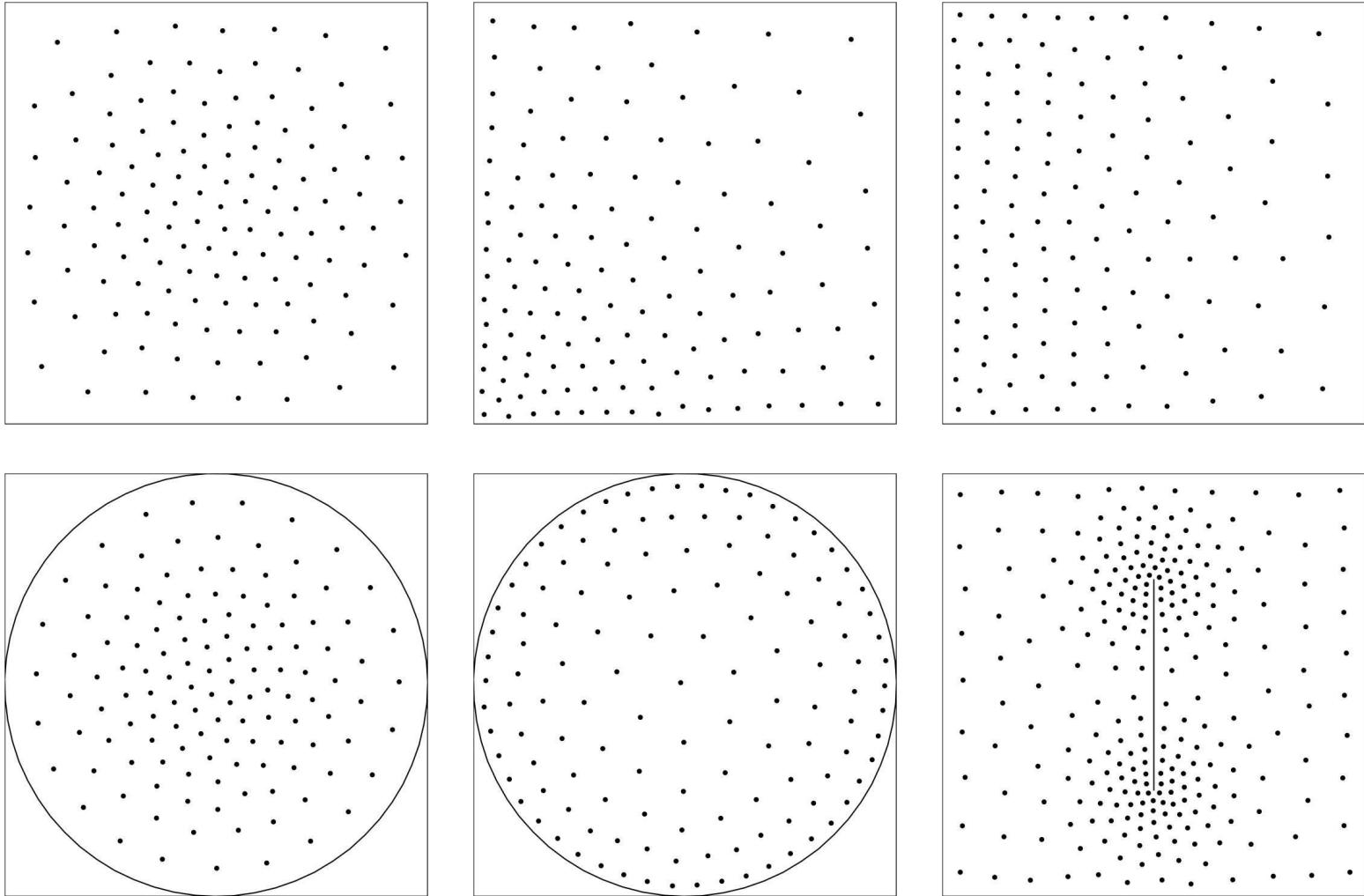
From: N.-K. Khumbah, *Mathematical Quantization for Massive Data Sets*, Ph.D dissertation, George Mason University, 2000

## Boundary conformity of CVT points

- Under the CVT algorithm, points are repelled by other points
  - this results in good spacing between points
- CVT points are also repelled by the boundary
  - this results in the points conforming to the boundary



*Uniform CVT point distributions*



*Non-uniform CVT point distributions*

## ALGORITHMS FOR CONSTRUCTING CVT'S

---

### Lloyd's method

---

0. Start with some initial set of  $K$  points  $\{z_i\}_{i=1}^K$
  1. Construct the Voronoi tessellation  $\{V_i\}_{i=1}^K$  of  $\Omega$  associated with the points  $\{z_i\}_{i=1}^K$
  2. Construct the centers of mass of the Voronoi regions  $\{V_i\}_{i=1}^K$  found in Step 1; these centroids are the new set of points  $\{z_i\}_{i=1}^K$
  3. Go back to Step 1, or, if you are happy with what you have, quit
- 

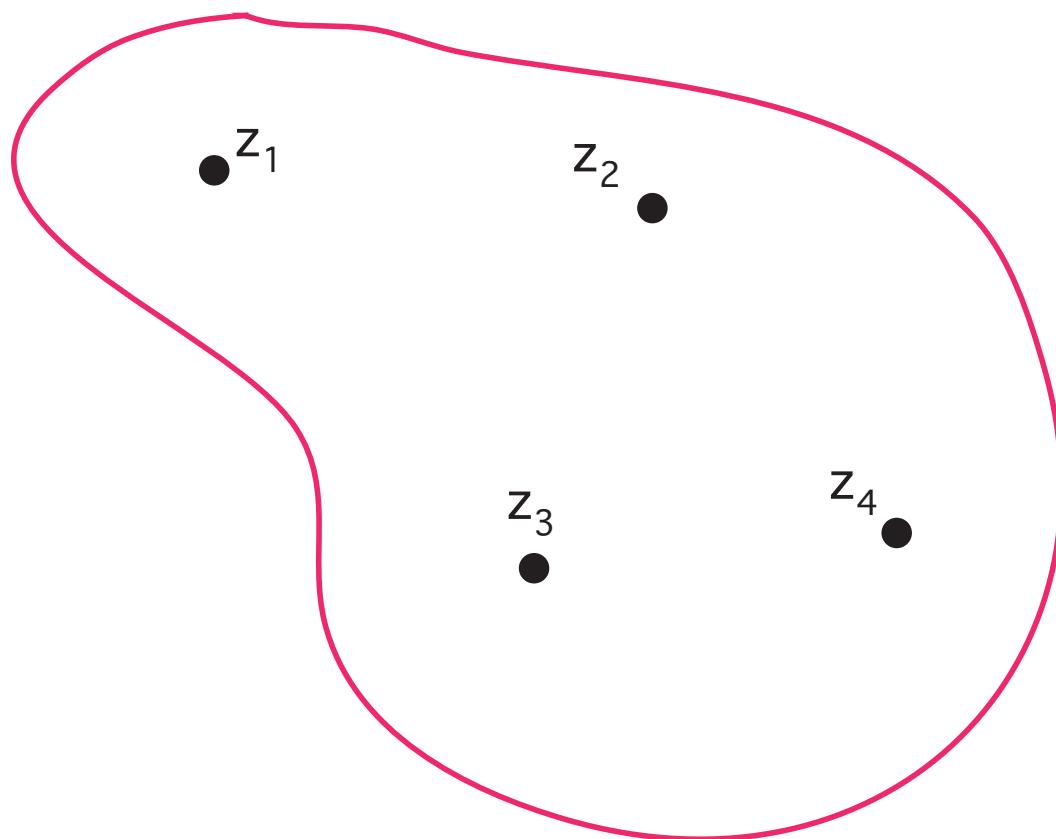
- Steps 1 and 2 can both be costly to effect

---

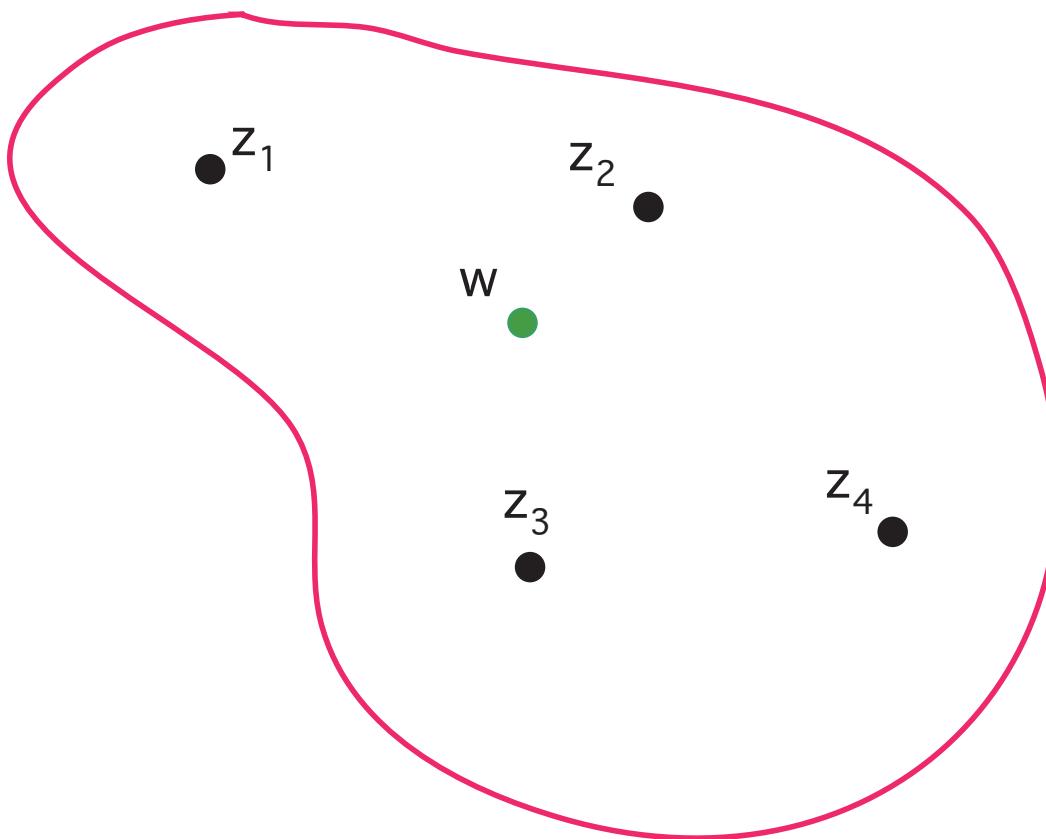
## McQueen's method (random sampling and averaging)

---

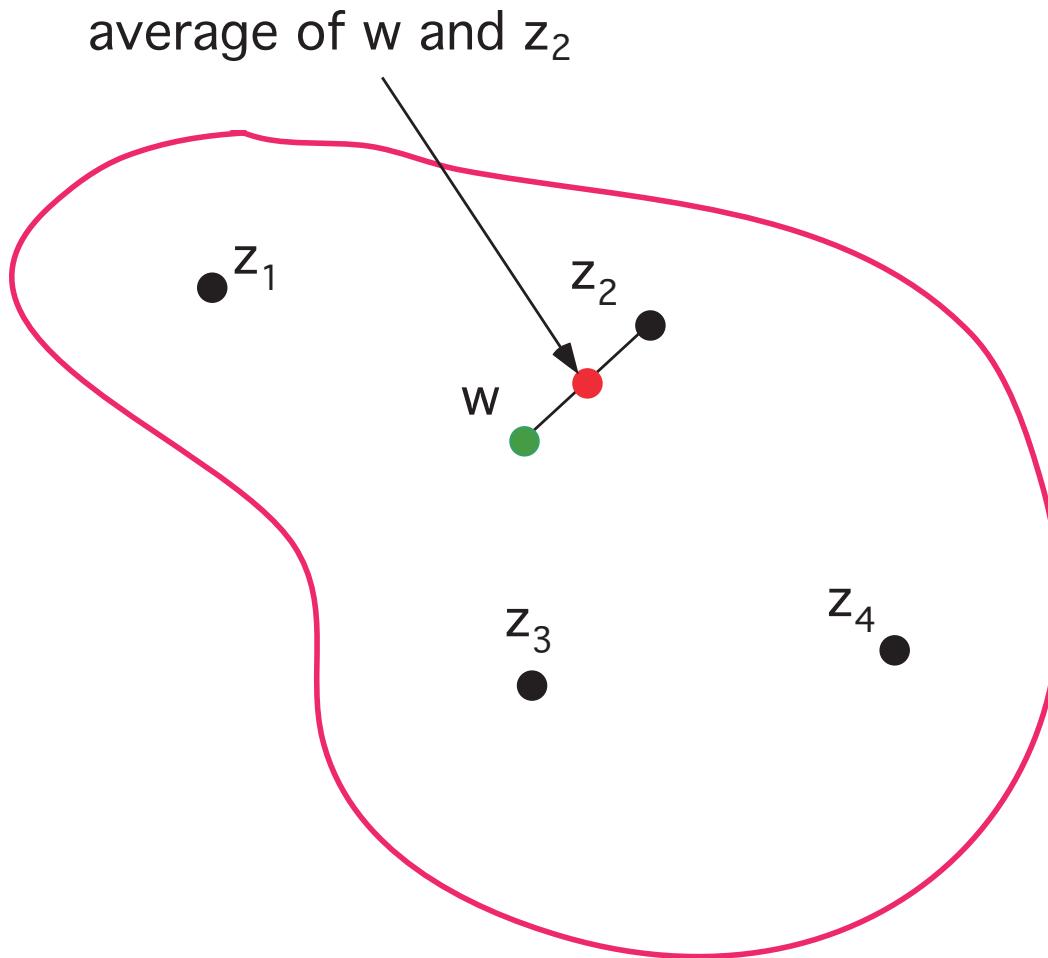
- Start with some initial set of  $K$  points  $\{z_i\}_{i=1}^K$  ( $K = 4$  in the sketch)



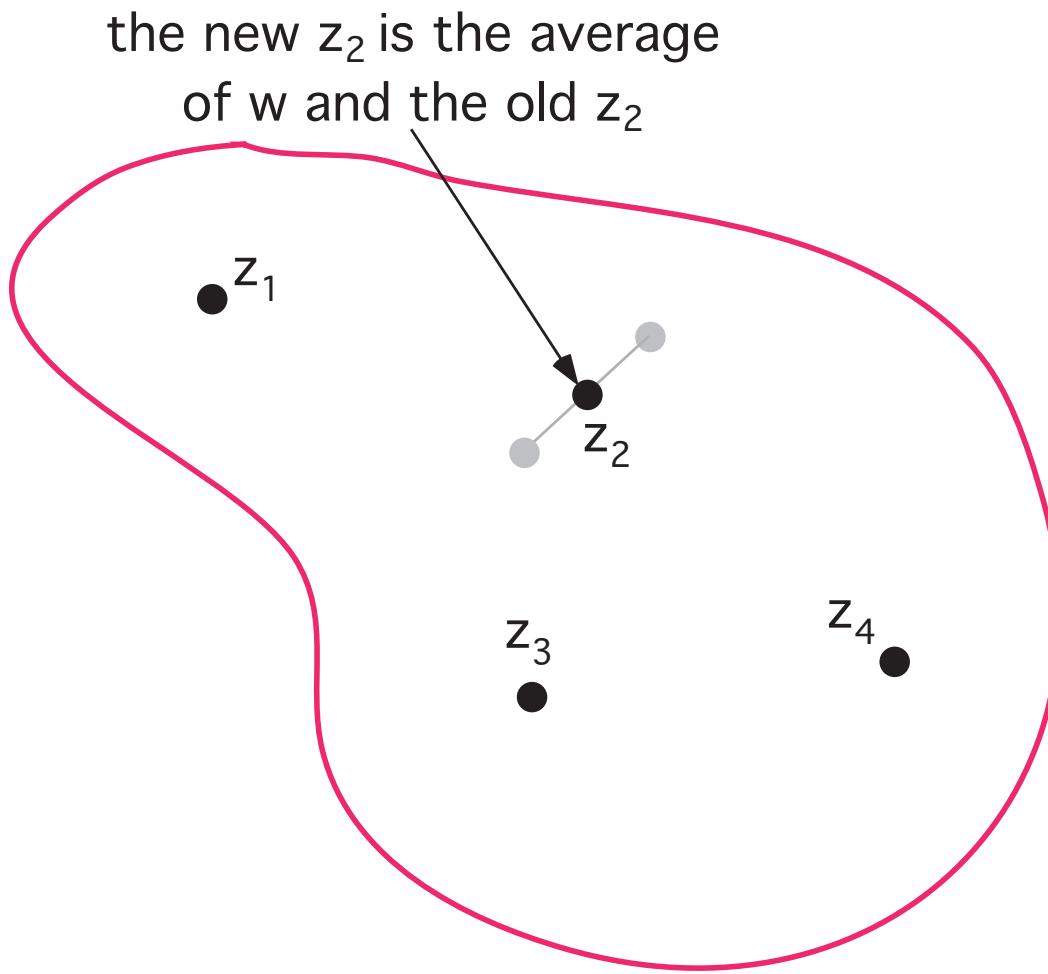
- Sample another point  $w$
- Determine which of the  $z_i$ 's is closest to  $w$  (it is  $z_2$  in the sketch)



- Find the average of  $w$  and the  $z_i$  closest to it



- Replace the  $z_i$  by the average point

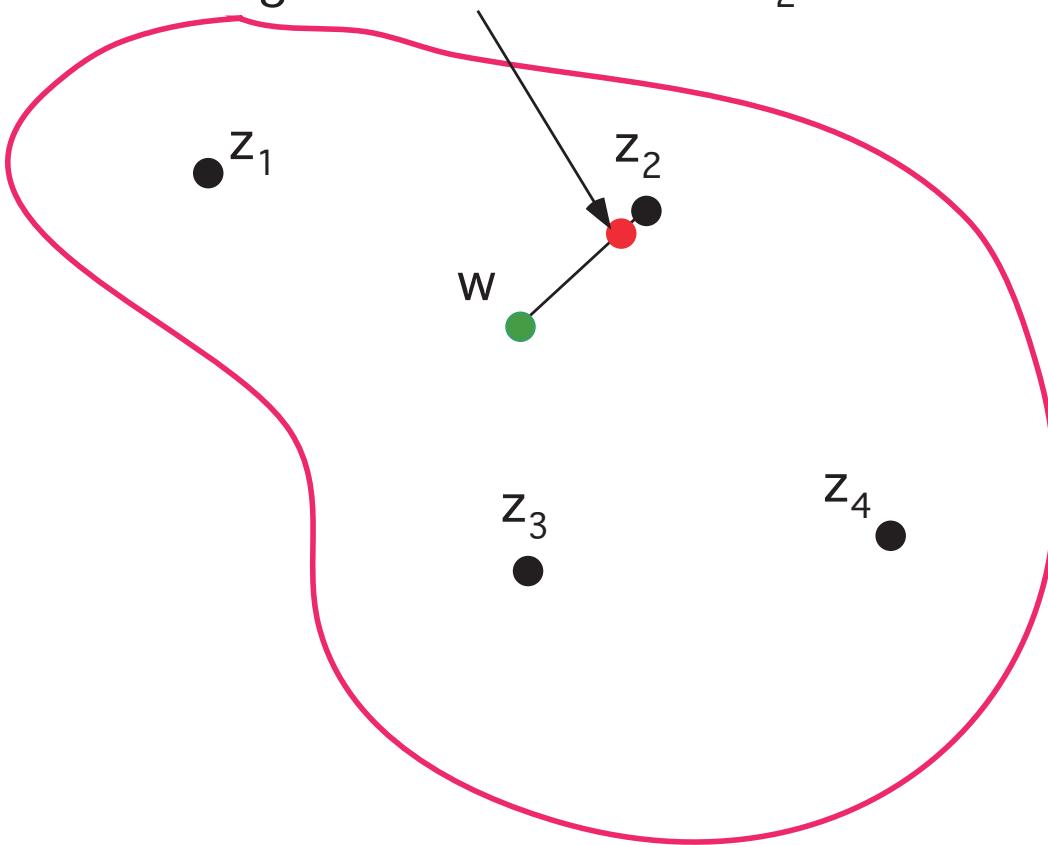


- Continue the process, that is,
  - sample points  $w$
  - find the closest  $z_i$
  - average  $w$  and that  $z_i$
  - replace that  $z_i$  by the average

except

- we keep track of how many times each point has been previously updated
- when we do the averaging, we weight the old point according the number of times it has been previously updated
- For example, suppose  $z_2$  had already been updated 12 times (counting the initial positions as the first update); then
  - instead of the new  $z_2 \leftarrow \frac{w + z_2}{2}$
  - we have the new  $z_2 \leftarrow \frac{w + 12z_2}{13}$

since  $z_2$  had previously updated,  
the new  $z_2$  is the weighted  
average of  $w$  and the old  $z_2$



- McQueen's method doesn't require the construction of Voronoi sets or centers of mass
  - despite this, the points produced by McQueen's method converge to the generators of a centroidal Voronoi tessellation
- The convergence of McQueen's method is very slow, that is, it takes many steps (millions) to obtain a set of CVT points from an initial set of points
  - the problem with McQueen's method is that it samples only a single point before it averages

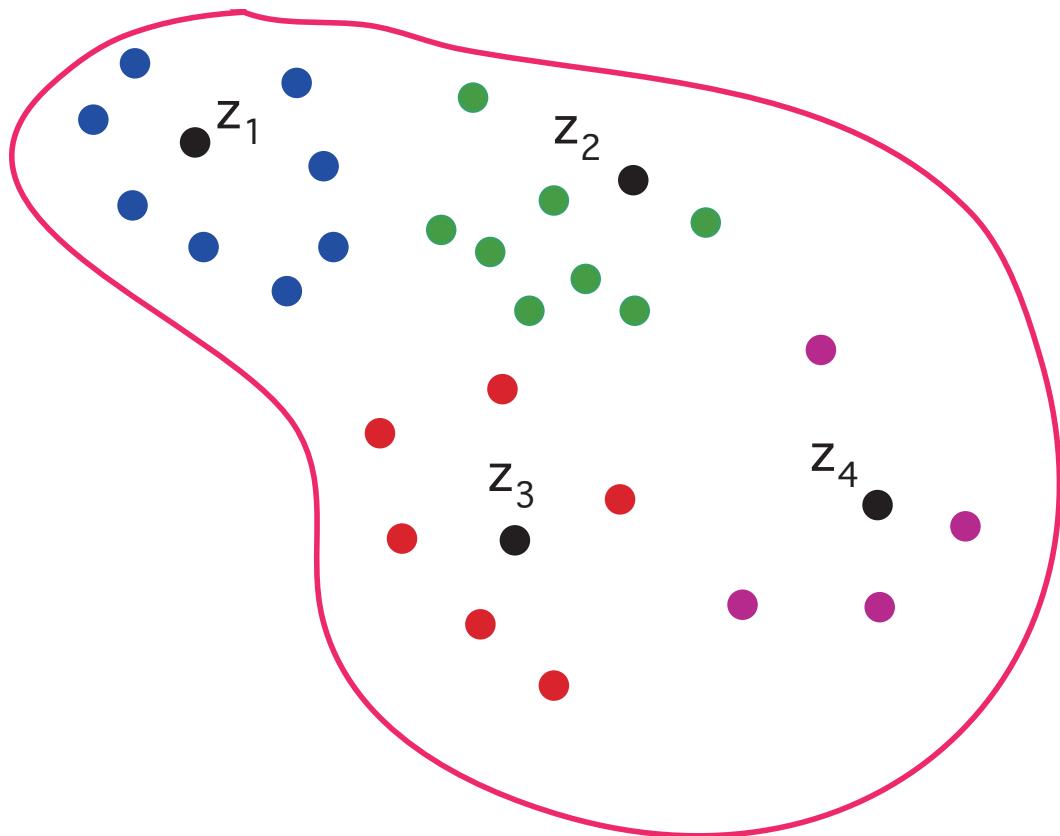
---

## New method

---

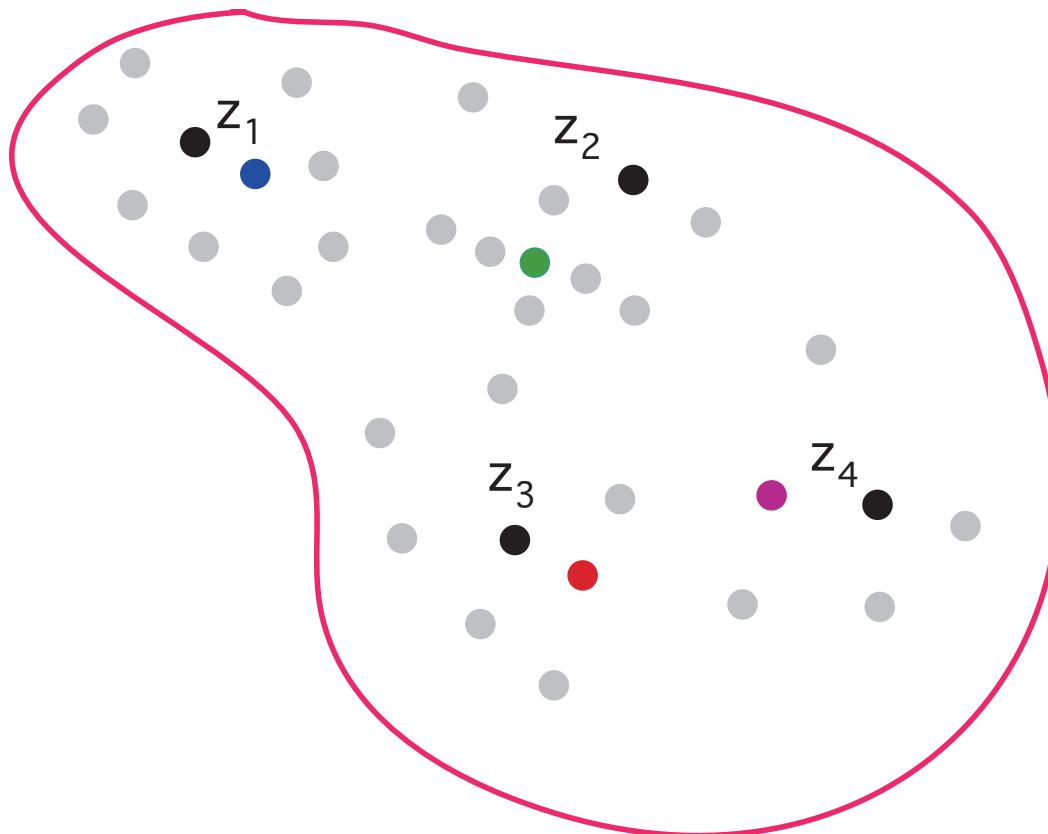
- sample lots of points before averaging
- good for parallel computing

- Sample lots of points (thousands) and group (cluster) them according to which is the nearest  $z_i$



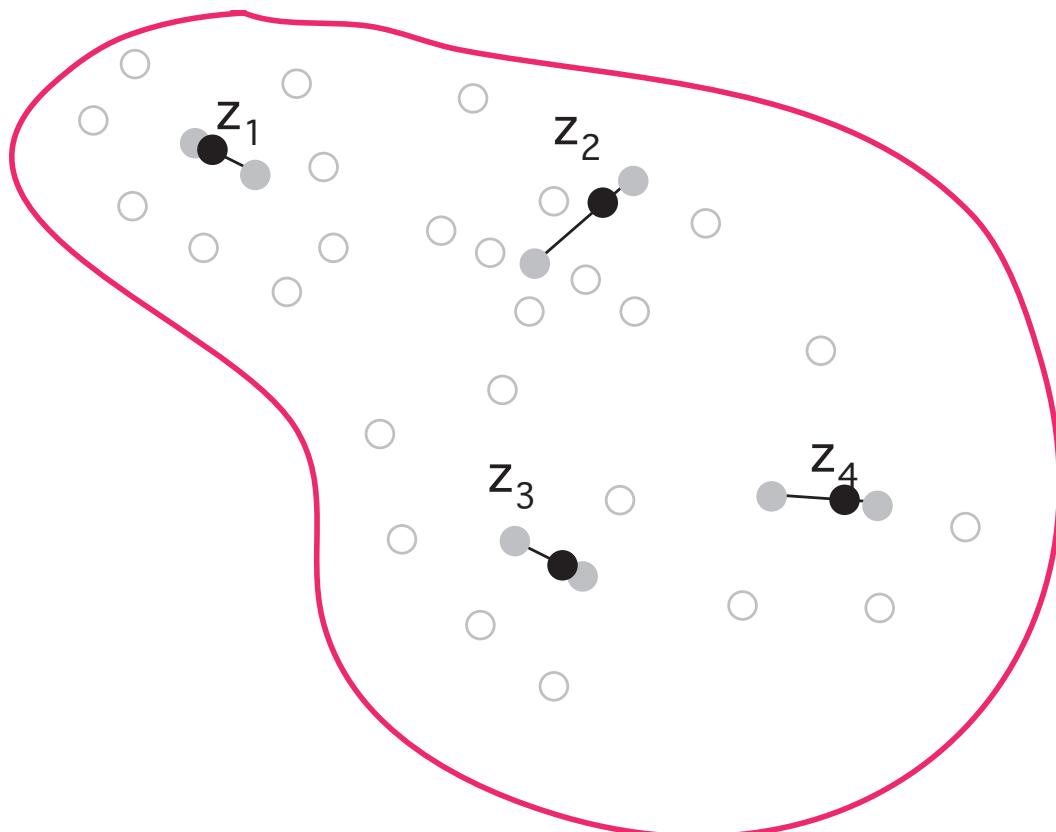
- sampled points nearest  $z_1$
- sampled points nearest  $z_2$
- sampled points nearest  $z_3$
- sampled points nearest  $z_4$

- Find the average of each of the clusters

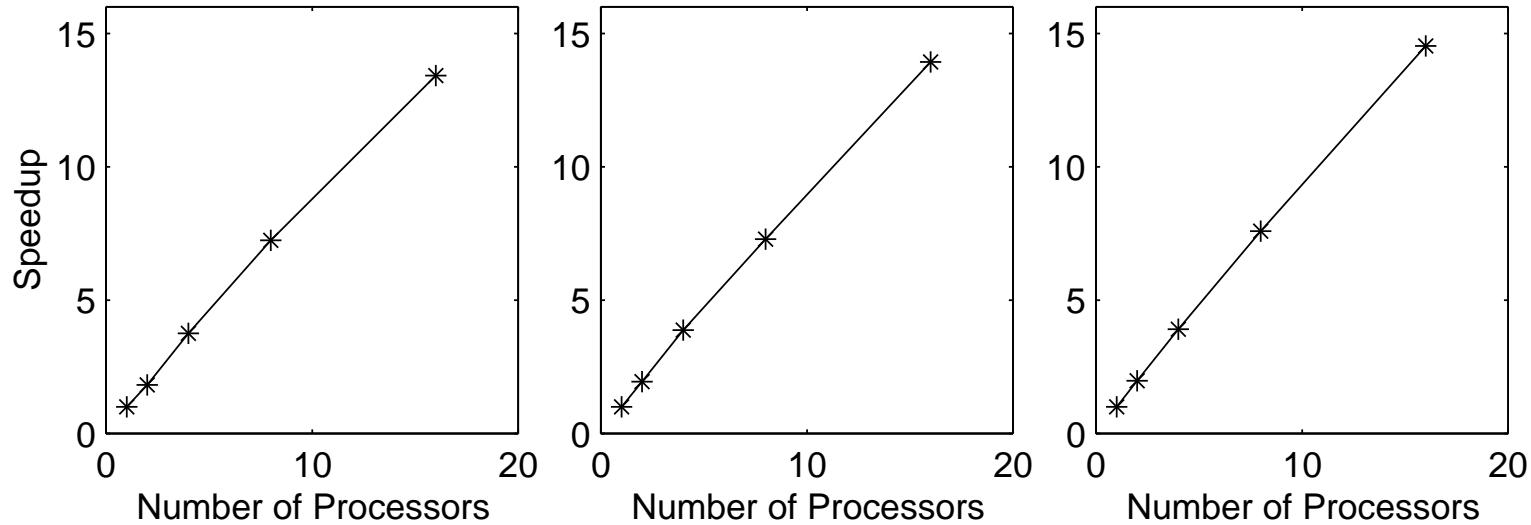


- average of points nearest  $z_1$
- average of points nearest  $z_2$
- average of points nearest  $z_3$
- average of points nearest  $z_4$

- The new  $z_i$ 's are a weighted average of the old  $z_i$ 's and the corresponding cluster averages



- Parallel versions of the new method exhibit near-perfect scalability



*Speedup of a parallel implementation of the new algorithm for three different density functions*

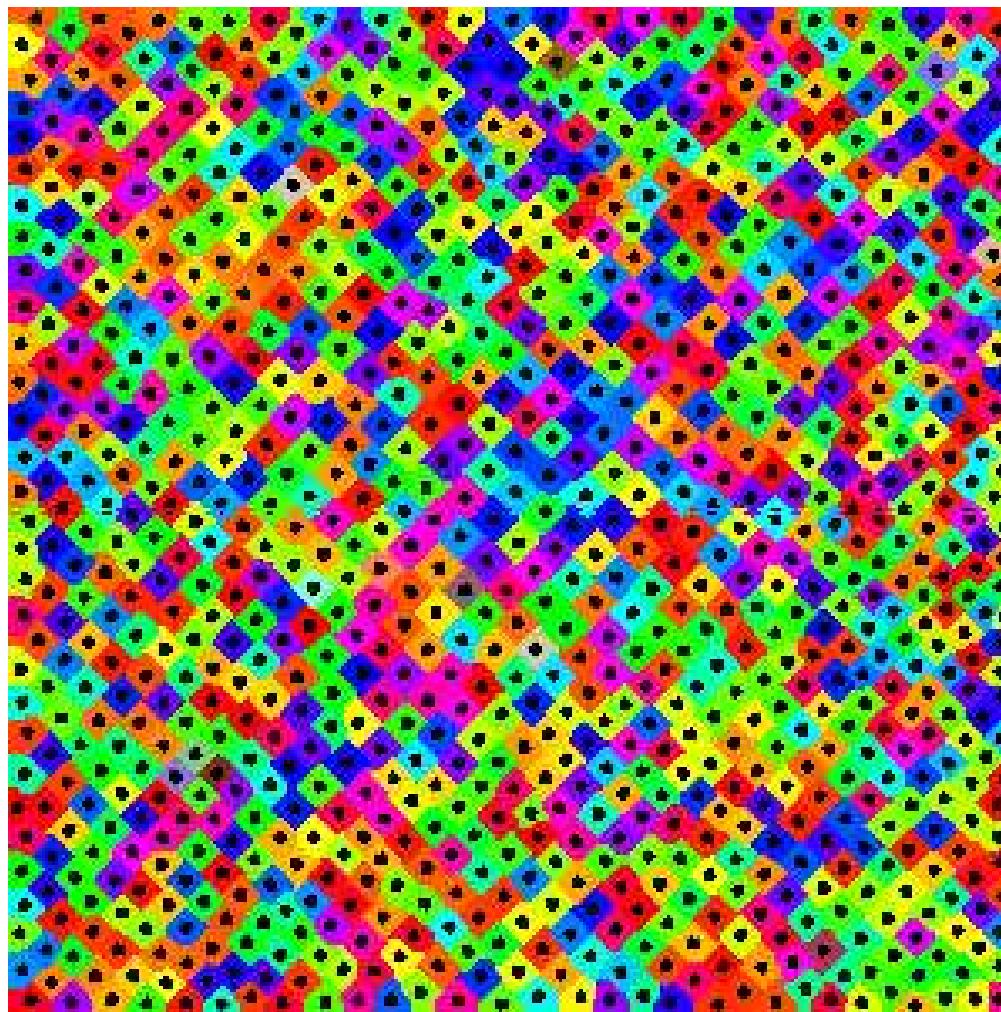
- We have, in fact, developed a two parameter family of effective, probabilistic methods for generating CVT's in general regions and with general point density distributions

- There exist other deterministic method for constructing CVTs, including Newton and multigrid methods

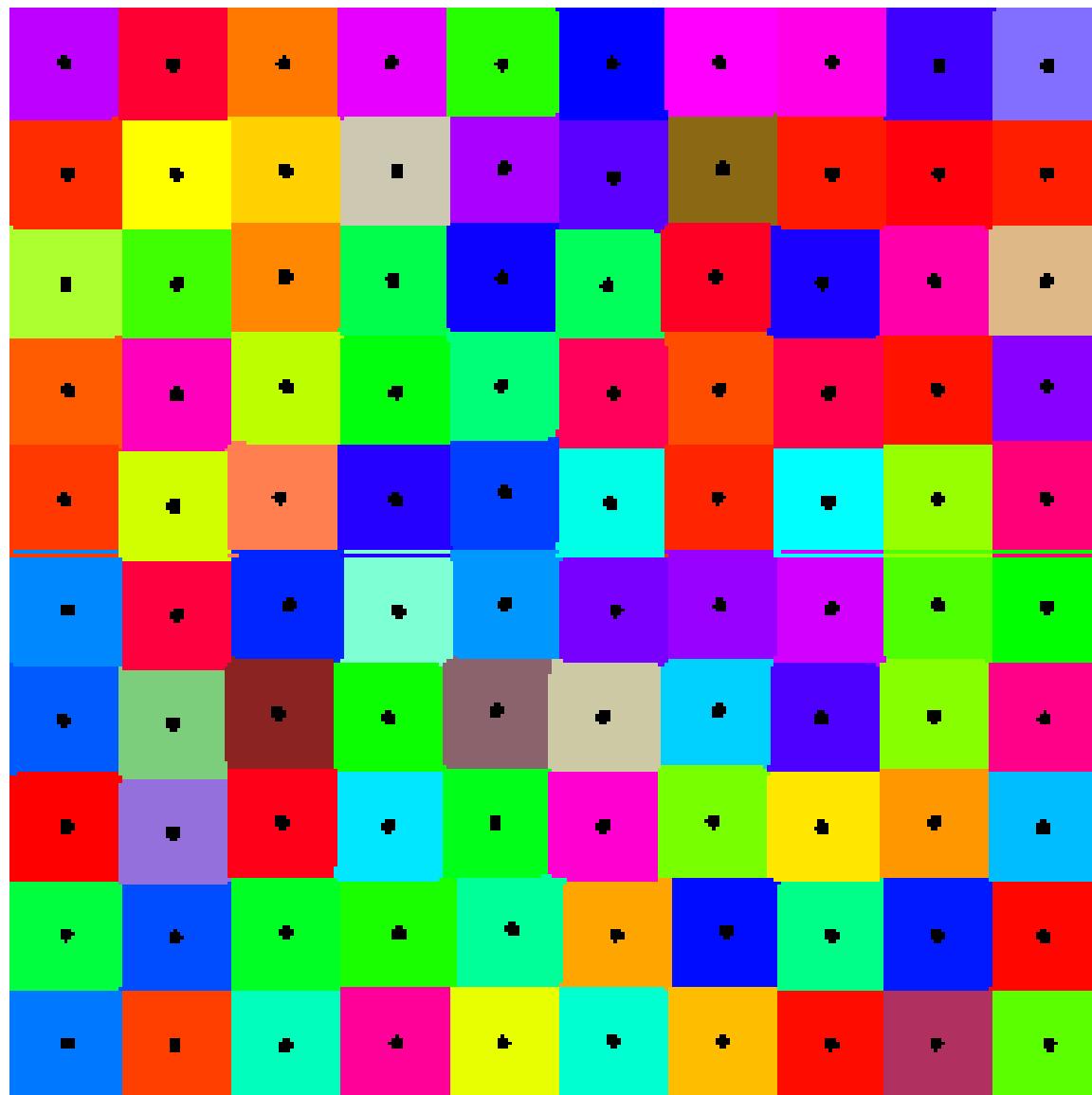
## GENERALIZATIONS OF CVT

- CVT's for other metrics
- CVT's for other types of generators
- Constrained CVT's for placing some or all the points on a surface
- Constrained CVT's for fixing the position of some points
- CVT's for anisotropic point distributions

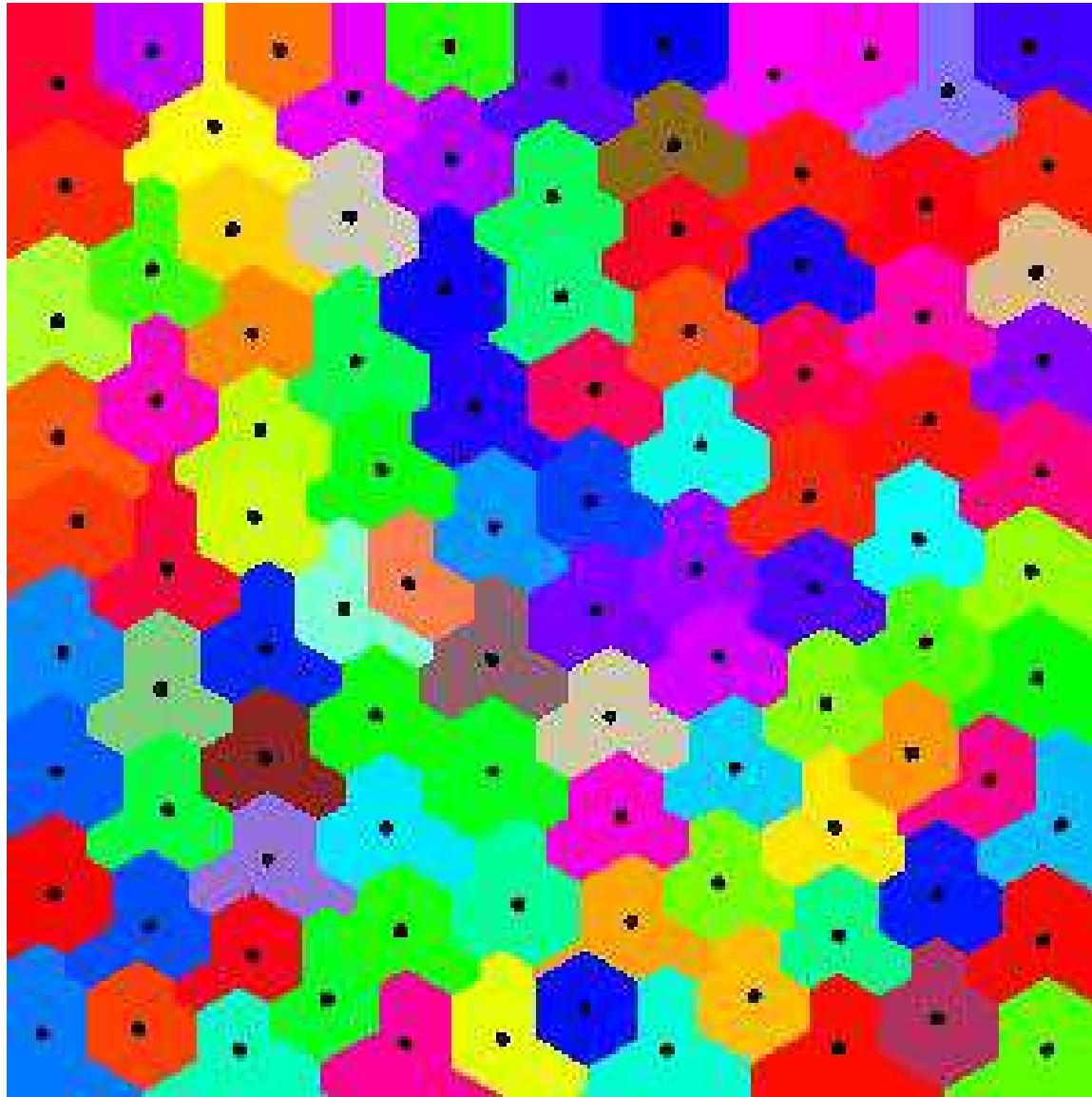
## CVT'S OF THE SQUARE FOR OTHER METRICS



*CVT for the  $L^1$  metric*

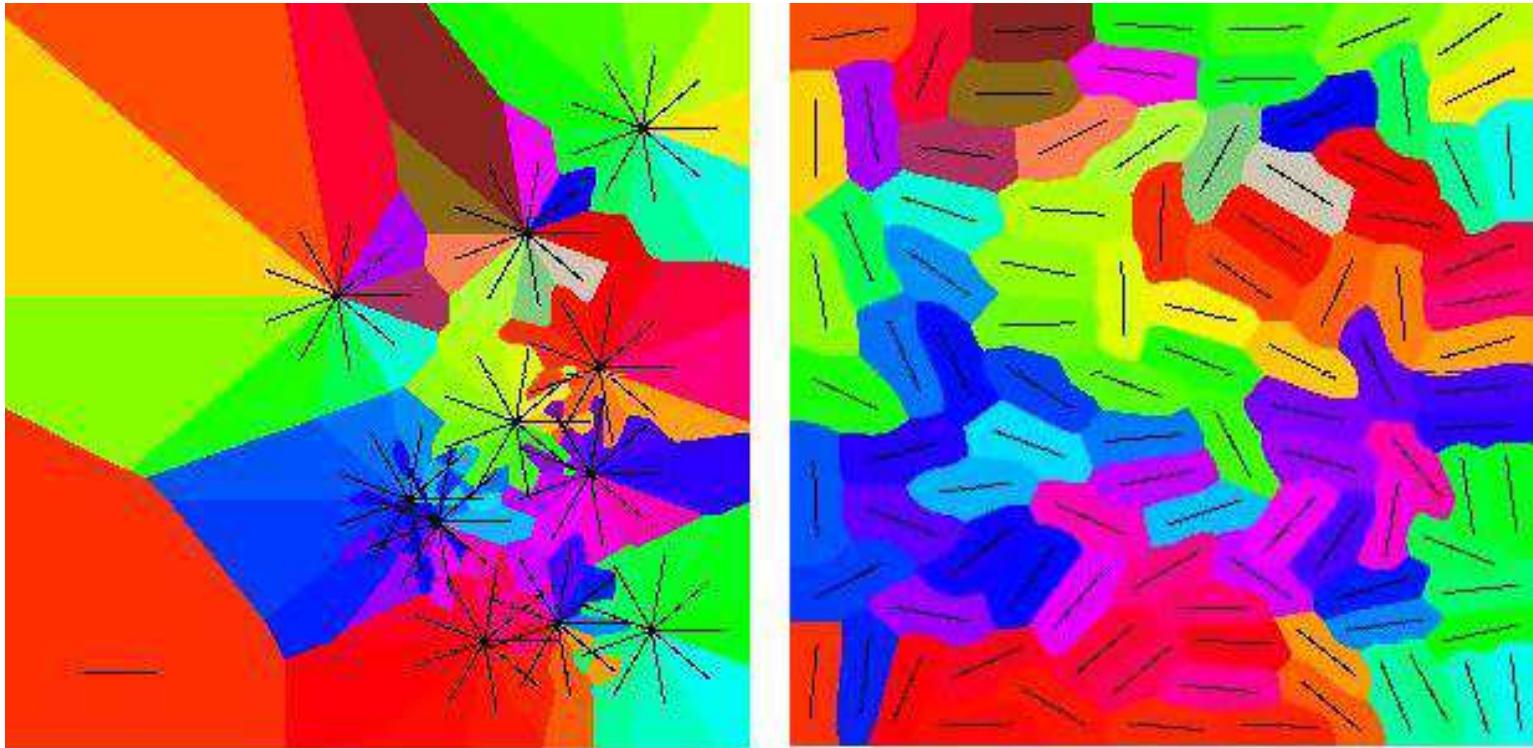


*CVT for the  $L^{20}$  metric (approximating the  $L^\infty$  metric)*

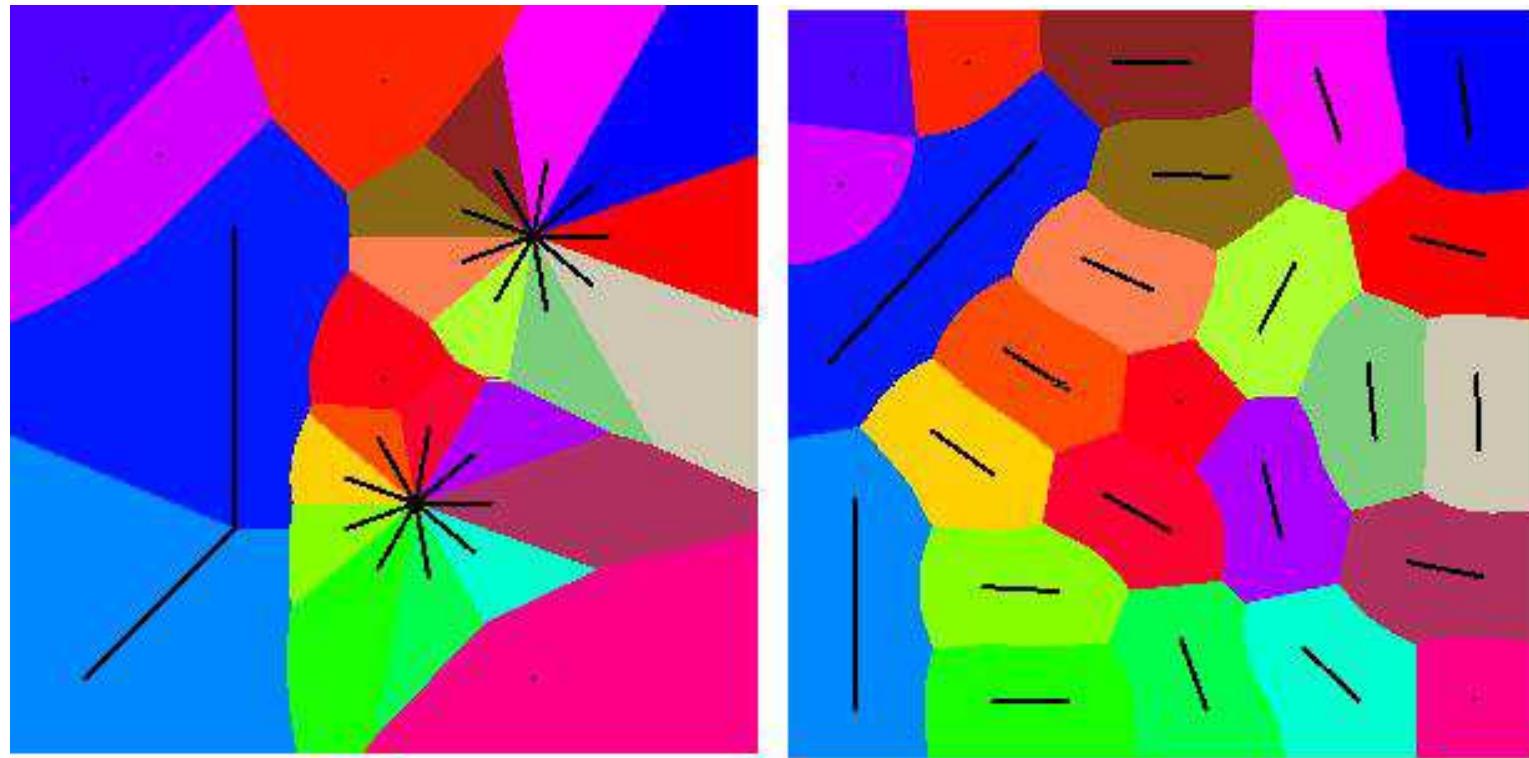


*CVT for the triangle metric*

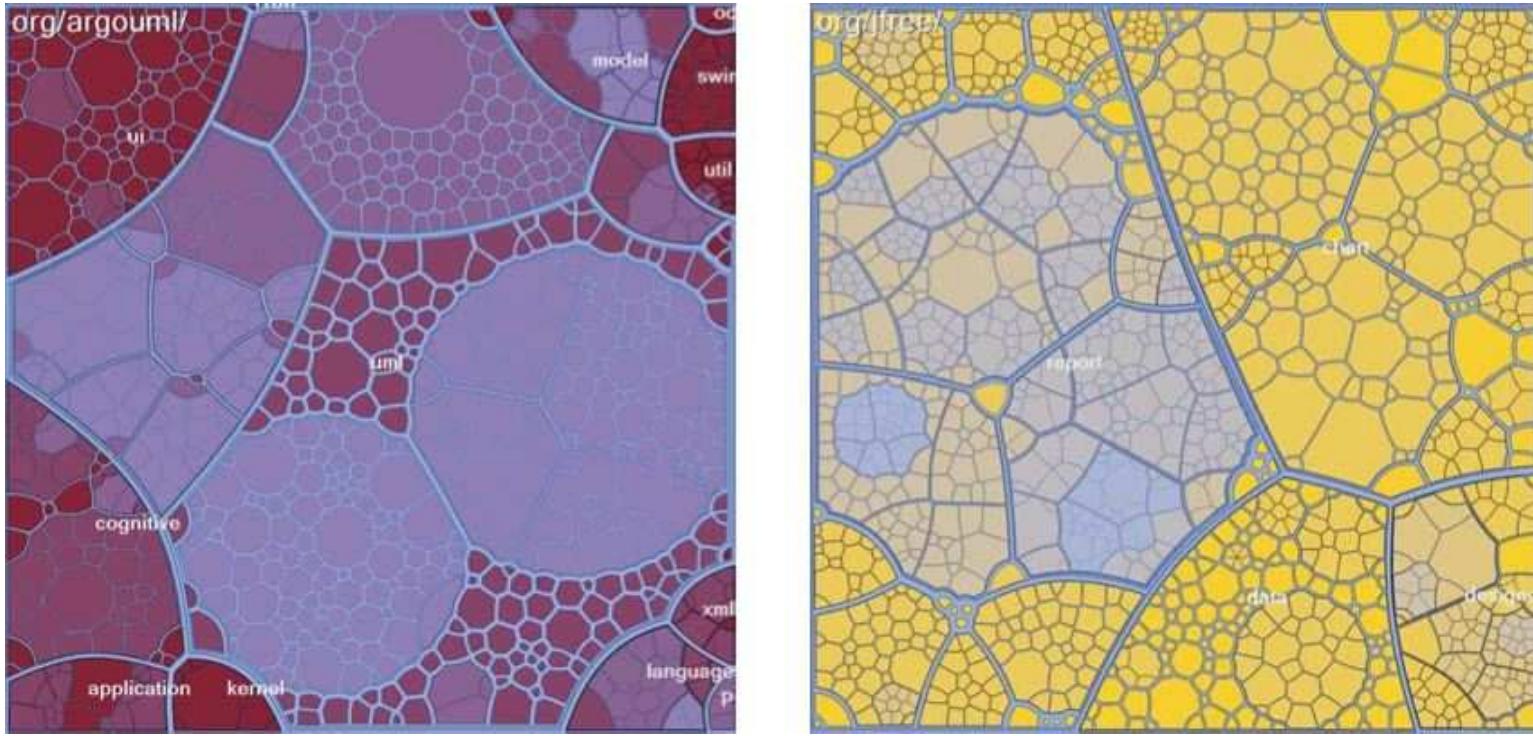
## CVT'S OF THE SQUARE FOR GENERALIZED GENERATORS



*CVT with lines; left: initial configuration; right: CVT configuration*



*CVT with points and lines; left: initial configuration; right: CVT configuration*

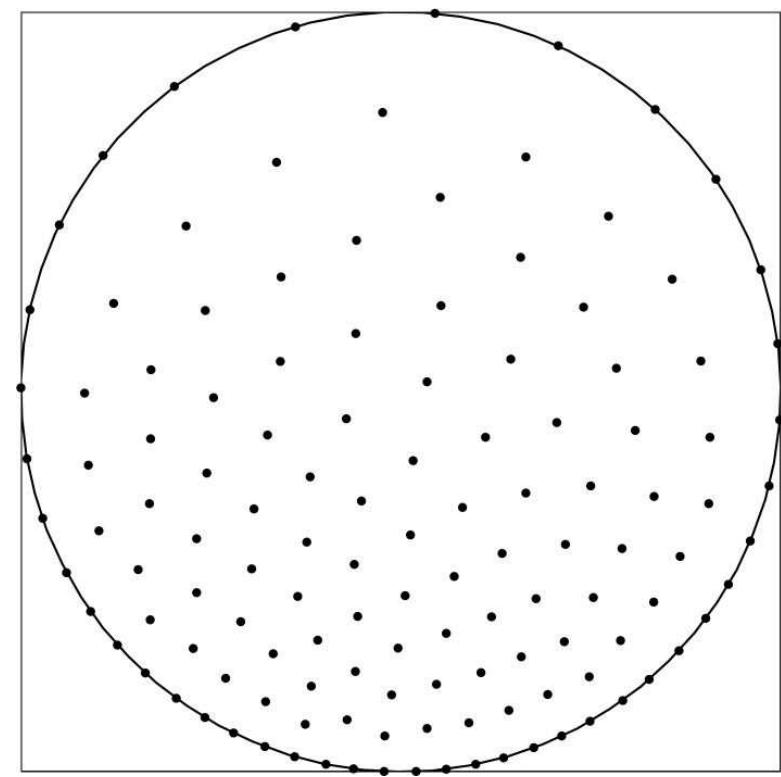
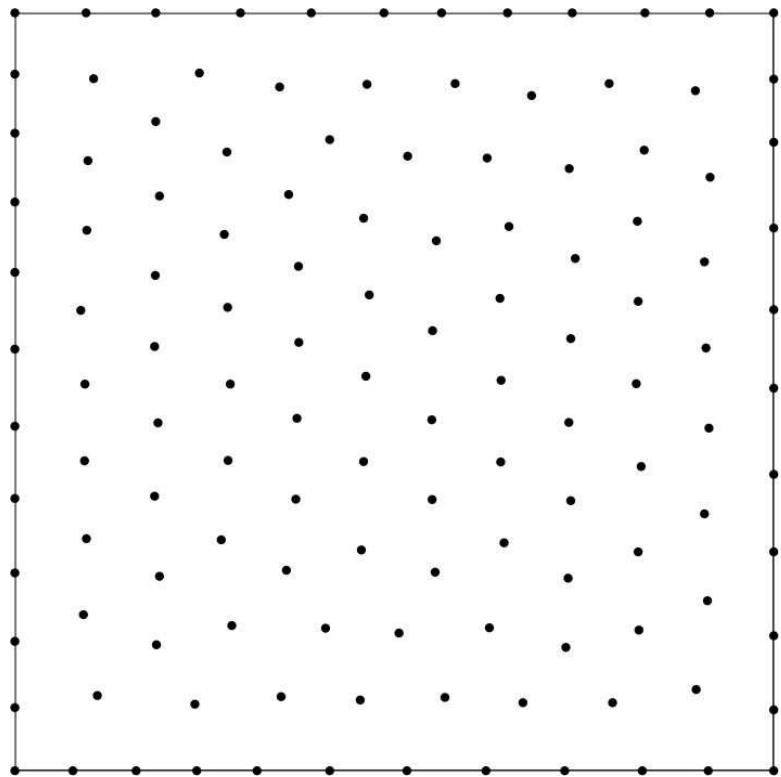


*Examples of a CVT with generators being circles, i.e., having nonzero area*

## CVT'S WITH POINTS ON THE BOUNDARY

- For most grid generation applications, one needs points on the boundary
  - for the basic CVT method, the boundary repels points so that CVT points never lie on the boundary
- However, generalized CVTs can be defined to have some of the points lie on the boundary — there are three ways to do this
  1. a set of fixed points on the boundary can be specified, and then a generalized CVT method will optimally place the interior points
  2. a set of points on the boundary can be specified, and then another generalized CVT method will optimally place points in the interior and optimally slide the boundary points
  3. no points on the boundary are specified, and then another generalized CVT method will optimally place points in the interior and on the boundary

- We usually take the third approach because it produces the best point distributions
- The key steps in, e.g., a Lloyd's method approach, are to:
  - after each iteration, determine the boundary Voronoi regions, i.e., those Voronoi regions that intersect the boundary, then the following steps are carried out, with later steps excluding points that have been treated in previous steps
  - if a Voronoi region contains a vertex or corner of the domain, move the generator to that corner
  - if a Voronoi region intersects an edge, project the generator to that edge
  - if a Voronoi region intersects the boundary, project the generator to that boundary
  - do not change the positions of the interior Voronoi regions



*CVTs with points on the boundary*

## CVT'S ON SURFACES

- Generalizations of the basic CVT method can be defined so that all the points end up on the boundary or on a surface
- These generalization can be implemented so that ordinary Euclidean metrics and center of mass definitions can be used
  - instead of using complicated geodesic distances along surfaces and/or centers of mass with respect to surfaces

## APPLICATIONS OF CVT'S

- optimal distribution of resources
- cell division
- finite volume methods for PDE's
- territorial behavior of animals
- data compression
- climate modeling

○

○

○

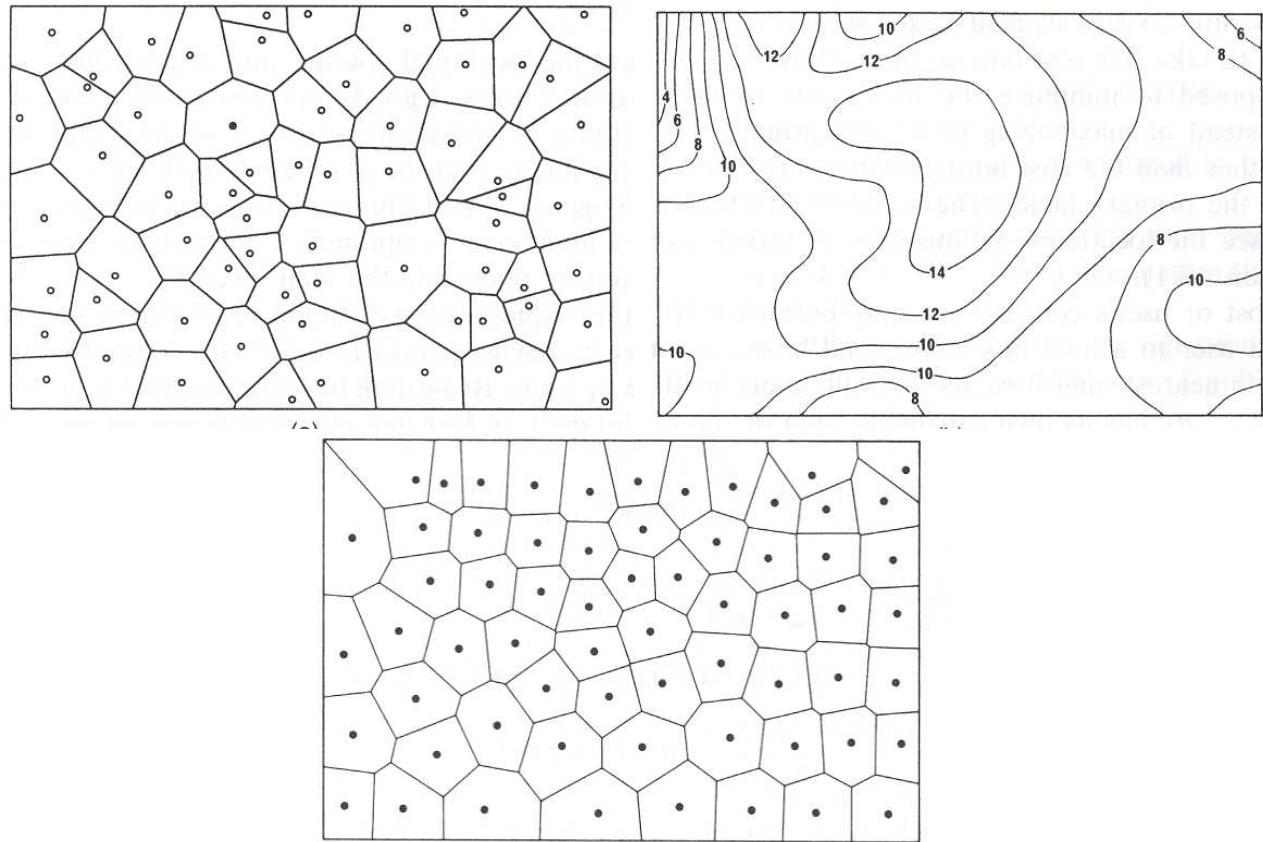
- Homer Simpson

## OPTIMAL DISTRIBUTION OF RESOURCES

What is the optimal placement of **mailboxes** in a given region?

- A user will use the mailbox nearest to their home
- The cost (to the user) of using a mailbox is proportional to the distance from the user's home to the mailbox
- The total cost to users as a whole is measured by the distance to the nearest mailbox averaged over all users in the region
- The optimal placement of mailboxes is defined to be the one that minimizes the total cost to the users

The optimal placement of the mail boxes is at the centroids of a centroidal Voronoi tessellation, using the population distribution as the density function in the center of mass definition

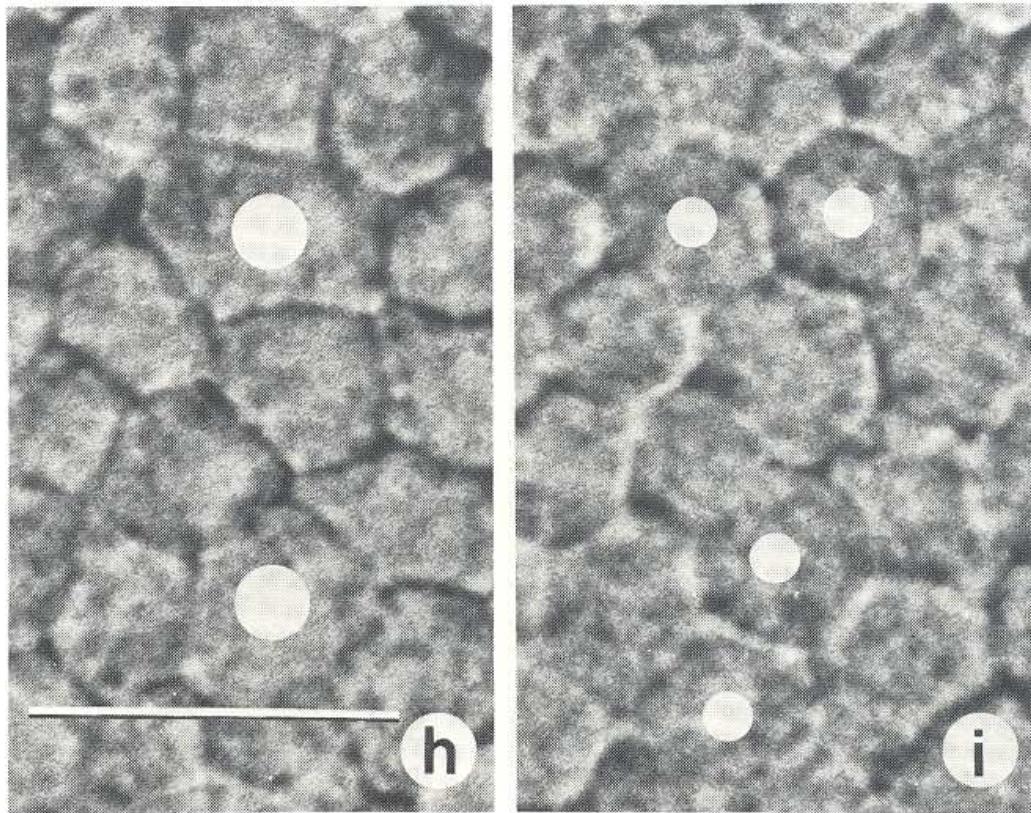


*Top left: Actual distribution of mailboxes in a part of Tokyo. Top right: population distribution in that district. Bottom: CVT distribution of mailboxes*

From: A. Okabe, B. Boots, K. Sugihara, and S. Chiu, *Spatial Tessellations*, Wiley, Chichester, 2002

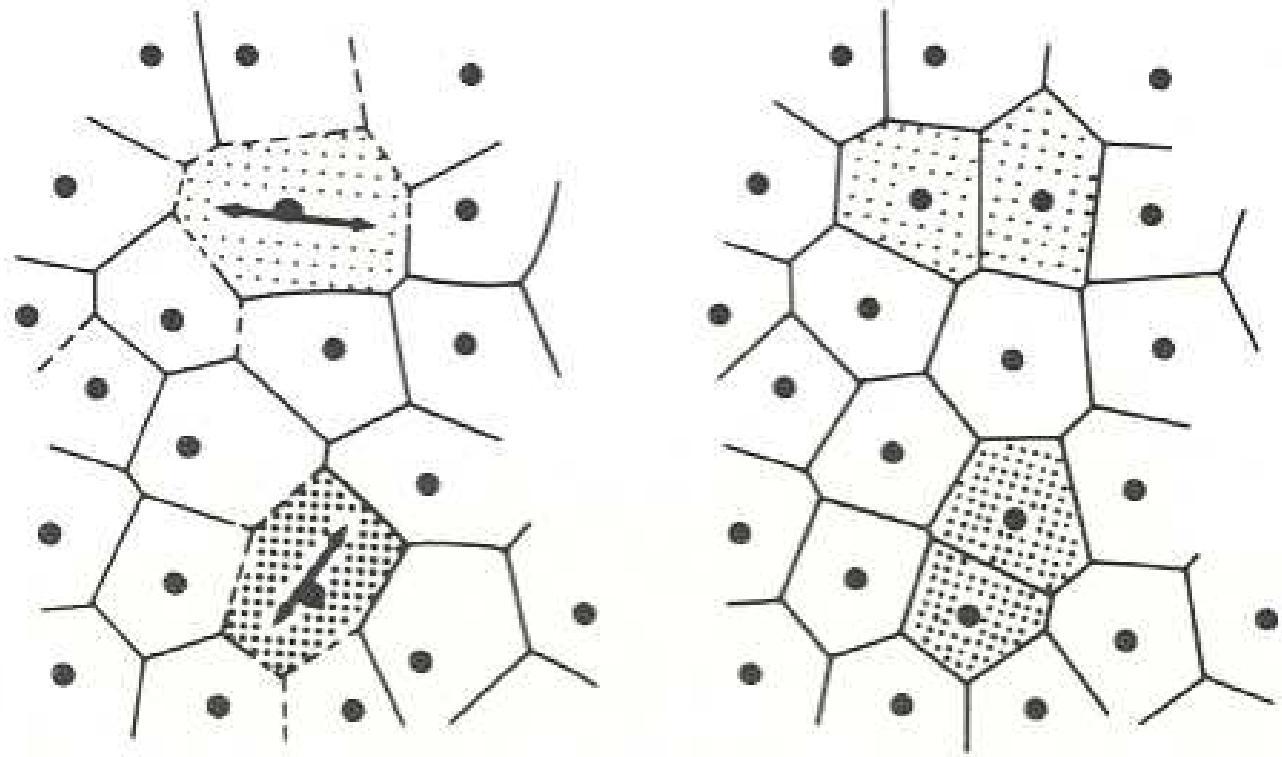
## CELL DIVISION

- There are many examples of cells that are polygonal – often they can be identified with a Voronoi, indeed, a centroidal Voronoi tessellation
  - this is especially evident in monolayered or columnar cells, e.g., as in the early development of a starfish (*Asteria pectinifera*)
- Cell division
  - Start with a configuration of cells that, by observation, form a Voronoi tessellation (this is very commonly the case)
  - After the cells divide, what is the shape of the new cell arrangement?
  - It is observed that the new cell arrangement is closely approximated by a centroidal Voronoi tessellation



*Actual cellular patterns of a starfish embryo before (left) and after (right) cell division. White circles on the left are the parent cells that divide into the four daughter cells on the right indicated by white dots*

From: H. Honda, Geometric models for cells in tissues, *International Review of Cytology* 81 1983, pp. 191–248



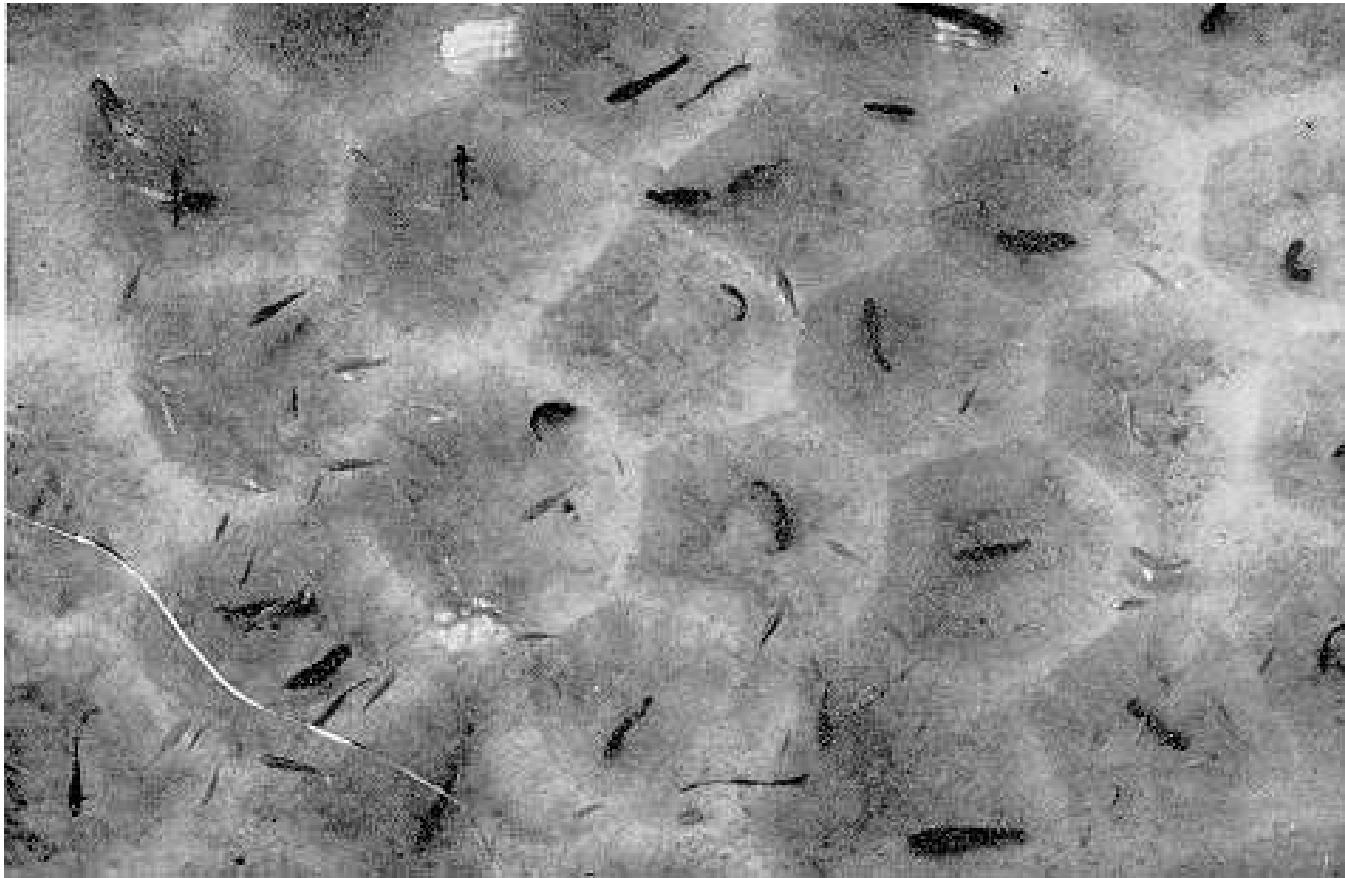
*Left: Actual cellular patterns before cell division traced from photograph. Right: CVT-based cellular patterns after two parent cell generators are allowed to separate. The CVT-based cellular pattern can be shown to be close to the actual cellular pattern after cell division*

From: H. Honda, Geometric models for cells in tissues, *International Review of Cytology* 81 1983, pp. 191–248

## TERRITORIAL BEHAVIOR OF ANIMALS

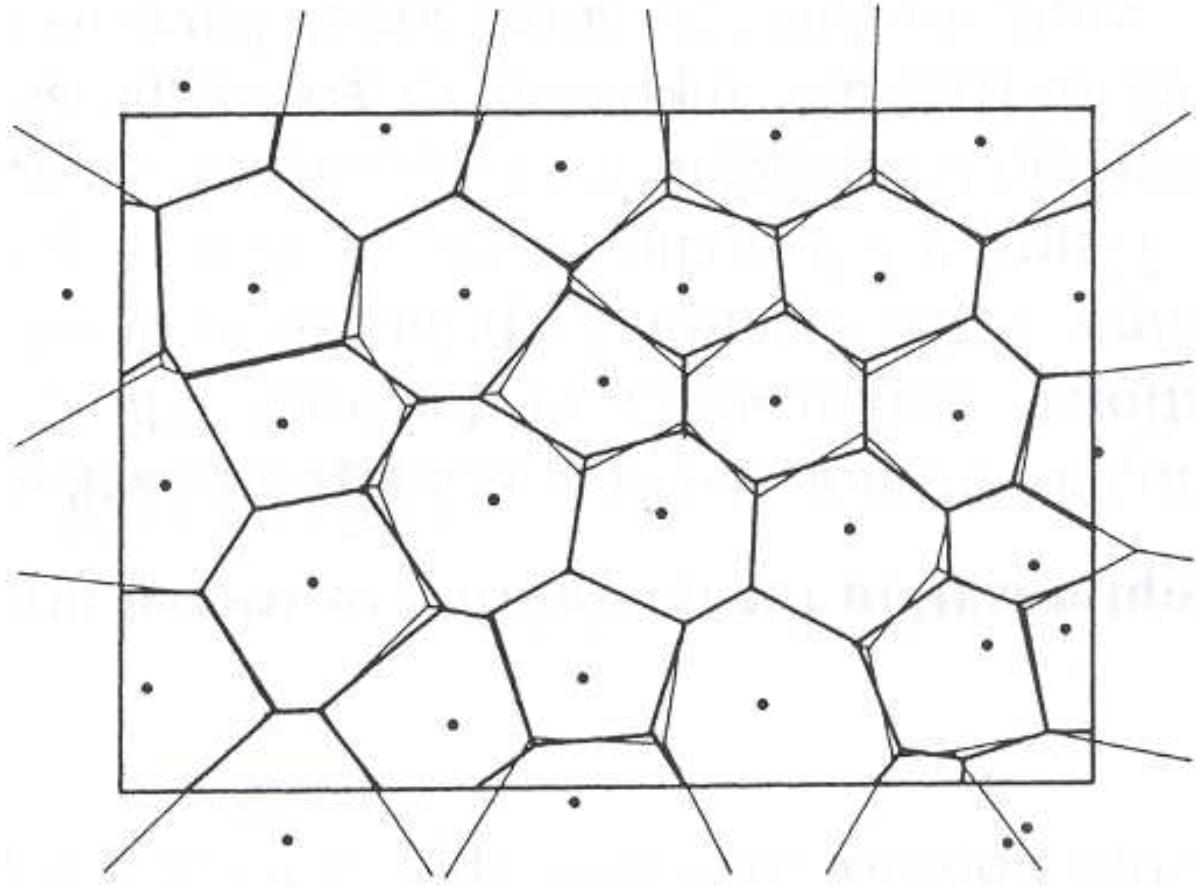
Male moutbreeder **fish** – *Tilapia mossambica*

- Fishes dig nesting pits in sandy bottoms
- They adjust the centers and boundaries of the pits so that the final configuration of territories is a centroidal Voronoi tessellation



*A top view photograph, using a polarizing filter, of the territories of the male  
Tilapia mossambica*

From: George Barlow; Hexagonal territories, *Animal Behavior* 22 1974, pp. 876–  
878

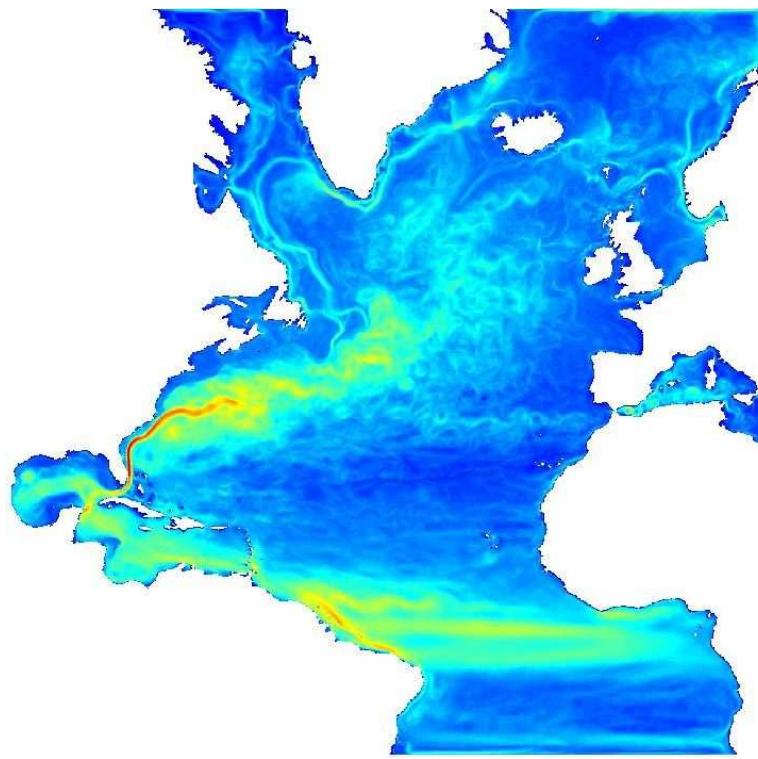


*Actual territories and Voronoi tessellation for generators located at nesting pits.*

From: A. Susuki and M. Iri, Approximation of a tessellation of the plane by a Voronoi diagram, *J. Operations Research Society of Japan* **29** 1986, pp. 69–96

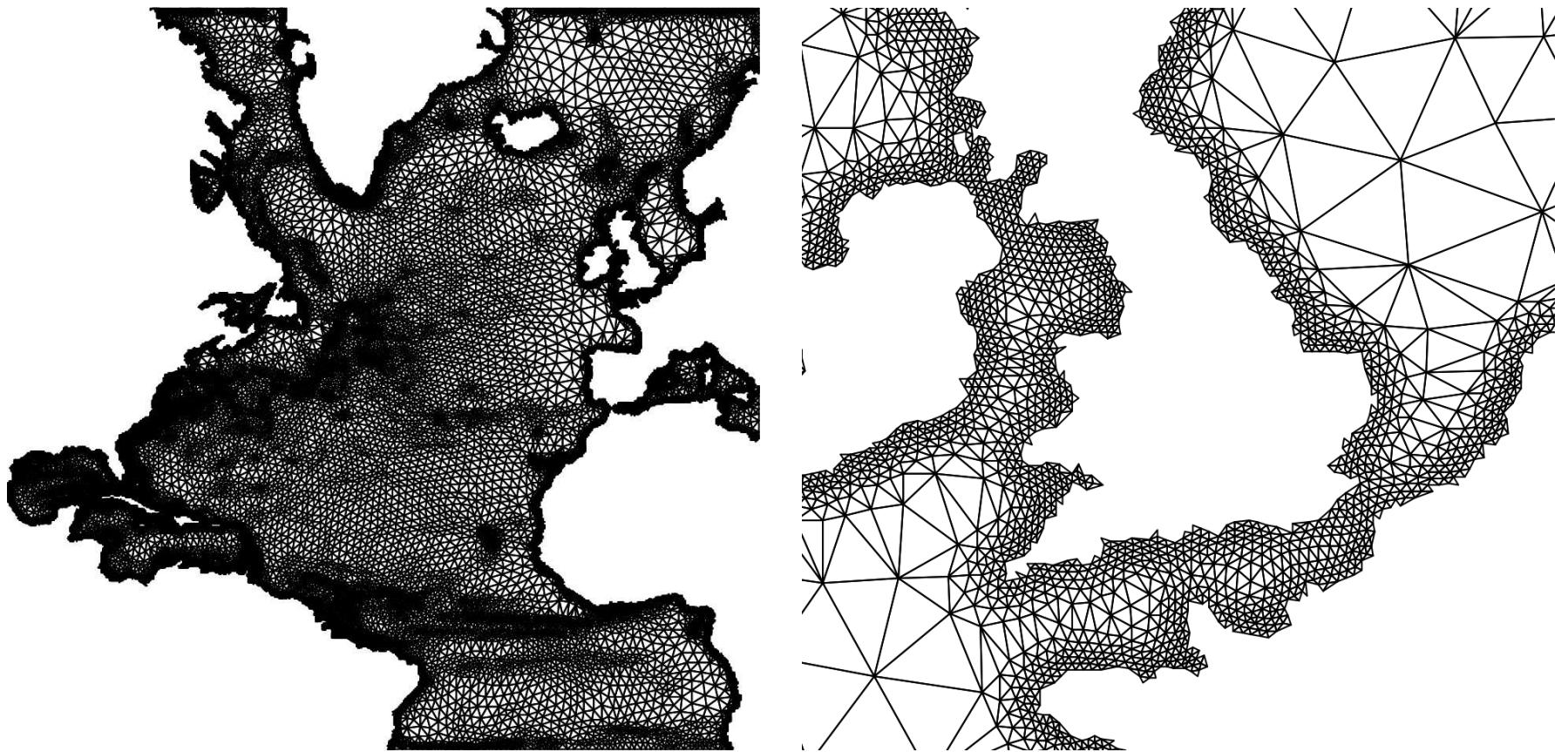
## OCEAN, ATMOSPHERE, AND LAND ICE GRIDS

- CVT grids are been adopted as the grid generation technique for the next generation climate model being developed at NCAR and Los Alamos
  - new ocean model developed by LANL
  - new atmospheric model developed at NCAR
- It appears likely that CVT grids will also be used for the next generation land ice model, e.g., for Greenland and Antarctica

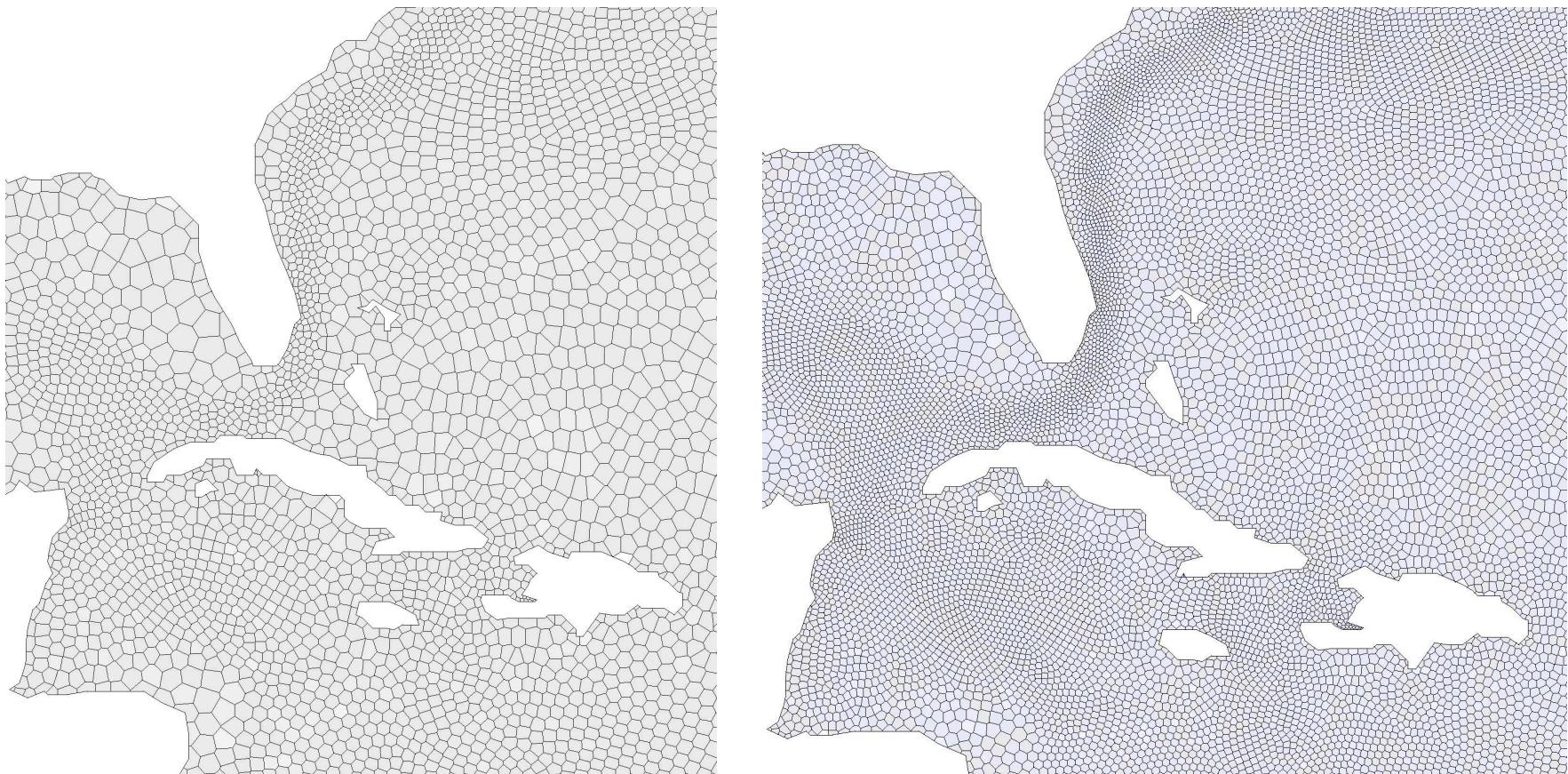


*Satellite data for the kinetic energy distribution in the North Atlantic; note that the boundary is quite complicated*

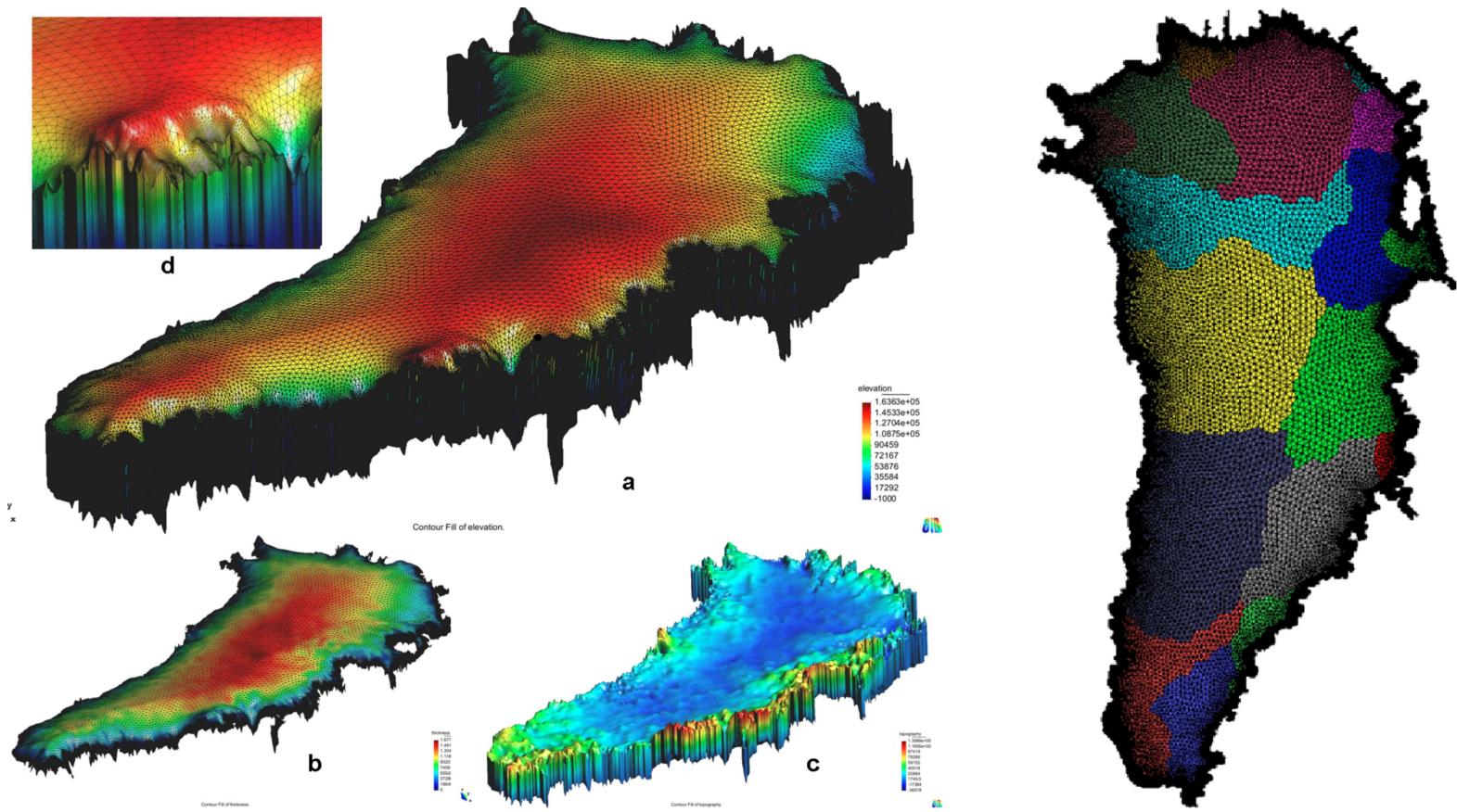
- The grid should be refined
  - where the kinetic energy is relatively large (red and yellow areas)
  - to resolve the boundary



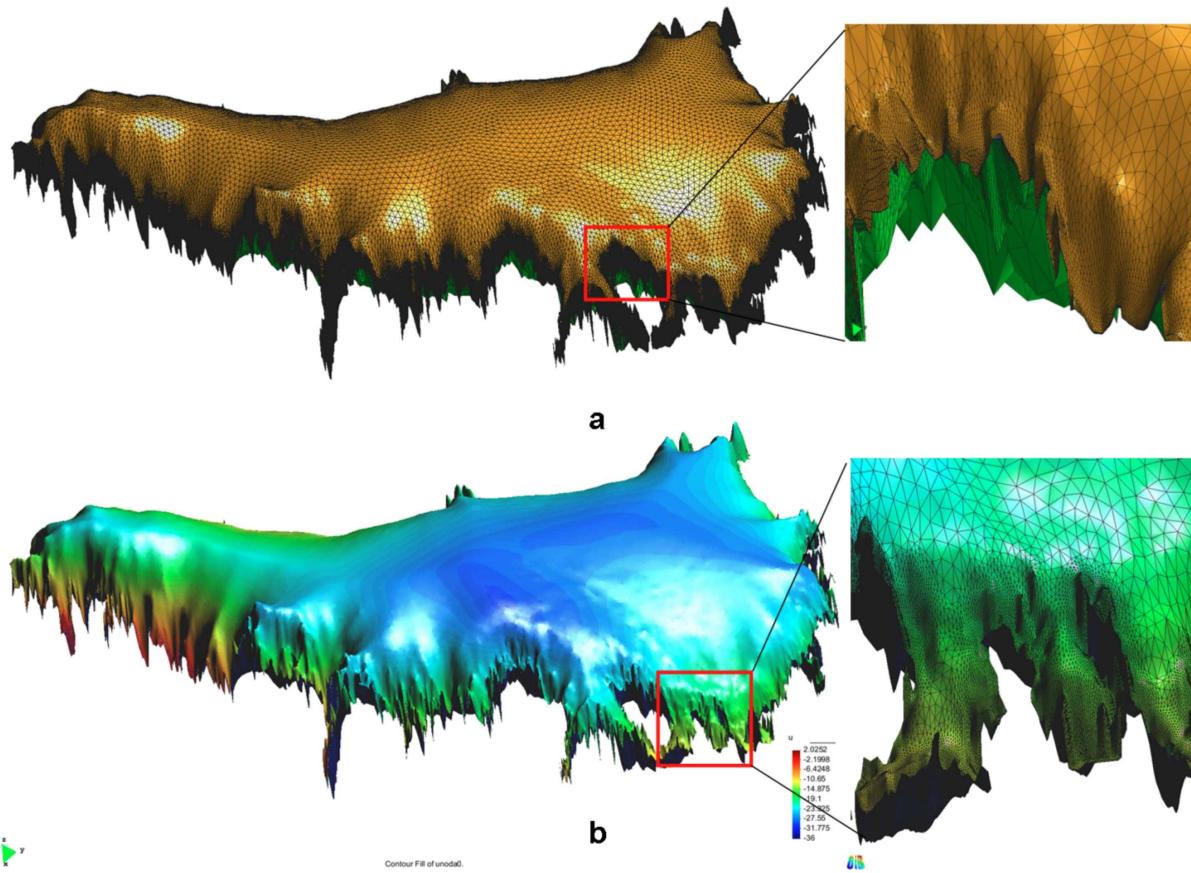
*Left: a CVDT of the North Atlantic; Right: a zoomed in portion of that CVDT; grid refinement is based on both distance from the boundary and measured kinetic energy distribution*



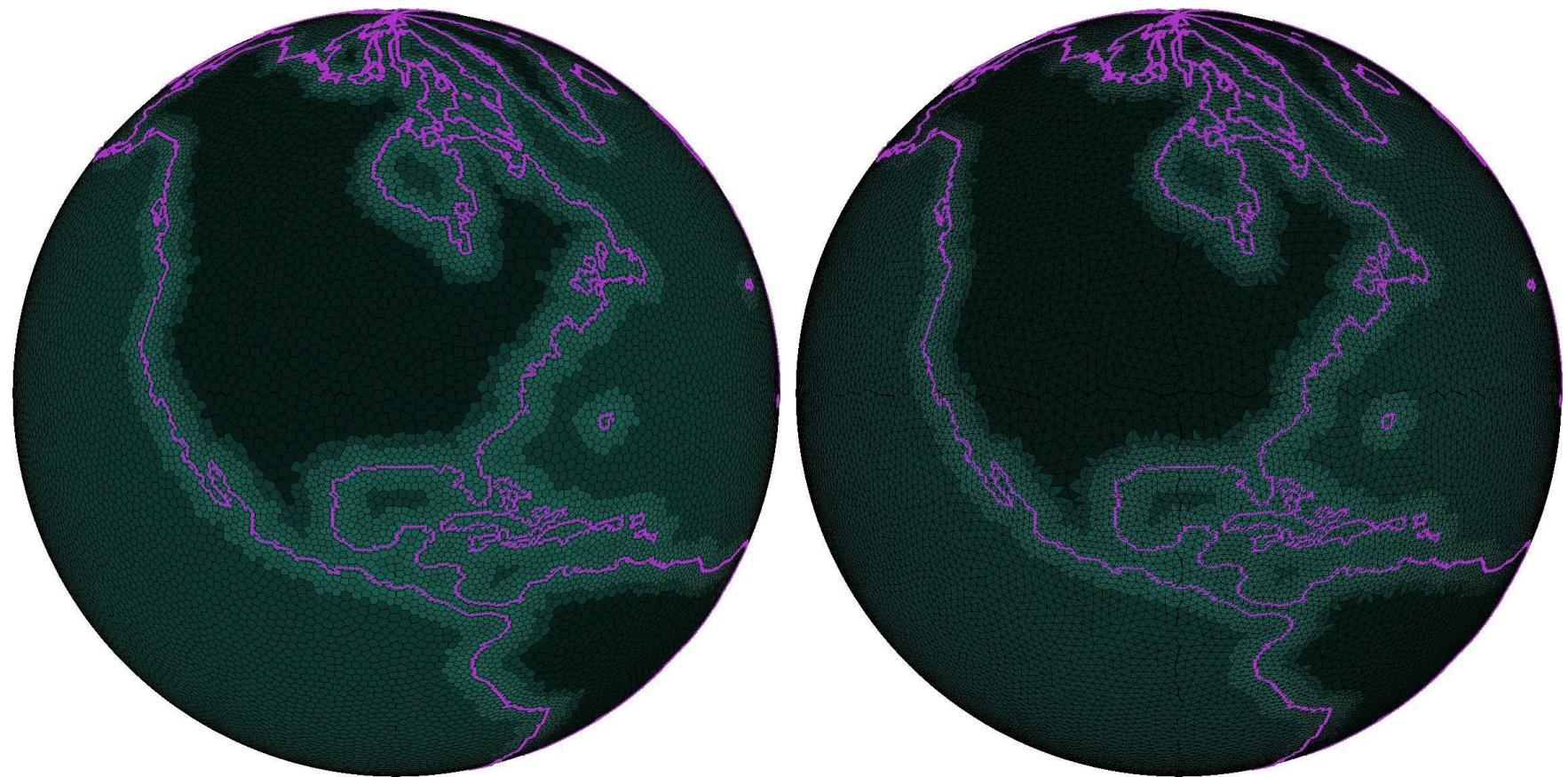
*Left: a zoom-in of the CVT mesh with 47305; right: a zoom-in of the same region of a CVT mesh with 183907 nodes*



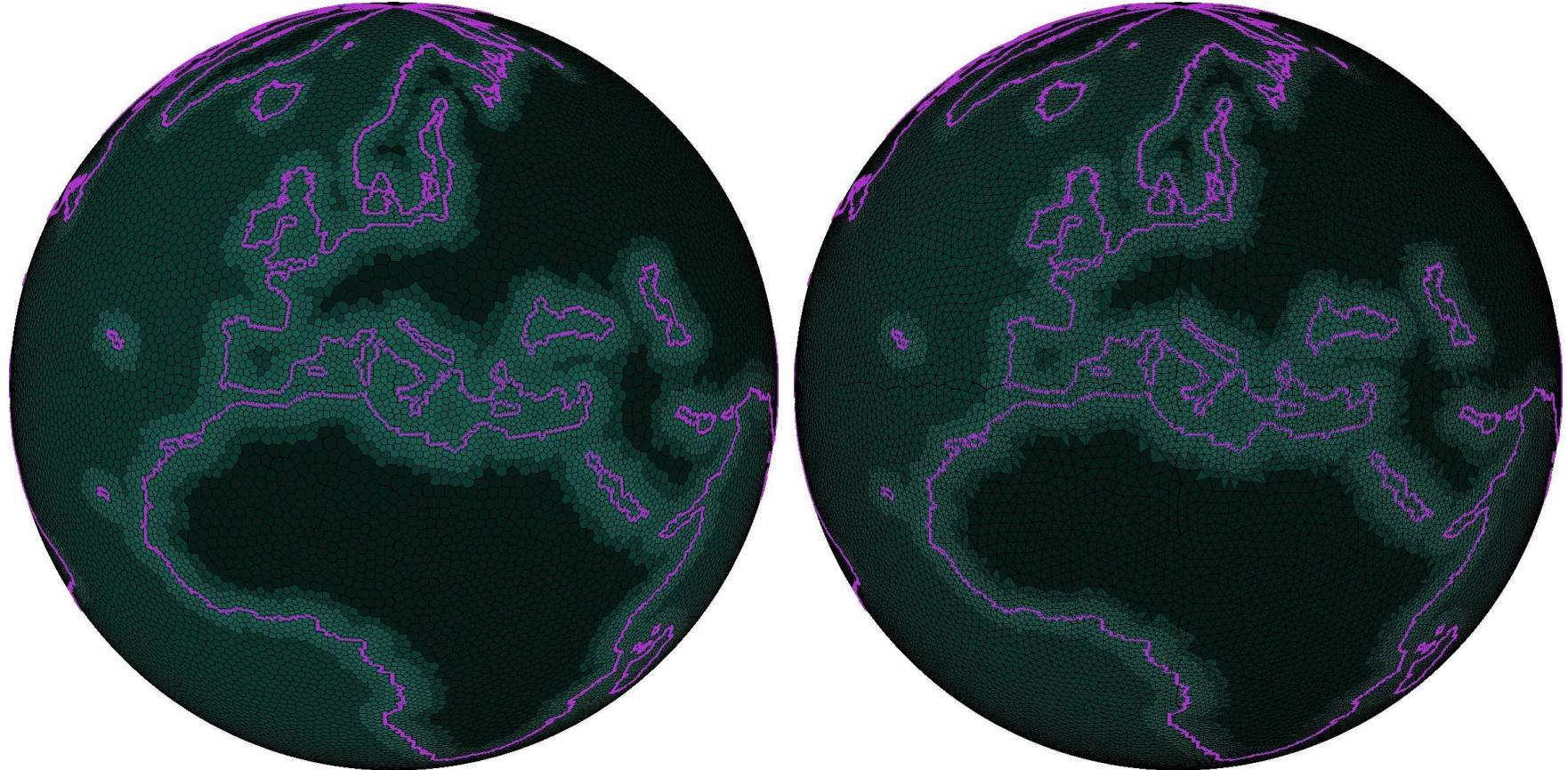
*Left: A 3D prismatic mesh of the Greenland ice sheet (the vertical direction of the ice sheet is stretched) (a) ice sheet surface elevation; (b) ice sheet thickness; (c) bed elevation; (d) zoom-in of a region close to the ice sheet edge. Right: Schematic diagram of the Greenland mesh partition for parallel processing*



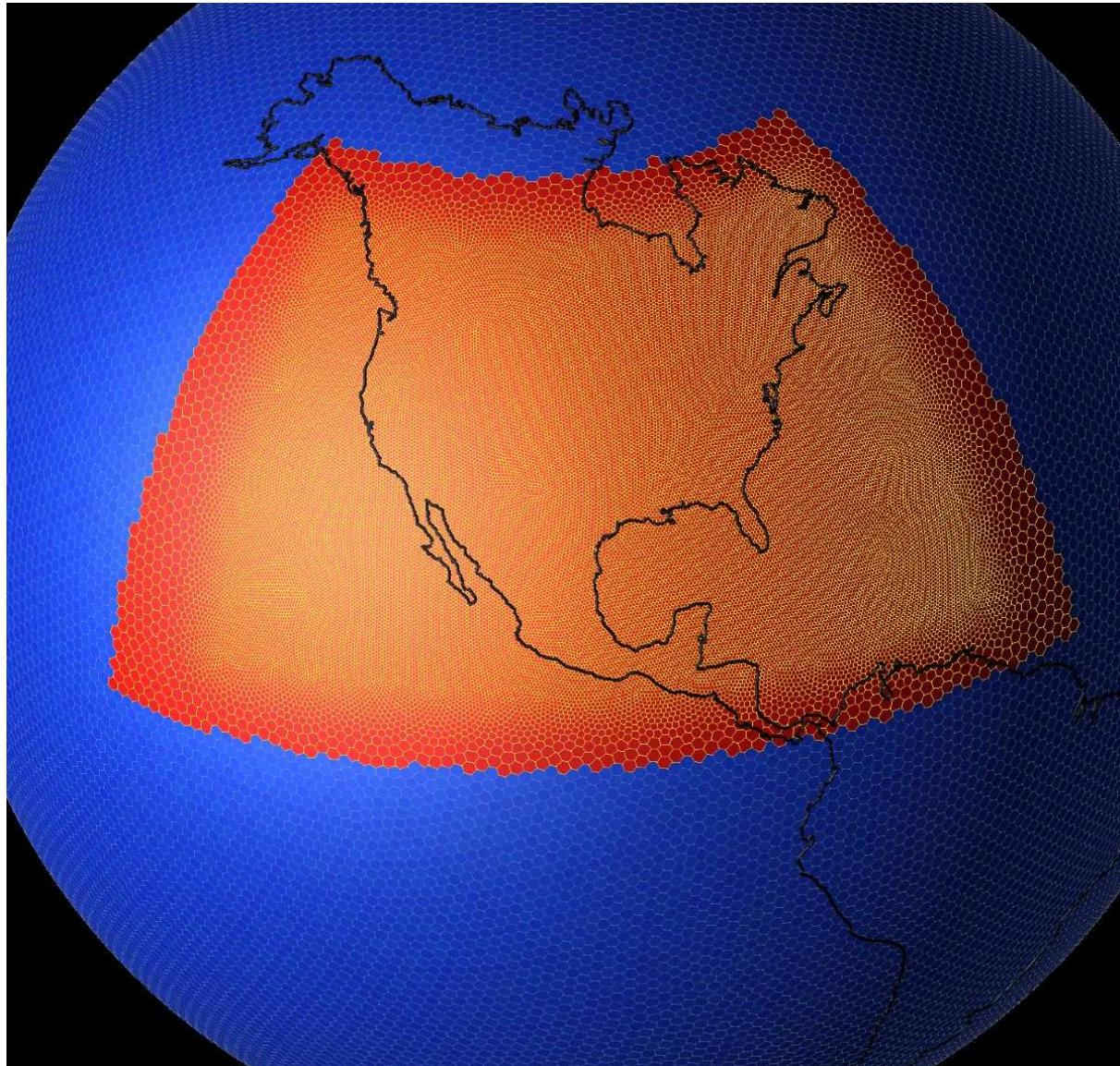
*Simulation results for the temperature evolution of the Greenland ice sheet (the vertical direction of the ice sheet is stretched). (a) the 3D prismatic mesh used for the computation; (b) the ice sheet temperature at the ice-atmosphere and ice-ocean boundaries after ten years*



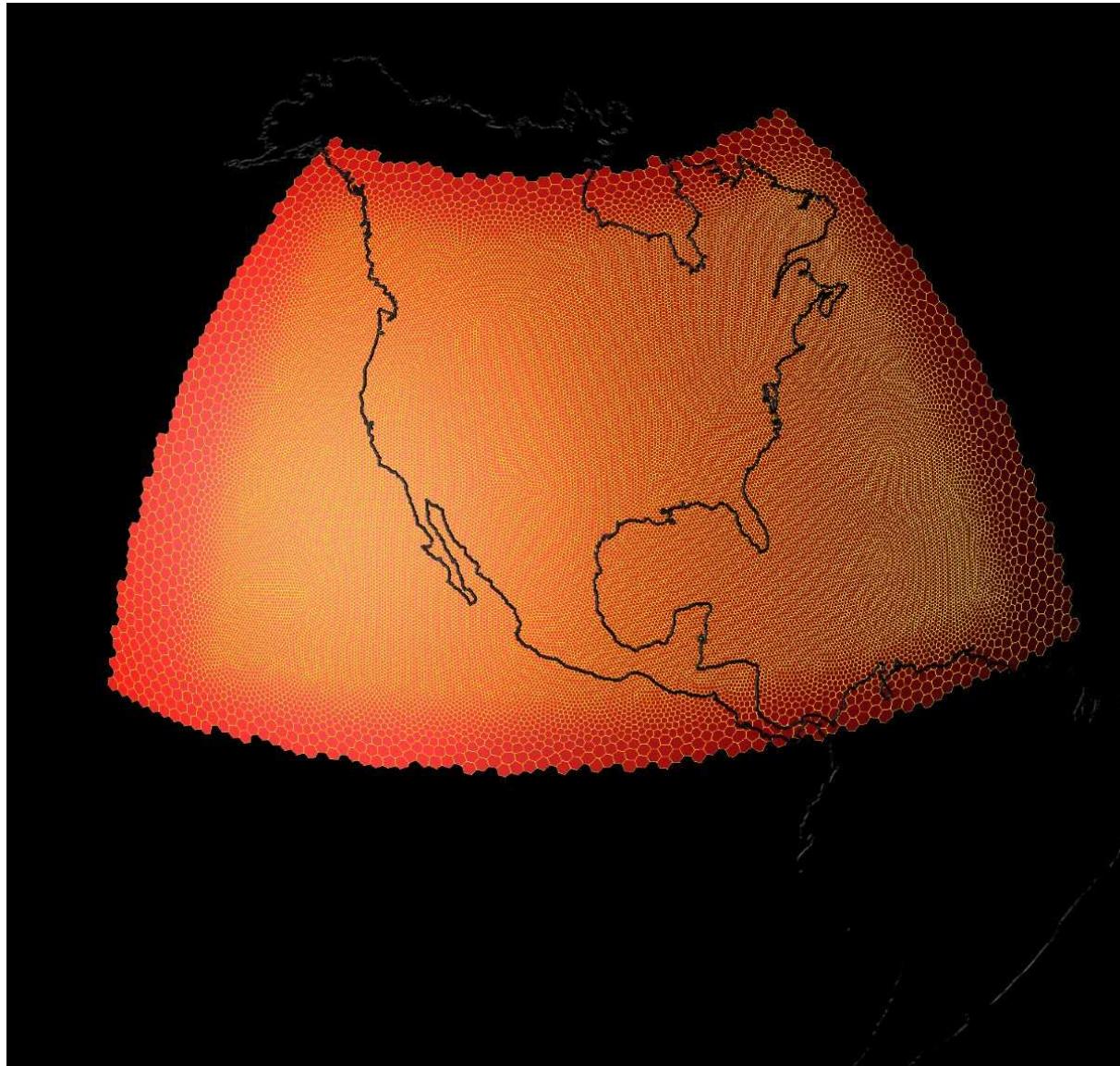
*Global CCVT of the globe and the corresponding Delauney triangulation with points automatically placed on ocean/land boundaries*



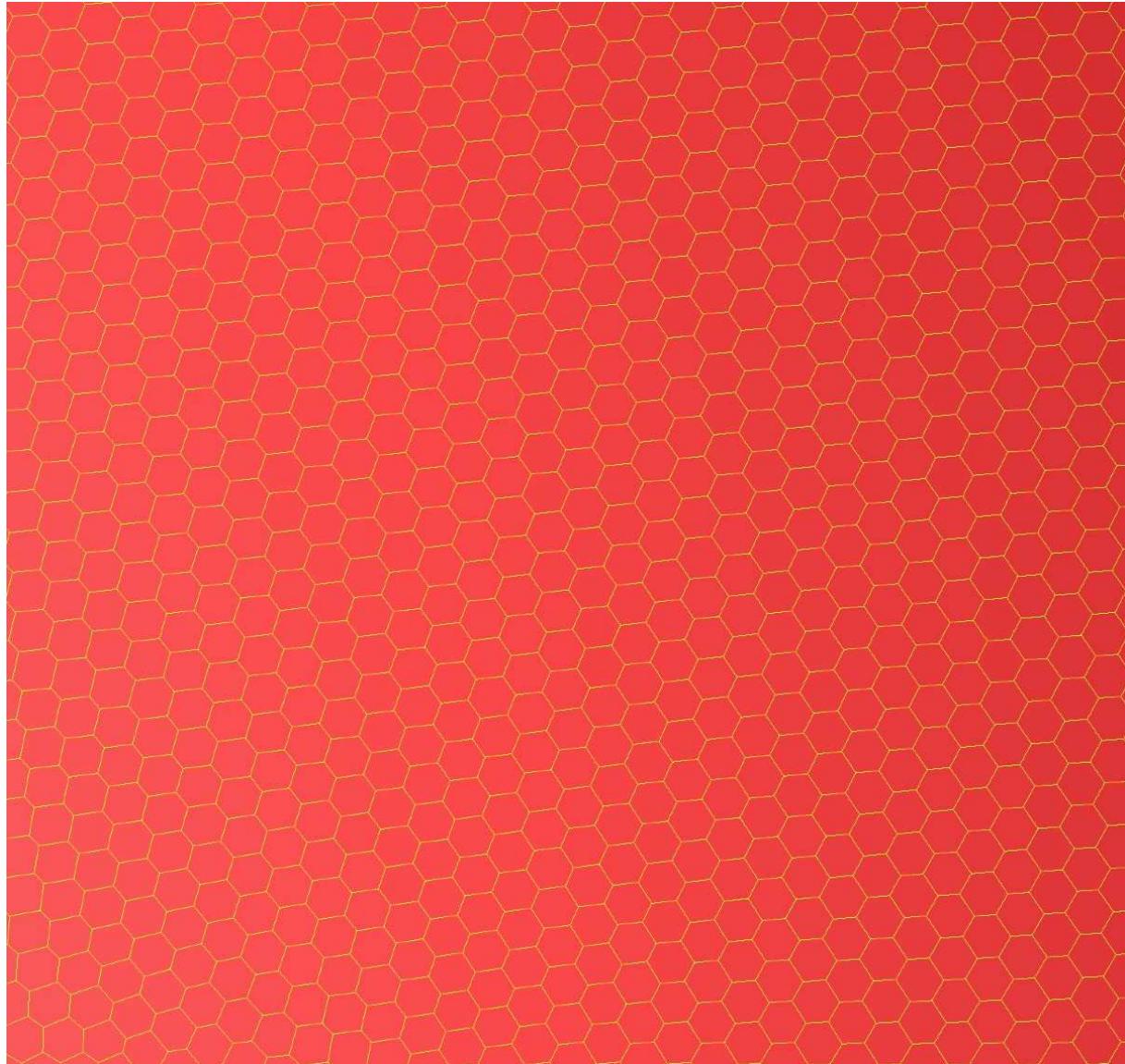
*Global CCVT of the globe and the corresponding Delaunay triangulation with points automatically placed on ocean/land boundaries*



*Global regionally refined grid*



*Grid in refined region*



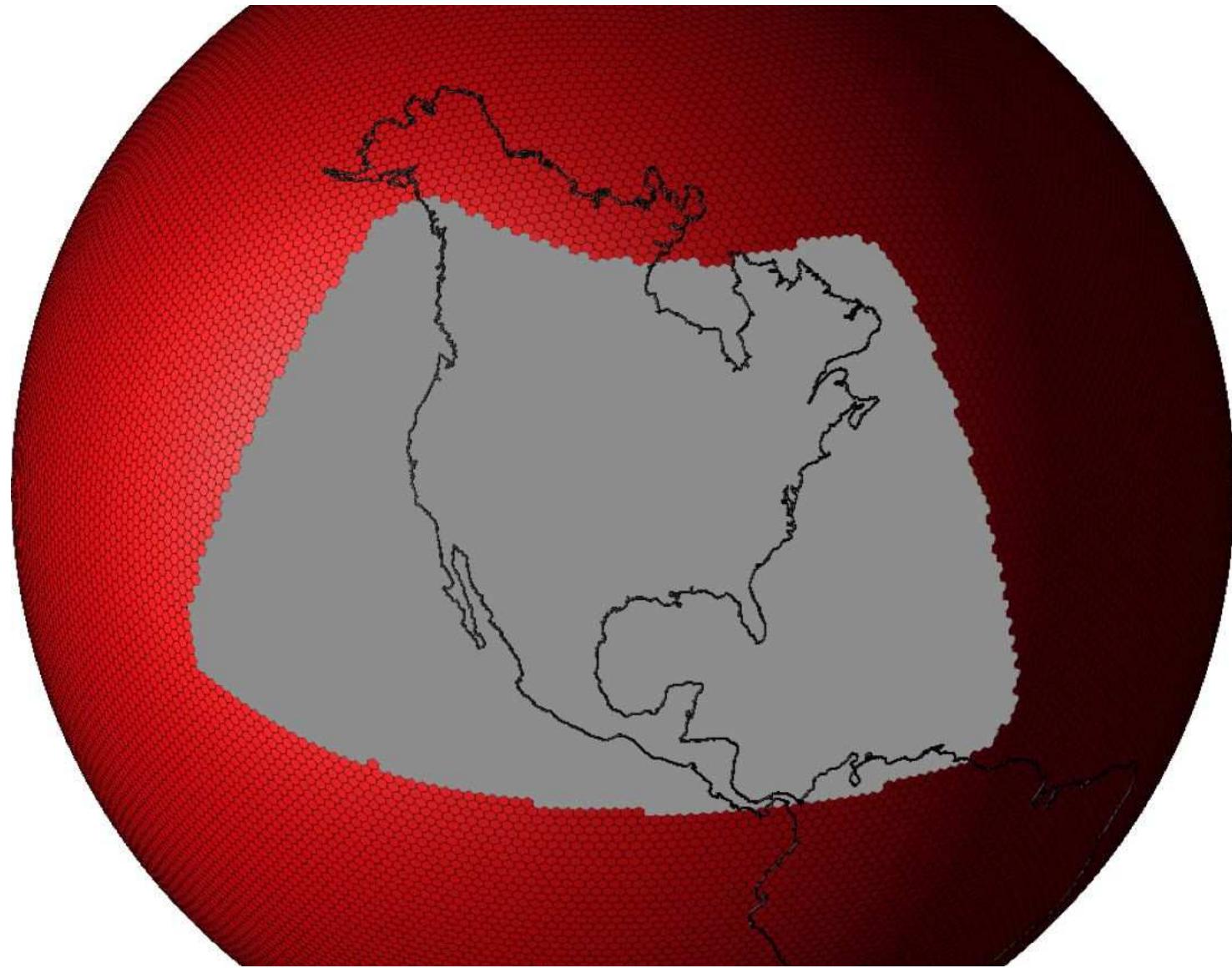
*Grid in interior of refined region*



*Refined grid in transition region*



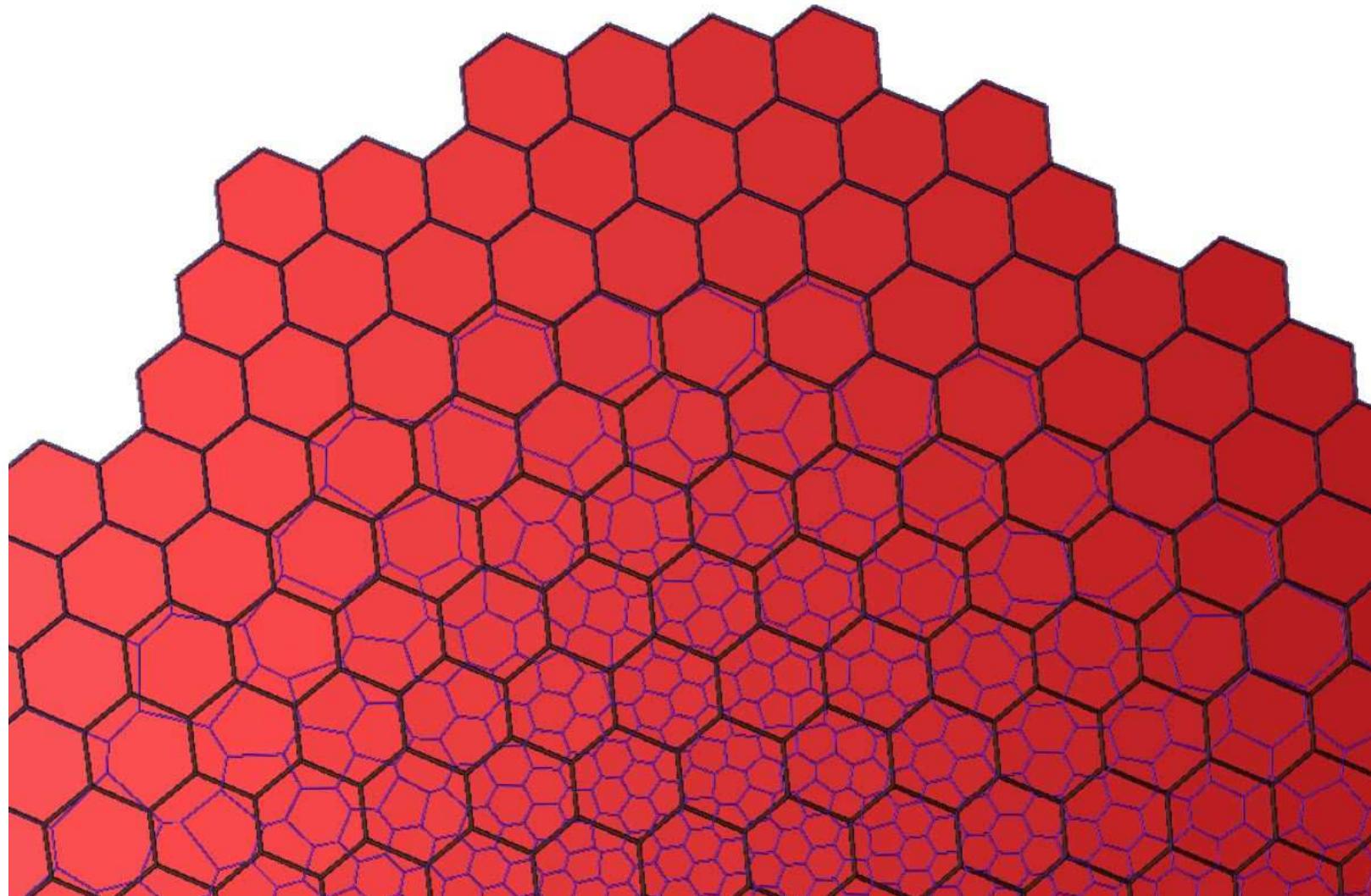
*Global uniform grid*



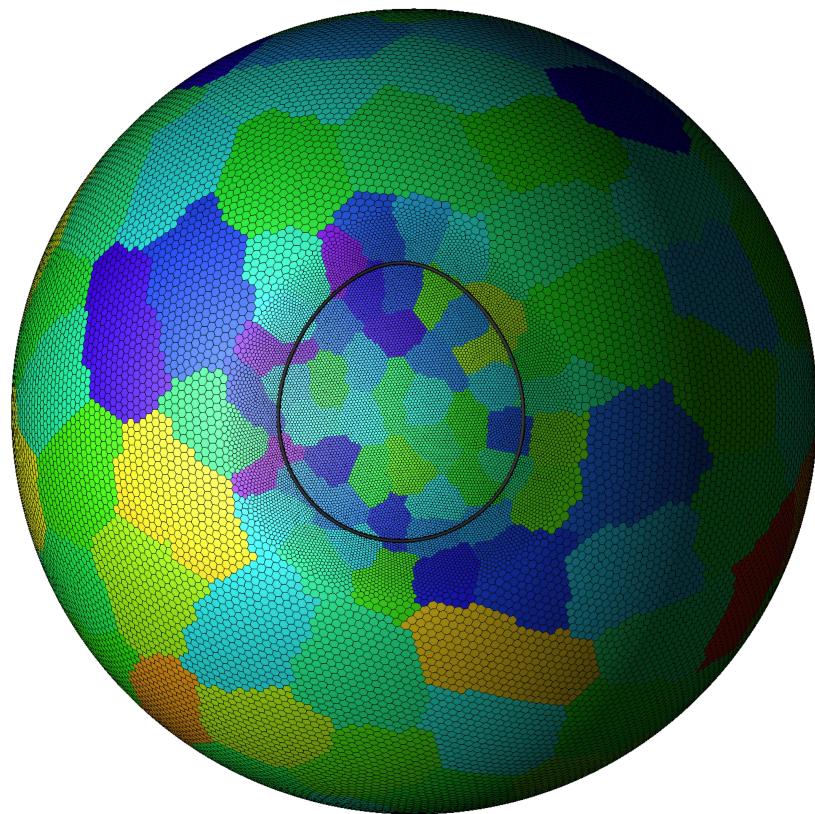
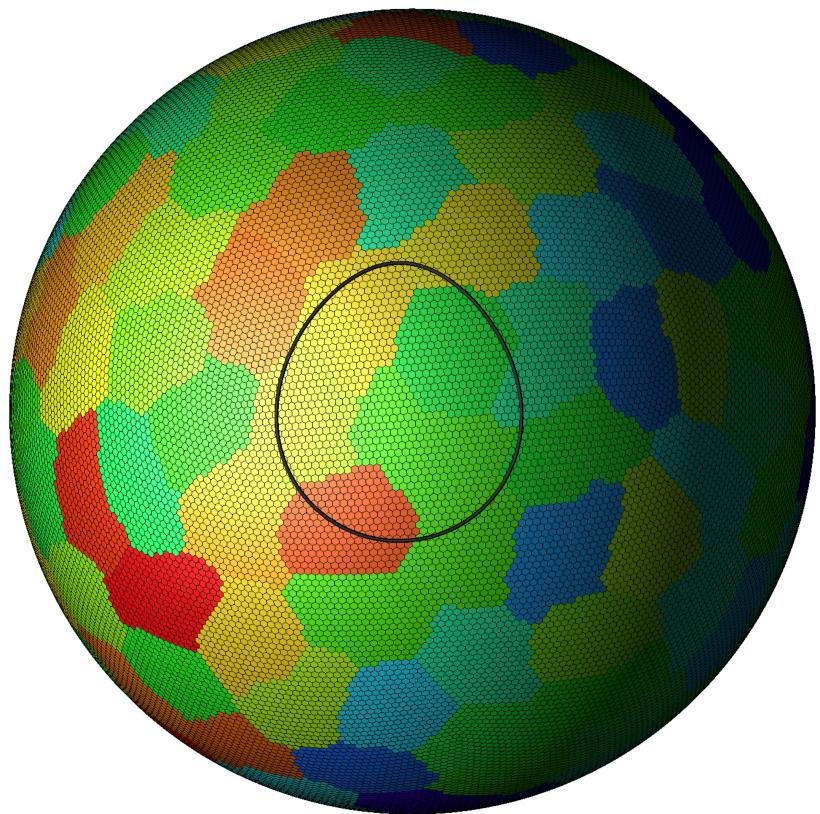
*Region to be refined*



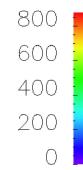
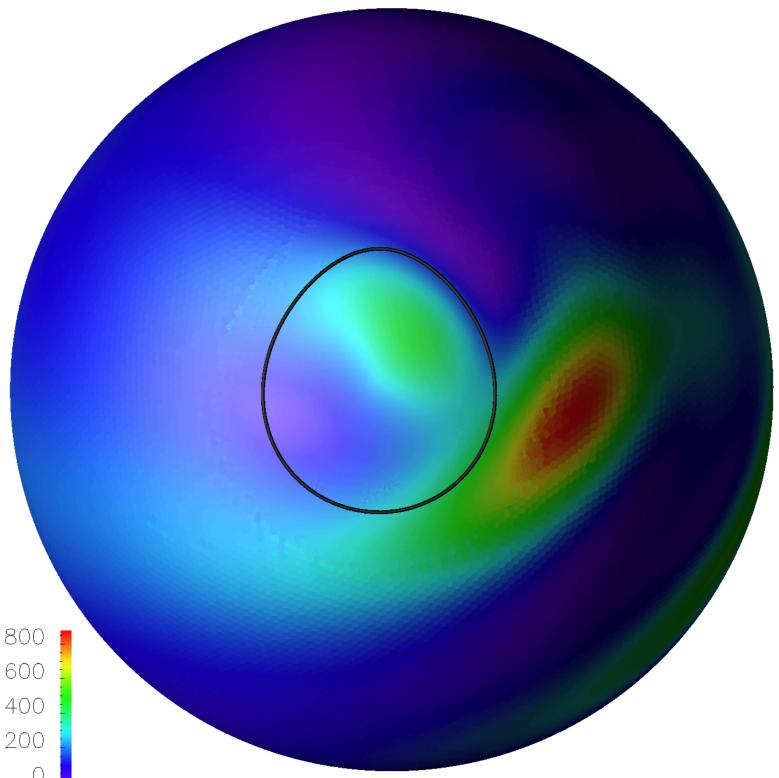
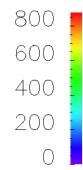
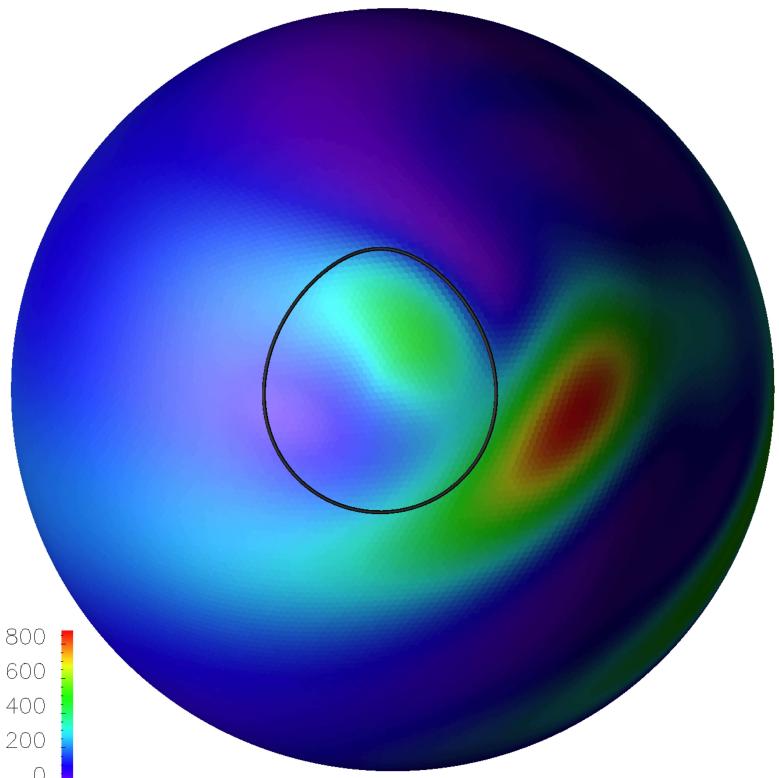
*Refined local grid*



*Match between global grid and refined local grid*



*Quasi-uniform (left) and variable resolution (right) grids on the sphere; the grid on the right is refined in the vicinity of an orographic feature (a mountain) that is the sole forcing in the simulation; the colors indicate a domain-decomposition strategy for efficient implementation on distributed memory systems – each block represents a different computational processor*



*Kinetic energy field at day 10 of simulation; left: simulation using quasi-uniform mesh; right: simulation using variable-resolution mesh*

## DATA COMPRESSION (IMAGE PROCESSING)

- Each point in a picture has a specific color
- Each color is a combination of basic (primary, RGB, CMY) colors
- Let the components of a vector  $w$  represent a possible combination of the basic colors
- Let  $\rho(w)$  denote the number of times the particular combination  $w$  occurs in the picture
- Let  $\Omega$  denote the set of admissible color combinations
- There are **billions** of different colors in a given picture
- One would like to approximate the picture using just a **few** combinations of the basic colors

- Questions:

1. How to choose the few colors that are to be used to represent the picture?
2. How to assign the colors in the picture to the few chosen colors?
  - one usually assigns all colors in the Voronoi region (in color space) associated with one of the few chosen colors to that color

- Why data compression?

- the number of pixels remains the same, but the information associated with each pixel is reduced

- “Obvious” method for choosing the reduced set of colors
  - use random sampling to determine the colors
  - even with 256 approximating colors, doesn’t give good approximate pictures

•  
•  
•

- Better method for choosing the reduced set of colors
  - choose the colors by doing a CVT in color space
  - produces great approximate pictures
  - method used on some (early) HP ink jet **color printers**



*Original 8-bit grayscale image*



*Centroidal Voronoi 6-bit approximate picture*



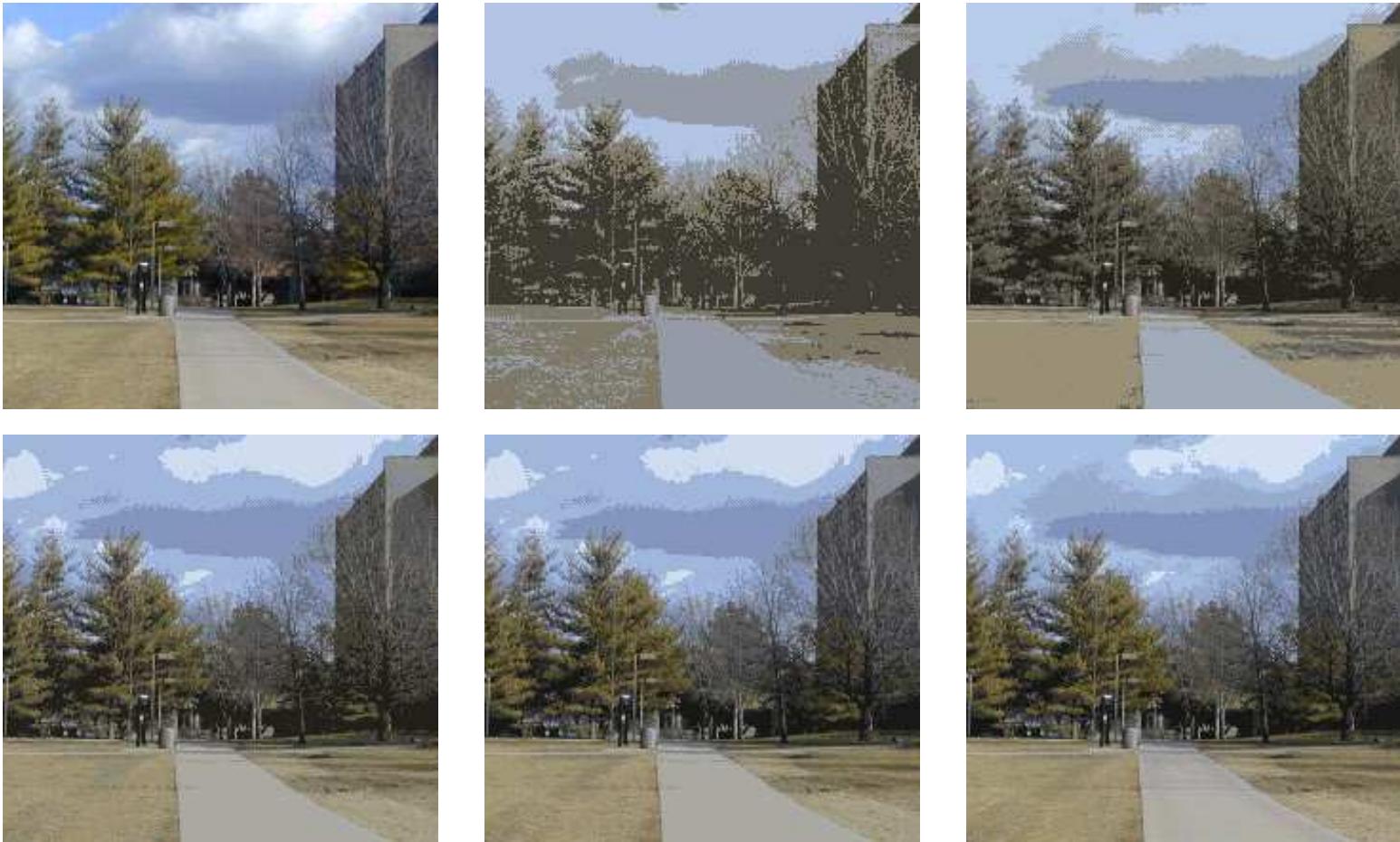
*Centroidal Voronoi 5-bit approximate picture*



*Centroidal Voronoi 4-bit approximate picture*

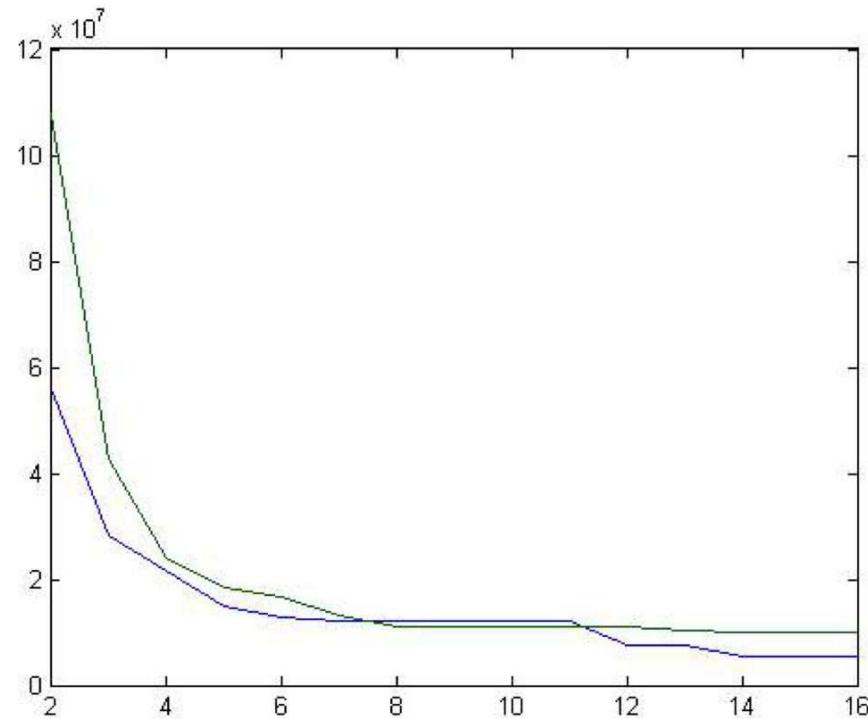


*Centroidal Voronoi 2-bit approximate picture*



*From left to right and top to bottom: original image containing 1434 different colors and CVT-approximate images containing 4, 8, 16, 32, and 64 colors, respectively*

- **Elbowing effect:** the CVT-energy vs. number of generators (reduced set of colors) decreases rapidly at first, but then, as one increases the number of generators, reductions in the energy become less pronounced
  - can be used to determine the number of generators that should be used

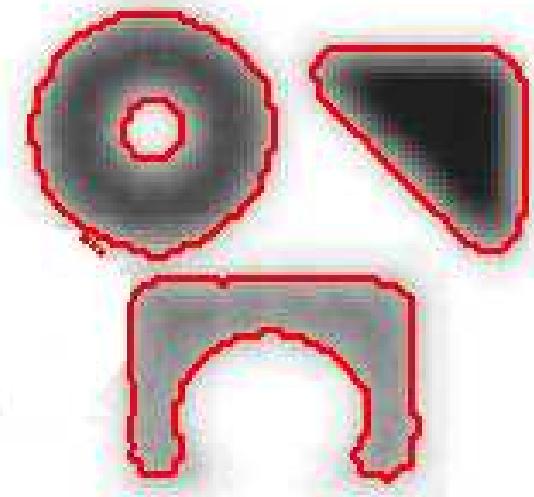
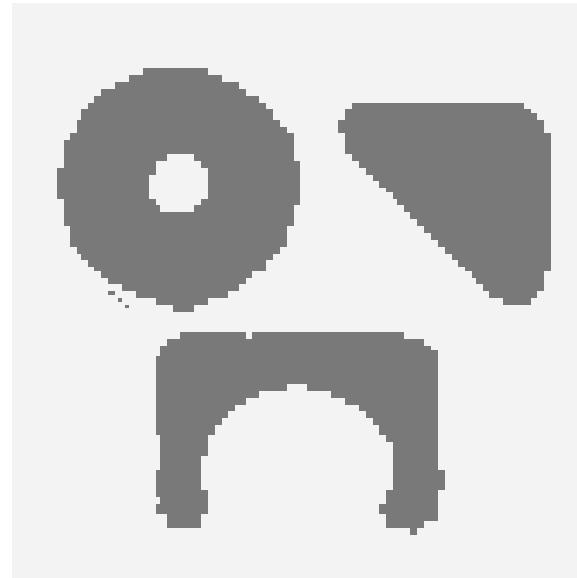


*The CVT-energy of CVT-approximate images for two images vs. the number of replacement colors*

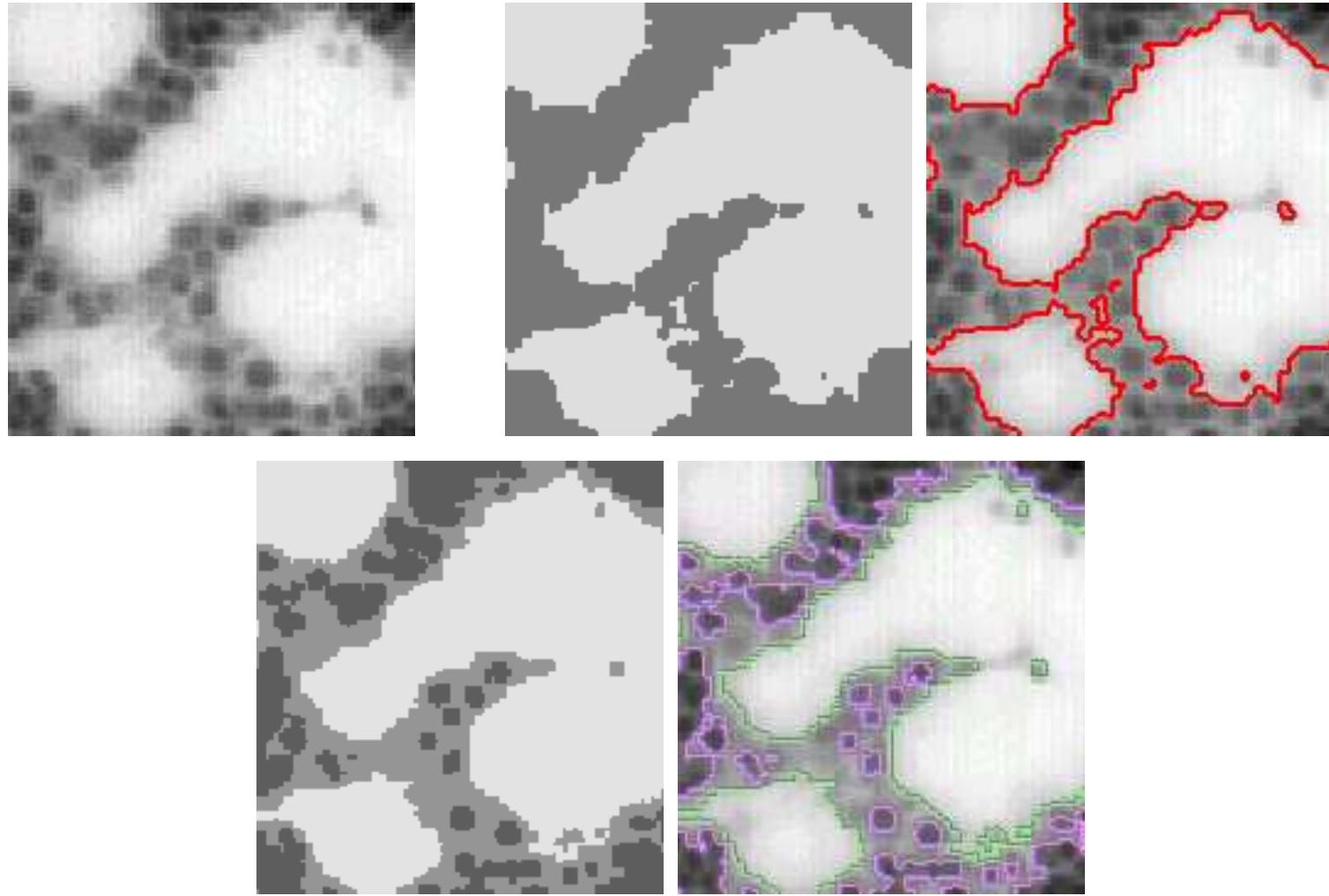
## IMAGE SEGMENTATION AND EDGE DETECTION

- One can regard a rectangular digital image as a function defined over the set of the integral points  $(x, y) = (i, j)$ , where  $i = \{1, 2, \dots, I\}$  and  $j = \{1, 2, \dots, J\}$  for two positive integers  $I$  and  $J$
- We denote that function by  $u$ , with the values of  $u$  representing some attribute of the picture, e.g., color or brightness.
- Image segmentation is the process of identifying the parts of an image that have a common attribute,
  - e.g., that are roughly the same color or have the same brightness,
  - and to also identify edges,
  - i.e., the boundaries between the different segments of the image
- The segmentation is done on the physical picture

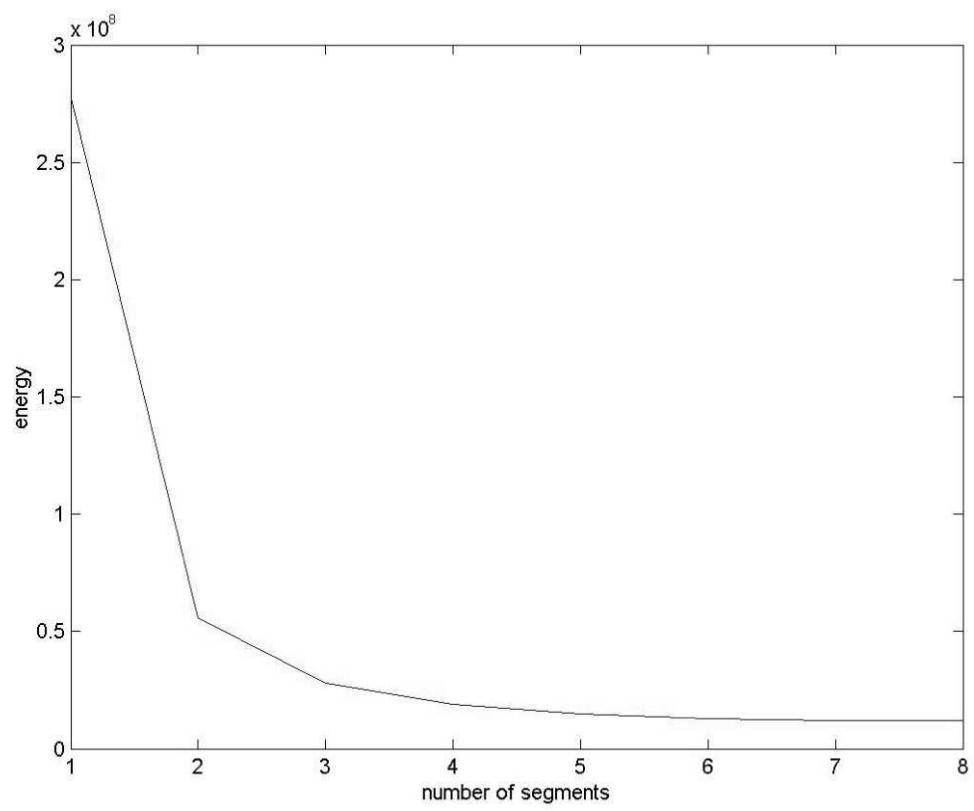
- CVT clustering of an image can be easily used for image segmentation
  - recall that CVT clustering is the partitioning of an image in color space
- Suppose that we have partitioned an image into the CVT clusters  $\{V_\ell\}_{\ell=1}^L$ 
  - in physical space, this corresponds to the segmentation of the image into the  $L$  segments  $\{\mathcal{V}_\ell\}_{\ell=1}^L$ , where
$$\mathcal{V}_\ell = \{(i, j) : u(i, j) \in V_\ell\}.$$
- Edges can be detected by seeing if neighboring points belong to a different cluster
  - the point  $(i, j)$  is an edge point of the segment  $\mathcal{V}_\ell$  if one of its neighboring points belongs to a different segment  $\mathcal{V}_k$ ,  $k \neq \ell$
- The elbowing effect of the CVT “energy” can be used to determine a good number of segments to use



*CVT-based segmentation and edge detection into two segments*



*CVT-based segmentation and edge detection of a bone tissue image (top-left) into two segments (top-right) and three segments (bottom)*



*CVT-energy vs. number of segments for CVT-based segmentations of the bone tissue image*

- CVT-based image segmentation is very cheap compared to, e.g., PDE and variationally based methods
- Simple, CVT-based image segmentation, as just illustrated, is well known
  - it is simply k-means clustering applied to image segmentation
- However, the CVT context easily allows for several useful generalizations

## Weighted CVT's

- Give some cells more weight than others
  - can help enhance the separation between colors



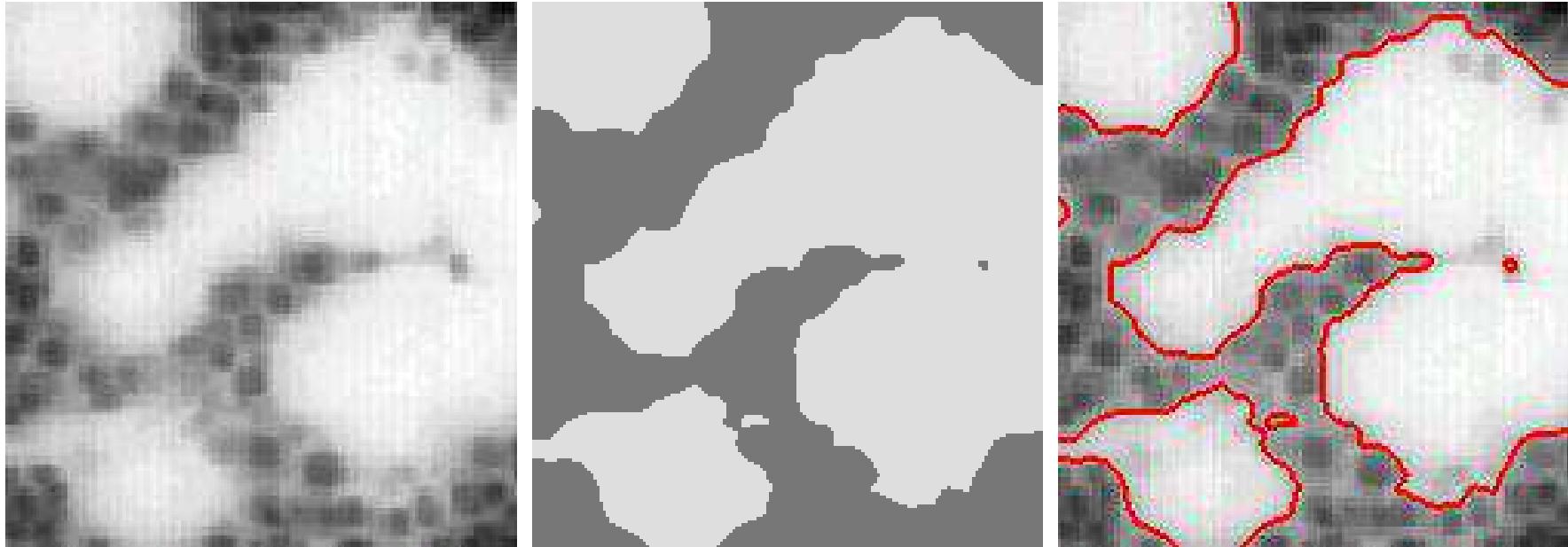
*Original image (left) and weighted CVT-based compressed images with, from left to right, equal weight functions, weight functions proportional to cell volume and square of the cell volume*

## CVT with averaging for segmenting noisy or irregular images

- Averaging can be used to smooth out noisy or irregular images
  - we effect averaging through discrete convolutions



*Original image (left); averaged image (middle); image with stronger averaging (right)*

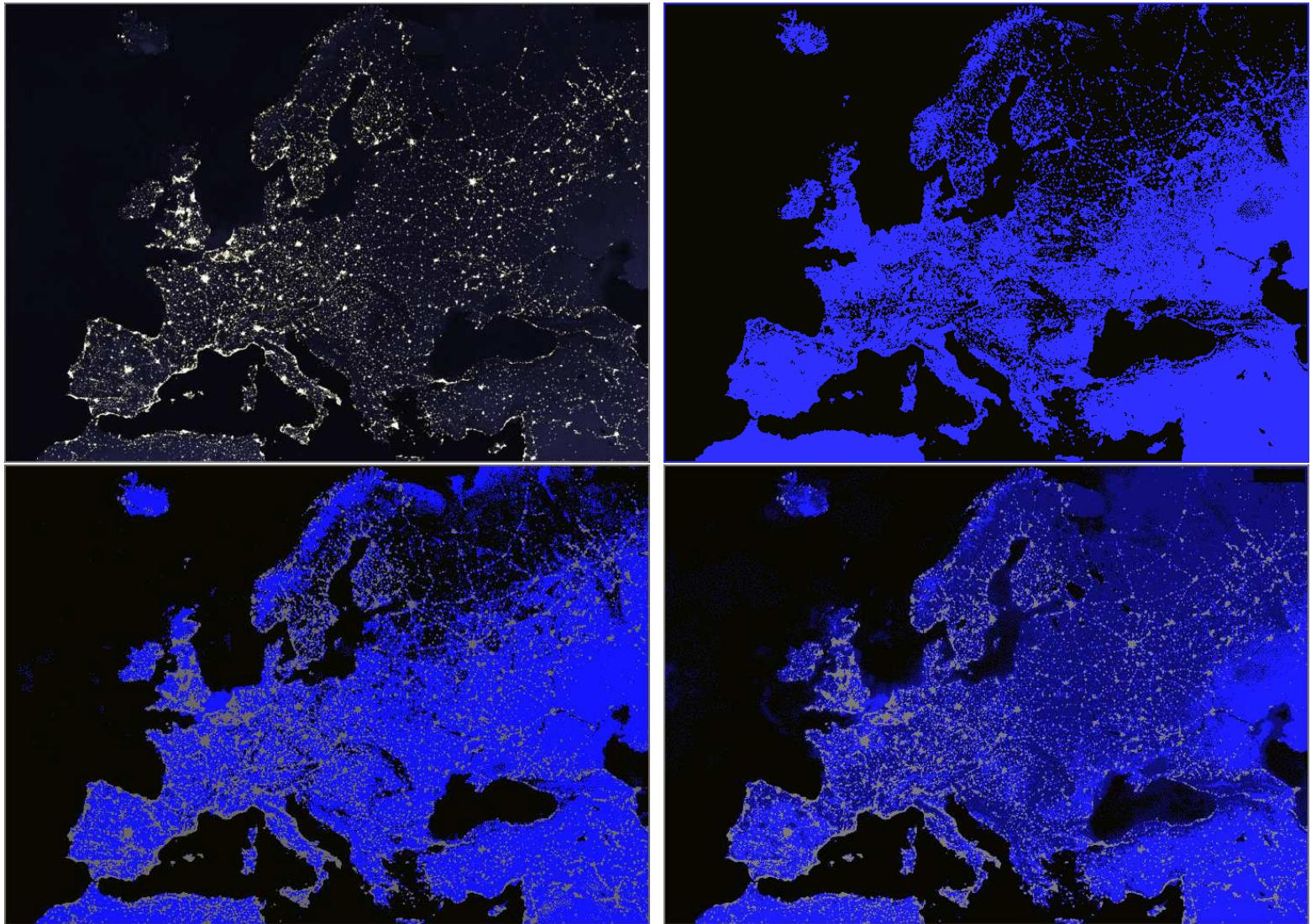


*Original grayscale bone tissue image (left); CVT-based segmentation into two segments with subsequent averaging (middle); the corresponding edges (right)*



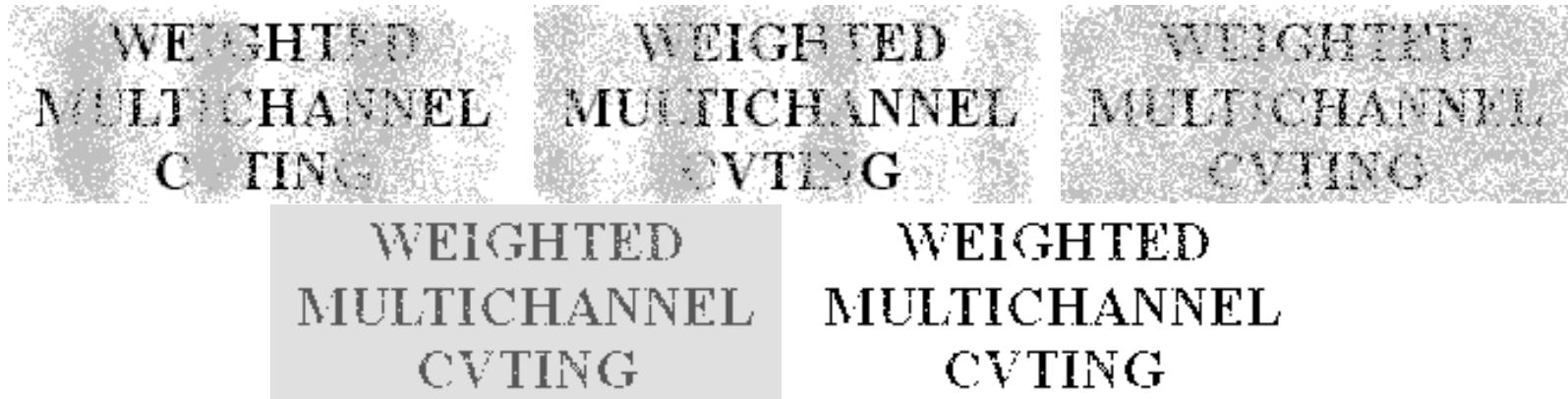
*Original grayscale image of a house (left); CVT-based segmentation into two segments with subsequent averaging (middle); the corresponding edges (right)*

## **Europe-by-night**– segmented into 2, 3, and 4 segments



## MULTICHANNEL IMAGE RECONSTRUCTION

- Suppose one has in hand several versions of a picture, none of which contains all the information necessary to recover the complete image
  - how can the information contained in the different versions be combined so as to recover the whole image?
  - this is a natural task for CVT-based image processing



*Three noisy and incomplete versions of the same image (top); multichannel CVT-based approximate image using 2 replacement colors (bottom-left); restored image using renormalization (bottom-right)*

- Can combine multi-channel CVT with averaging and weighting

## GRID GENERATION

- The dual Delaunay meshes corresponding to CVT's are well suited for triangular and tetrahedral mesh generation
  - CVT and the dual Delaunay mesh pair are well suited to finite volume and edge element type discretizations (as well as nodal elements)
- In order to render them useful for mesh generation, CVT concepts and algorithms have been extended in several directions so that the CVT methodology can handle
  - automatically locating a subset of the points on the boundary
  - generating interior points for a specified surface point distribution
  - nonuniform point distributions
- CVT point generation is quite cheap

- The key and the curse of the effectiveness of CVT mesh generation can be found in Gersho's conjecture
  - the fact that the point distribution is always locally uniform makes it unlikely that weird shaped elements, e.g., slivers, will be generated
  - on the other hand, it seems that the local uniformity of the mesh also means that anisotropic meshes, e.g., for fluid boundary layers, cannot be generated
- Gersho's conjecture assumes that one uses the standard Euclidean norm to measure the distance between two points
  - if one instead uses anisotropic metrics, e.g.,
 
$$\mu_1(y_1, y_2, y_3)(z_1 - y_1)^2 + \mu_2(y_1, y_2, y_3)(z_2 - y_2)^2 + \mu_3(y_1, y_2, y_3)(z_3 - y_3)^2$$
 one can indeed generate anisotropic CVT grids
- The CVT density function and metric function coefficients can be connected to solutions and to a posteriori error estimators to enable CVT adaptive grid generation

## CVTs with points on boundary

- For grid generation applications, one needs points on the boundary
  - for the basic CVT method, the boundary repels points so that CVT points never lie on the boundary
- However, generalized CVTs can be defined to have some of the points lie on the boundary — there are three ways to do this
  1. a set of fixed points on the boundary can be specified, and then a generalized CVT methodology will optimally place the interior points
  2. a set of points on the boundary can be specified, and then another generalized CVT methodology will optimally place points in the interior and optimally slide the boundary points
  3. no points on the boundary are specified, and then another generalized CVT methodology will optimally place points in the interior and on the boundary

## UNIFORM GRID COMPARISONS WITH OTHER 2D MESH GENERATORS

- We will use the eyeball norm  $\| \cdot \|_{\text{eye}}$  to compare different methods for generating uniform triangulations of regions in the plane
  - we also want to make comparisons based on quantitative measures
- In an effort to quantify the process of assessing and comparing the uniformity of point sets and grids, we have examined 12 measures of uniformity
  - 2 are based on just the coordinates of the points
  - 6 are based on the Voronoi tessellation of the region, with the points serving as generators
  - 4 are based on the Delaunay triangulation of the point set
- We will only consider the 8 most useful ones

## Measures of grid uniformity

### The point distribution measure

- Given a Voronoi tessellation  $\mathcal{V} = \{\mathbf{z}_i, V_i\}_{i=1}^N$  of a point set, the **point distribution measure** is defined by

$$h = \max_{i=1,\dots,N} h_i \quad \text{where} \quad h_i = \max_{\mathbf{y} \in V_i} |\mathbf{z}_i - \mathbf{y}|$$

- Thus,
  - $h_i$  gives the maximum distance between the particular generator  $\mathbf{z}_i$  and the points in its associated Voronoi cell  $V_i$
  - $h$  gives the maximum distance between any generator and the points in its associated cell
- For an ideal tessellation into congruent regular hexagons,  $h_i = h$  for all  $i$ 
  - thus, the smaller  $h$  is, the more uniform is the mesh

## The regularity measure

- Given a Voronoi tessellation  $\mathcal{V} = \{\mathbf{z}_i, V_i\}_{i=1}^N$  of a point set, we define the **regularity measure** by

$$\chi = \left( \max_{i=1,\dots,N} \chi_i \right) \quad \text{where} \quad \chi_i = \frac{\sqrt{3}h_i}{\gamma_i}$$

and where

$$\gamma_i = \min_{j \neq i} |\mathbf{z}_i - \mathbf{z}_j| \quad \text{for } i = 1, \dots, N$$

so that  $\gamma_i$  is the minimum distance from the point  $\mathbf{z}_i$  to any of the other points

- For an ideal uniform hexagonal mesh,  $h_i = h$  and  $\gamma_i = \sqrt{3}h$  for all  $i$  so that  $\chi_i = 1$  for all  $i$  and then  $\chi = 1$ 
  - thus, the smaller  $\chi$  is, the more uniform is the mesh.

## The second moment trace measure

- Given a Voronoi mesh  $\mathcal{V} = \{\mathbf{z}_i, V_i\}_{i=1}^N$ , let  $t_i$  denote the trace of the second moment tensor (about the region generator)

$$\mathbb{M}_i = \frac{1}{|V_i|} \int_{V_i} (\mathbf{x} - \bar{\mathbf{z}})(\mathbf{x} - \bar{\mathbf{z}}_i)^T d\mathbf{x}$$

associated with each Voronoi region  $V_i$  having volume  $|V_i|$

- Let  $\bar{t} = \frac{1}{N} \sum_{i=1}^N t_i$  denote the average of the traces

- Then, we define the **second moment trace measure** by

$$\tau = \max_{i=1,\dots,n} |t_i - \bar{t}|$$

- For a perfectly uniform point distribution,  $t_i = \bar{t}$  for all  $i$  so that  $\tau = 0$ 
  - thus, the smaller  $\tau$  is, the more uniform is the mesh

## The second moment determinant measure

- Given a Voronoi mesh  $\mathcal{V} = \{\mathbf{z}_i, V_i\}_{i=1}^N$ , let  $d_i$  denote the determinant of the deviatoric tensor

$$\mathbb{D}_i = \mathbb{M}_i - \frac{1}{N} t_i \mathbb{I}$$

associated with each Voronoi region  $V_i$

- Then, the **second moment determinant measure** is given by

$$d = \max_{i=1,\dots,n} |d_i|$$

- For a perfectly uniform point distribution,  $d_1 = d_2 = \dots = d_N = 0$  so that  $d = 0$ 
  - thus, the smaller  $d$  is, the more uniform is the mesh

## Maximum area measure (Shewchuk)

- Given a Delaunay tessellation  $\{\triangle_j\}_{j=1}^{\tilde{N}}$ , let  $|\triangle_j|$  denote the area of the triangle  $\triangle_j$
- Then, the maximum area measure is defined by

$$\alpha = \left( \frac{\tilde{N}}{|\Omega|} \max_{j=1, \dots, \tilde{N}} |\triangle_j| \right)$$

- For an ideal uniform mesh,  $|\triangle_1| = |\triangle_2| = \dots = |\triangle_{\tilde{N}}| = |\Omega|/\tilde{N}$  so that, ideally,  $\alpha = 1$ 
  - thus, the smaller  $\alpha$  is, the more uniform is the mesh

## Minimum angle measure (Shewchuk)

- Given a Delaunay tessellation  $\{\triangle_j\}_{j=1}^{\tilde{N}}$ , let  $\beta_j$  denote the minimum angle of the triangle  $\triangle_j$ 
  - note that  $\beta_j \leq \pi/3$  radians
- Then, the **minimum angle measure** is defined by

$$\beta = \left( \frac{\pi/3}{\min_{j=1,\dots,\tilde{N}} \beta_j} \right)$$

- For an ideal uniform mesh,  $\beta_1 = \beta_2 = \dots = \beta_{\tilde{N}} = \beta = \pi/3$  radians so that, ideally,  $\beta = 1$ 
  - thus, the smaller  $\beta$  is, the more uniform is the mesh

## Circle ratio measure (Persson and Strang)

- Given a Delaunay tessellation  $\{\Delta_j\}_{j=1}^{\tilde{N}}$ , let

$$q_j = \frac{R_j}{2r_j} = \frac{abc}{(b+c-a)(c+a-b)(a+b-c)} \quad \text{for } j = 1, \dots, \tilde{N},$$

where  $a$ ,  $b$ , and  $c$  denote the lengths of the sides of the triangle  $\Delta_j$

- $q_j$  is half the ratio of the radius  $r_j$  of the largest inscribed circle to the radius  $R_j$  of the smallest circumscribed circle of the triangle  $\Delta_j$
  - note that  $q_j \geq 1$
- Then, the **circle ratio measure** is given by

$$\textcolor{red}{q} = \left( \max_{j=1, \dots, \tilde{N}} q_j \right)$$

- For an ideal uniform mesh,  $q_1 = q_2 = \dots = q_{\tilde{N}} = 1$  so that, ideally,  $q = 1$ 
  - thus, the smaller  $q$  is, the more uniform is the mesh

## The normalized standard deviation measure (Persson and Strang)

- Given a Delaunay tessellation  $\{\triangle_j\}_{j=1}^{\tilde{N}}$ , let  $R_j$  denote the radius of the smallest circumscribed circle
- Let

$$\bar{R} = \frac{1}{\tilde{N}} \sum_{j=1}^{\tilde{K}} R_j \quad \text{and} \quad R_{std} = \text{standard deviation of } R_j \text{ over } j = 1, \dots, \tilde{N}$$

- Then, the normalized standard deviation measure is given by

$$p = \frac{R_{std}}{\bar{R}}.$$

- For an ideal uniform mesh,  $R_1 = R_2 = \dots = R_{\tilde{N}} = \bar{R}$  so that  $p = 0$ 
  - thus, the smaller  $p$  is, the more uniform is the mesh

## Methods for triangulating planar regions

### TRIANGLE

J. SHEWCHUK, Delaunay Refinement Algorithms for Triangular Mesh Generation, *Comput. Geom.: Theory Appl.* **22** 2002, pp. 21-74

J. SHEWCHUK, Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator, *Lecture Notes in Comput. Sci.* **1148**, Springer, New York, 1996, pp. 203-222

- A Delaunay triangulation is refined by halving edges and/or inserting circumcenters in such a way that
  - triangles having an area greater than a specified area are subdivided
  - angles smaller than a specified angle are eliminated

## DISTMESH

P.-O. PERSSON AND G. STRANG, A simple mesh generator in Matlab, *SIAM Review* **46** 2004, pp. 329-345

- A triangulation is viewed as a system of point masses connected by springs
  - the point masses are moved until a static equilibrium is achieved

## CVT

- The set of vertex points that are the generators of a CVT of the domain that is constrained to have some points on the boundary

## MESHGEN

- A variant of the CVT optimization process in which Voronoi regions are replaced by the easier to construct region formed by joining
  - the circumcenters of acute triangles
  - the mid-sides of the longest sides of obtuse triangles

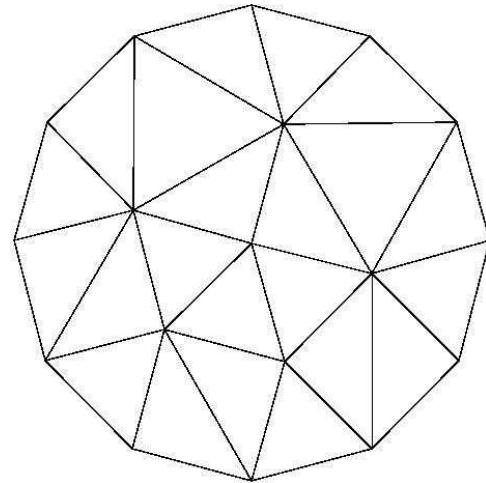
that surround a vertex in the triangulation

## VTM

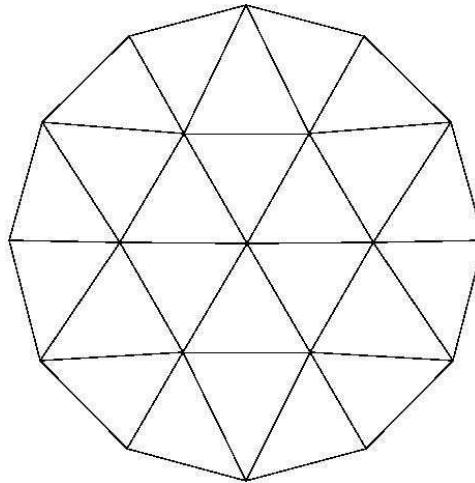
P. ALLIEZ, D. COHEN-STEINER, M. YVINEC, AND M. DESBRUN, Variational tetrahedral meshing, *Proceedings of ACM SIGGRAPH'05*, 2005, pp. 617-625

- Can be viewed as another variant of CVT in which the Voronoi regions are replaced by patches of Delaunay triangles that surround a vertex
  - works directly with the Delaunay triangulation
  - patches corresponding to two adjacent vertices overlap

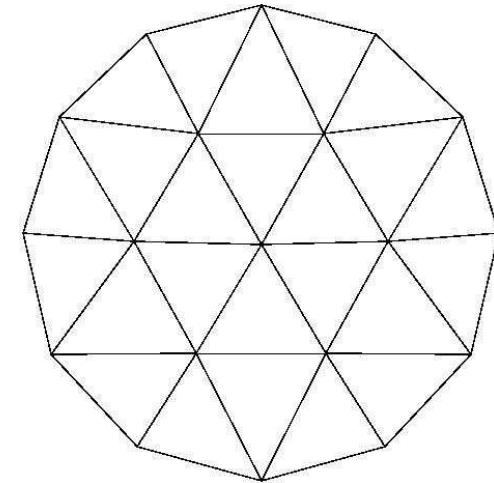
## Eyeball norm comparisons



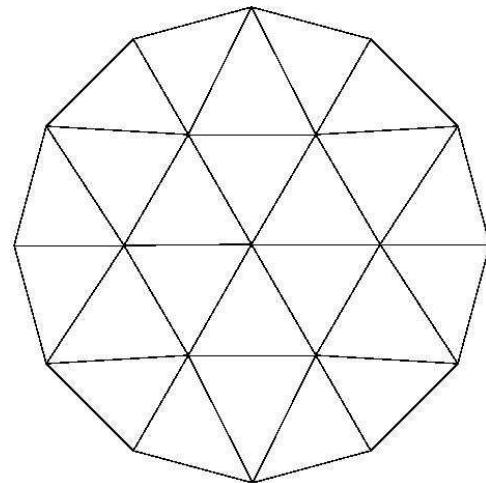
TRIANGLE



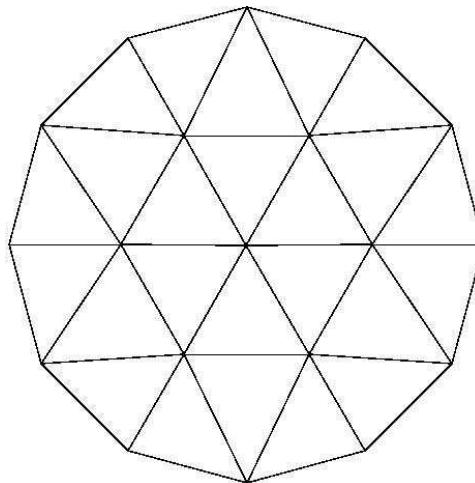
DISTMESH



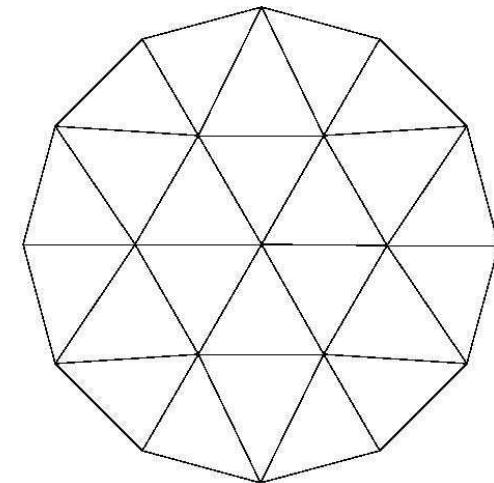
MESHGEN



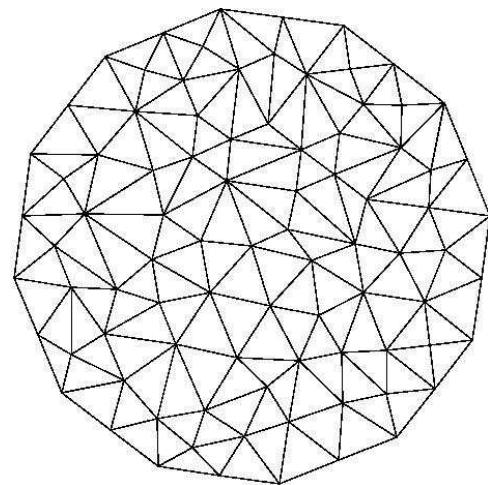
VTM



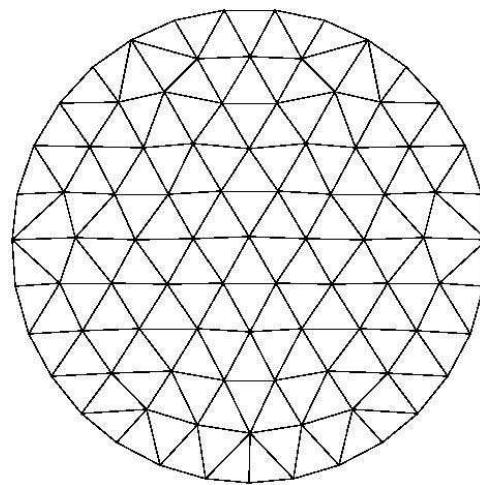
CVT1



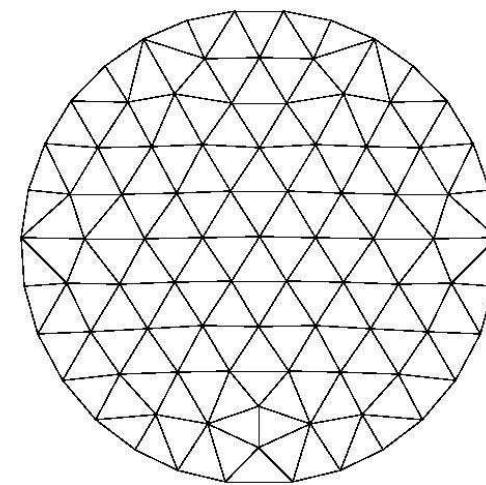
CVT2



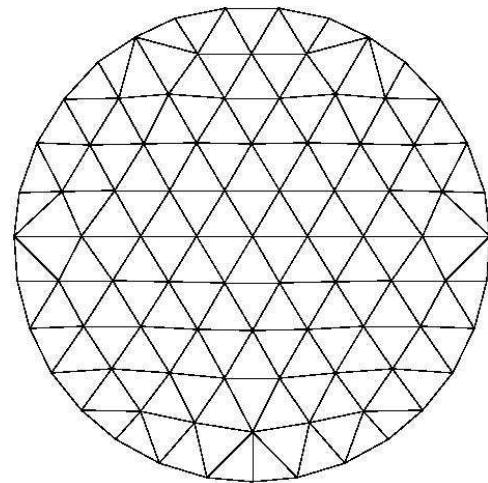
TRIANGLE



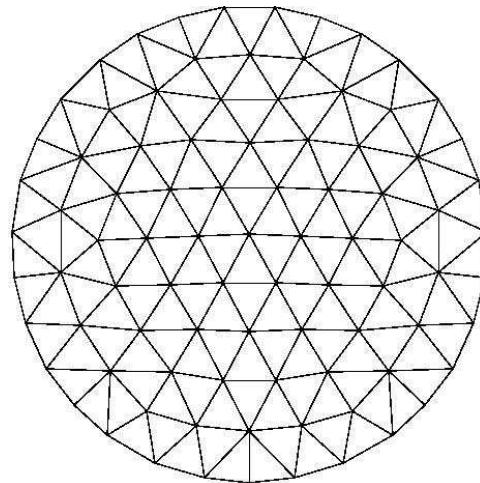
DISTMESH



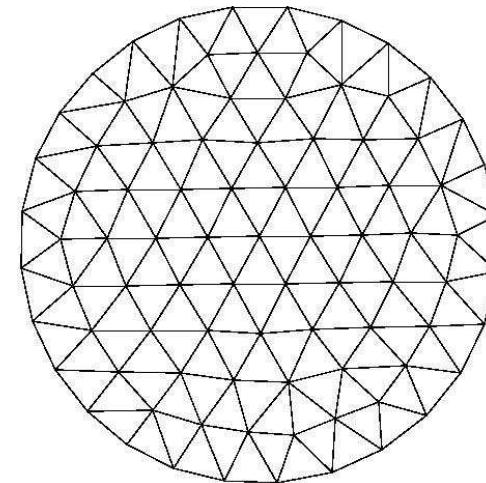
MESHGEN



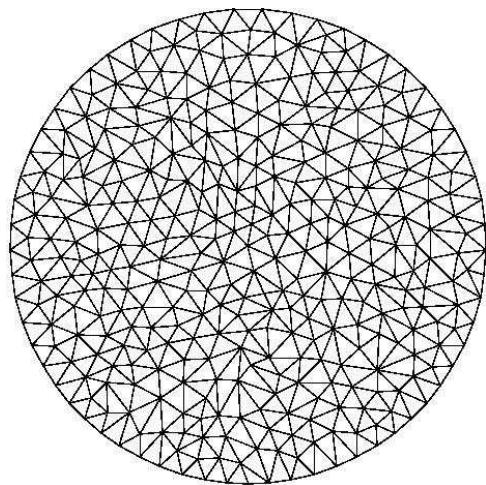
VTM



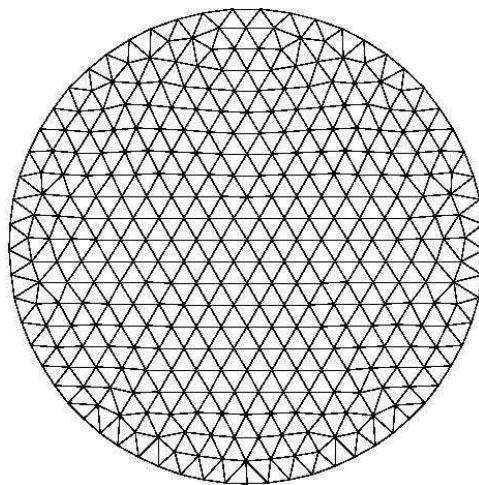
CVT1



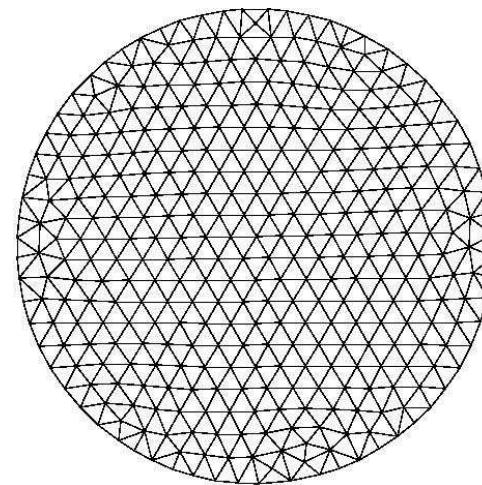
CVT2



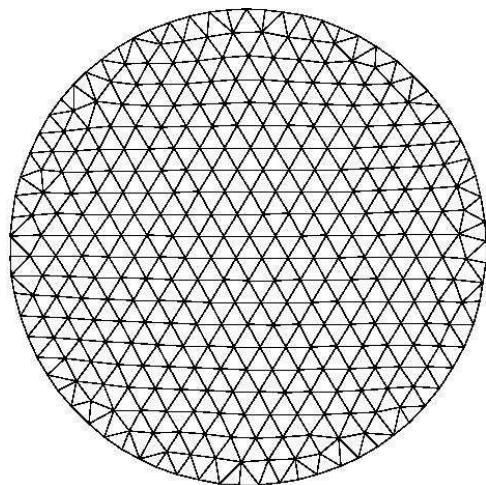
TRIANGLE



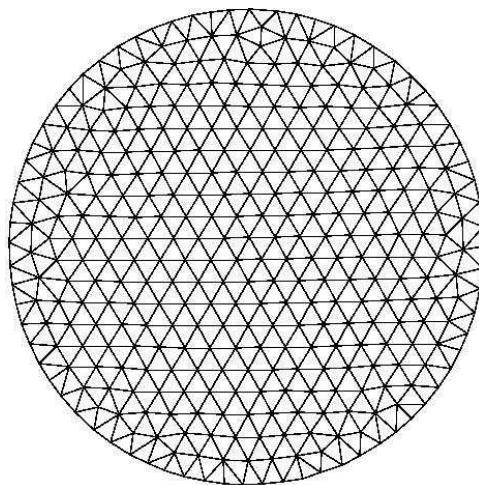
DISTMESH



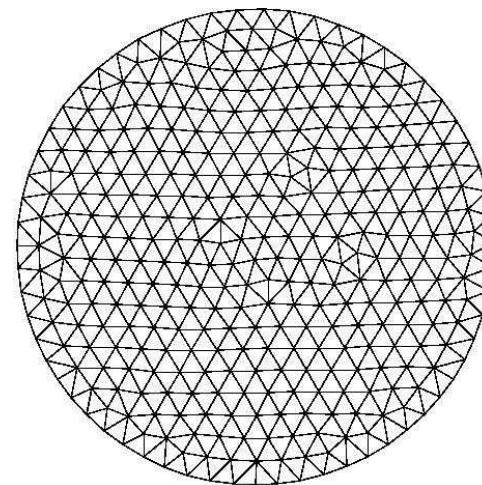
MESHGEN



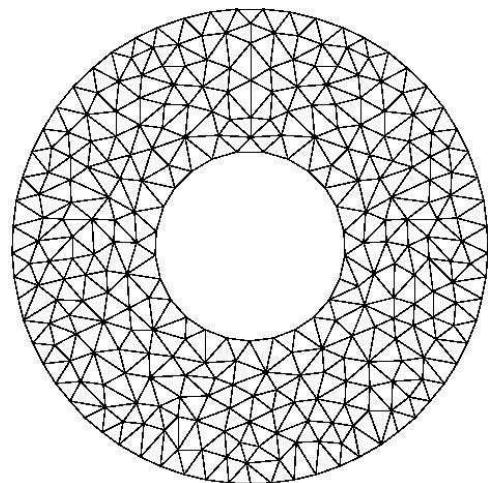
VTM



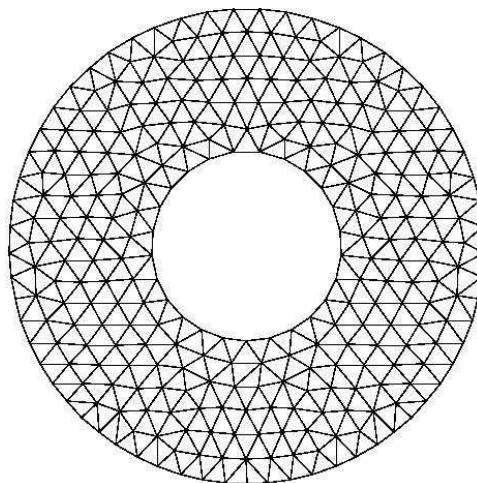
CVT1



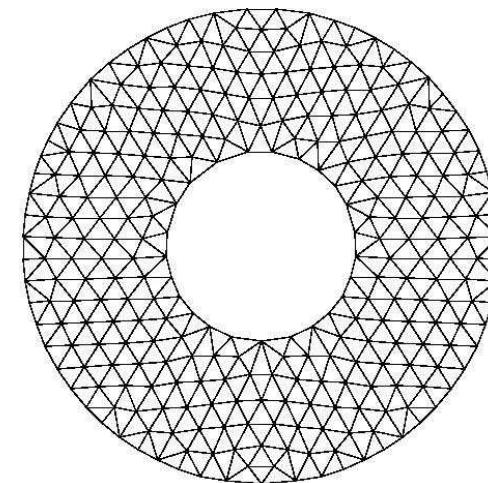
CVT2



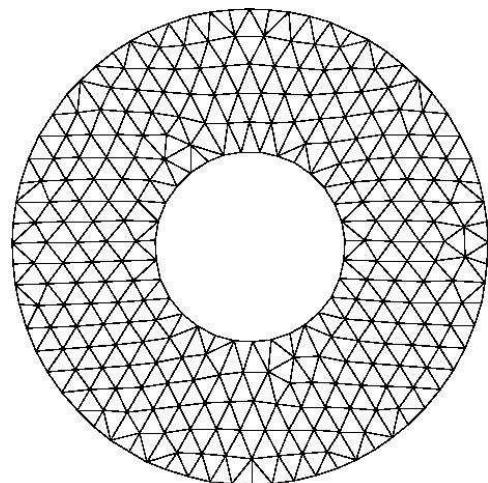
TRIANGLE



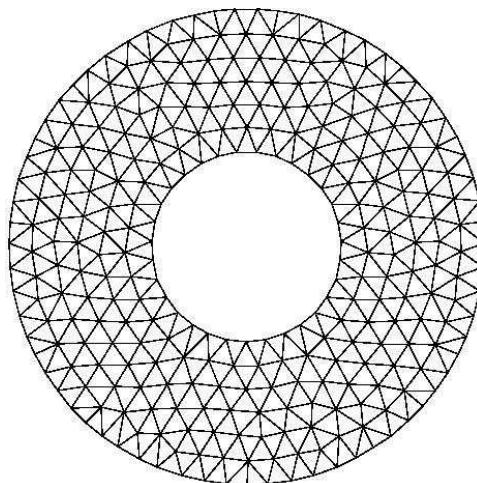
DISTMESH



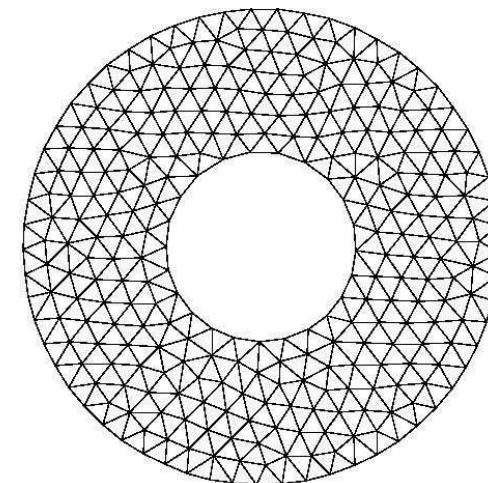
MESHGEN



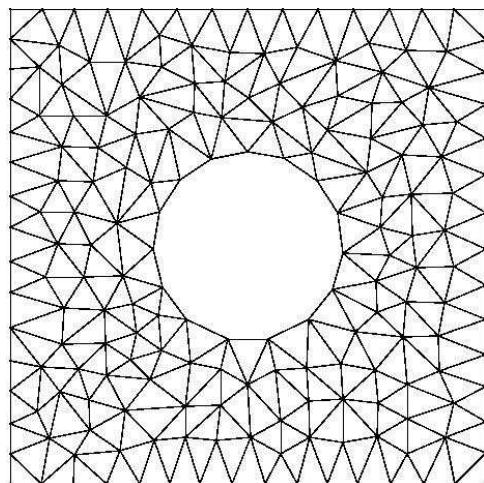
VTM



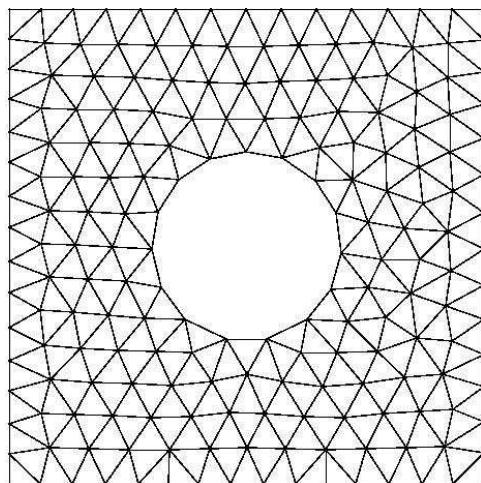
CVT1



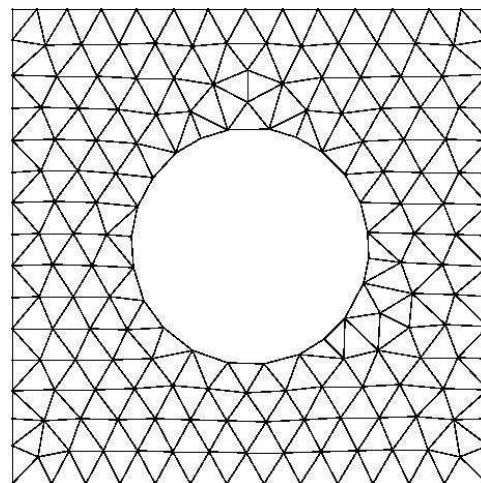
CVT2



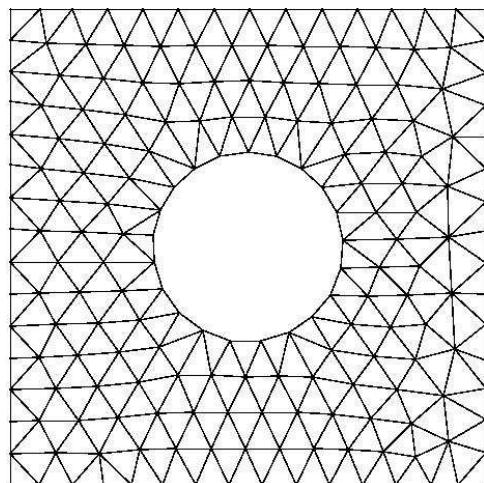
TRIANGLE



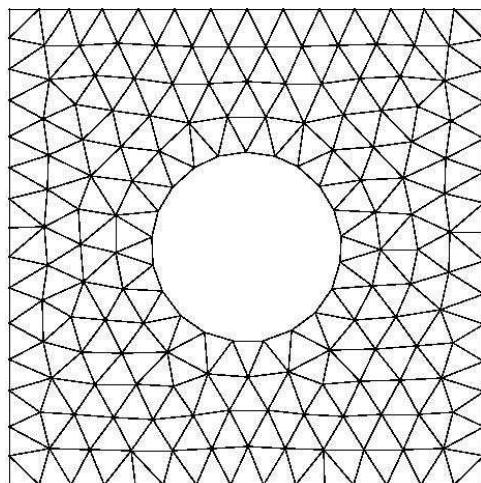
DISTMESH



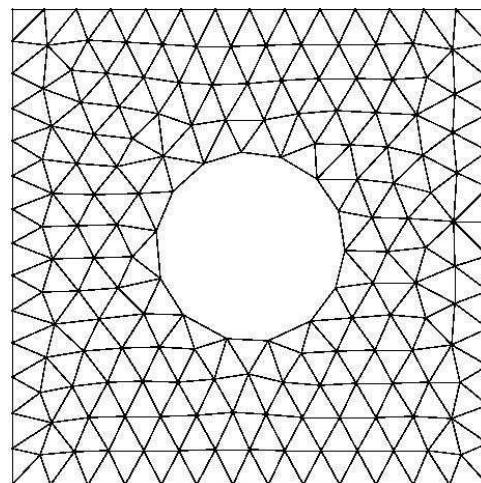
MESHGEN



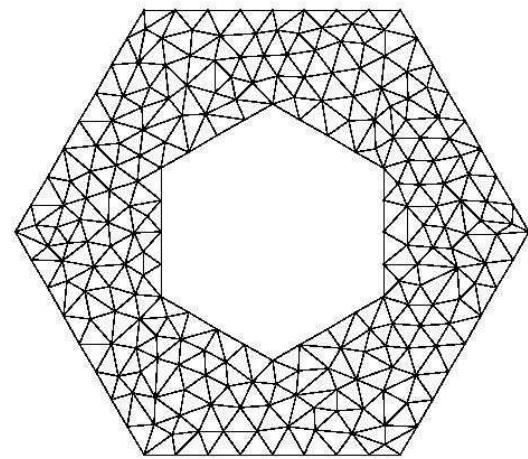
VTM



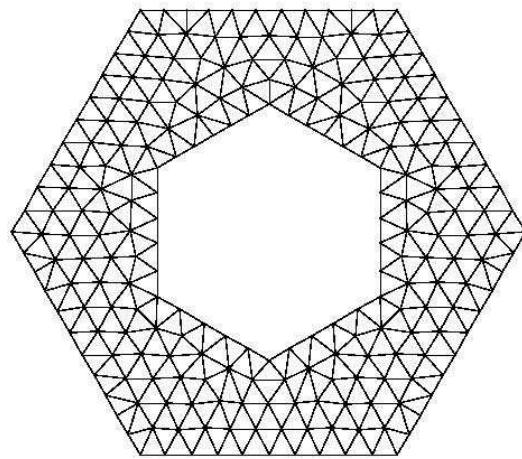
CVT1



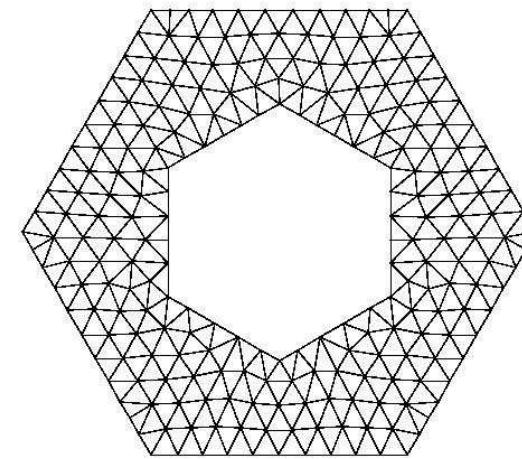
CVT2



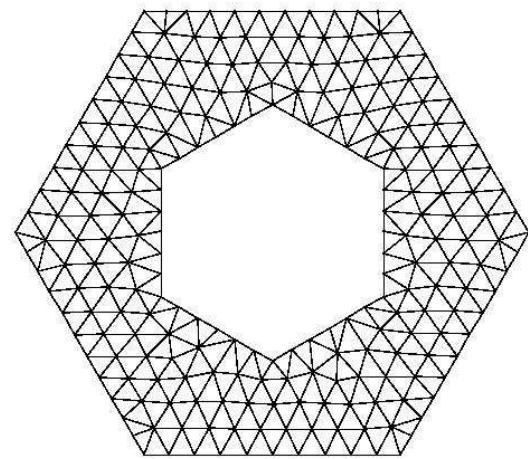
TRIANGLE



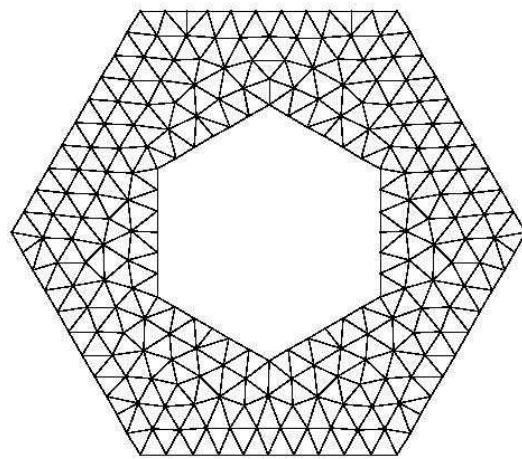
DISTMESH



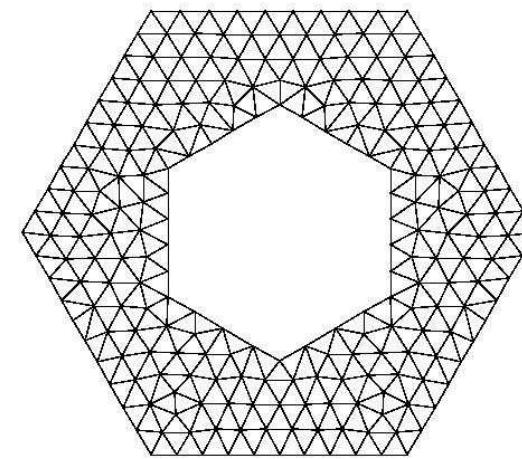
MESHGEN



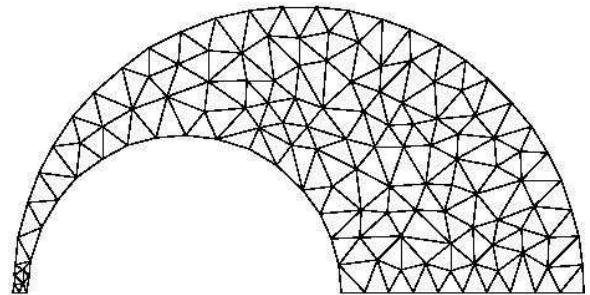
VTM



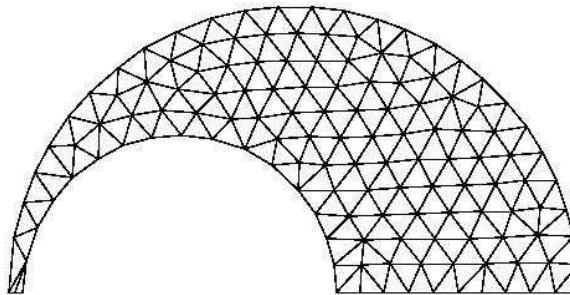
CVT1



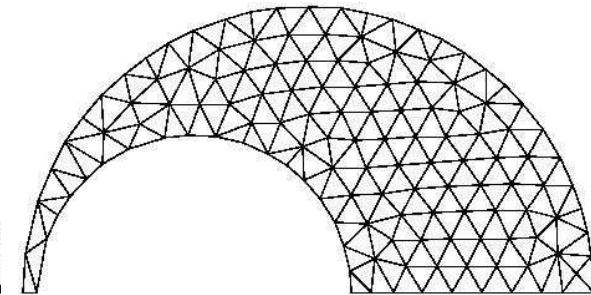
CVT2



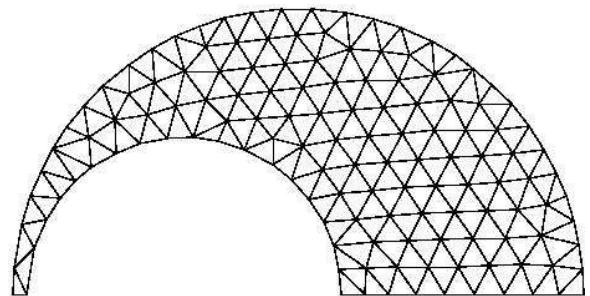
TRIANGLE



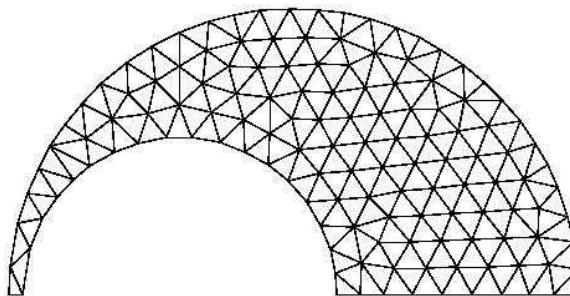
DISTMESH



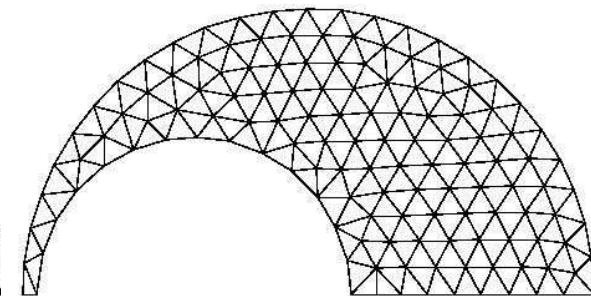
MESHPGEN



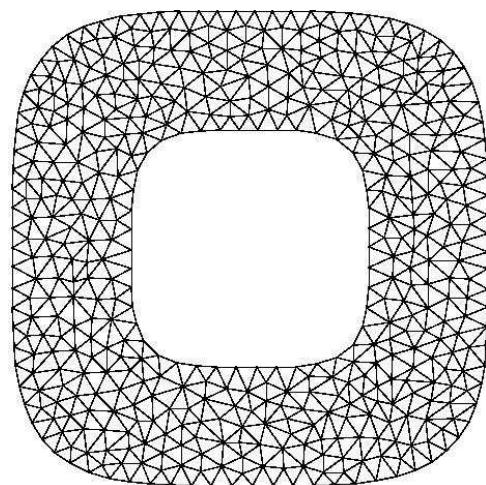
VTM



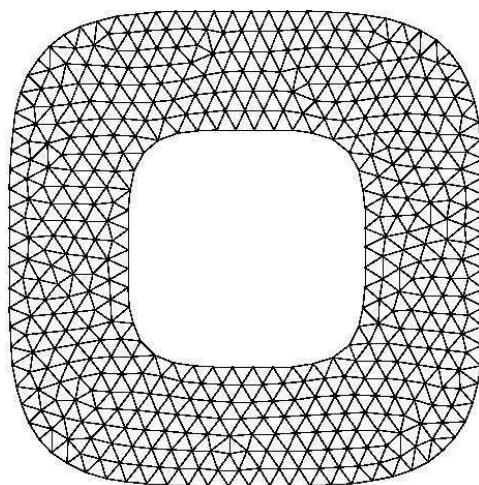
CVT1



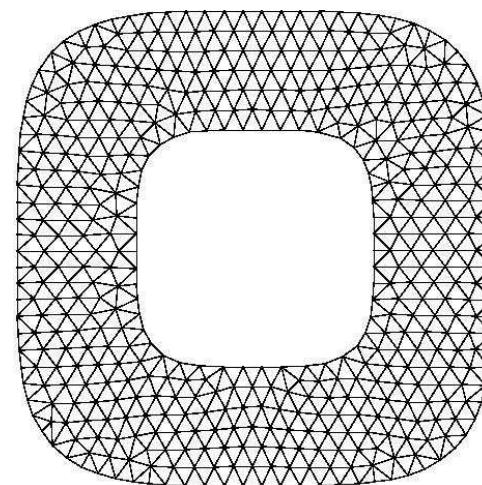
CVT2



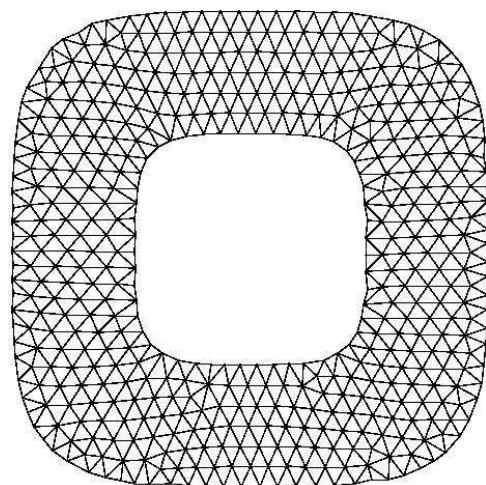
TRIANGLE



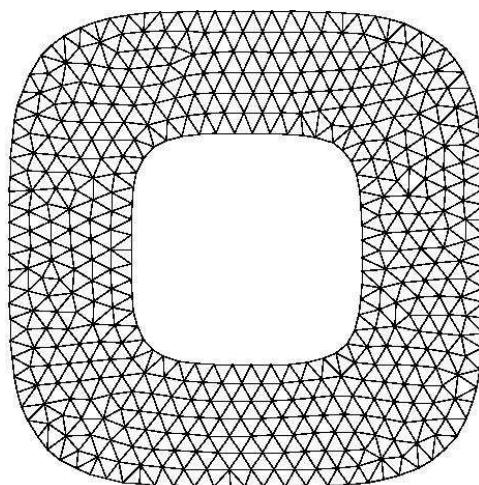
DISTMESH



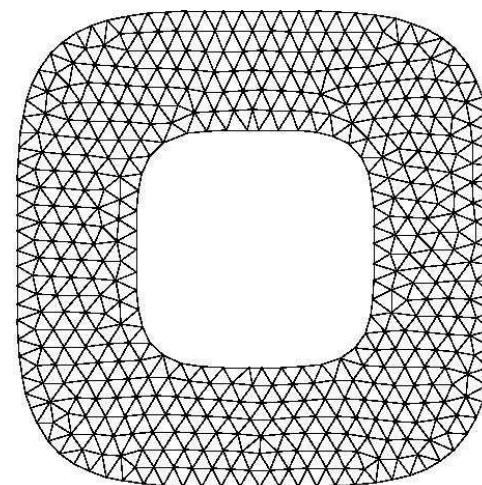
MESHGEN



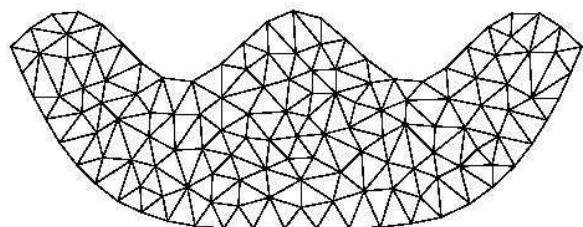
VTM



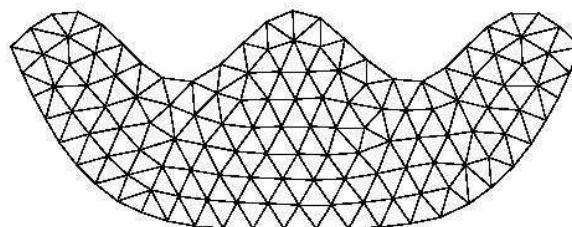
CVT1



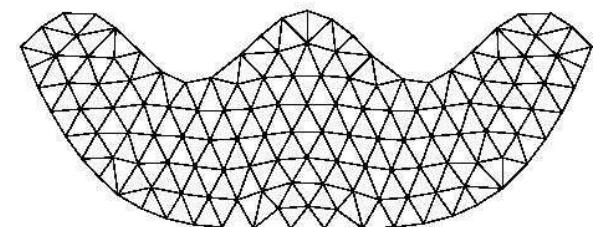
CVT2



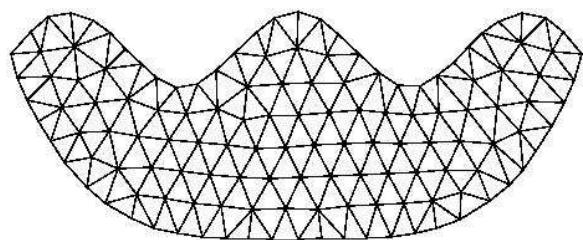
TRIANGLE



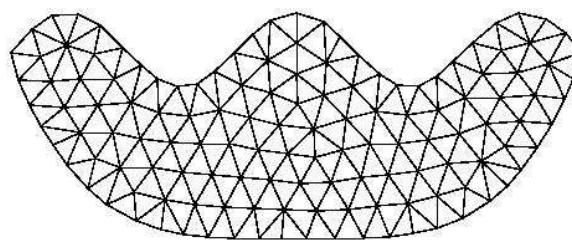
DISTMESH



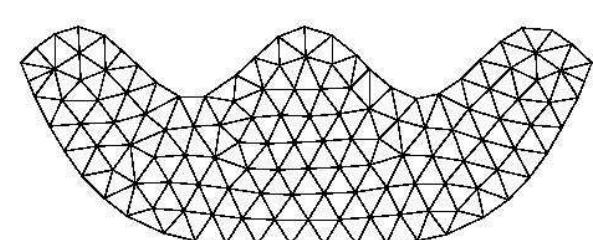
MESHGEN



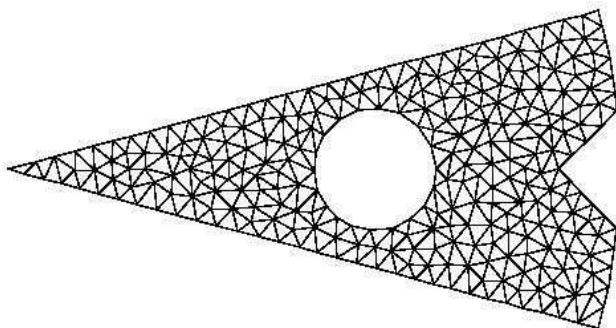
VTM



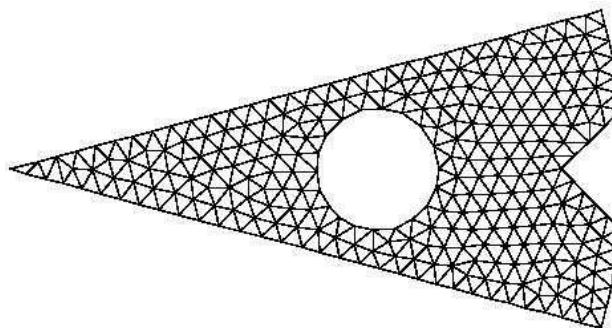
CVT1



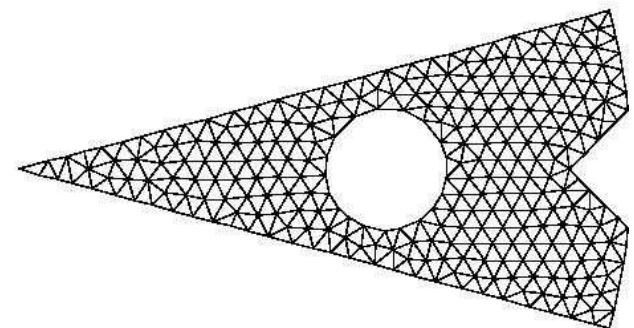
CVT2



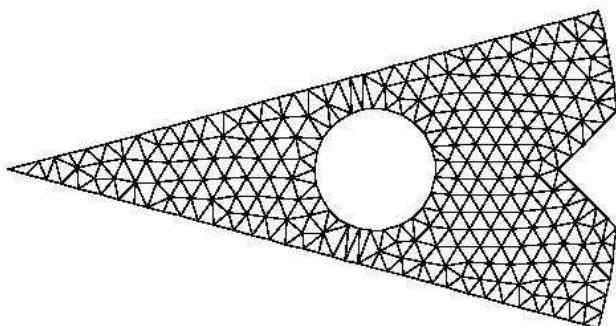
TRIANGLE



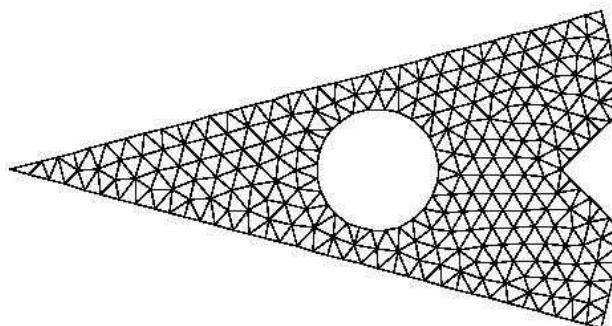
DISTMESH



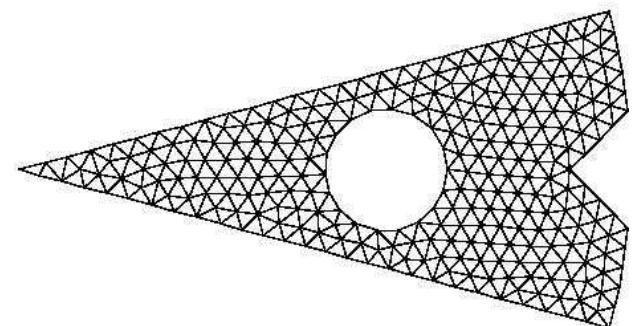
MESHGEN



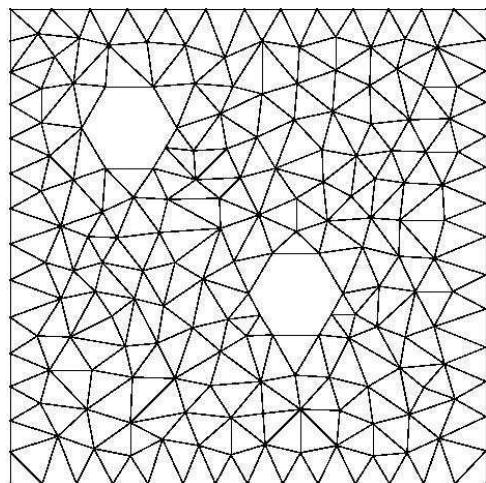
VTM



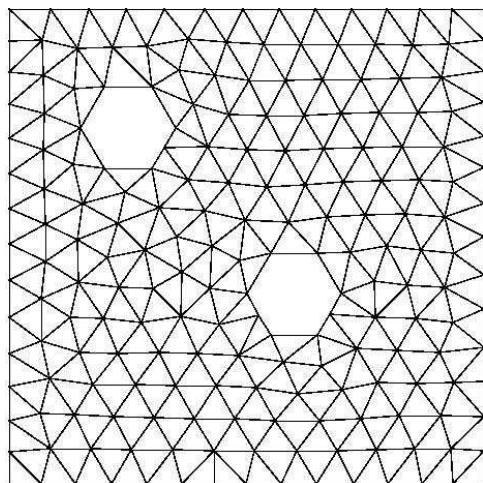
CVT1



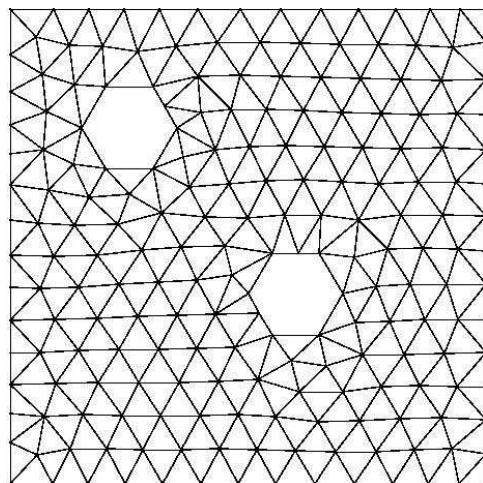
CVT2



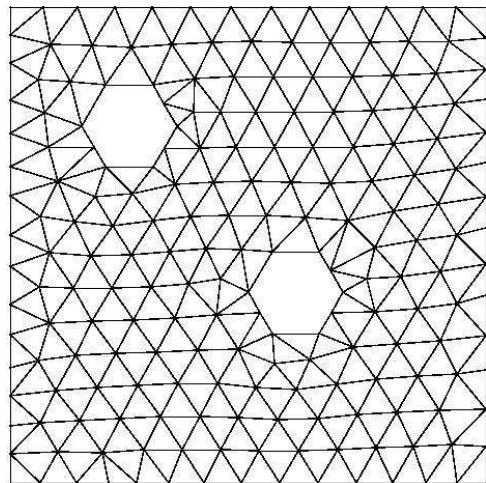
TRIANGLE



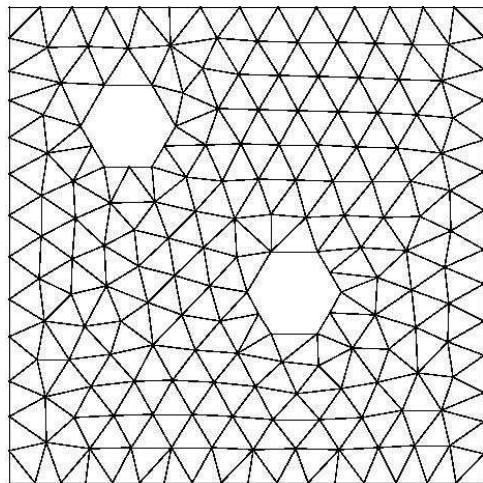
DISTMESH



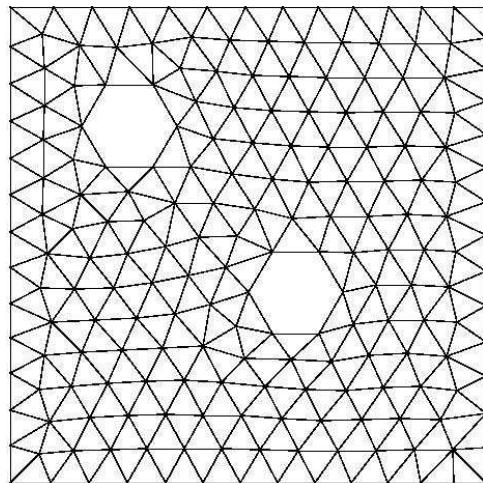
MESHGEN



VTM

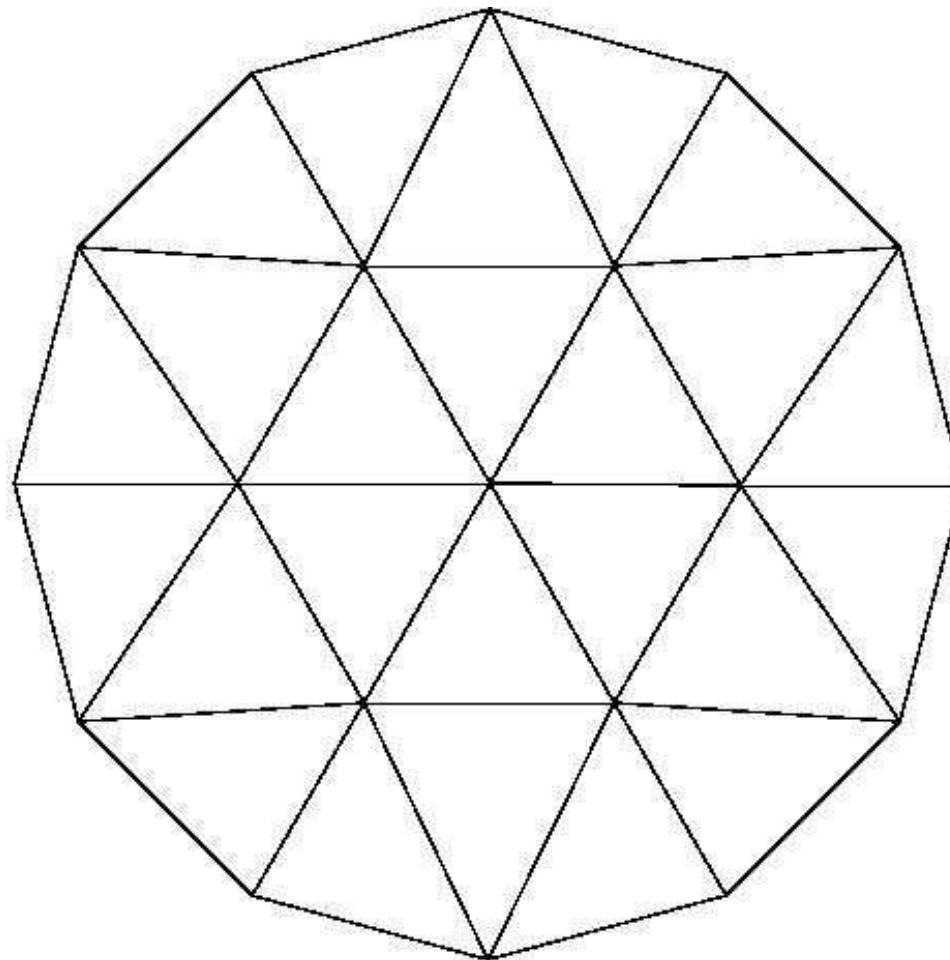


CVT1

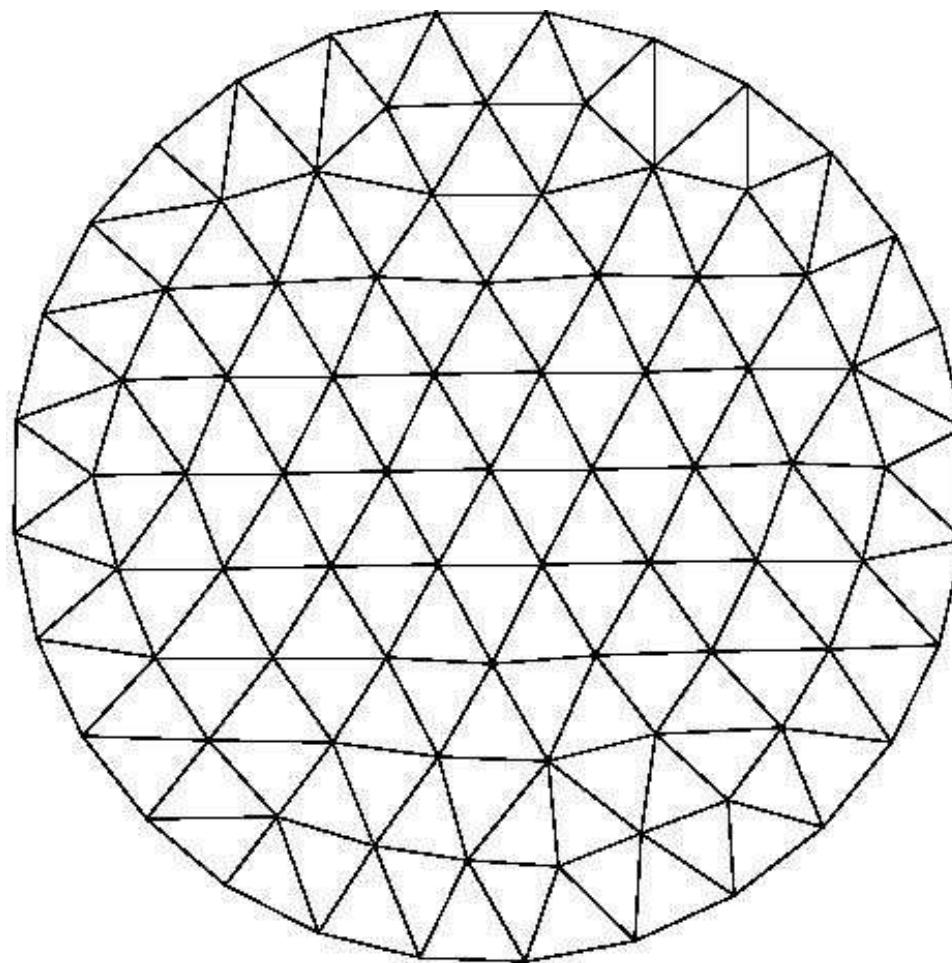


CVT2

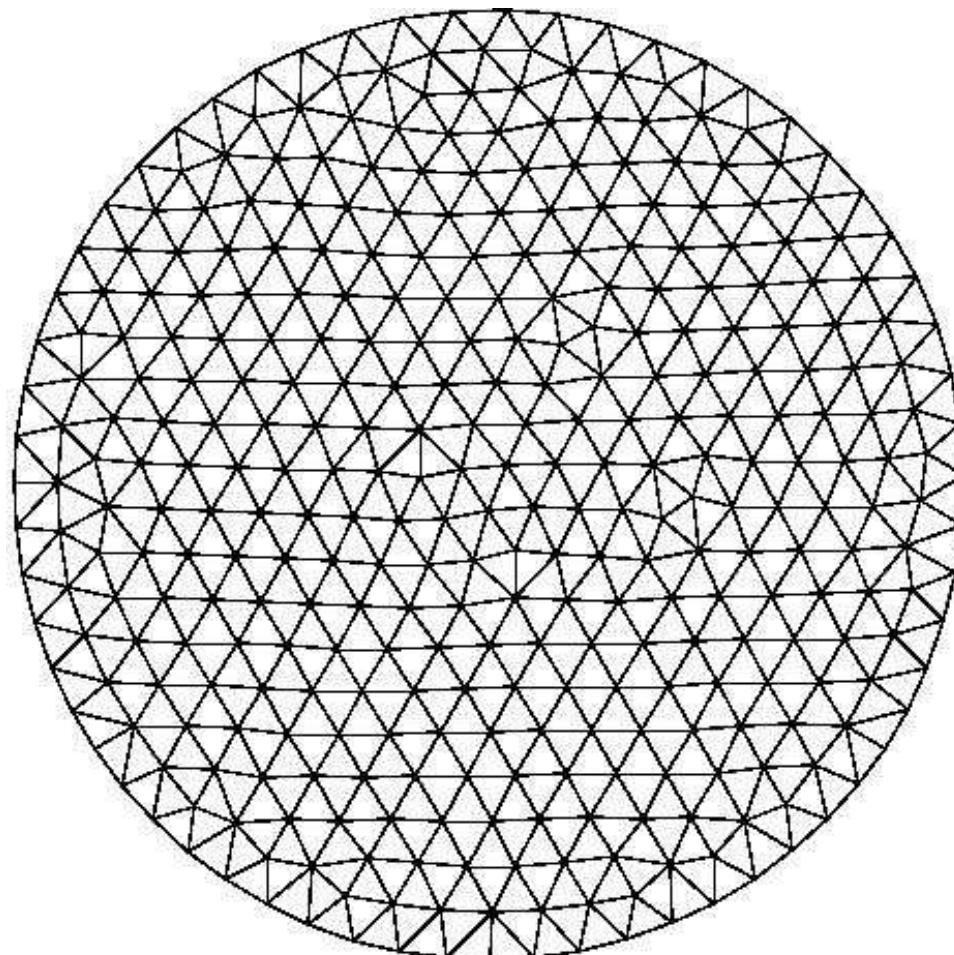
## Comparison using 8 quantitative measures



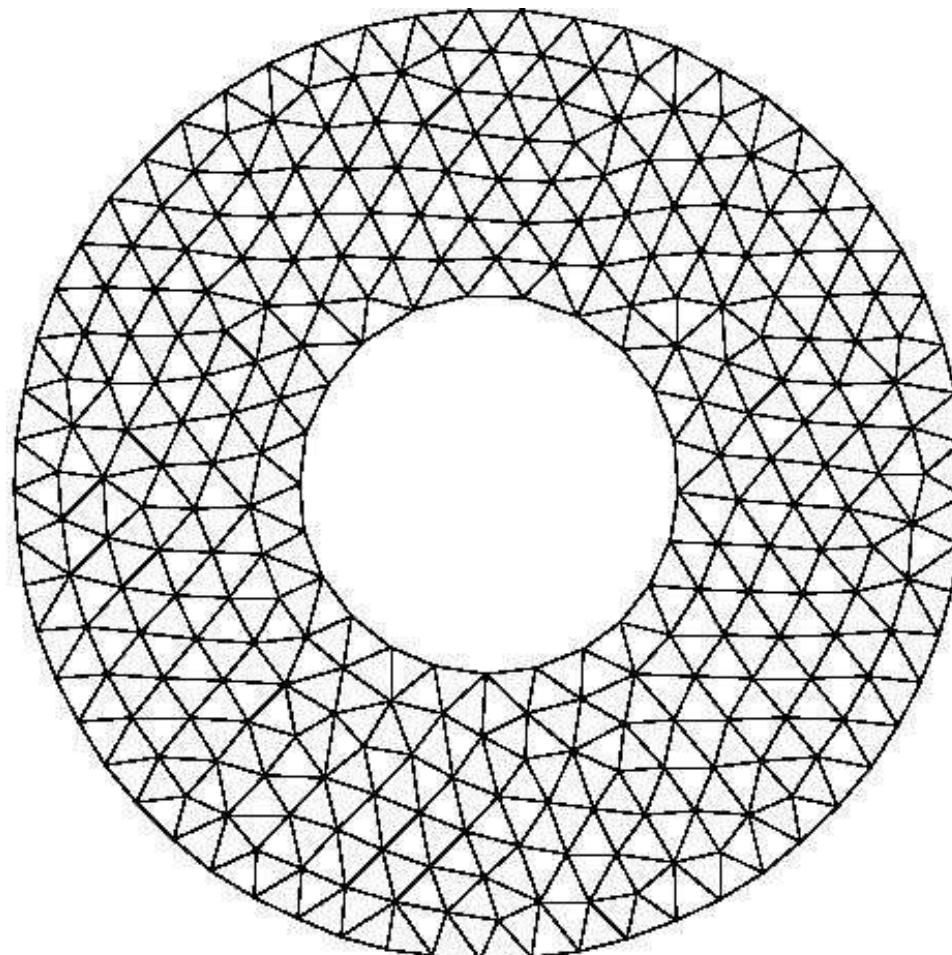
	TRIANGLE	DISTMESH	MESHGEN	TVM	CVT1	CVT2
no. points	18	19	19	19	19	19
no. triangles	22	24	24	24	24	24
$h$	0.21	0.17	0.17	0.17	0.17	0.17
$\chi$	1.62	1.42	1.43	1.43	1.41	1.41
$\tau \times 10^3$	6.49	3.29	3.24	3.36	3.33	3.34
$d \times 10^6$	2.50	3.41	3.90	3.33	3.31	3.30
$\alpha \times 10^2$	5.77	3.60	3.64	3.59	3.59	3.58
$\beta \times 10^2$	2.24	2.07	2.11	2.07	2.05	2.04
$q$	1.20	1.06	1.08	1.05	1.05	1.05
$p \times 10^2$	11.50	3.60	3.70	3.50	3.70	3.70



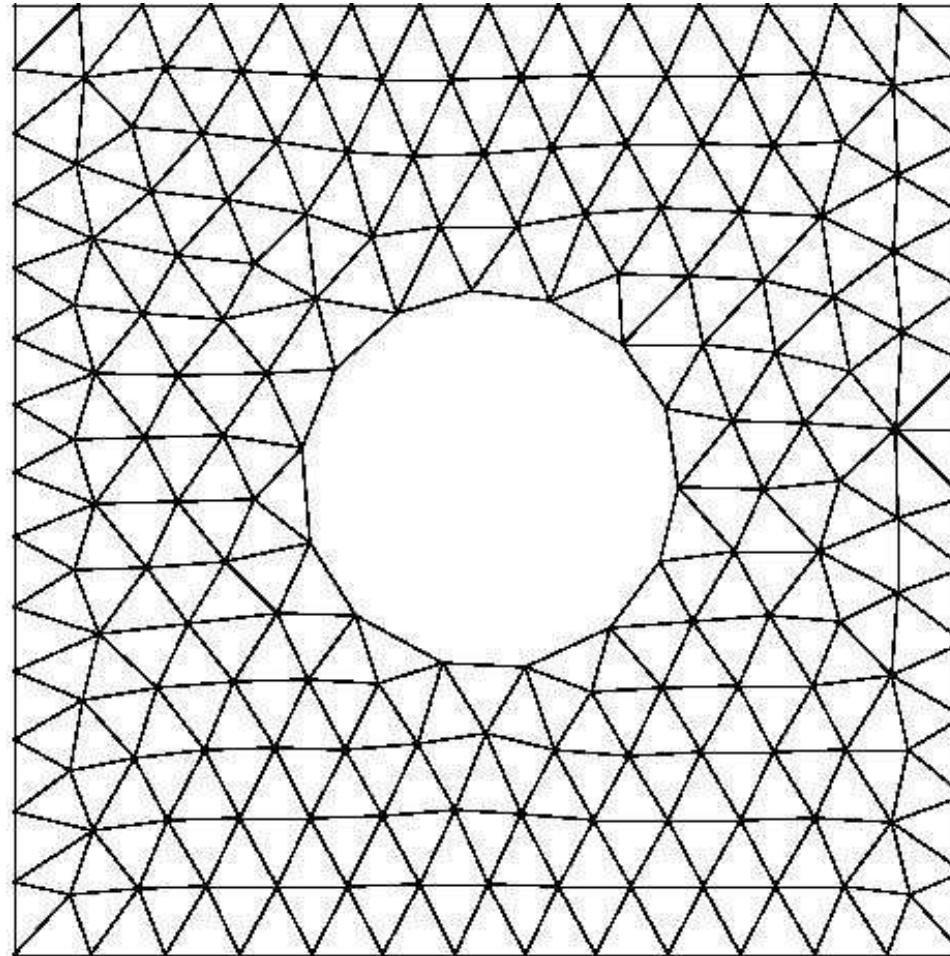
	TRIANGLE	DISTMESH	MESHGEN	TVM	CVT1	CVT2
no. points	86	88	88	88	88	88
no. triangles	146	143	144	143	143	146
$h \times 10^2$	8.85	7.42	7.22	7.42	7.24	7.31
$\chi$	2.02	1.56	1.62	1.68	1.50	1.54
$\tau \times 10^4$	6.87	6.33	6.16	6.36	6.51	5.02
$d \times 10^7$	2.29	0.91	1.05	0.98	1.00	1.30
$\alpha \times 10^3$	7.97	6.38	6.30	6.73	6.69	6.67
$\beta \times 10^2$	3.35	2.49	2.48	2.54	2.38	2.46
$q$	2.04	1.37	1.25	1.37	1.20	1.28
$p \times 10^2$	12.30	4.00	3.60	4.30	3.50	3.80



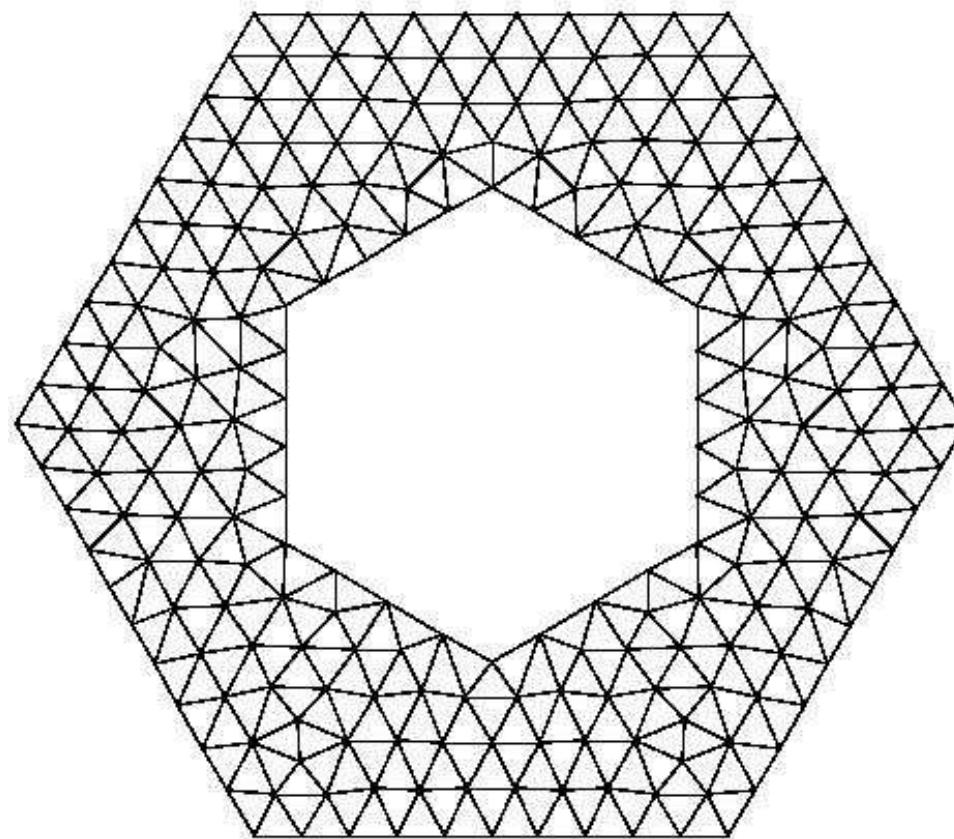
	TRIANGLE	DISTMESH	MESHGEN	TVM	CVT1	CVT2
no. points	362	362	362	362	362	362
no. triangles	661	661	657	650	651	661
$h \times 10^2$	4.29	3.30	3.43	3.37	3.41	3.37
$\chi$	2.18	1.51	1.70	1.78	1.76	1.48
$\tau \times 10^4$	2.06	1.54	1.84	2.03	1.85	1.51
$d \times 10^8$	1.09	0.60	0.63	0.47	0.45	0.63
$\alpha \times 10^3$	1.80	1.35	1.50	1.48	1.52	1.41
$\beta \times 10^2$	3.70	2.41	2.77	2.87	2.89	2.33
$q$	1.96	1.27	1.35	1.52	1.37	1.22
$p \times 10^2$	13.30	2.20	2.80	3.70	2.90	2.90



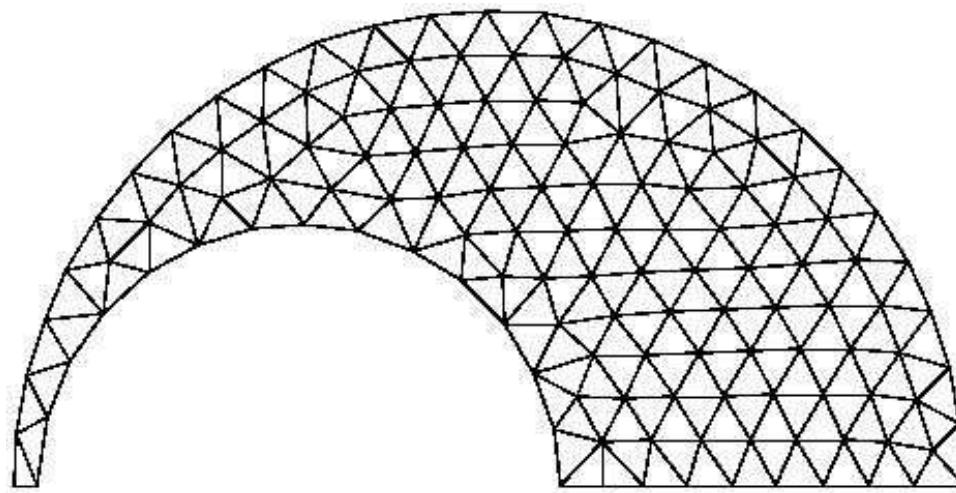
	TRIANGLE	DISTMESH	MESHGEN	TVM	CVT1	CVT2
no. points	303	303	303	303	303	303
no. triangles	520	520	519	507	508	524
$h \times 10^2$	<b>4.44</b>	3.64	3.76	3.99	3.65	<b>3.56</b>
$\chi$	2.17	1.55	1.75	<b>2.29</b>	1.99	<b>1.47</b>
$\tau \times 10^4$	<b>2.34</b>	1.59	1.98	1.93	1.70	<b>1.36</b>
$d \times 10^9$	<b>8.69</b>	6.80	8.44	<b>5.66</b>	6.00	8.41
$\alpha \times 10^3$	1.89	1.56	<b>1.51</b>	<b>1.91</b>	1.56	1.56
$\beta \times 10^2$	3.33	2.51	2.85	<b>3.86</b>	3.36	<b>2.33</b>
$q$	1.75	1.33	1.45	<b>1.96</b>	1.32	<b>1.22</b>
$p \times 10^2$	<b>12.20</b>	<b>3.20</b>	4.70	6.50	3.70	3.80



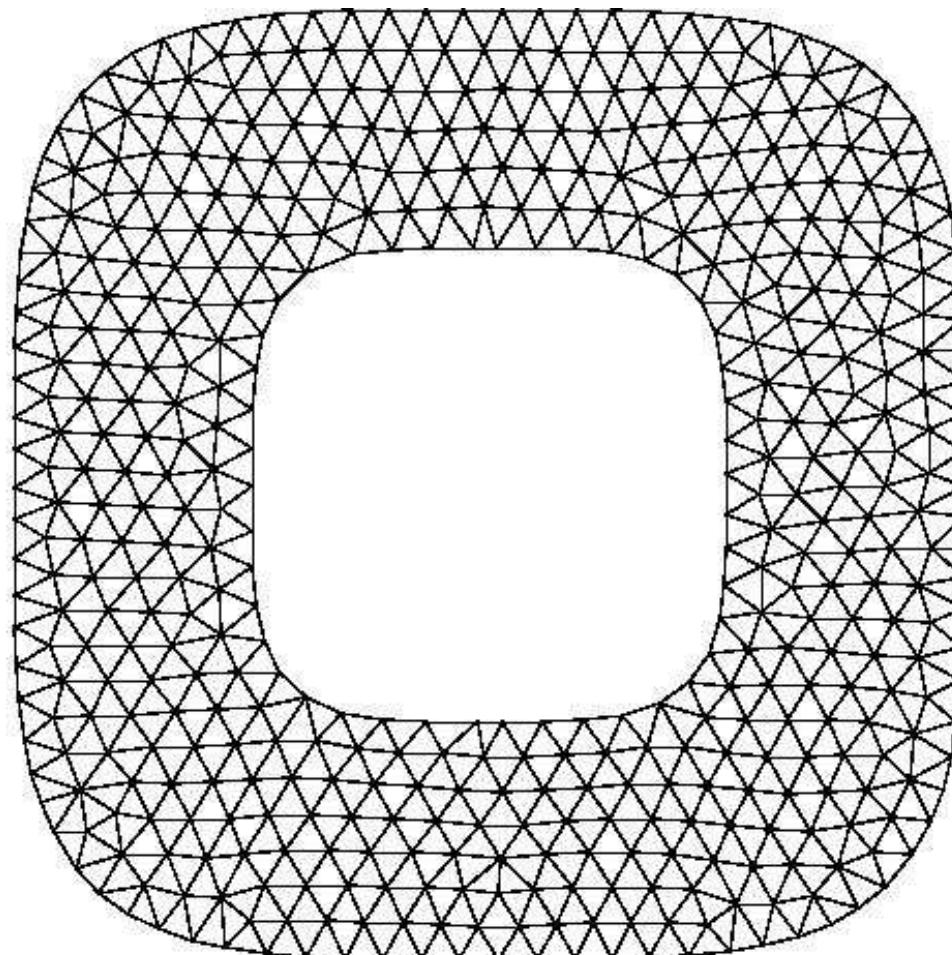
	TRIANGLE	DISTMESH	MESHGEN	TVM	CVT1	CVT2
no. points	193	194	194	194	194	194
no. triangles	313	315	308	308	308	320
$h$	0.13	0.10	0.13	0.12	0.11	0.11
$\chi$	2.11	1.64	2.35	2.10	1.80	1.60
$\tau \times 10^3$	2.50	2.39	2.48	2.44	2.39	2.35
$d \times 10^7$	9.02	5.77	5.94	2.75	1.88	7.44
$\alpha \times 10^2$	1.70	1.31	1.28	1.78	1.45	1.35
$\beta \times 10^2$	3.54	2.74	2.86	3.10	2.94	2.58
$q$	1.67	1.25	1.47	1.67	1.33	1.28
$p \times 10^2$	12.90	2.90	4.60	7.10	3.60	4.20



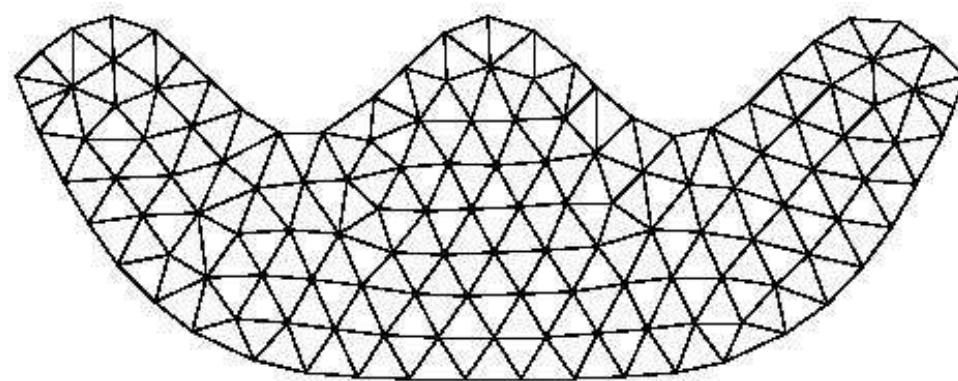
	TRIANGLE	DISTMESH	MESHGEN	TVM	CVT1	CVT2
no. points	242	240	240	240	240	240
no. triangles	412	385	385	381	381	393
$h \times 10^2$	<b>8.61</b>	7.32	7.30	7.78	<b>7.16</b>	7.37
$\chi$	<b>2.25</b>	1.75	1.85	2.02	1.80	<b>1.72</b>
$\tau \times 10^4$	8.83	<b>9.25</b>	9.19	8.98	8.94	<b>8.47</b>
$d \times 10^7$	<b>3.36</b>	1.53	1.09	0.94	<b>0.72</b>	1.72
$\alpha \times 10^3$	6.94	6.48	6.15	<b>7.49</b>	6.55	<b>6.00</b>
$\beta \times 10^2$	3.32	2.82	3.09	<b>3.33</b>	2.90	<b>2.56</b>
$q$	<b>2.00</b>	<b>1.37</b>	1.30	1.69	1.39	<b>1.37</b>
$p \times 10^2$	<b>11.60</b>	<b>3.60</b>	4.60	6.00	4.30	4.20



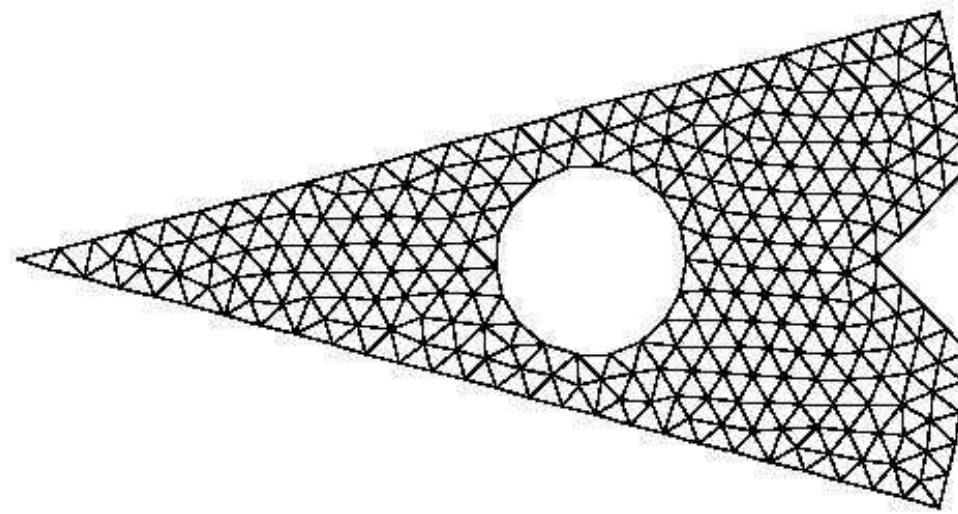
	TRIANGLE	DISTMESH	MESHGEN	TVM	CVT1	CVT2
no. points	137	138	138	138	138	138
no. triangles	213	219	216	214	214	224
$h \times 10^2$	<b>8.98</b>	7.22	7.77	7.57	7.70	<b>6.85</b>
$\chi$	3.10	<b>6.02</b>	2.93	<b>2.23</b>	<b>2.23</b>	2.50
$\tau \times 10^3$	<b>1.37</b>	1.09	1.13	1.09	1.09	<b>1.08</b>
$d \times 10^7$	<b>3.45</b>	1.37	2.99	0.95	<b>0.87</b>	1.61
$\alpha \times 10^3$	<b>8.87</b>	6.45	7.50	6.87	6.79	<b>6.15</b>
$\beta \times 10^2$	3.27	<b>10.27</b>	<b>4.84</b>	3.50	3.53	3.95
$q$	2.08	<b>3.57</b>	1.89	1.61	1.64	<b>1.49</b>
$p \times 10^2$	<b>20.30</b>	6.50	8.10	6.00	5.50	<b>3.70</b>



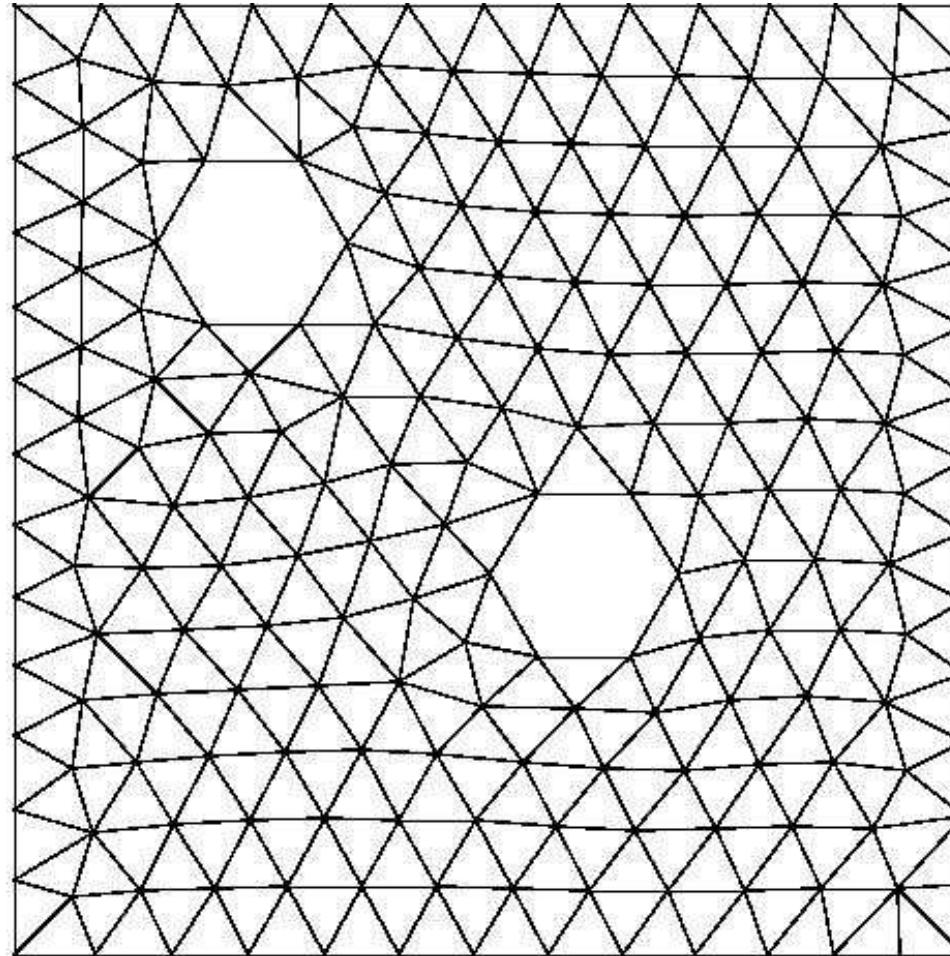
	TRIANGLE	DISTMESH	MESHGEN	TVM	CVT1	CVT2
no. points	505	505	505	505	505	505
no. triangles	882	882	868	864	862	873
$h \times 10^2$	<b>6.85</b>	<b>5.56</b>	5.84	6.54	6.03	5.82
$\chi$	2.07	<b>1.60</b>	2.00	2.12	2.13	<b>2.30</b>
$\tau \times 10^4$	5.28	<b>3.85</b>	4.82	<b>5.87</b>	5.50	4.57
$d \times 10^8$	<b>5.54</b>	5.24	3.44	4.76	3.27	<b>5.00</b>
$\alpha \times 10^3$	<b>4.80</b>	<b>3.77</b>	4.29	4.72	4.01	3.95
$\beta \times 10^2$	3.23	<b>2.43</b>	3.54	<b>3.57</b>	3.14	3.43
$q$	1.96	<b>1.27</b>	1.41	<b>2.08</b>	1.59	1.33
$p \times 10^2$	11.40	<b>3.00</b>	4.30	<b>6.00</b>	4.00	3.80



	TRIANGLE	DISTMESH	MESHGEN	TVM	CVT1	CVT2
no. points	137	136	136	136	136	136
no. triangles	222	220	220	210	213	219
$h \times 10$	7.00	5.62	5.61	6.21	5.81	5.47
$\chi$	2.10	1.60	1.71	2.36	1.91	2.15
$\tau \times 10^2$	5.55	5.30	5.54	5.69	4.98	6.08
$d \times 10^4$	3.99	3.83	4.51	3.26	2.78	3.46
$\alpha \times 10$	4.47	3.62	3.49	4.64	4.27	3.58
$\beta \times 10^2$	3.33	2.58	2.84	3.83	3.17	3.39
$q$	1.89	1.27	1.32	1.67	1.47	1.41
$p \times 10^2$	13.70	3.70	4.50	6.40	4.50	4.60



	TRIANGLE	DISTMESH	MESHGEN	TVM	CVT1	CVT2
no. points	285	287	287	287	287	287
no. triangles	470	474	479	452	451	484
$h \times 10^2$	2.62	2.28	2.36	2.93	2.30	2.18
$\chi$	2.13	1.64	1.84	3.32	2.25	1.82
$\tau \times 10^4$	1.24	1.20	1.13	1.14	1.12	1.14
$d \times 10^9$	2.81	1.84	1.19	0.97	0.51	1.37
$\alpha \times 10^4$	7.26	5.70	6.88	8.99	6.59	6.33
$\beta \times 10^2$	3.34	3.34	3.33	5.70	3.81	3.33
$q$	2.13	2.13	1.75	2.08	2.04	1.56
$p \times 10^2$	11.70	4.30	5.50	10.30	5.80	4.10



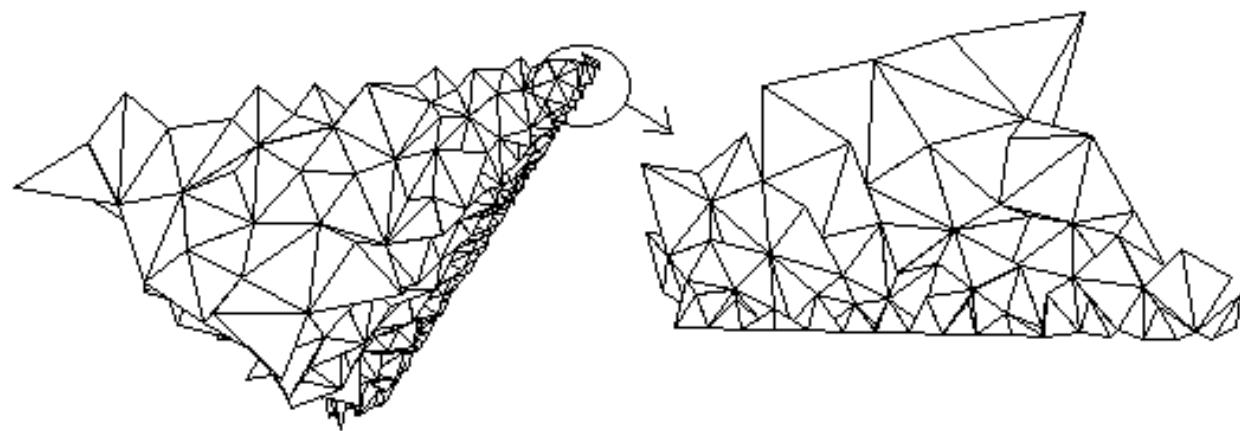
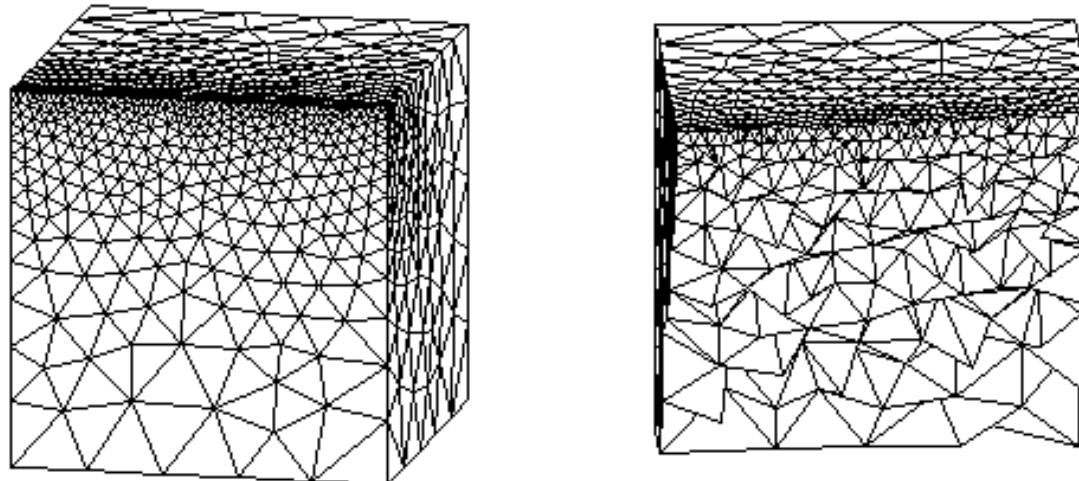
	TRIANGLE	DISTMESH	MESHGEN	TVM	CVT1	CVT2
no. points	202	203	203	203	203	203
no. triangles	336	338	339	332	343	334
$h$	6.65	5.87	5.46	5.77	5.58	5.65
$\chi$	2.40	2.11	2.37	2.25	1.70	2.51
$\tau \times 10^4$	6.30	6.12	6.53	6.42	6.13	6.28
$d \times 10^8$	5.80	8.56	3.56	5.79	8.27	4.49
$\alpha \times 10^3$	4.45	4.11	3.76	4.08	3.88	3.99
$\beta \times 10^2$	3.33	3.25	3.48	3.51	2.54	4.33
$q$	1.54	1.37	1.41	1.52	1.25	1.82
$p \times 10^2$	13.10	3.60	4.10	4.90	4.20	6.20

	TRIANGLE	DISTMESH	MESHGEN	TVM	CVT1	CVT2
$h$	0 - 8	3 - 0	1 - 0	1 - 2	1 - 0	4 - 0
$\chi$	0 - 3	3 - 1	0 - 1	1 - 3	2 - 0	3 - 1
$\tau$	0 - 5	3 - 0	0 - 1	0 - 2	1 - 0	5 - 0
$d$	0 - 7	0 - 1	1 - 1	2 - 1	5 - 0	1 - 0
$\alpha$	0 - 4	3 - 0	4 - 0	0 - 4	0 - 0	2 - 0
$\beta$	0 - 2	2 - 1	2 - 0	0 - 5	1 - 0	5 - 1
$q$	0 - 5	4 - 1	0 - 0	0 - 2	1 - 0	5 - 1
$p$	0 - 8	7 - 0	0 - 0	0 - 1	0 - 0	2 - 0
	0 - 42	25 - 4	8 - 3	4 - 20	11 - 0	27 - 3

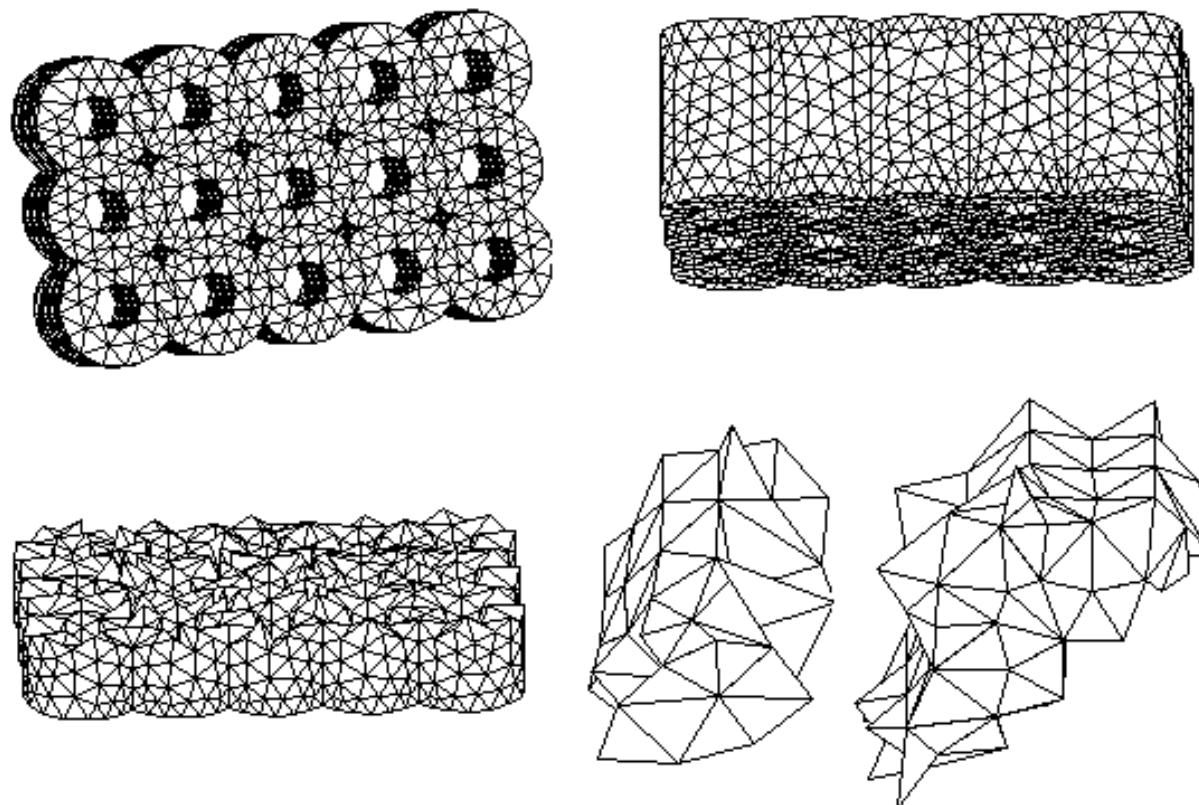
Scorecard

## 3D CVT MESH GENERATION

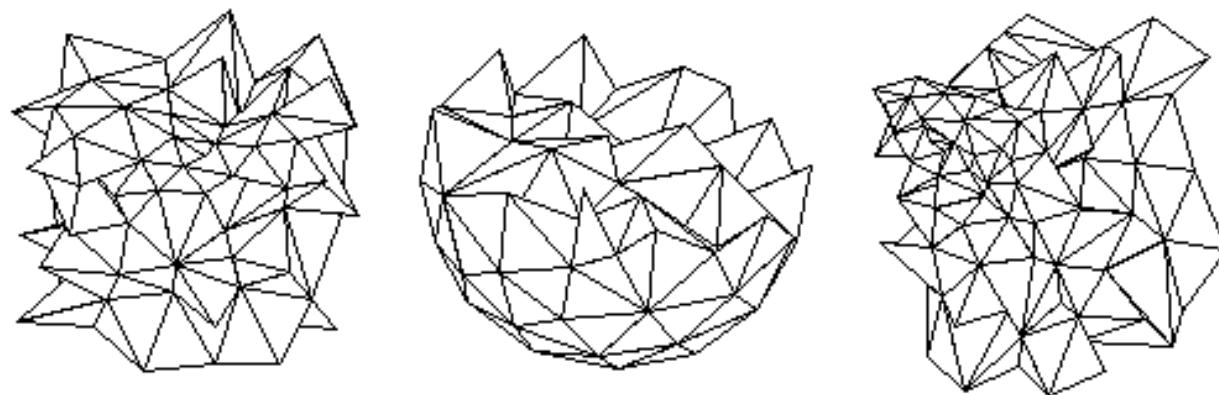
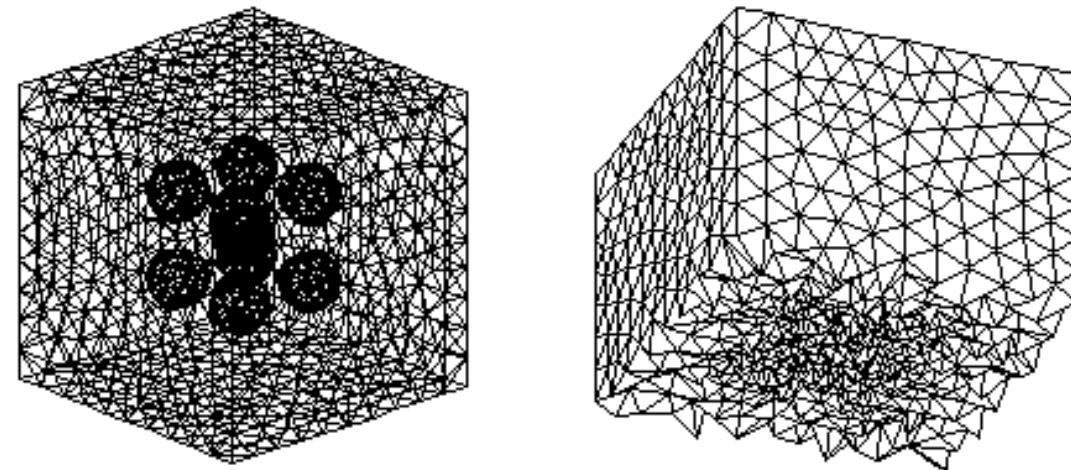
- CVT generators can be used to define the vertices of a Delaunay tetrahedral mesh in 3D
- Gersho's conjecture implies that bad-shaped tetrahedra, e.g., slivers, will be avoided
- Some examples from
  - Q. Du and D. Wang, Tetrahedral mesh generation and optimization based on centroidal Voronoi tessellations, *International Journal for Numerical Methods in Engineering* 56 2003, pp. 1355–1373



*CVT tetrahedral mesh in a cube*



*CVT tetrahedral mesh in a more complex region*

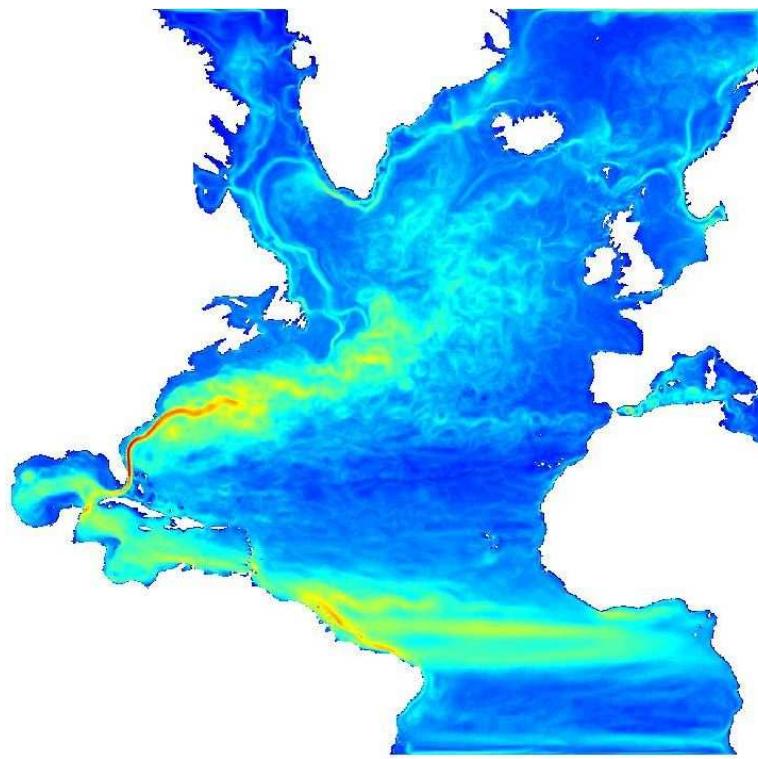


*CVT tetrahedral mesh for a composite material simulations*

## CVT MESH REFINEMENT

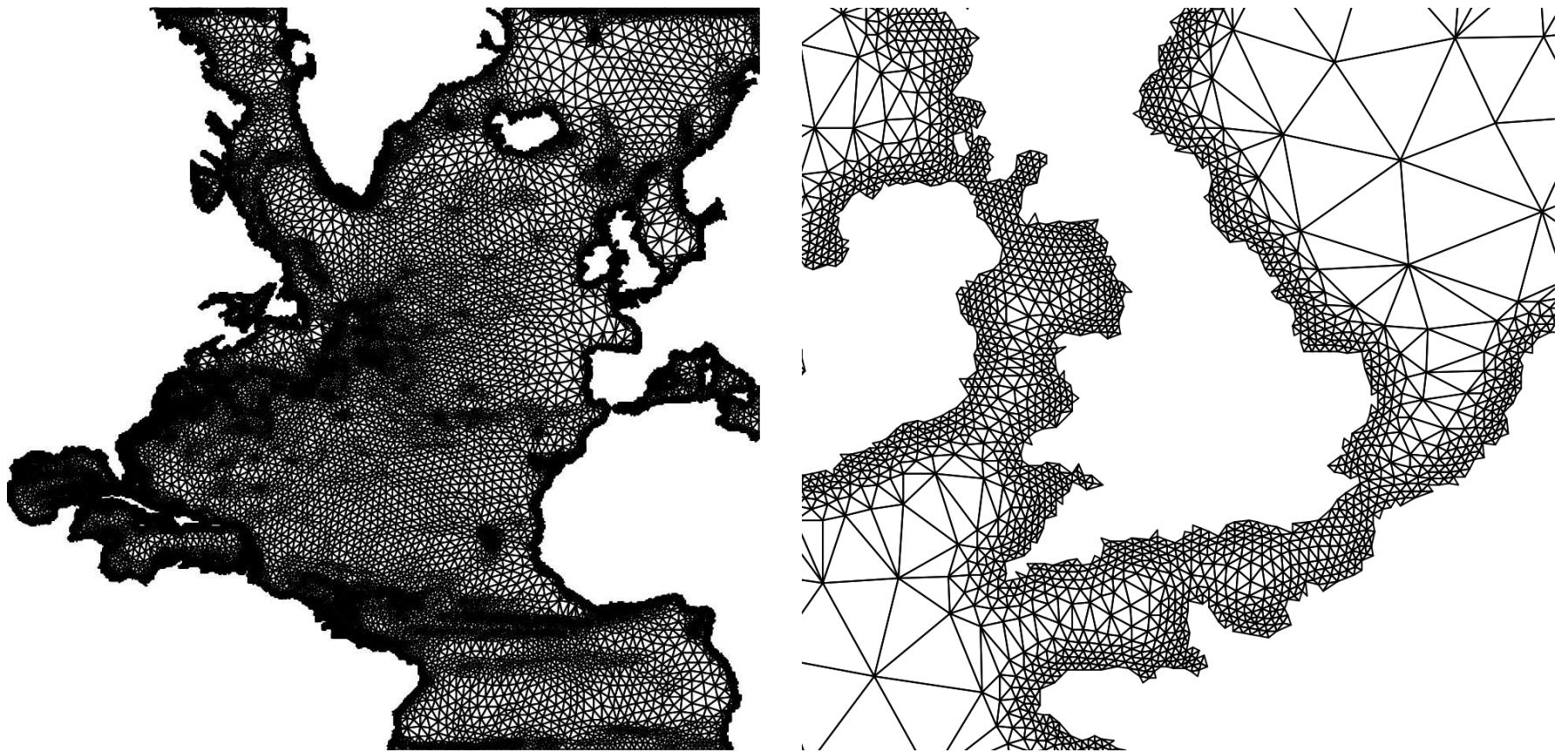
### Indicator-based (or passive) mesh refinement

- Mesh refinement is effected by relating the CVT density function to a given feature
  - the grid size will be smaller in regions where the CVT density is larger
- Good resolution of complicated boundaries can be effected by choosing the CVT density to be bigger near boundaries
  - e.g., choose the CVT density to decrease as the distance from the boundary increases
- Can also use measured data or some other known information to determine where the CVT density should be relatively larger

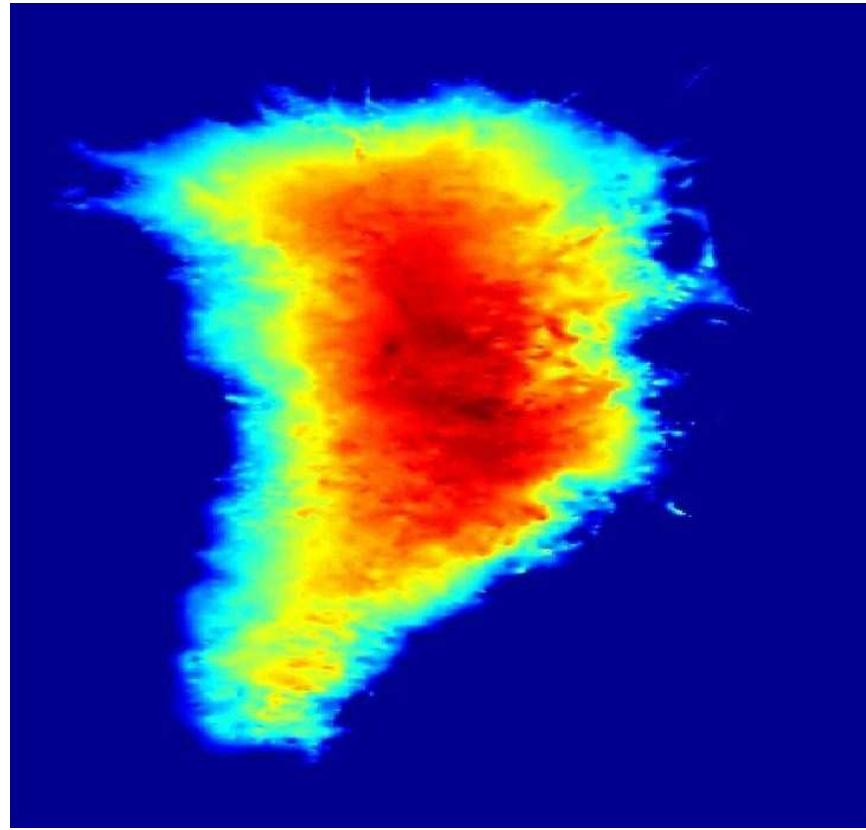


*Satellite data for the kinetic energy distribution in the North Atlantic; note that the boundary is quite complicated*

- The grid should be refined
  - where the kinetic energy is relatively large (red and yellow areas)
  - to resolve the boundary

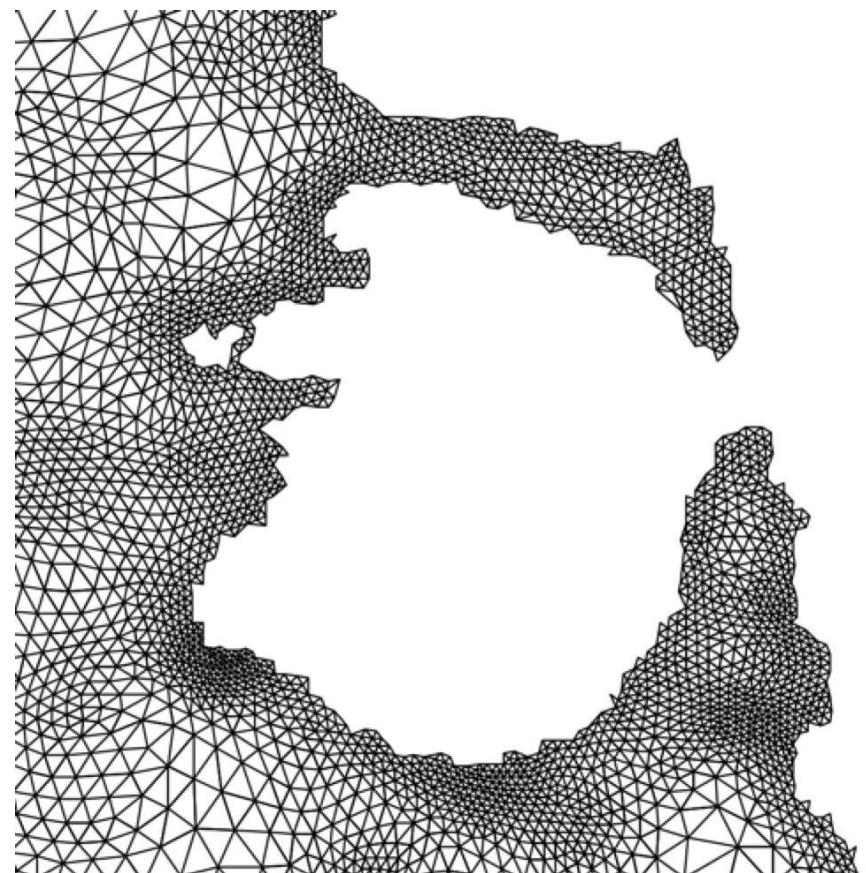


*Left: a CVDT of the North Atlantic; Right: a zoomed in portion of that CVDT; grid refinement is based on both distance from the boundary and measured kinetic energy distribution*



*Measured ice thickness of Greenland; note that the boundary is quite complicated*

- The grid should be refined
  - where the ice thickness is relatively small (light blue areas)
  - to resolve the boundary



*Left: a CVDT of the Greenland ice sheet. Right: a zoomed in portion of that CVDT; grid refinement is based on both distance from the boundary and measured ice thickness*

## Adaptive mesh generation

- One starts with the approximate solution of a PDE on a coarse grid
- A posteriori error estimators are determined based on the computed approximate solution of the PDE on the chosen grid
  - two types of error estimators are developed
    - one is based on  $L^2$  norms
    - the other is based on  $H^1$  norms
- The a posteriori error estimator is used to determine a CVT density function
  - the CVT density function is relative large in triangles where the error estimator is large
  - it is important to keep in mind that the relation between the density function and the local spacing between points is, in  $d$  dimensions,

$$\rho \longleftrightarrow \frac{1}{h^{(d+2)}}$$

- Using the CVT density function, the points are moved to generate a CVT of the region
- Starting with the latest grid, the process is repeated until the error indicator in every triangle is smaller than some prescribed tolerance
- Grid refinement and coarsening is then automatically effected
  - points will move away from regions where the CVT density is relatively small
  - points will aggregate in regions where the CVT density if relatively large
- The final grid obtained will be an optimal grid in the sense that the CVT energy is minimized

- We present results for a finite volume discretization of the problem

$$\begin{cases} -\nabla \cdot (a \nabla u) + bu = f & \text{in } \Omega \\ u = g & \text{on } \partial\Omega \end{cases}$$

## Smooth solution with large gradients

- Set

$$\Omega = [-1, 1] \times [-1, 1]$$

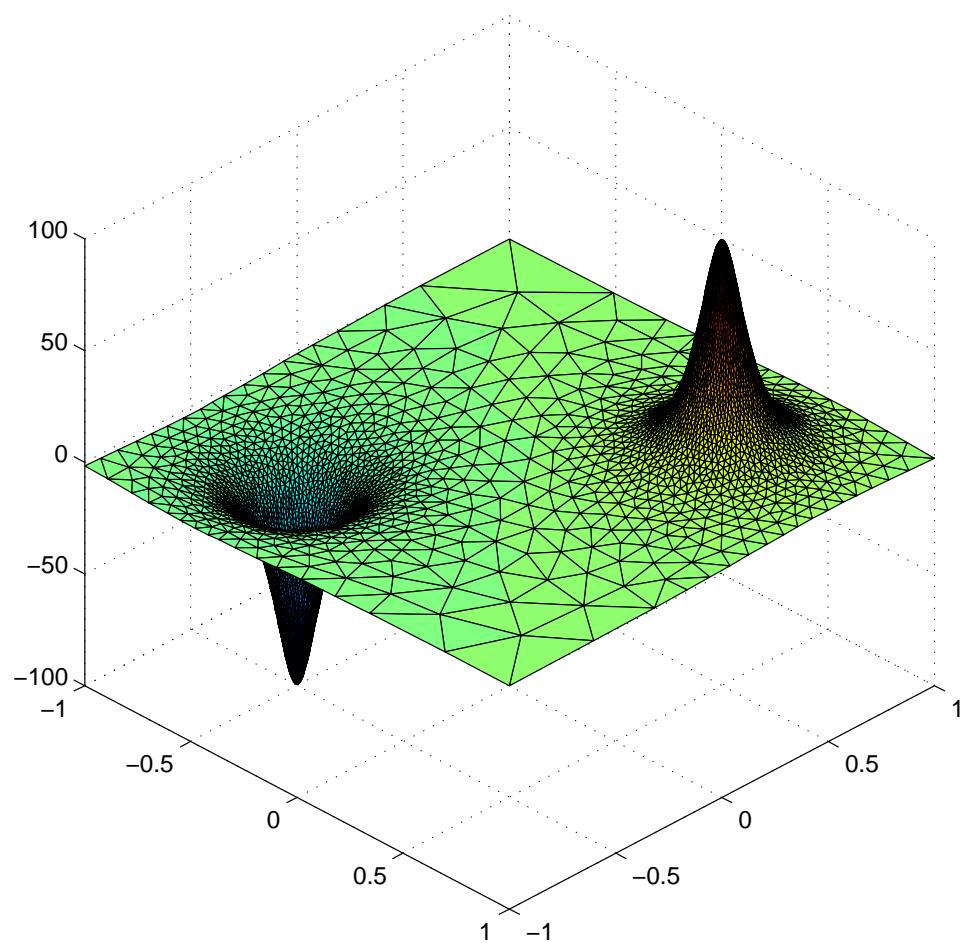
$$a(x, y) = 10.0 \cos(y)$$

$$b(x, y) = x^2 + y^2$$

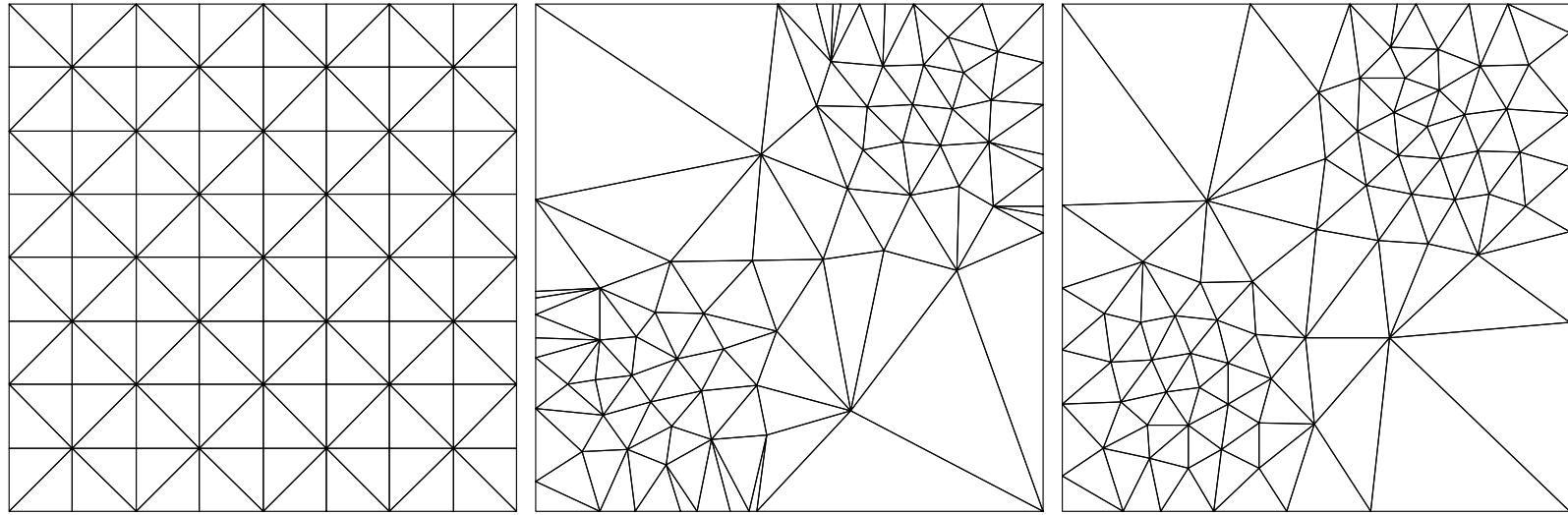
- Choose the exact solution

$$u(x, y) = \frac{1.0}{(x - 0.5)^2 + (y - 0.5)^2 + 0.01} + \frac{1.0}{(x + 0.5)^2 + (y + 0.5)^2 + 0.01}$$

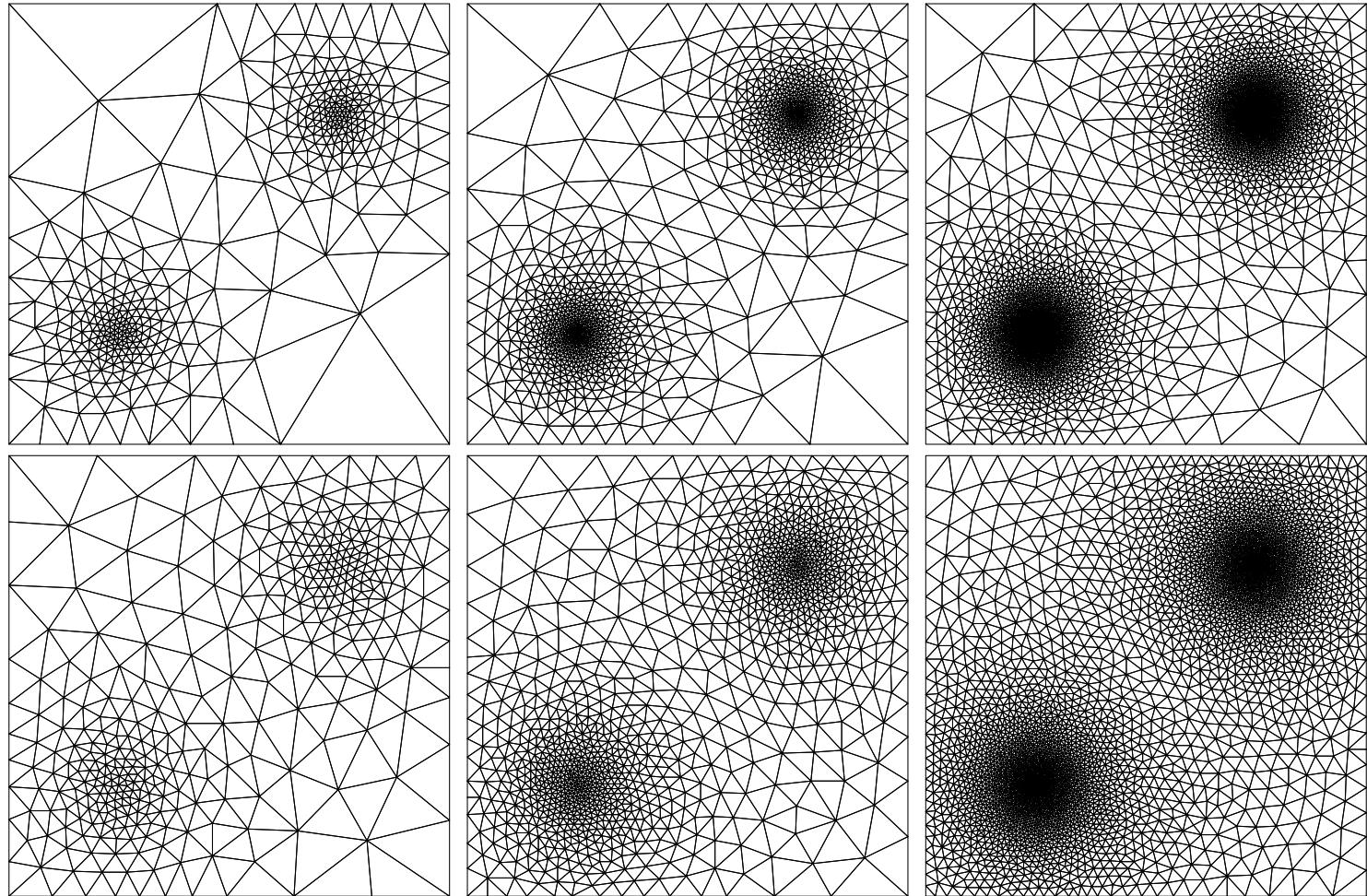
- Determine  $f$  and  $g$  from  $u$  so that the PDE problem is satisfied
- The solution is smooth, i.e.,  $u \in C^2(\Omega)$ , achieves its maximum value near the two points  $(0.5, 0.5)$  and  $(-0.5, -0.5)$ , and decays very quickly away from its maxima



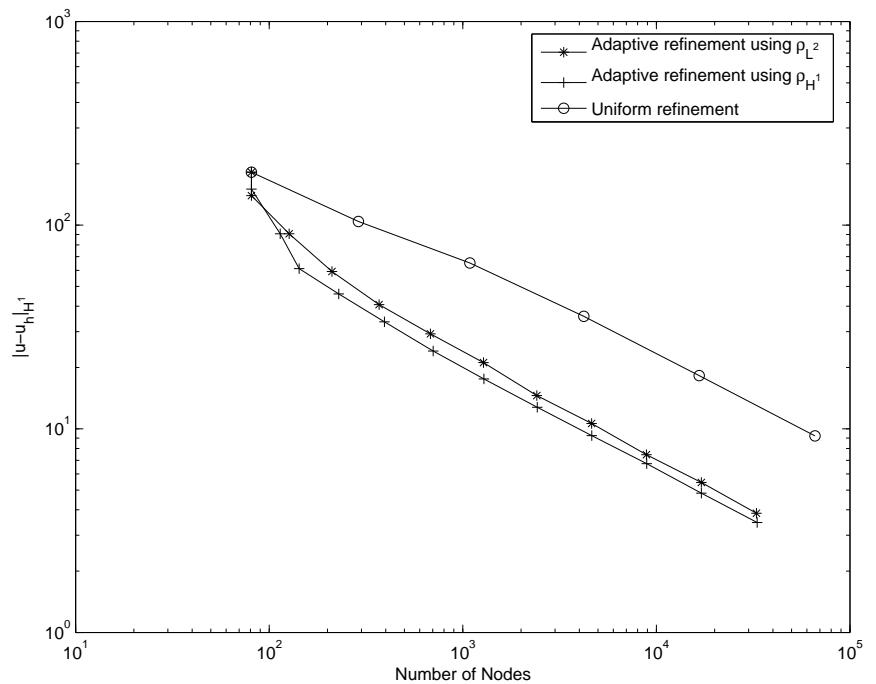
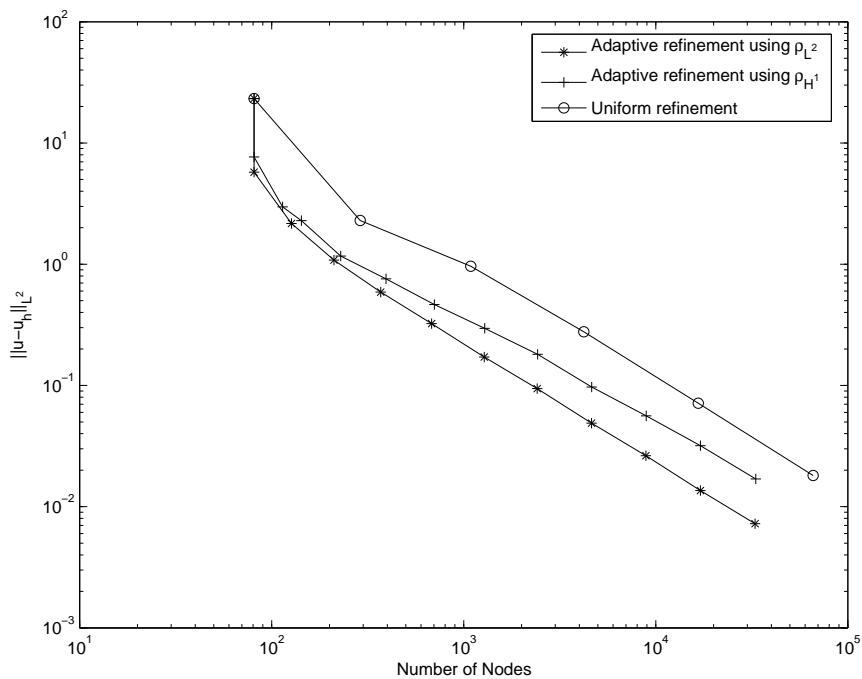
Solution



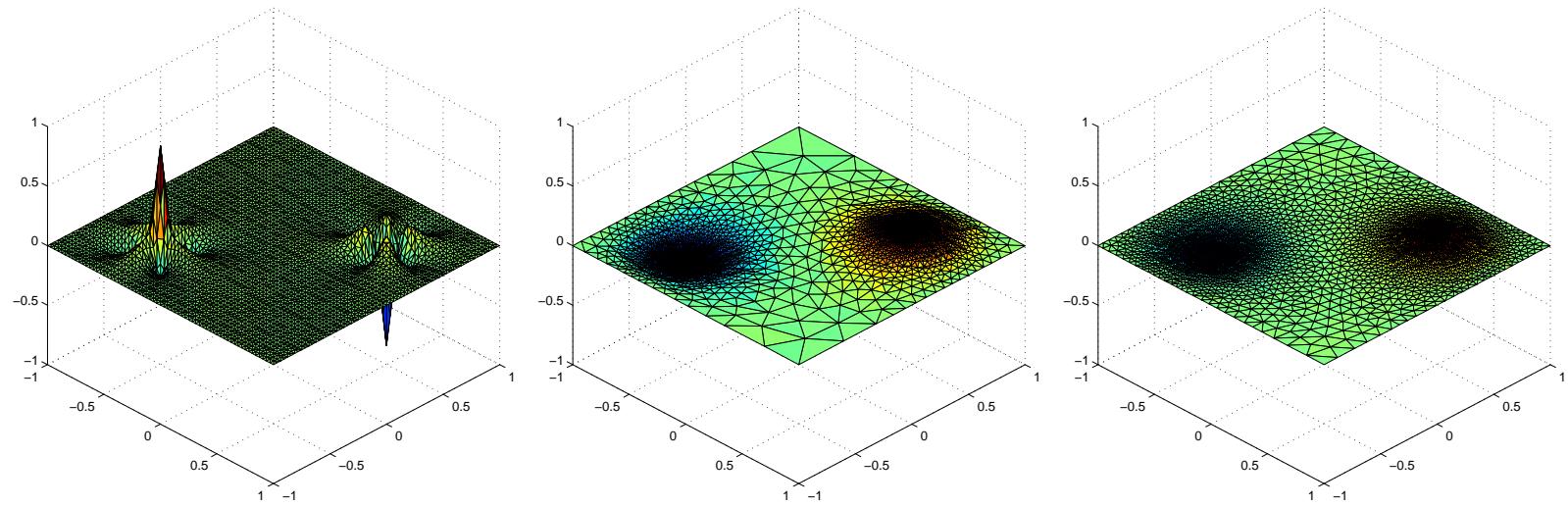
*Initial meshes; left: a uniform Cartesian grid with 81 nodes; middle and right: the corresponding CVDT meshes with the same number of nodes generated using the density functions related to  $H^1$  and  $L^2$  error estimators, respectively.*



*Repeatedly refined adaptive meshes at some levels generated by the CVDT-based adaptive method; top: 394, 1287, 4644 nodes using the  $H^1$  error estimator; bottom: 370, 1280, 4629 nodes using the  $L^2$  error estimator.*



*Error norms vs. number of nodes for different refinement strategies; left:  $L^2$  error; right:  $H^1$  error*



*Equilibration of error using different adaptive refinement strategies; left: error on a uniform mesh with 4225 nodes; middle: error on the mesh with 4644 nodes obtained using the CVDT-based adaptive method with the  $H^1$  error estimator; right: error on the mesh with 4629 nodes obtained using the CVDT-based adaptive method with the  $L^2$  error estimator.*

## Problem with a geometric singularity

- Set

$\Omega = \text{the L-shaped domain } [-1, 0] \times [0, 1] \cup [0, 1] \times [-1, 1]$   
with a re-entrant corner at the point  $(0, 0)$

$$a(x, y) = 1$$

$$b(x, y) = 0$$

- Choose the exact solution

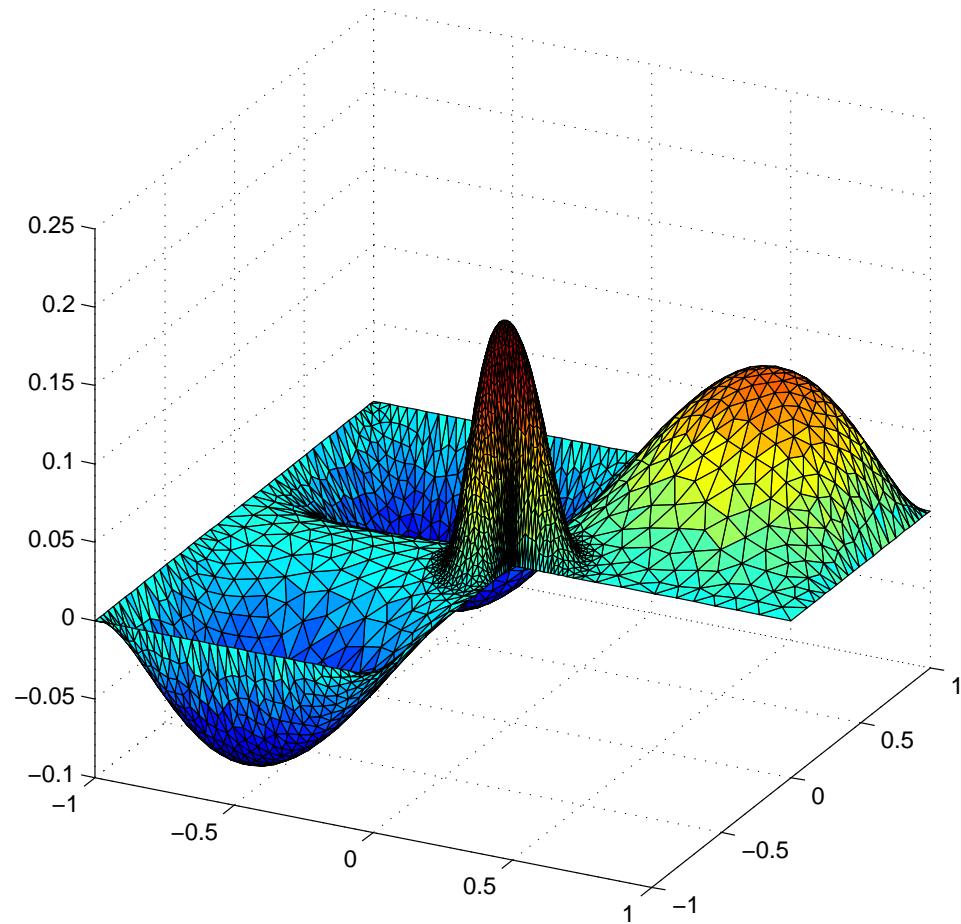
$$u(r, \theta) = s \left( \frac{r - \delta_1}{\delta_2 - \delta_1} \right) r^{2/3} \sin \left( \frac{2}{3} \theta \right) + w(r \cos \theta, r \sin \theta)$$

with  $\delta_1 = 0.02$  and  $\delta_2 = 0.25$ , where  $s$  is the cut-off function

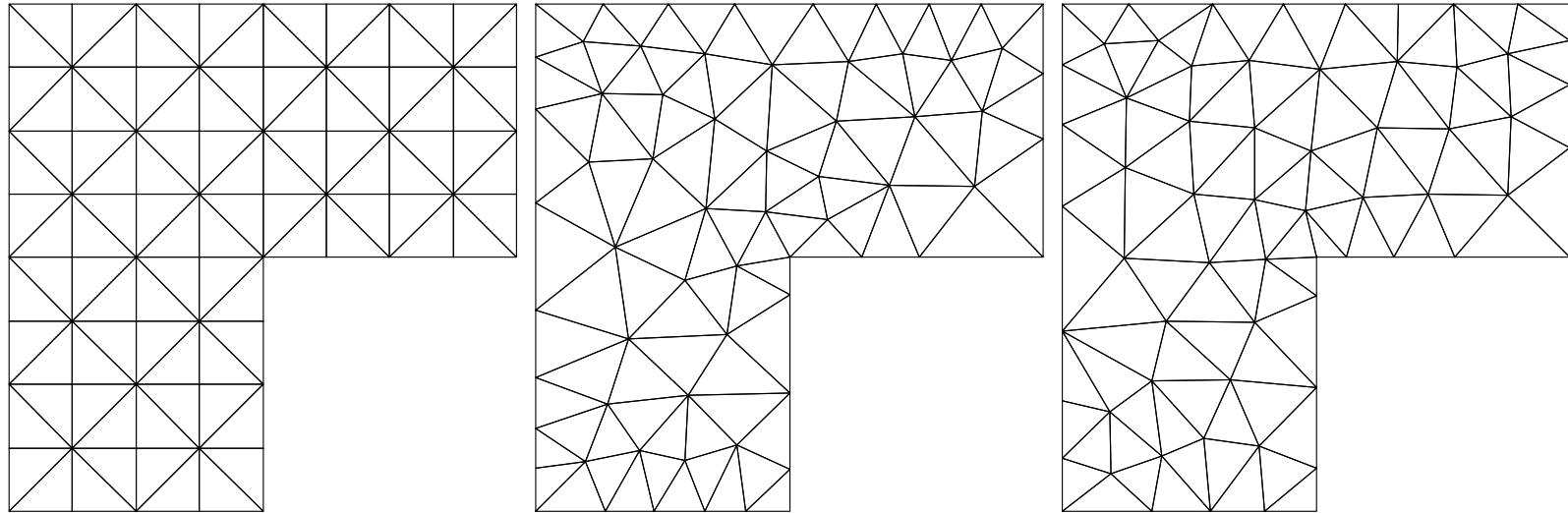
$$s(t) = \begin{cases} 1, & t < 0, \\ -6t^5 + 15t^4 - 10t^3 + 1, & 0 \leq t \leq 1, \\ 0, & t > 1, \end{cases}$$

and  $w(x, y) = (x - x^3)(y^2 - y^4)$ ; note that  $w$  is a smooth function with  $w|_{\partial\Omega} = 0$

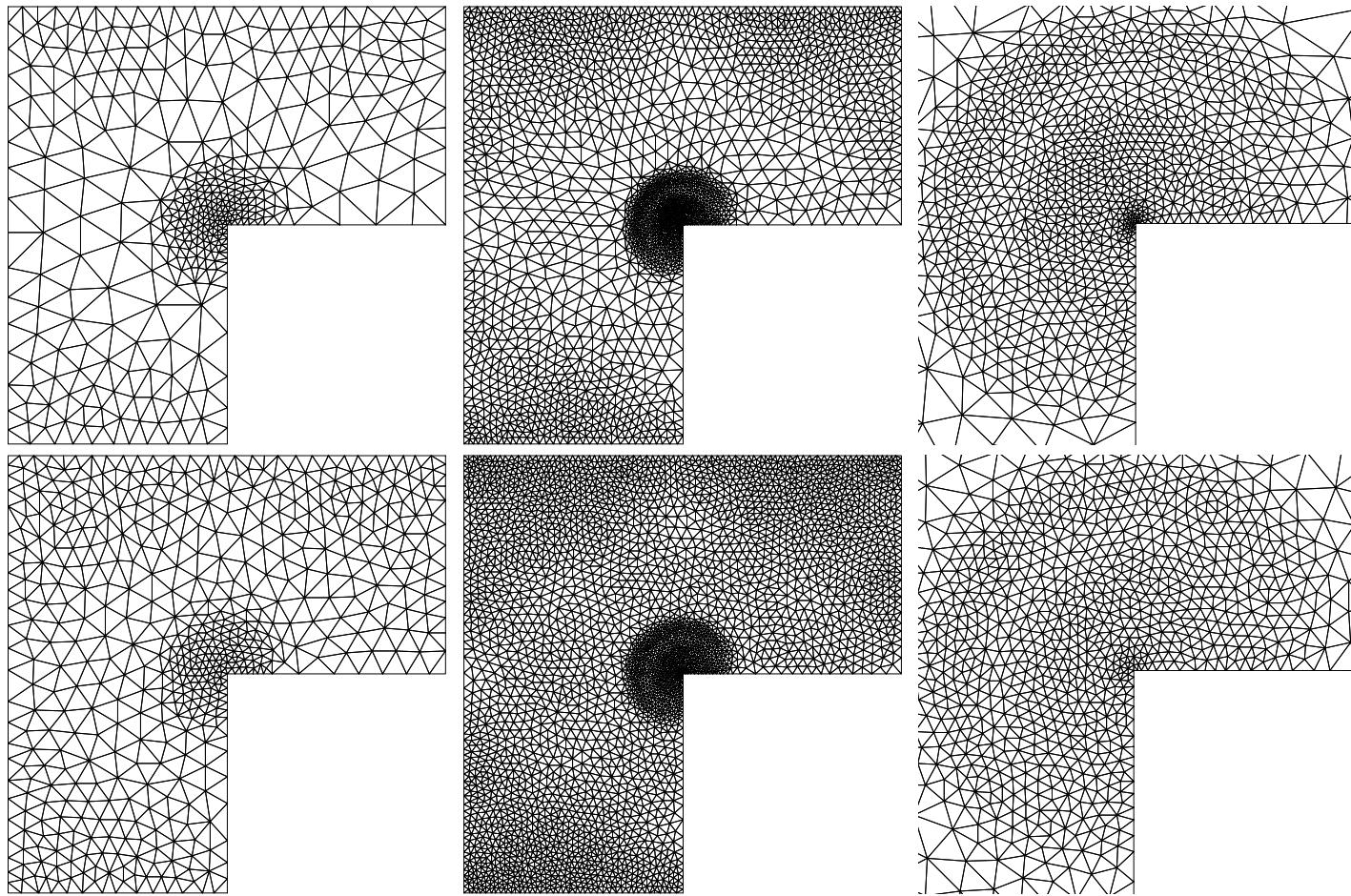
- Determine  $f$  and  $g$  from  $u$  so that the PDE problem is satisfied
- The solution  $u$  mimics the singularity in the solution at the re-entrant corner
  - $u$  merely belongs to  $H^{\frac{5}{3}-\epsilon}(\Omega)$  for any  $\epsilon > 0$



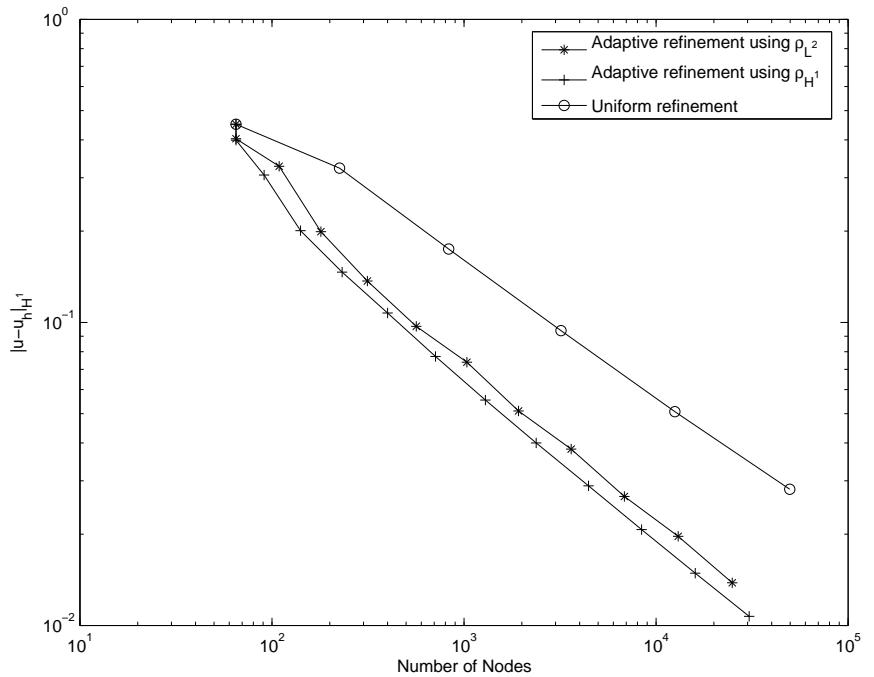
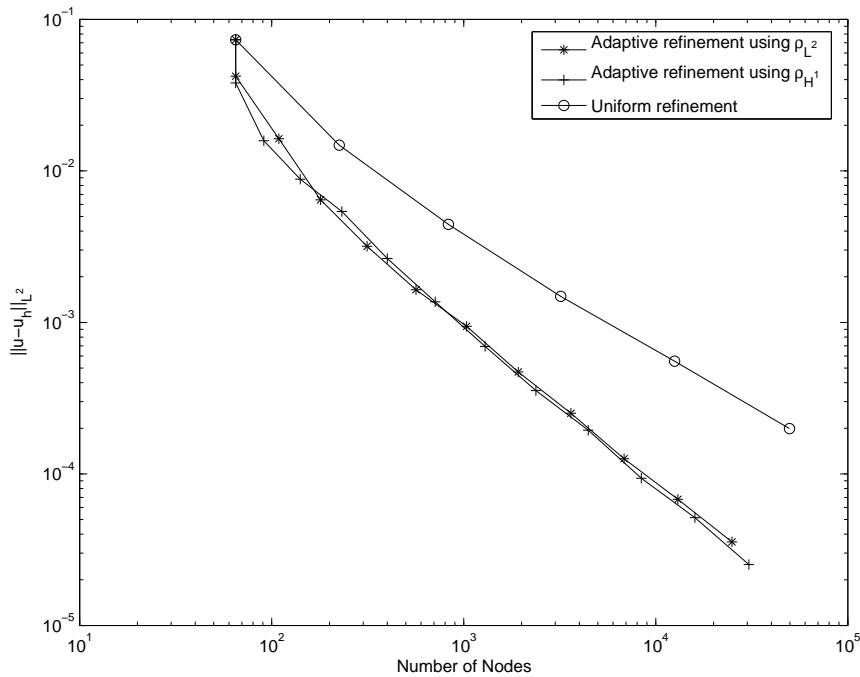
Solution



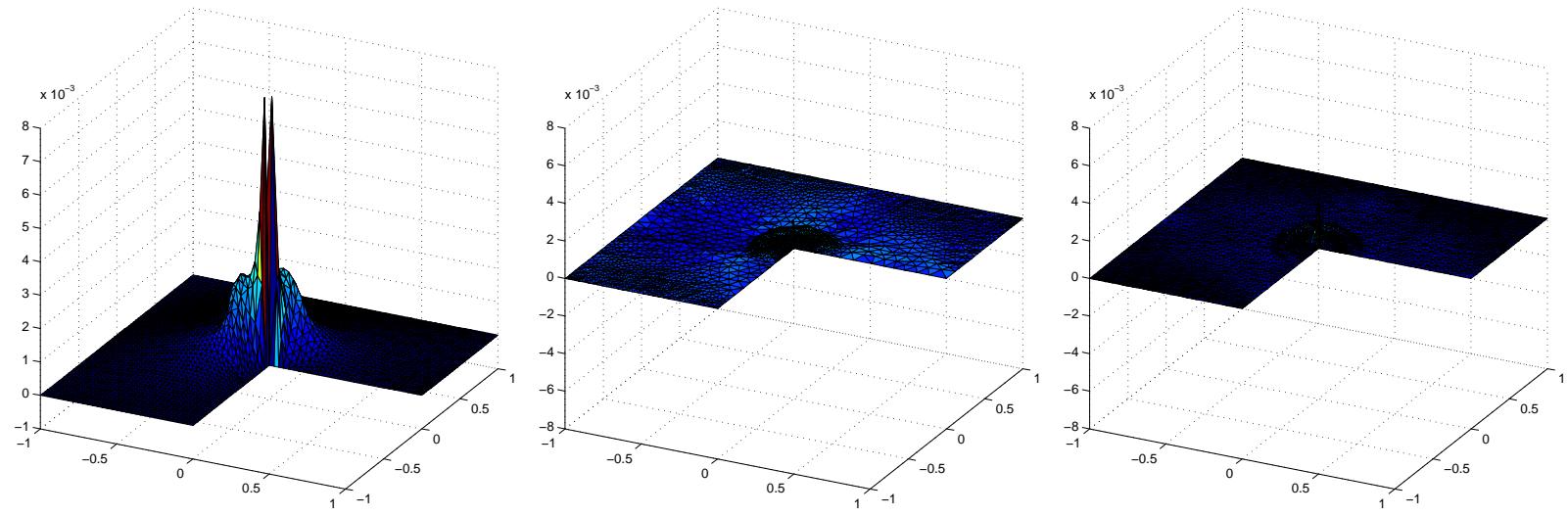
Initial meshes; left: a uniform Cartesian grid with 65 nodes; middle and right: the corresponding CVDT meshes with the same number of nodes generated using the density functions related to  $H^1$  and  $L^2$  error estimators, respectively.



*Refined adaptive meshes at some levels generated by the CVDT-based adaptive method; top: 400 and 2379 nodes and a zoom-in of the latter case near the re-entrant corner using the  $H^1$  error estimator; bottom: 565 and 3617 nodes and a zoom-in of the latter case near the re-entrant corner using the  $L^2$  error estimator.*



*Error norms vs. number of nodes for different refinement strategies; left:  $L^2$  error; right:  $H^1$  error*

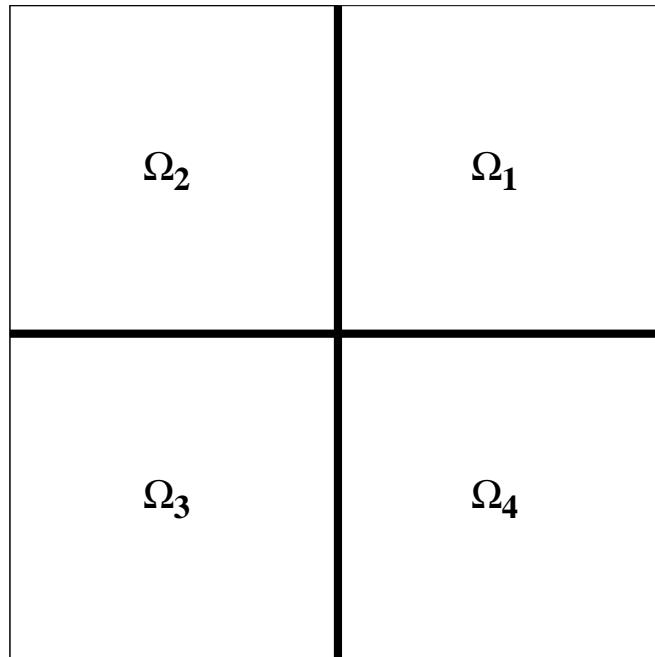


*Equilibration of error using different adaptive refinement strategies; left: error on a uniform mesh with 3201 nodes; middle: error on the mesh with 2379 nodes obtained using the CVDT-based adaptive method with the  $H^1$  error estimator; right: error on the mesh with 3617 nodes obtained using the CVDT-based adaptive method with the  $L^2$  error estimator.*

## Problem with interface singularities

- Set  $\Omega = [-1, 1] \times [-1, 1]$        $b(x, y) = 1$       and

$$a(x, y) = \begin{cases} 1 & (x, y) \in \Omega_1 \cup \Omega_3 \\ 5 & (x, y) \in \Omega_2 \cup \Omega_4 \end{cases}$$



note that  $a$  is discontinuous across  $x = 0$  and  $y = 0$

- Choose the exact solution

$$u(r, \theta) = r^\alpha (p_i \cos(\alpha\theta) + q_i \sin(\alpha\theta)) \quad \text{in } \Omega_i$$

with  $\alpha \approx 0.53544094560$  and

$$\{(p_i, q_i)\}_{i=1}^4 \approx \{(14.535673, -0.839562), (0.429608, 1.621108), (-13.043759, 6.469236), (-0.478922, -0.693383)\}$$

- Then, for  $i = 1, 2, 3, 4$ ,  $u$  satisfies

$$\nabla \cdot (a \nabla u) = 0 \quad \text{in } \Omega_i$$

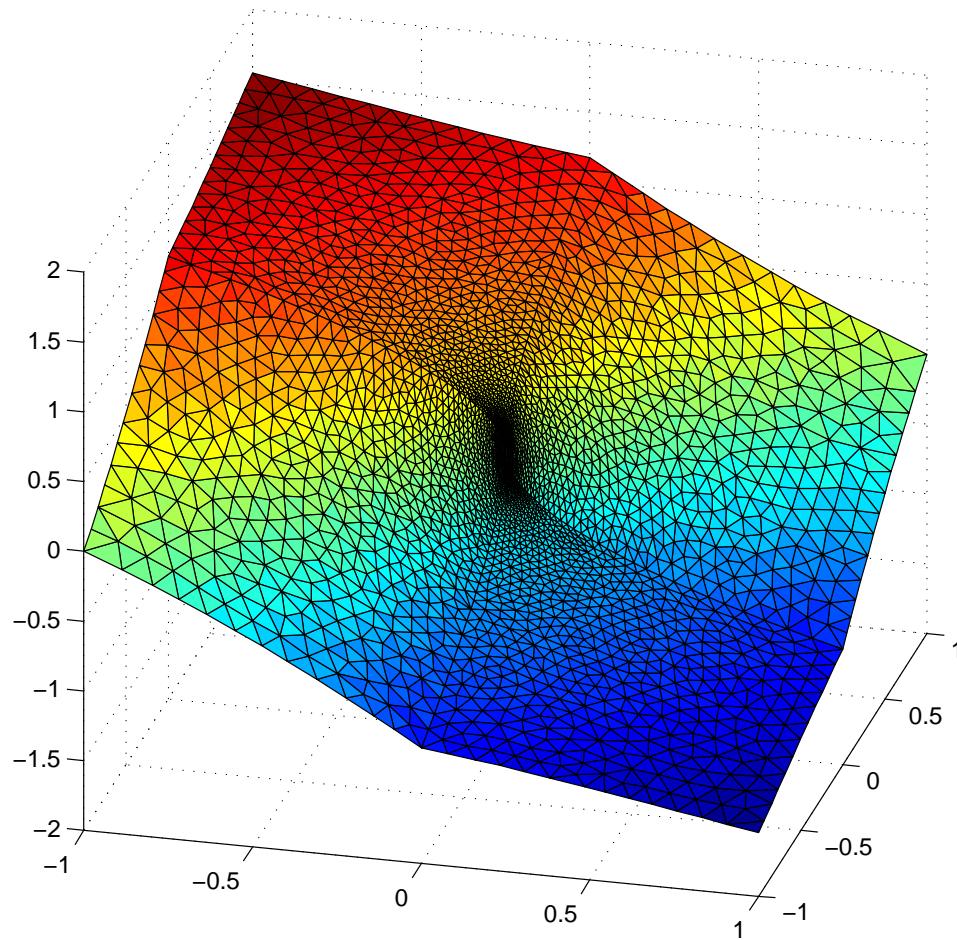
and the interface conditions

$$\lim_{\theta \rightarrow (i\pi/2)^+} u(r, \theta) = \lim_{\theta \rightarrow (i\pi/2)^-} u(r, \theta)$$

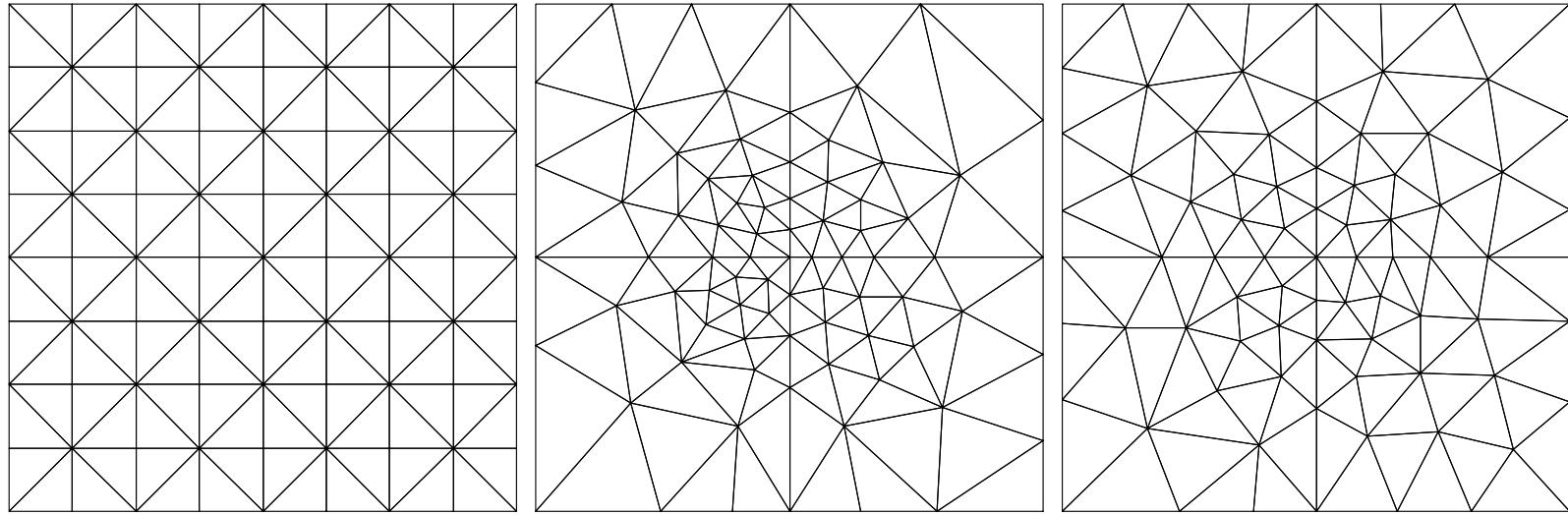
and

$$\lim_{\theta \rightarrow (i\pi/2)^+} a \frac{\partial u(r, \theta)}{\partial \theta} = \lim_{\theta \rightarrow (i\pi/2)^-} a \frac{\partial u(r, \theta)}{\partial \theta}$$

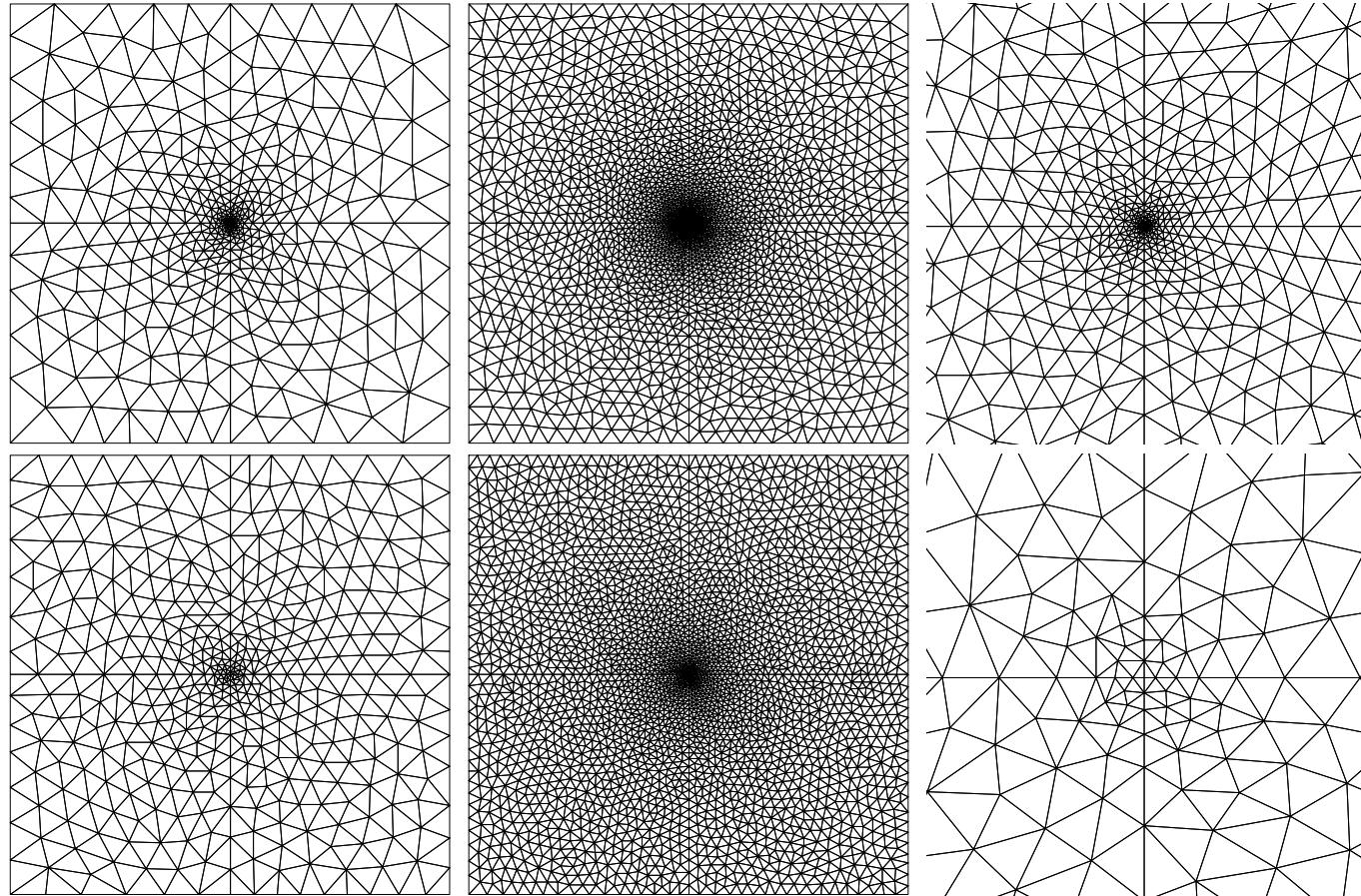
– note that  $u \in H^{1+\alpha-\epsilon}(\Omega)$  for any  $\epsilon > 0$  and has a strong singularity at the origin



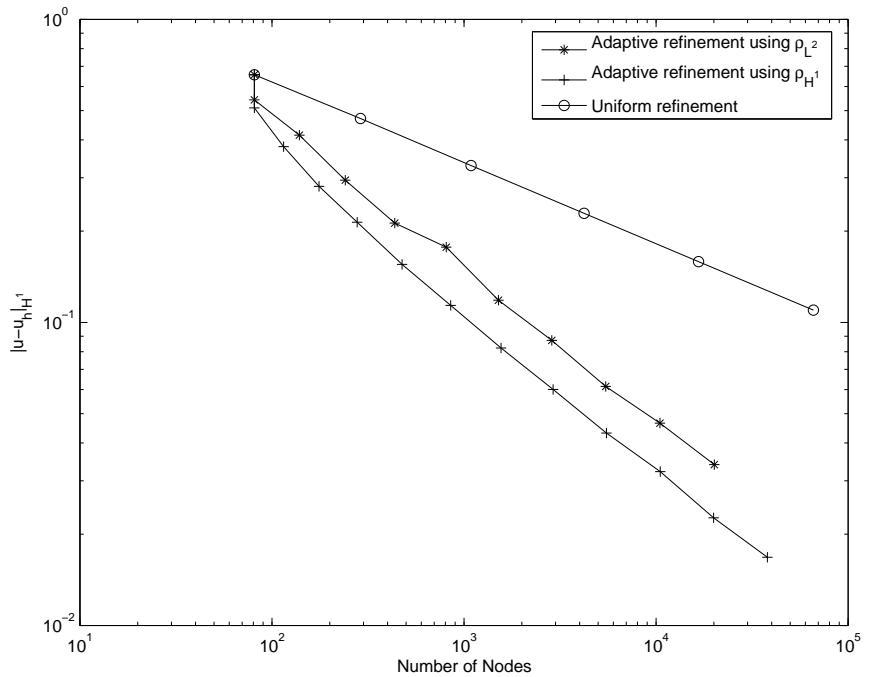
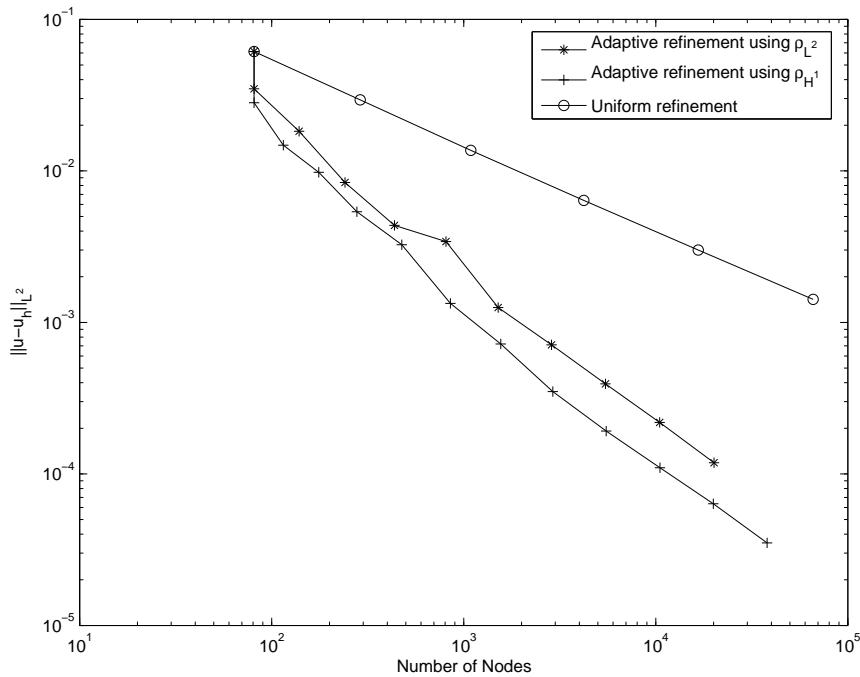
Solution



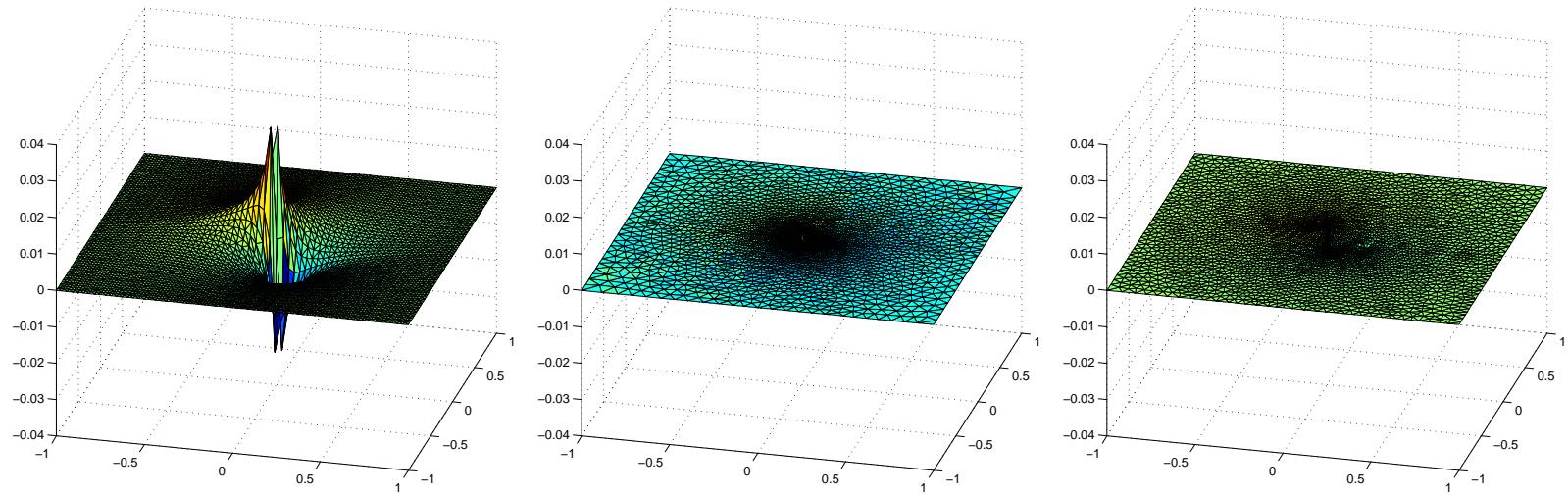
*Initial meshes; left: a uniform Cartesian grid with 81 nodes; middle and right: the corresponding CVDT meshes with the same number of nodes generated using the density functions related to  $H^1$  and  $L^2$  error estimators, respectively*



*Refined adaptive meshes at some levels generated by the CVDT-based adaptive method; top: 476 and 2910 nodes and a zoom-in of the latter case near the singular point using the  $H^1$  error estimator; bottom: 436 and 2868 nodes and a zoom-in of the latter case near the singular point using the  $L^2$  error estimator.*



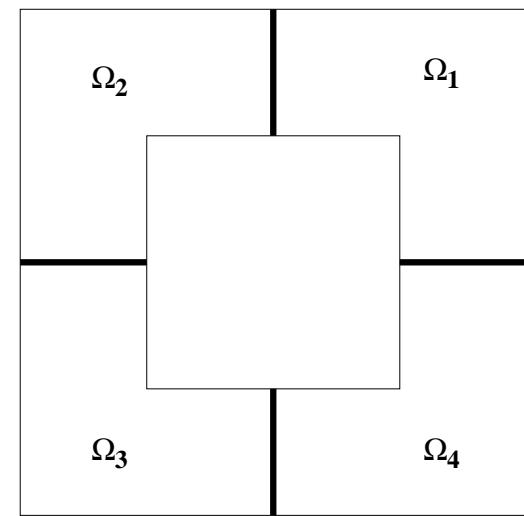
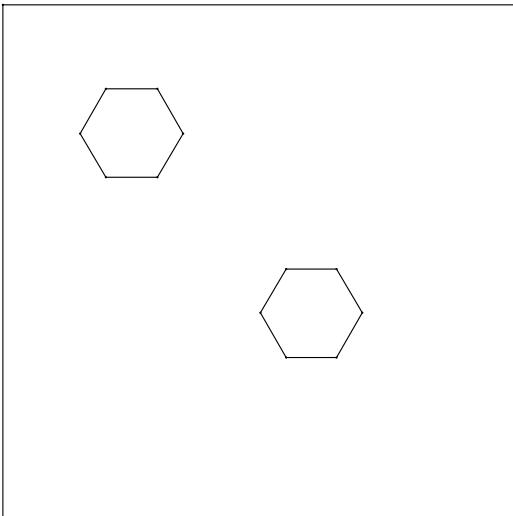
*Error norms vs. number of nodes for different refinement strategies; left:  $L^2$  error; right:  $H^1$  error*



*Equilibration of error using different adaptive refinement strategies; left: error on a uniform mesh with 4225 nodes; middle: error on the mesh with 2910 nodes obtained using the CVDT-based adaptive method with the  $H^1$  error estimator; right: error on the mesh with 2868 nodes obtained using the CVDT-based adaptive method with the  $L^2$  error estimator.*

## Problems with more complicated geometries

- The last two examples are posed on the domains



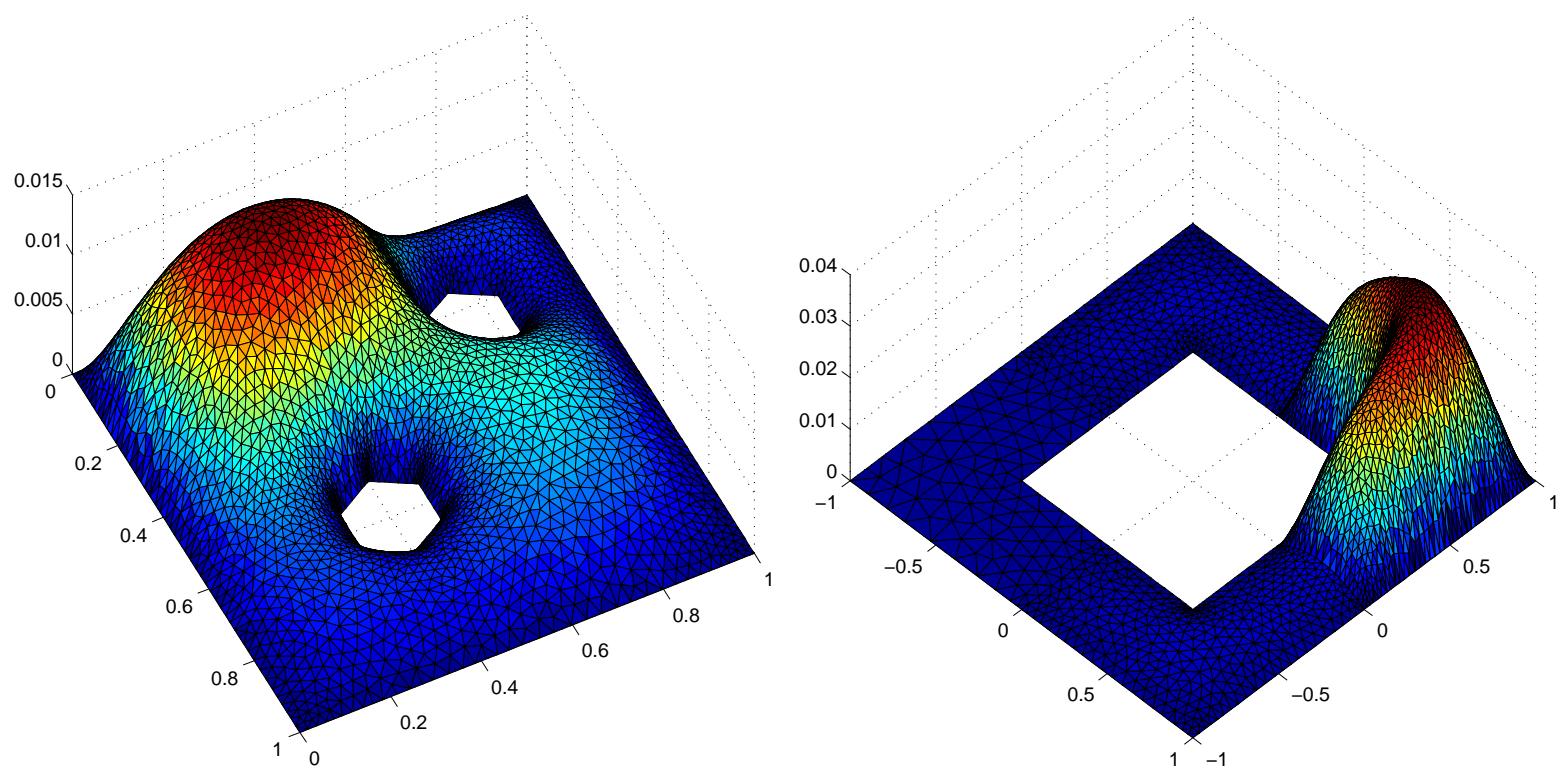
- We set  $f(x, y) = 1$  and  $g(x, y) = 0$  for both examples and

$$a(x, y) = 1 + 10x^2 + y^2$$

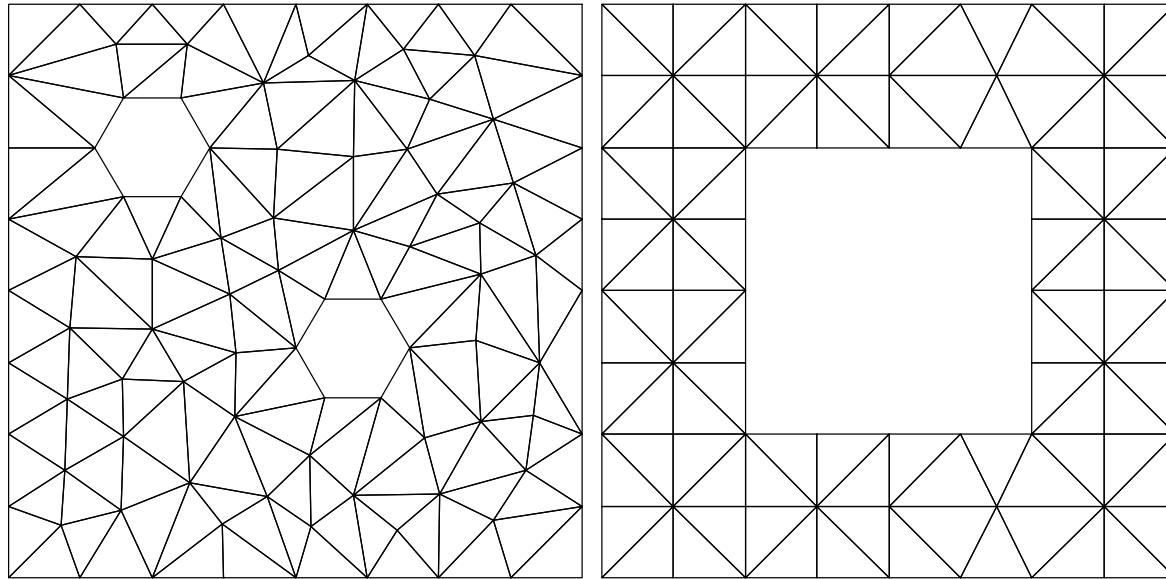
$$b(x, y) = 1 + x^2$$

$$a(x, y) = \begin{cases} 1 & \text{in } \Omega_1 \\ 20 & \text{in } \Omega_2 \\ 400 & \text{in } \Omega_3 \\ 20 & \text{in } \Omega_4 \end{cases}$$
$$b(x, y) = 0$$

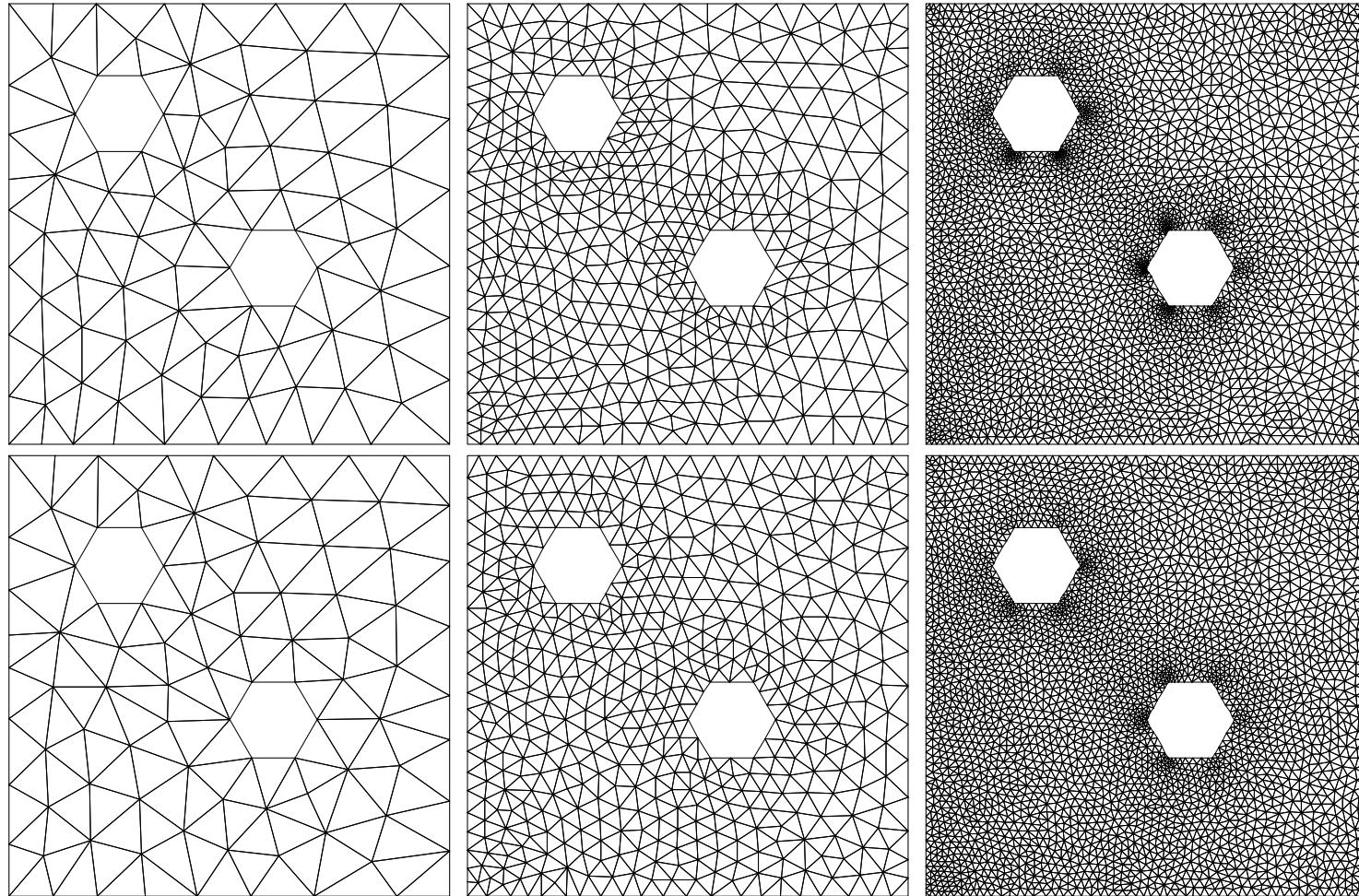
- Analytical solutions are not known; however
  - for the two hexagonal whole problem, we have that  $u$  belongs to  $H^{\frac{7}{4}-\epsilon}(\Omega)$  for any  $\epsilon > 0$ , but not to  $H^2(\Omega)$
  - for the problem with a square hole and interfaces,  $u$  only belongs to  $H^1(\Omega)$ , but not to  $H^2(\Omega)$ ; however,  $u|_{\Omega_i} \in H^2(\Omega_i)$  for  $i = 1, 2, 3, 4$



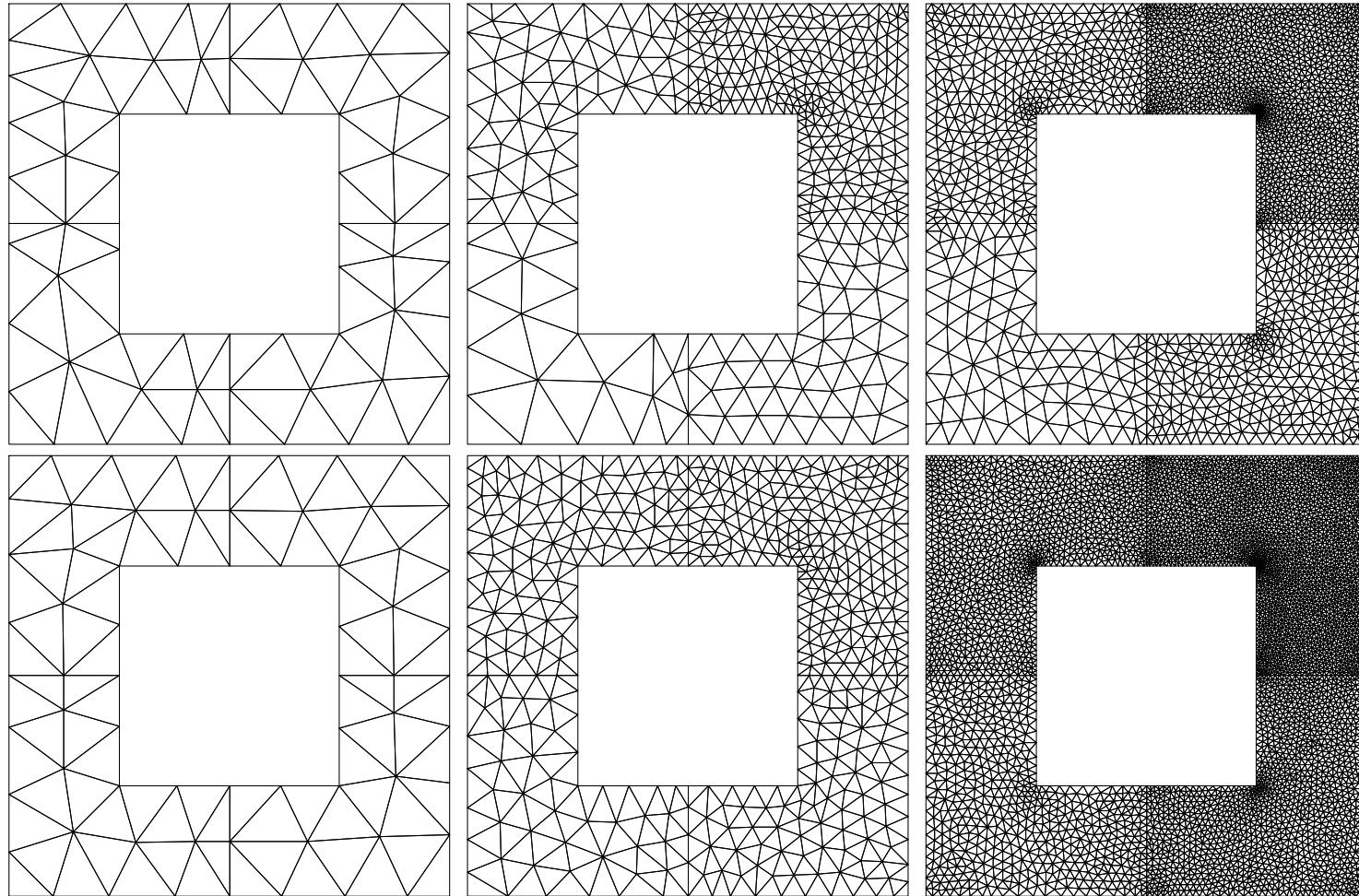
Computed solutions



*Initial meshes for the two examples*



*Refined adaptive meshes at some levels generated by the CVDT-based adaptive method; top: 94, 507, and 3096 nodes using the  $H^1$  error estimator; bottom: 94, 524, and 3373 nodes using the  $L^2$  error estimator*



*Refined adaptive meshes at some levels generated by the CVDT-based adaptive method; top: 70, 408, and 2474 nodes using the  $H^1$  error estimator; bottom: 70, 492, and 3020 nodes using the  $L^2$  error estimator*

## ANISOTROPIC CVT MESH GENERATION

- Anisotropic CVT's are defined with respect to a metric tensor  $\mathbb{M}$ 
  - distances are measured using the norm  $\|\mathbf{x}\|_M^2 = \mathbf{x}^\top \mathbb{M} \mathbf{x}$
- We consider metric tensors that are defined with respect to a function  $u(\mathbf{x})$ 
  - $u$  could be the a given function
  - in the grid generation case,  $u$  is an approximate solution of a partial differential equation
- Many definitions for metric tensors have been used for anisotropic grid generation
  - here, we will only discuss one particular metric tensor that is defined in W. Huang, Metric tensors for anisotropic mesh generation, *J. Comp. Phys.* **204** 2005, 633-665

- That metric tensor is given by

$$\mathbb{M}(\mathbf{x}) = \left( \frac{N\rho}{\sigma} \right)^{2/d} \left( \det \left( \mathbb{I} + \frac{1}{\alpha} \mathbb{H}(u) \right) \right)^{-1/d} \left( \mathbb{I} + \frac{1}{\alpha} \mathbb{H}(u) \right)$$

where

$d$  = space dimension

$N$  = number of triangles in the grid

$\mathbb{H}(u)$  = Hessian matrix for  $u(\mathbf{x})$

$$\sigma = \int_{\Omega} \rho d\Omega$$

$$\rho = \left( \det \left( \mathbb{I} + \frac{1}{\alpha} \mathbb{H}(u) \right) \right)^{1/3}$$

$$\alpha \text{ is defined implicitly by } \int_{\Omega} \rho d\Omega = 2^{4/3} |\Omega|$$

- for a finite element function, an approximation to the Hessian matrix can be determined from its nodal values

- So far, we have used metric tensors such as this one to both
  - make the grid anisotropic
  - determine local grid size, using the “size” of the metric tensor as an implicit CVT density function
- Although this seems to work well, we will also study adding the use of a posteriori error estimators to determine the local grid size

## Computational example

- From: V. John, A numerical study of a posteriori error estimators for convection-diffusion equations, *Comput. Meth. Appl. Mech. Engrg.* **190** 2000, 757-781.

$$-a\Delta u + \mathbf{v} \cdot \nabla u + bu = f \quad \text{in } \Omega$$

$$u = 0 \quad \text{on boundary of } \Omega$$

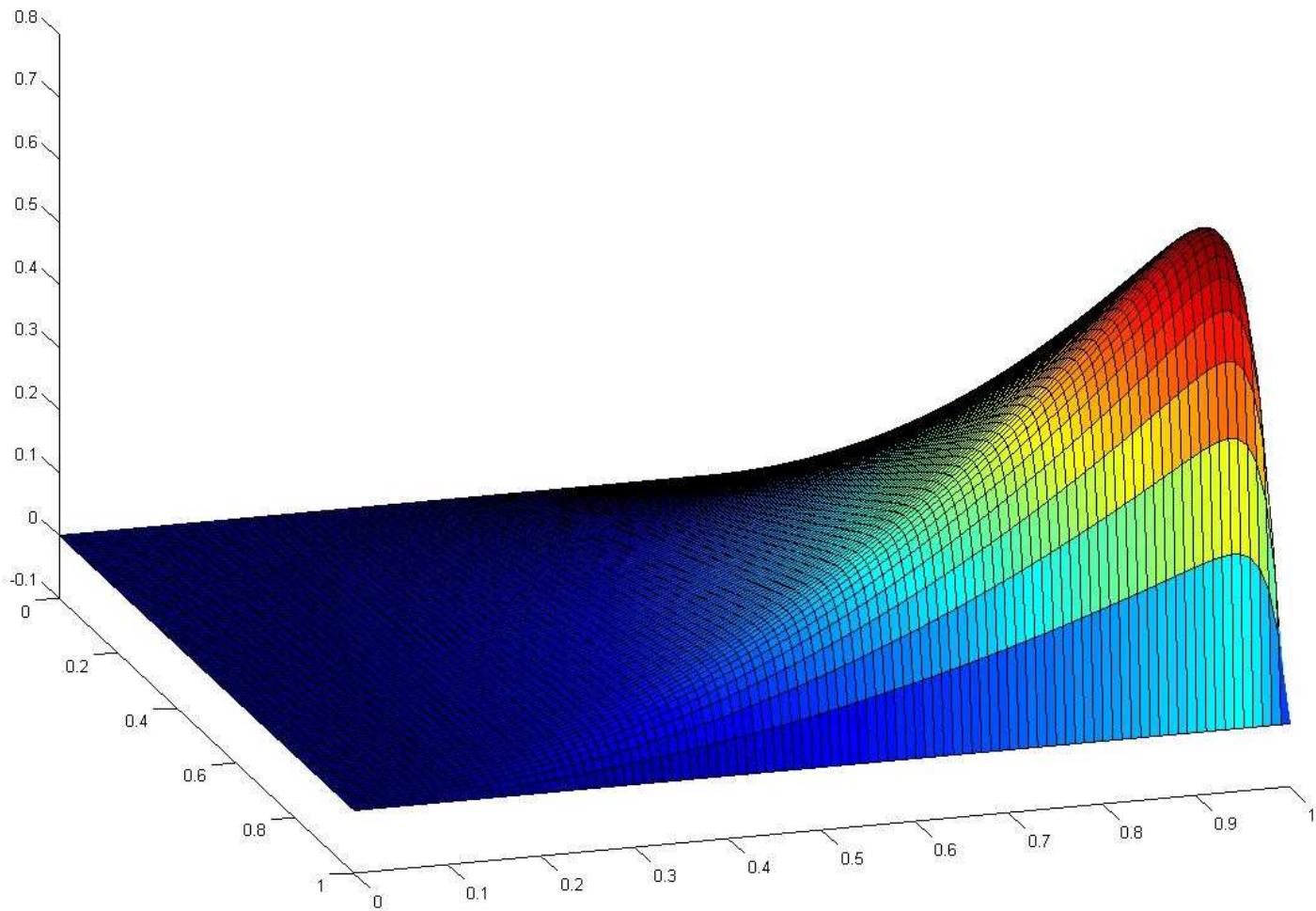
where

$$\Omega = (0, 1)^2 \quad a = 0.01 \quad \mathbf{v} = (2, 3) \quad b = 1$$

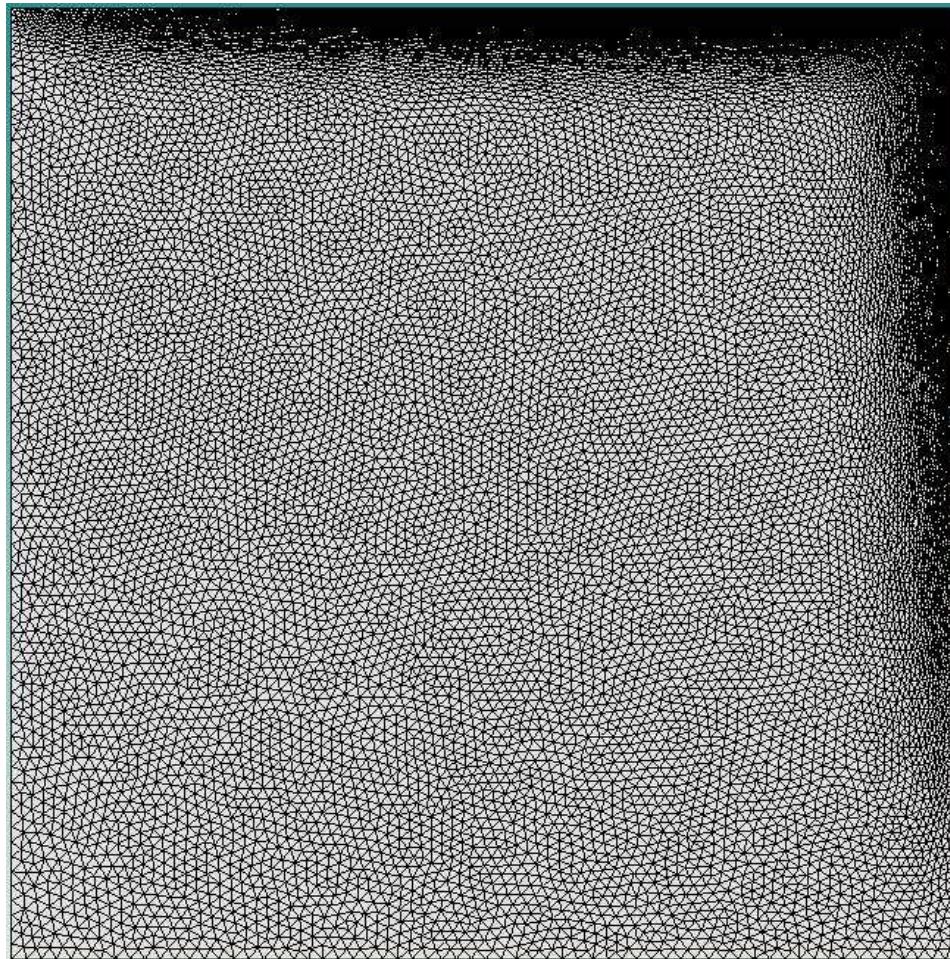
- $f$  is chosen so that the exact solution is given by

$$u = xy^2 - y^2 \exp\left(\frac{2(x-1)}{0.05}\right) - x \exp\left(\frac{3(y-1)}{0.05}\right) + \exp\left(\frac{2(x-1) + 3(y-1)}{0.05}\right)$$

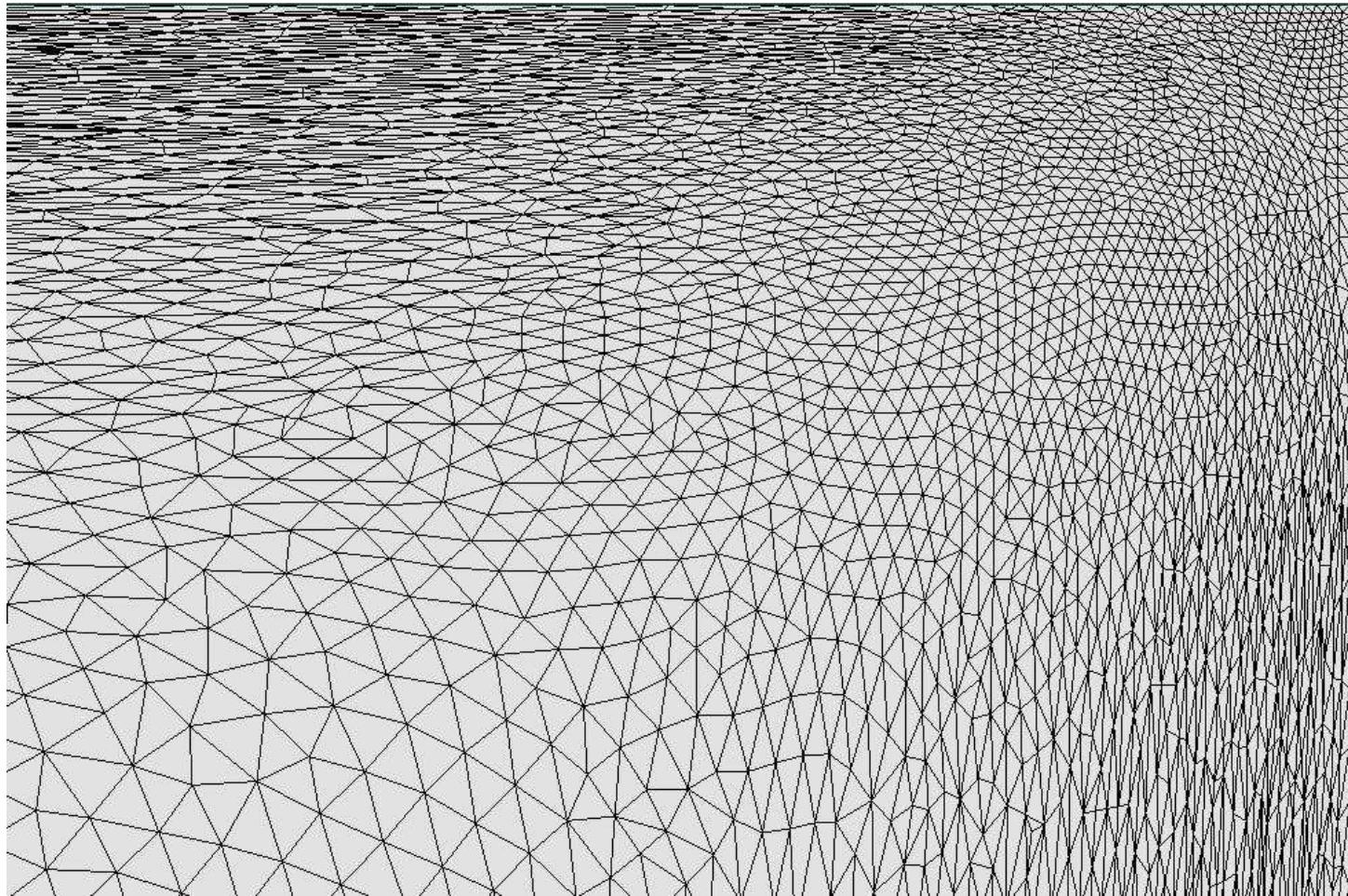
- The solution has boundary layers near  $x = 1$  and  $y = 1$ 
  - i.e., near the outflow boundaries where  $\mathbf{v} \cdot \mathbf{n} > 0$



*Exact solution with boundary layers near  $x = 1$  and  $y = 1$*



*Adaptively refined anisotropic CVT grid*



*Zoom in near  $(x, y) = (1, 1)$  of the adaptively refined anisotropic CVT grid*

refinement level	number of vertices	number of triangles	$L^\infty$ error	$L^2$ error	$H^1$ error
0	25	32	1.4541e+00	4.9449e-01	6.3923e+00
1	63	98	1.0291e+00	2.0689e-01	5.9529e+00
2	170	301	3.9171e-01	4.1308e-02	3.2622e+00
3	515	953	6.6131e-02	3.2532e-03	8.6938e-01
4	1524	2928	1.9862e-02	4.9886e-04	3.0390e-01
5	4817	9429	5.5310e-03	1.4333e-04	1.5924e-01
6	14299	28224	1.7945e-03	4.0194e-05	8.2427e-02

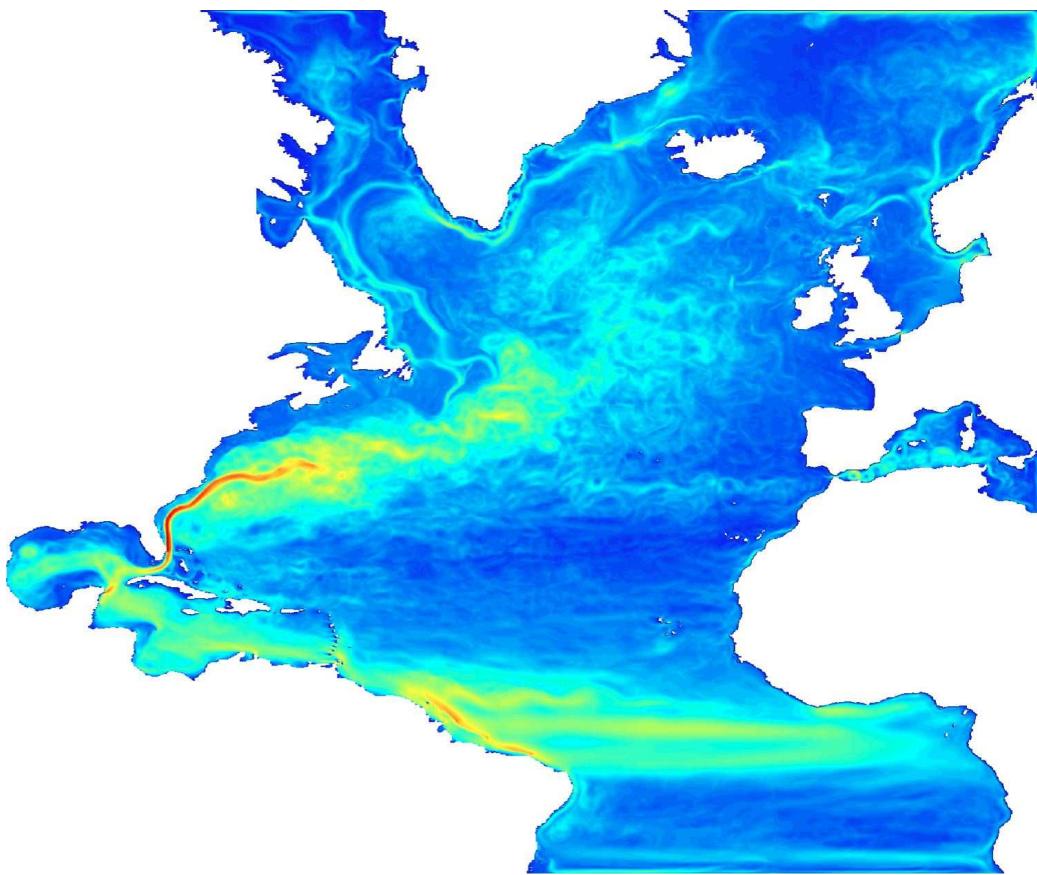
*Errors vs. level of refinement using adaptive anisotropic meshes*

level	$L^\infty$ error	$L^2$ error	$H^1$ error
1	0.7481	1.8855	0.1541
2	1.9461	3.2461	1.2118
3	3.2099	4.5859	2.3862
4	2.2173	3.4566	1.9376
5	2.2218	2.1675	1.1232
6	2.0691	2.3371	1.2104

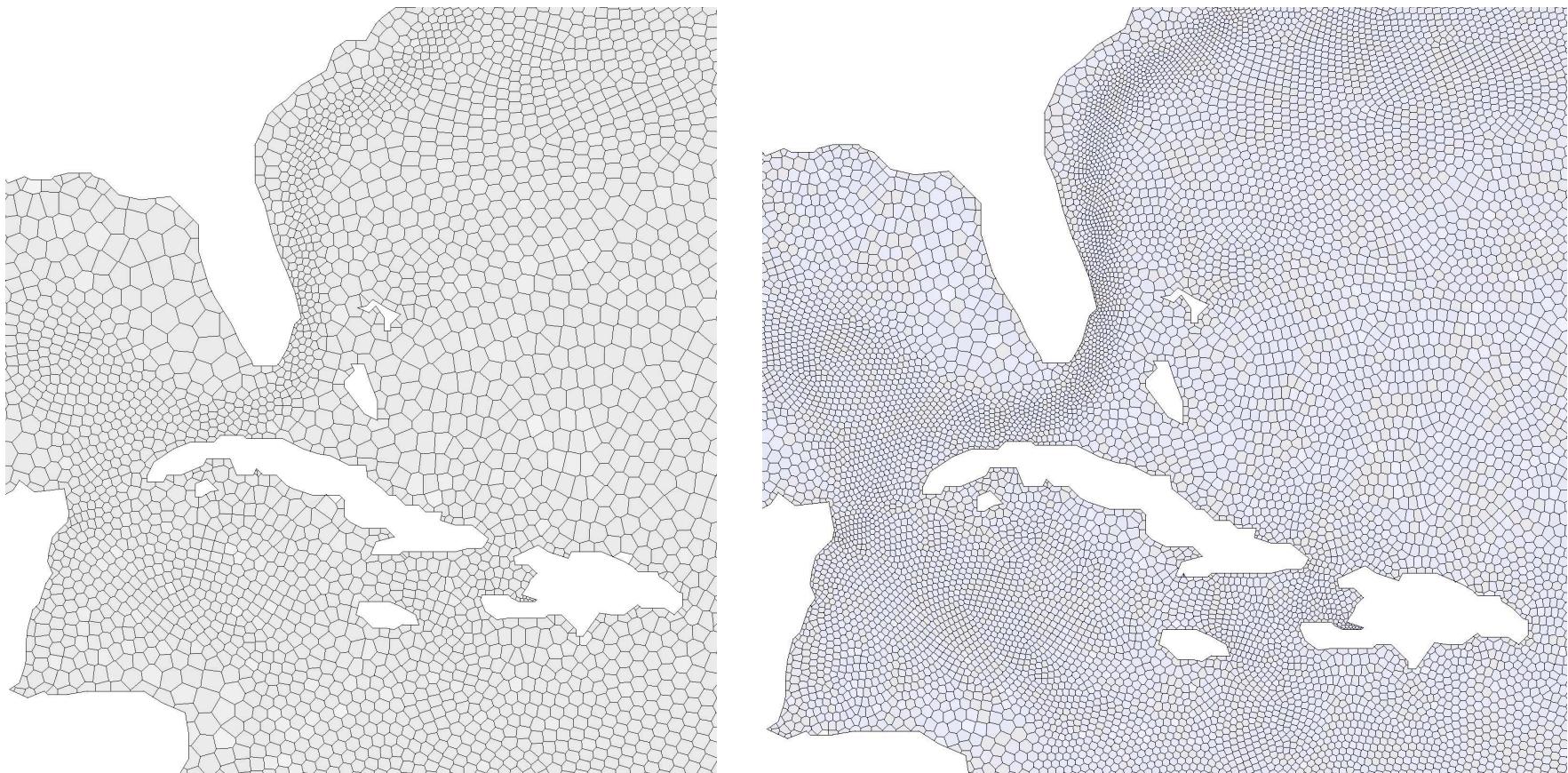
*Convergence rates vs. level of refinement*

## APPLICATION TO OCEAN GRIDS

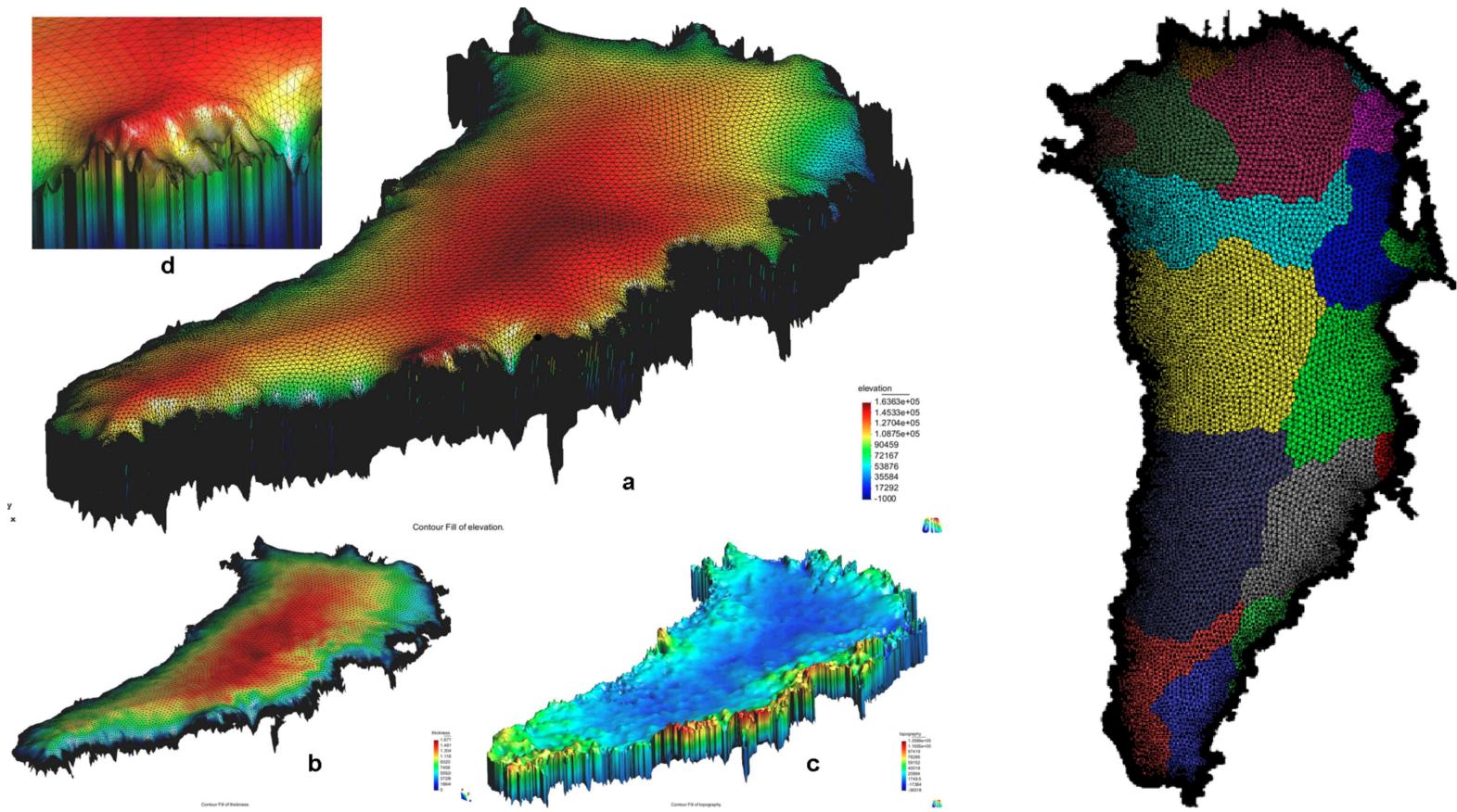
- CVT grids are been adopted as the grid generation technique for the next generation climate model being developed at NCAR and Los Alamos
  - new ocean model developed by LANL
  - new atmospheric model developed at NCAR
- It appears likely that CVT grids will also be used for the next generation land ice model, e.g., for Greenland and Antarctica



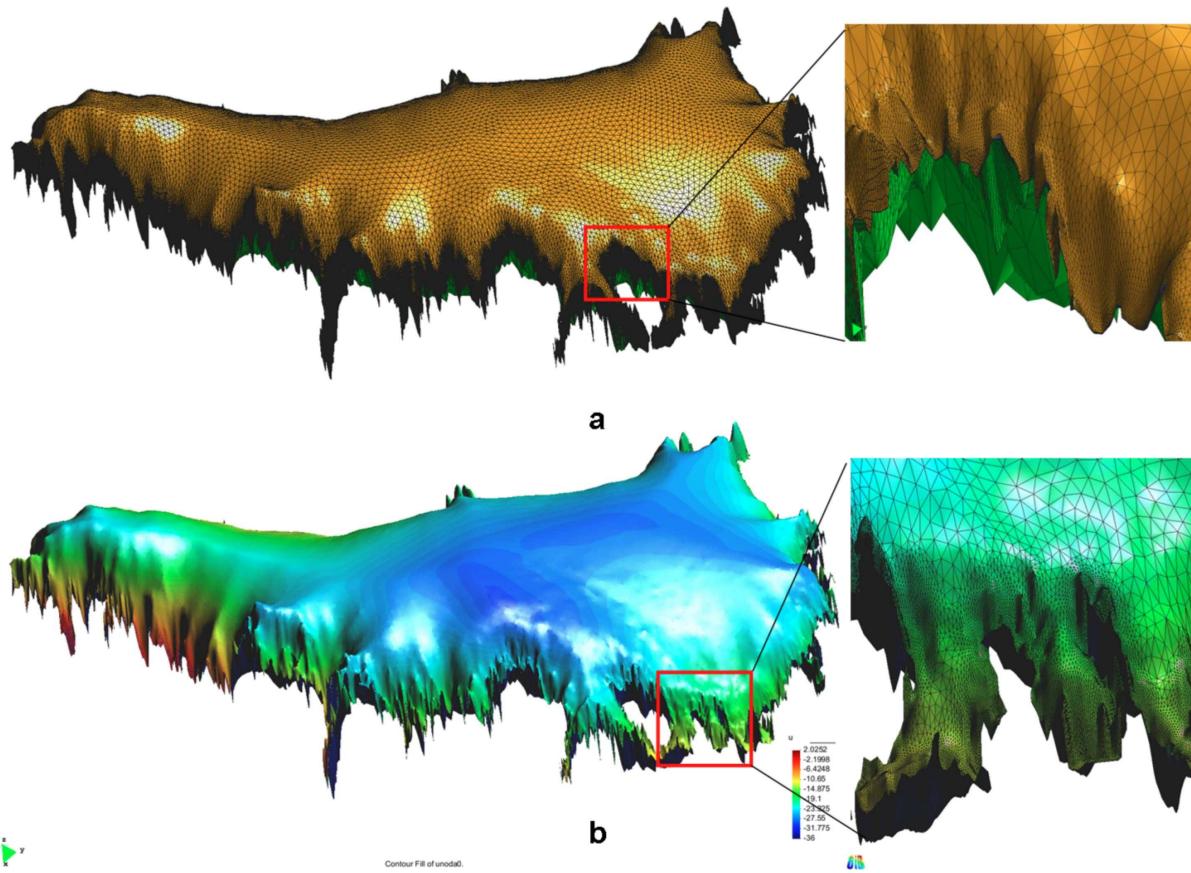
*Left: time-mean kinetic energy of the North Atlantic Ocean that is used as the CVT density function; right: a CVT mesh with 47305 nodes of the North Atlantic*



*Left: a zoom-in of the CVT mesh with 47305; right: a zoom-in of the same region of a CVT mesh with 183907 nodes*



*Left: A 3D prismatic mesh of the Greenland ice sheet (the vertical direction of the ice sheet is stretched) (a) ice sheet surface elevation; (b) ice sheet thickness; (c) bed elevation; (d) zoom-in of a region close to the ice sheet edge. Right: Schematic diagram of the Greenland mesh partition for parallel processing*



*Simulation results for the temperature evolution of the Greenland ice sheet (the vertical direction of the ice sheet is stretched). (a) the 3D prismatic mesh used for the computation; (b) the ice sheet temperature at the ice-atmosphere and ice-ocean boundaries after ten years*

## POINT DISTRIBUTIONS AND GRID GENERATION ON SURFACES

- In many applications, point distributions on surfaces are needed
- In order to generalize CVT's to surfaces, two main ingredients are needed
  - the generalization of the concept of Voronoi regions to surfaces
  - the generalization of the concept of mass centroids to surfaces
- There are a number of ways to do each of these
  - we choose generalizations which are “easy” to use
- We consider compact and continuous surfaces  $S \subset \mathbb{R}^N$

- Given a set of points  $\{\mathbf{z}_i\}_{i=1}^K \in \mathbf{S}$ , we define their corresponding **Voronoi regions** on  $\mathbf{S}$  by

$$V_i = \{ \mathbf{x} \in \mathbf{S} \mid |\mathbf{x} - \mathbf{z}_i| < |\mathbf{x} - \mathbf{z}_j| \text{ for } j = 1, \dots, K, j \neq i \}$$

for  $i = 1, \dots, K$

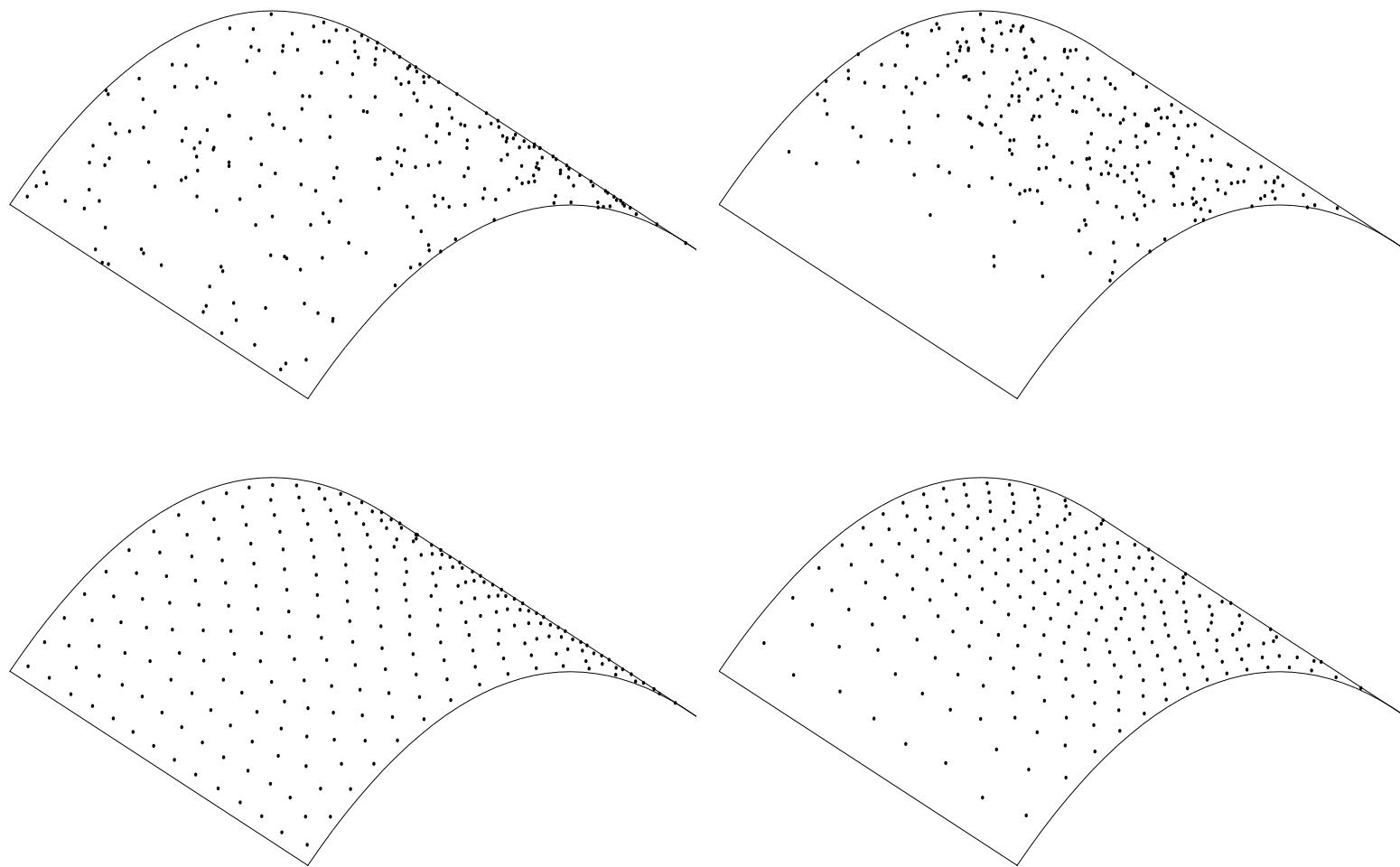
- For each Voronoi region  $V_i$ , we call  $\mathbf{z}_i^c$  the **constrained mass centroid** of  $V_i$  on  $\mathbf{S}$  if  $\mathbf{z}_i^c$  is a solution of the following problem:

$$\min_{\mathbf{z} \in \mathbf{S}} F_i(\mathbf{z}), \quad \text{where} \quad F_i(\mathbf{z}) = \int_{V_i} \rho(\mathbf{x}) |\mathbf{x} - \mathbf{z}|^2 d\mathbf{x}$$

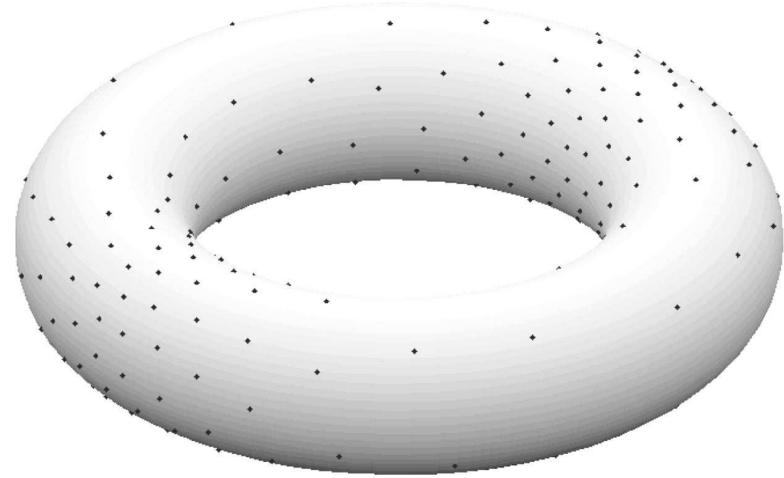
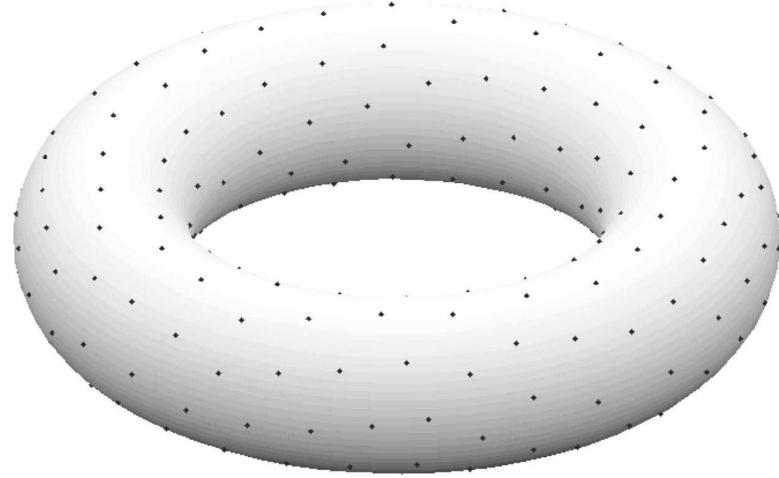
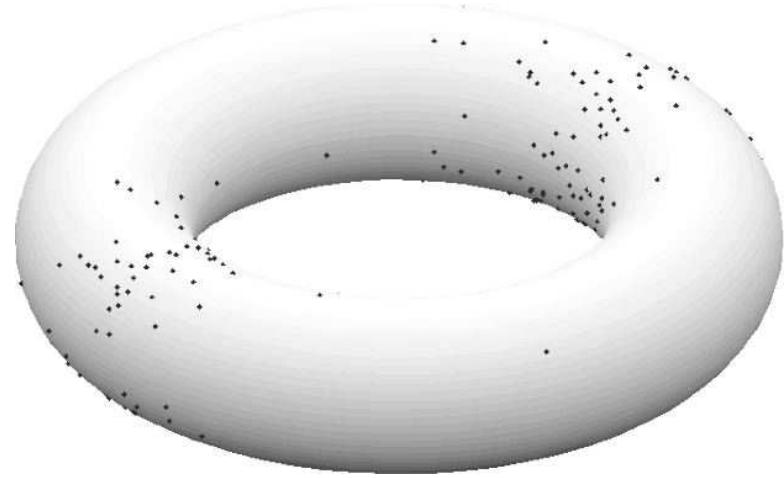
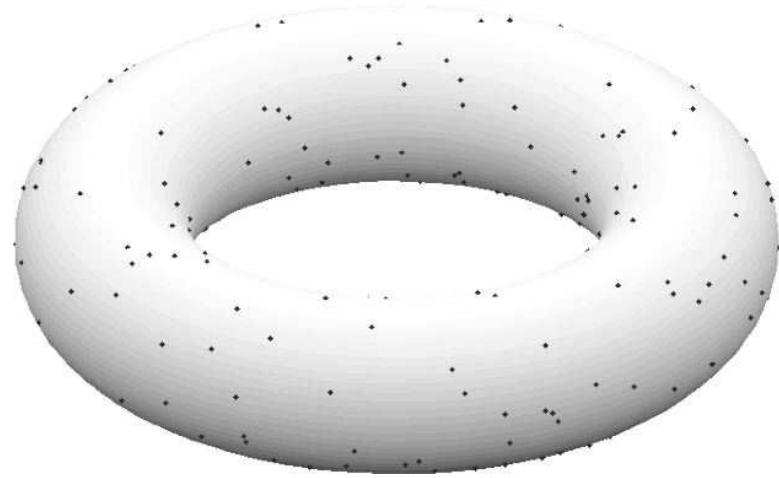
- We call a Voronoi tessellation a **constrained centroidal Voronoi tessellation (CCVT)** if and only if the points  $\{\mathbf{z}_i\}_{i=1}^K$  which serve as the generators of the Voronoi regions  $\{V_i\}_{i=1}^K$  are also the constrained mass centroids of those regions

- Note that the definition of CCVT implies that
  - generators are constrained to the surfaces
  - but distances are still the standard Euclidean distances, not the more general geodesic distances
- Constrained centroids are “easy” to construct by normal projection
  - first construct the 3D centroid which, in general, is not on the surface
  - then project the 3D centroid to the surface
  - it turns out (under mild assumptions) that the projected point is the constrained mass centroid

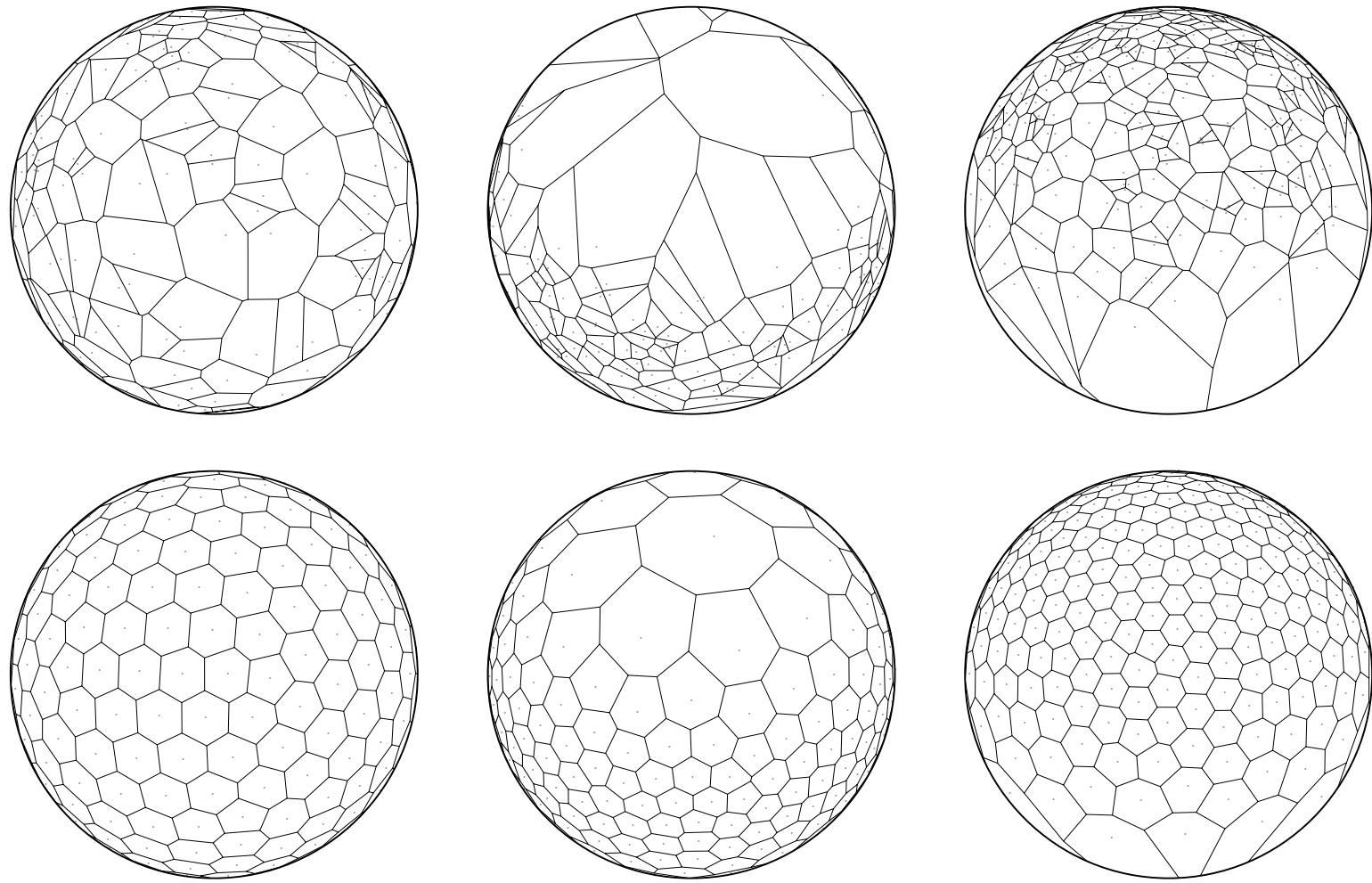
- Constrained CVT's defined in the manner above enjoy an [optimization property](#) in much the same way as do CVT's
- Algorithms (deterministic and probabilistic, serial and parallel) for CVT's may be then easily generalized to the case of CCVT's



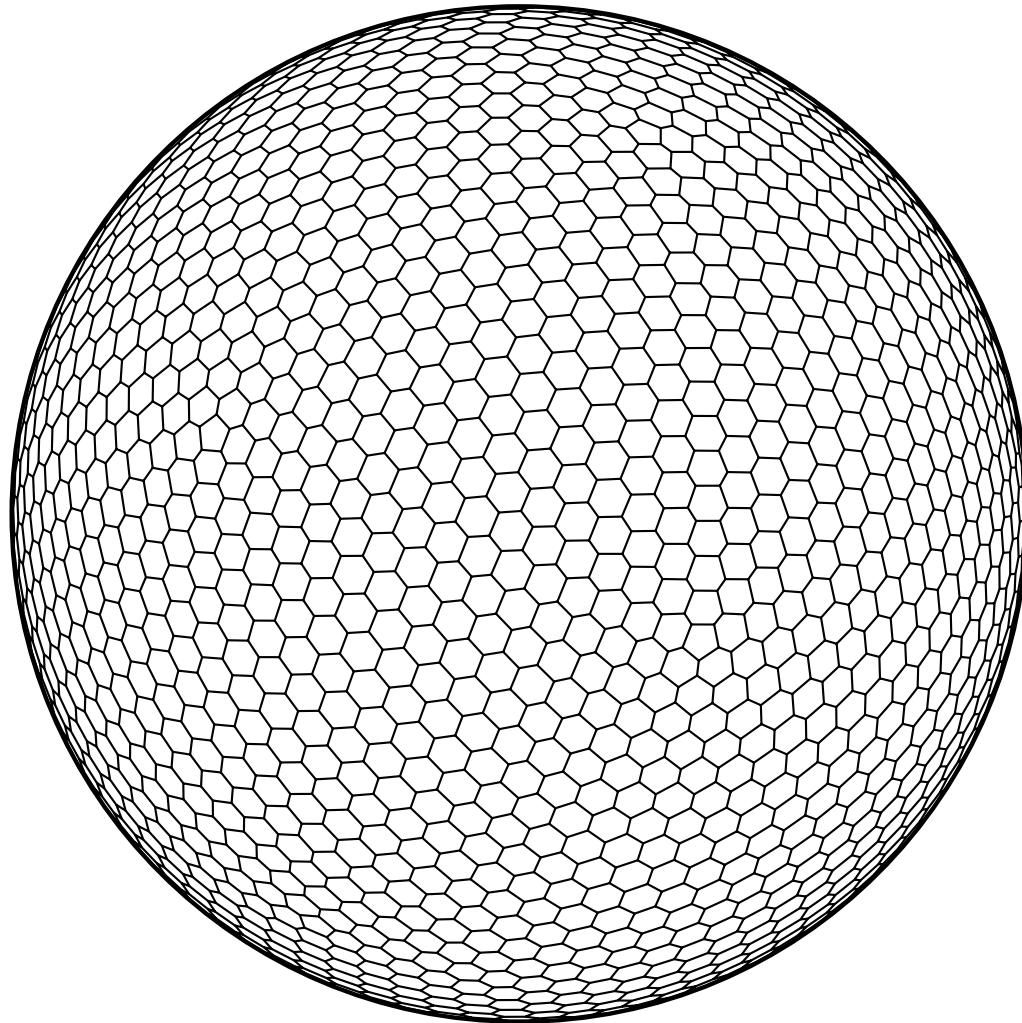
*Voronoi diagrams for 256 generators on a developable surface*  
*random sampling (top)    constrained CVT (bottom)*  
 $\rho(x, y, z) = 1$  (*left*)     $\rho(x, y, z) = e^{-20.0x^2}$  (*right*)



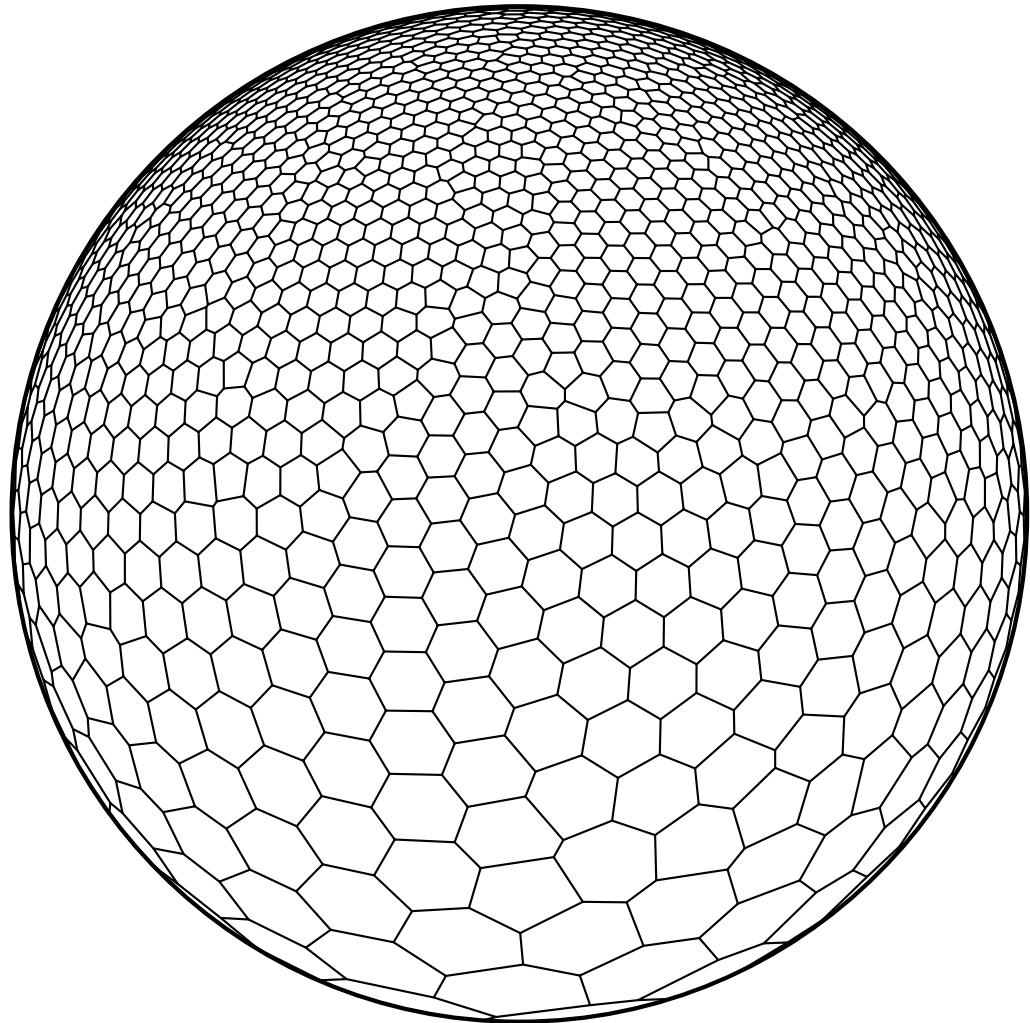
*Voronoi diagrams for 256 generators on a torus*  
*random sampling (top)*    *constrained CVT (bottom)*  
 $\rho(x, y, z) = 1$  (*left*)     $\rho(x, y, z) = e^{-5.0|y|}$  (*right*)



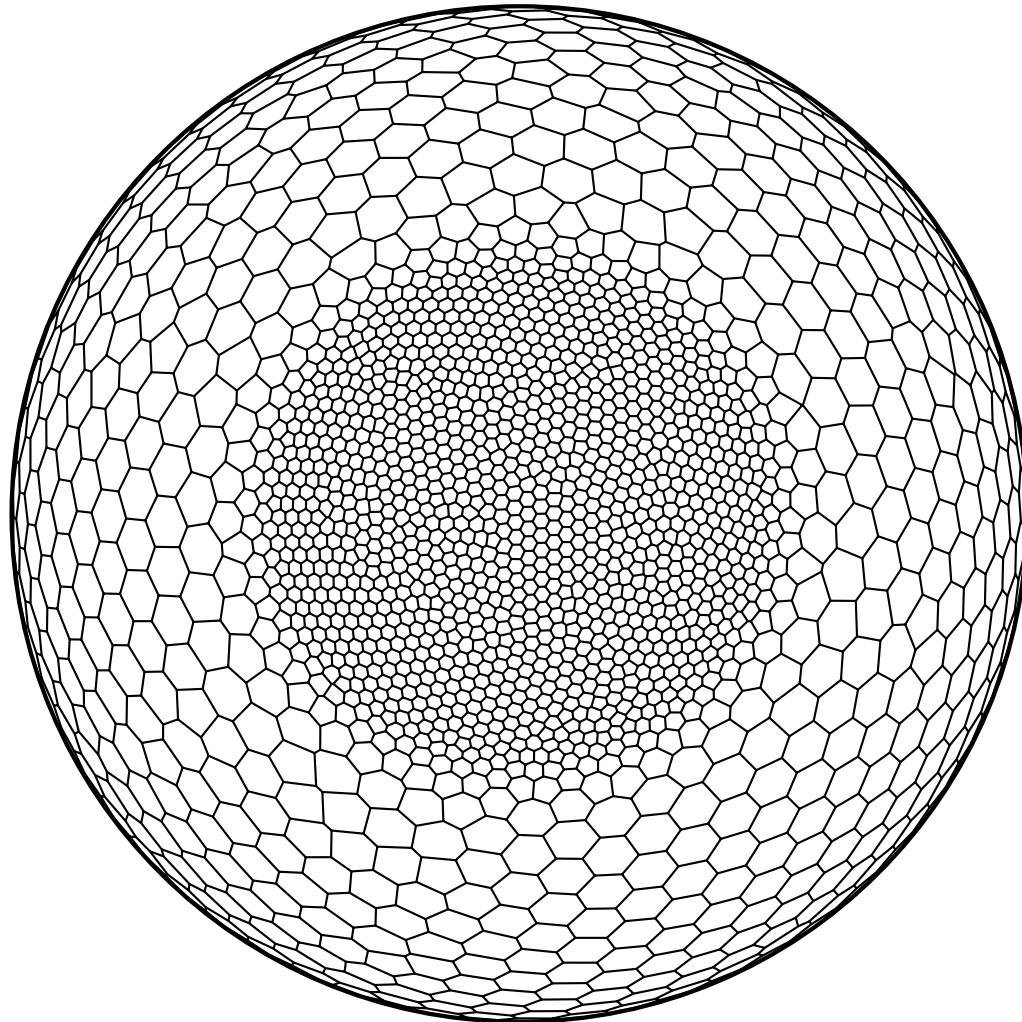
*Voronoi diagrams for 256 generators on the sphere*  
*random sampling (top)    constrained CVT (bottom)*  
 $\rho(x, y, z) = \begin{cases} 1 & (\text{left}) \\ e^{-6.0z^2} & (\text{middle}) \\ e^{-3.0(1-z)^2} & (\text{right}) \end{cases}$



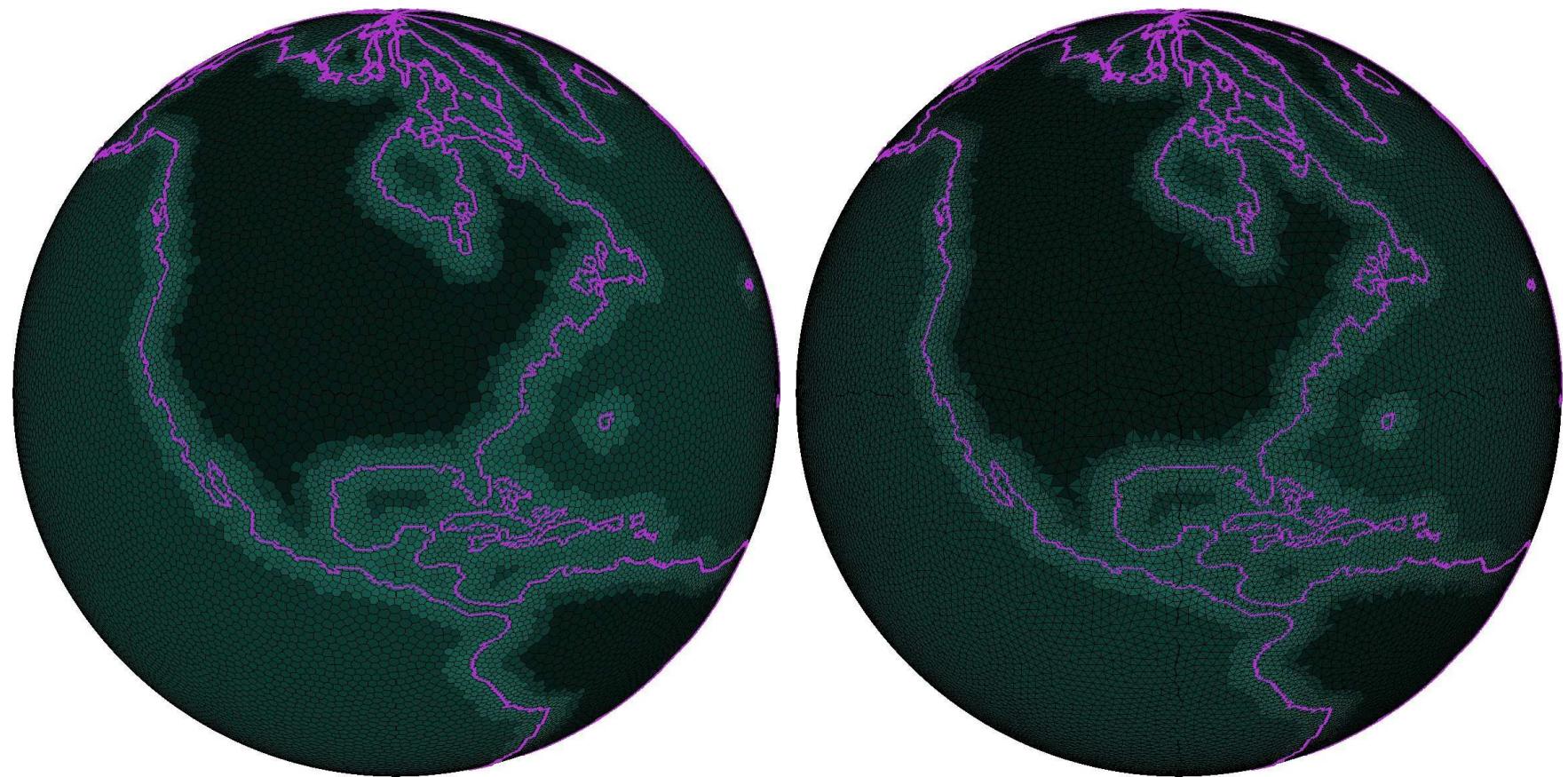
*CCVT on the sphere with uniform density*



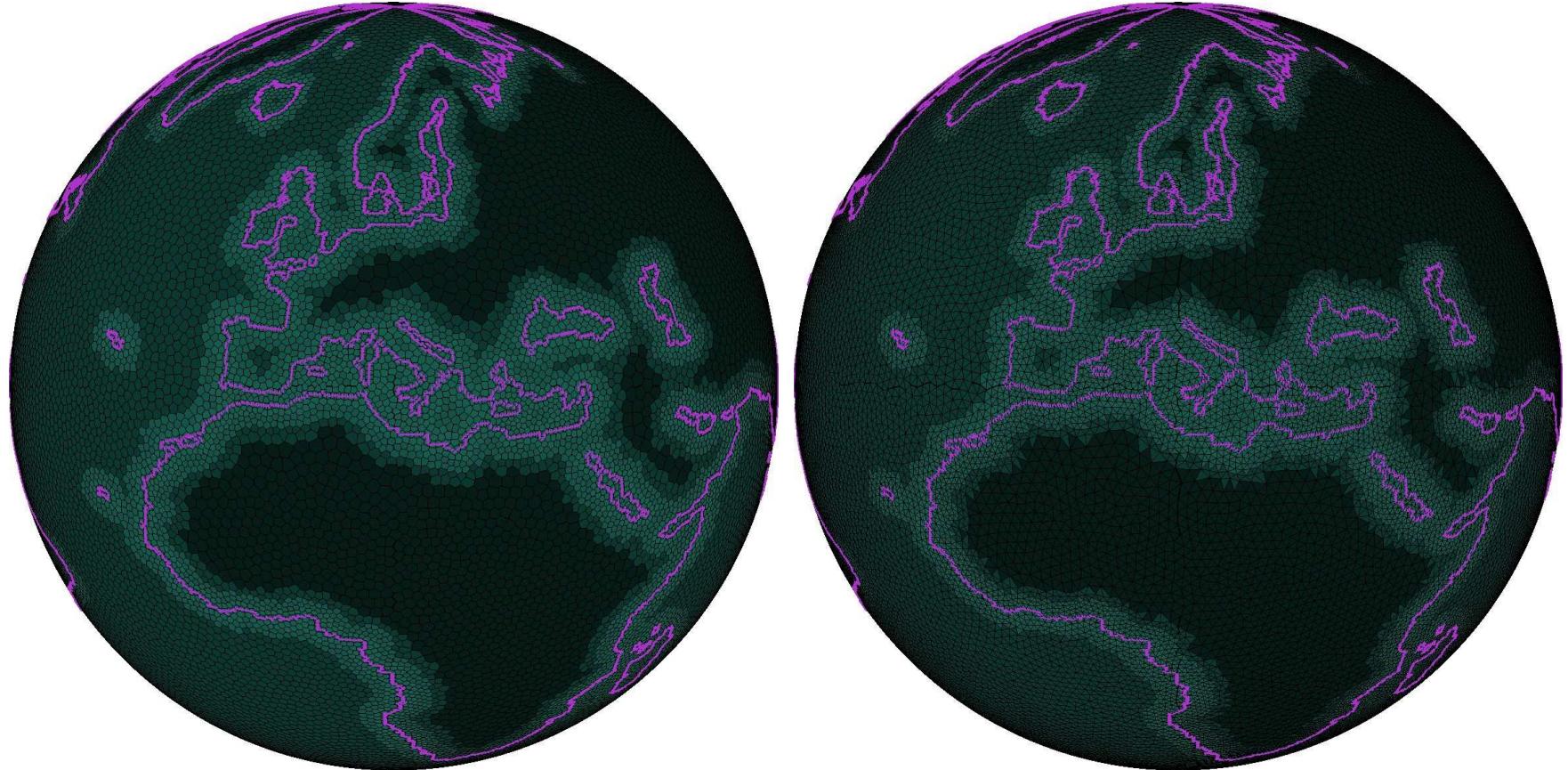
*CCVT on the sphere with nonuniform density*



*CCVT on the sphere with local refinement*



*Global CCVT of the globe and the corresponding Delauney triangulation with points automatically placed on ocean/land boundaries*



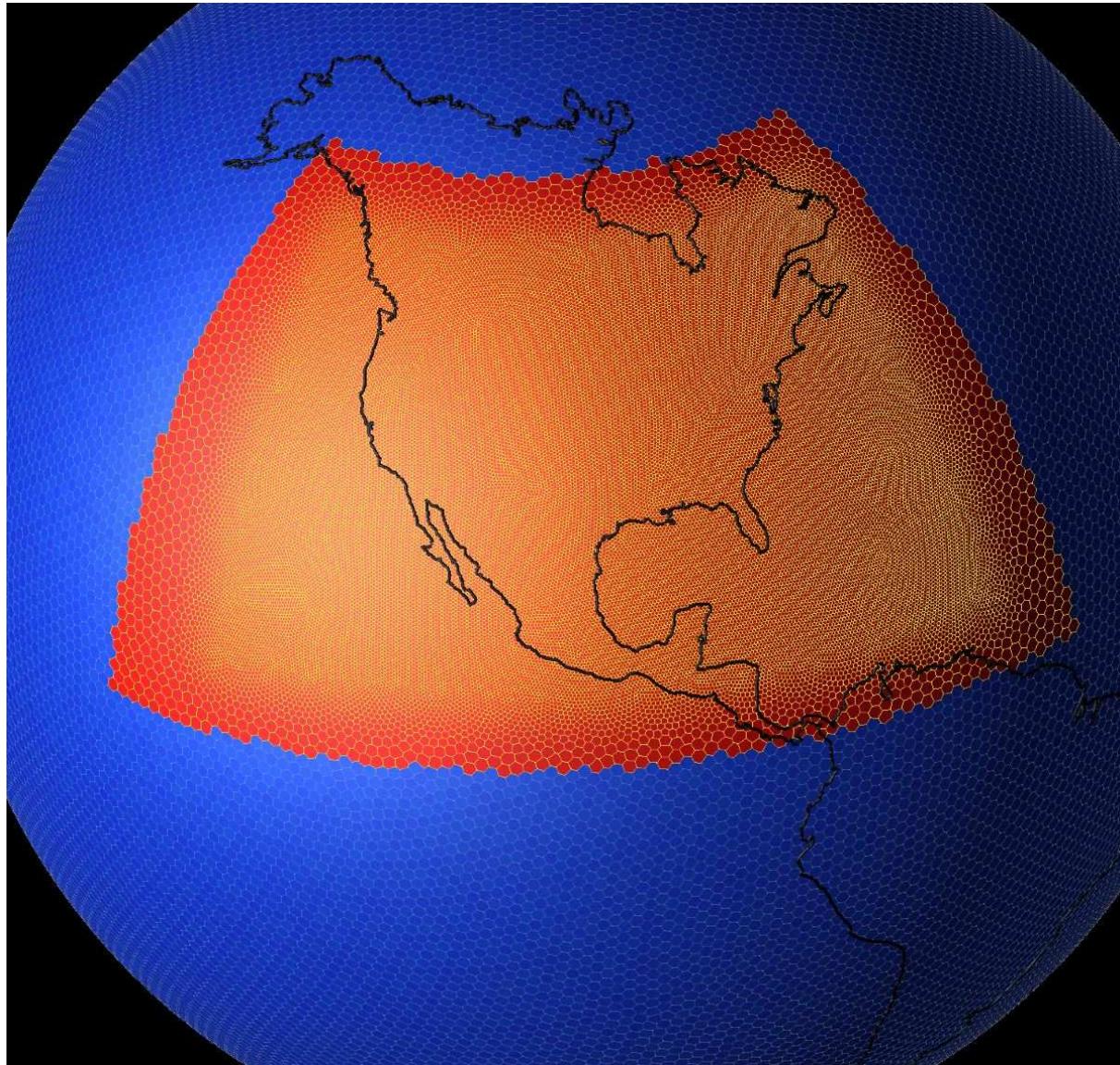
*Global CCVT of the globe and the corresponding Delaunay triangulation with points automatically placed on ocean/land boundaries*

## Regional refinement

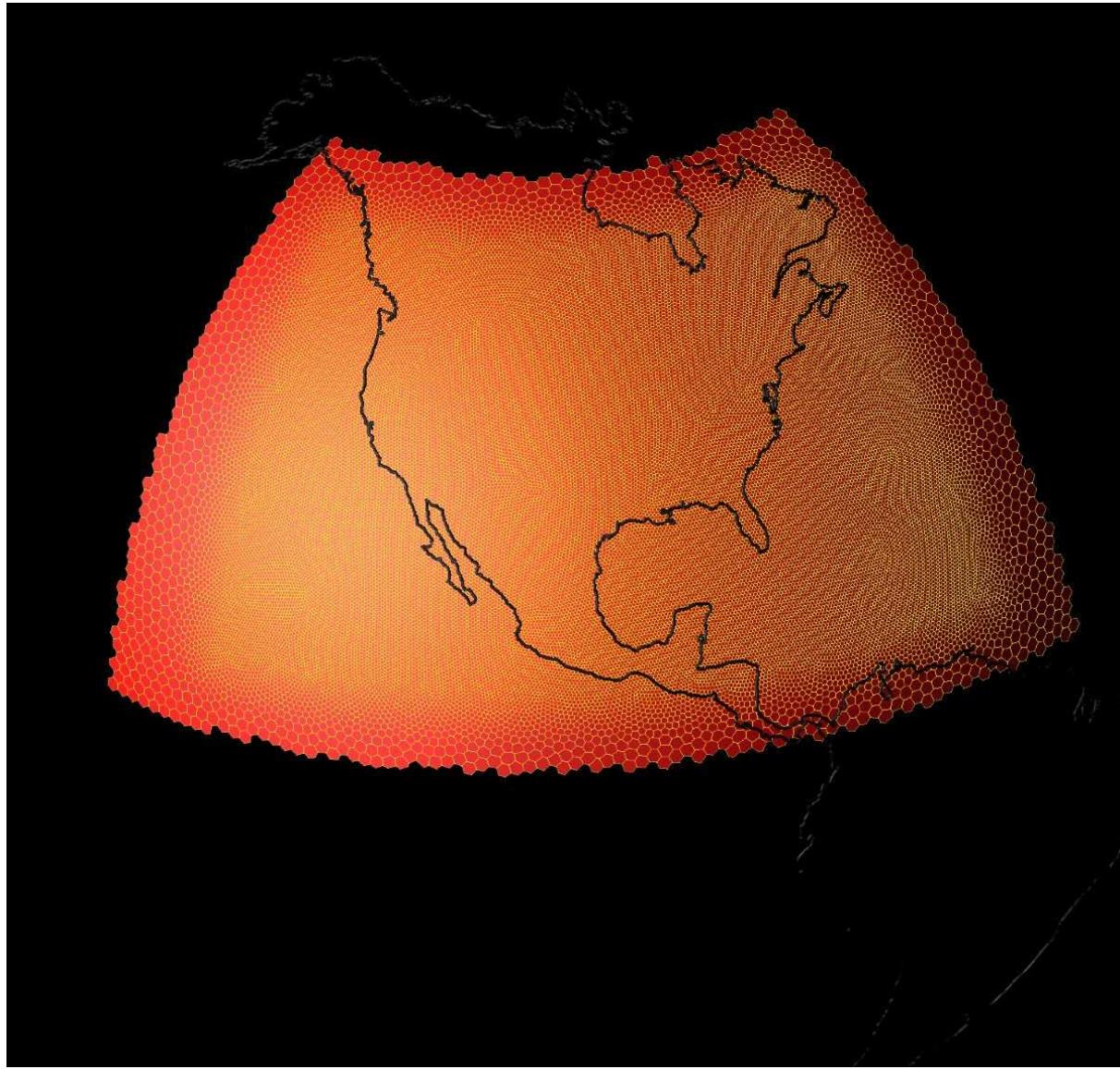
- Two scenarios for regional refinement:
    - a locally refined global simulation is run on the whole or large parts of the sphere or ocean
    - alternately, a coarse uniform grid simulation is run on the whole or large parts of the sphere or on the oceans
      - followed by a regional fine-grid simulation using the results of the coarse-grid simulation to provide boundary conditions
    - for both scenarios, compatibility between coarse and fine grids is important or at least useful for accuracy and conservation
  - We can generate regionally refined SCVT grids such that
    - the fine grid in the region of refinement matches exactly with the coarse uniform global grid
- and
- the transition from coarse to fine meshes is “smooth”

## **SCVT grids for the first scenario**

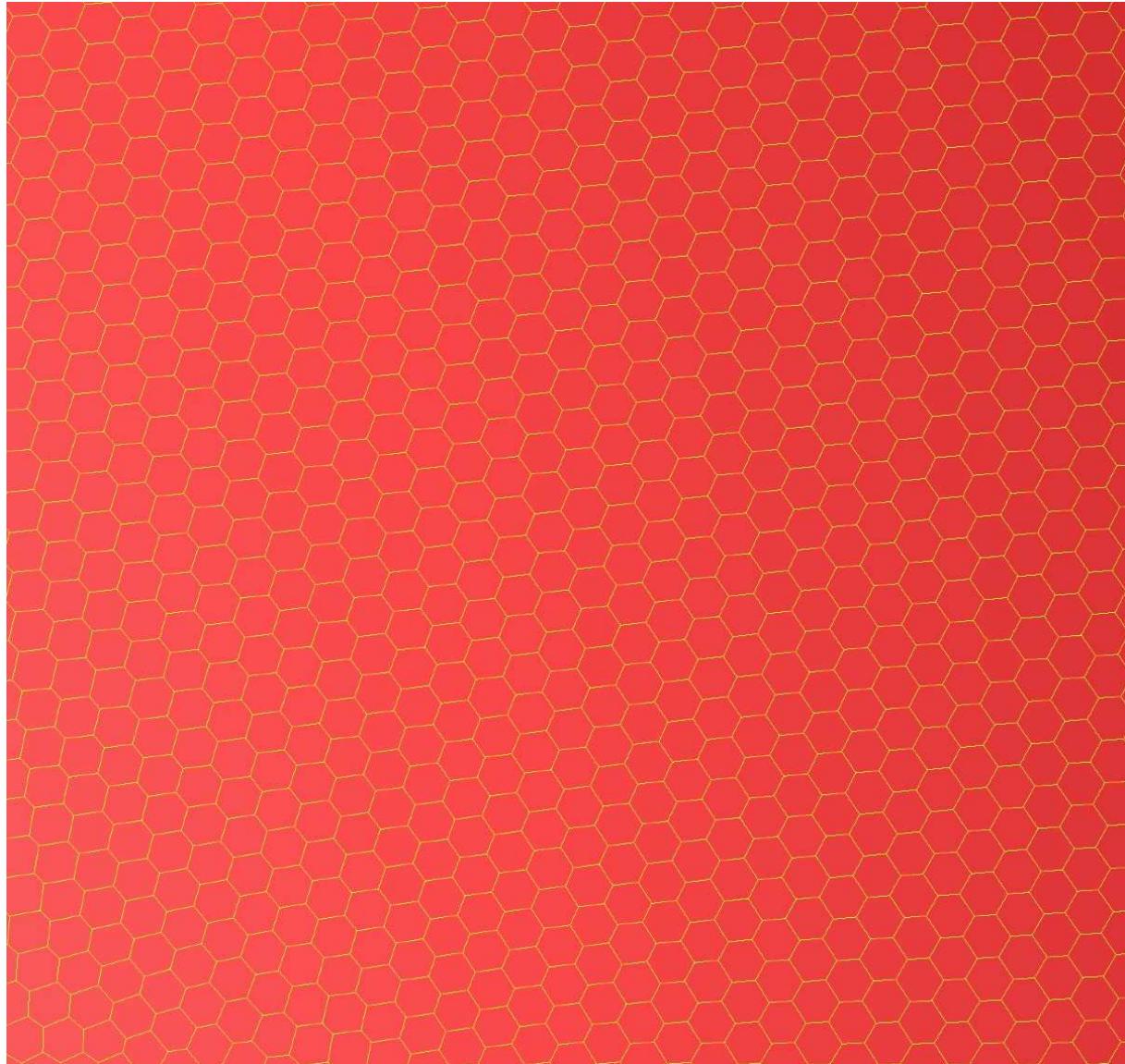
- The region of interest is the United States plus a little more
- 65,026 total nodes with 35,270 nodes in the nested region
- Outer domain grid is quasi-uniform with a resolution of  $\approx 120$  km
- Inner domain grid is quasi-uniform with a resolution of  $\approx 40$  km away from its boundary



*Global regionally refined grid.eps*



*Grid in refined region.eps*



*Grid in interior of refined region*



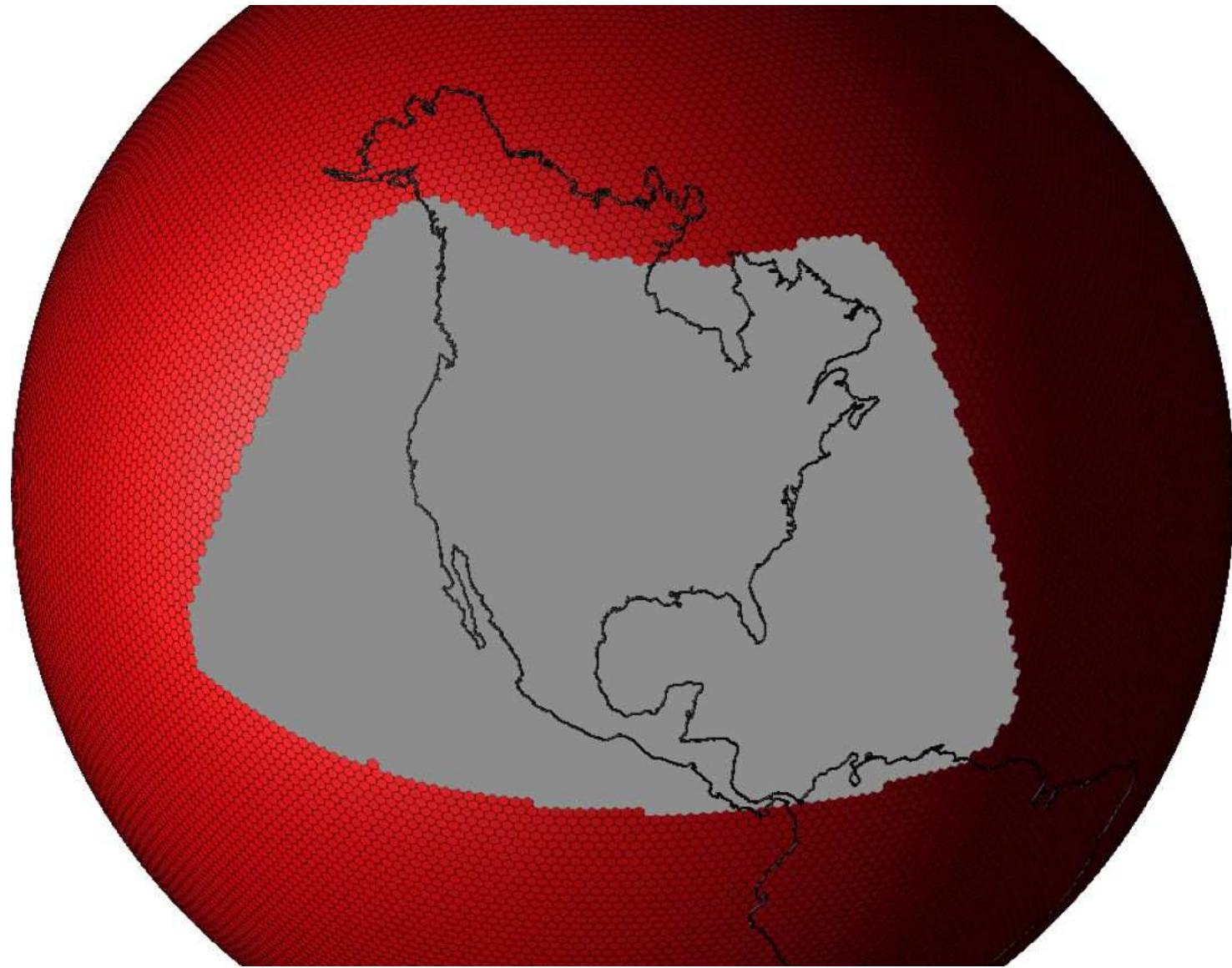
*Refined grid in transition region.eps*

## SCVT grids for the second scenario

- The region of interest is again the United States plus a little more
- The global uniform grid has 40,962 nodes and the local refined grid has 65,026 nodes
- The refined grid exactly matches with the parent global grid it replaced near the boundary between the two regions
- In the local refined grid, a “smooth” transition occurs from the fine to coarse quasi-uniform grids



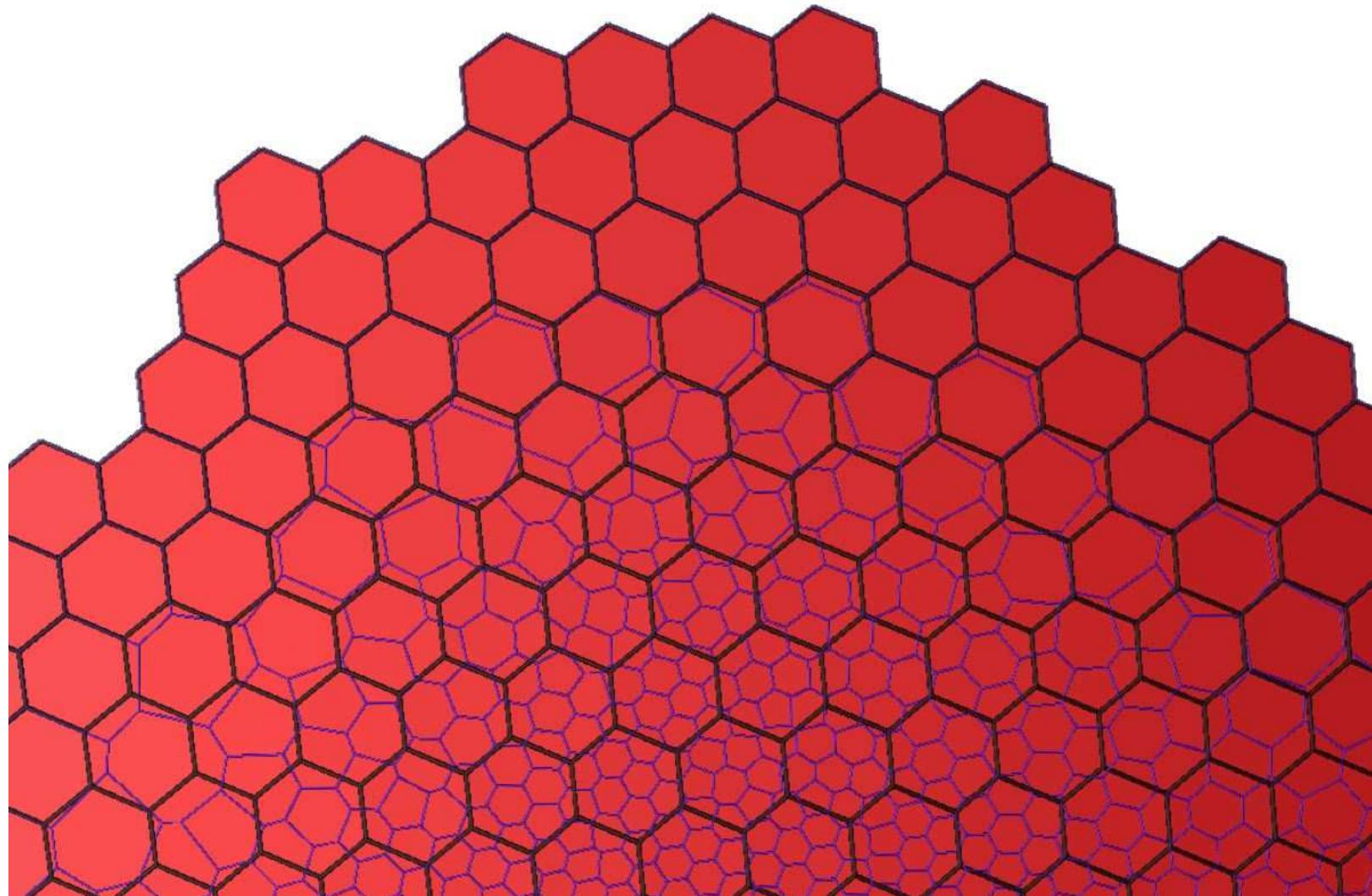
*Global uniform grid.eps*



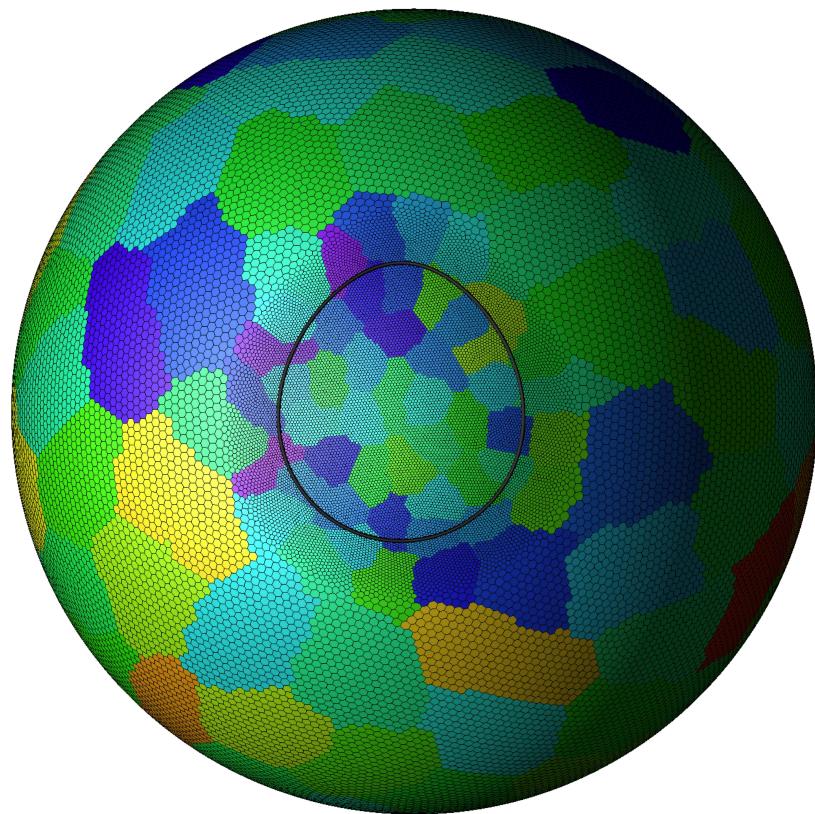
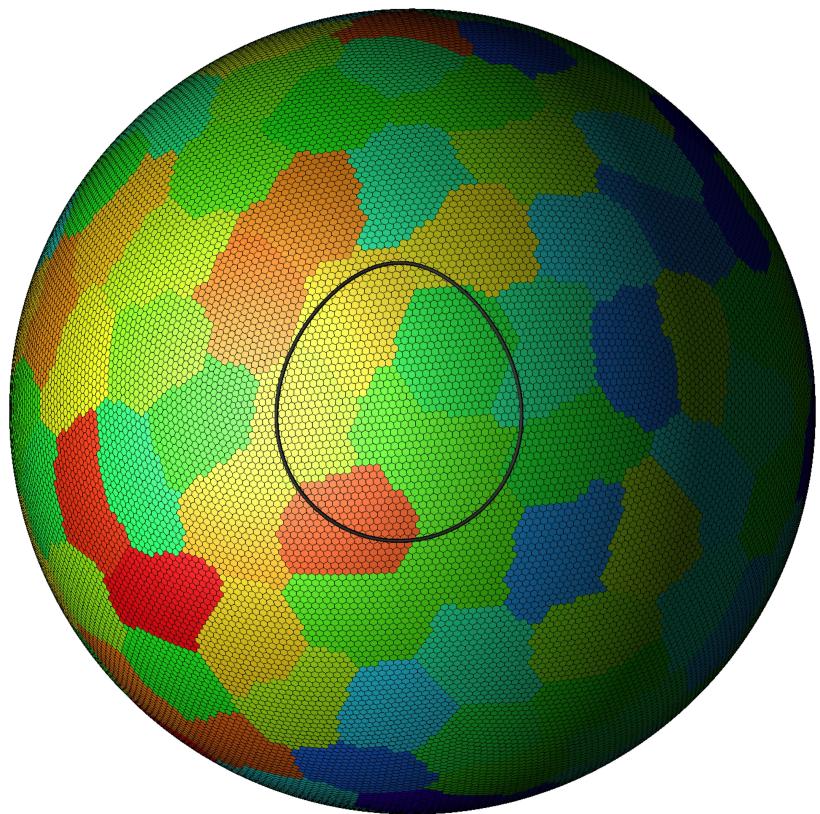
*Region to be refined*



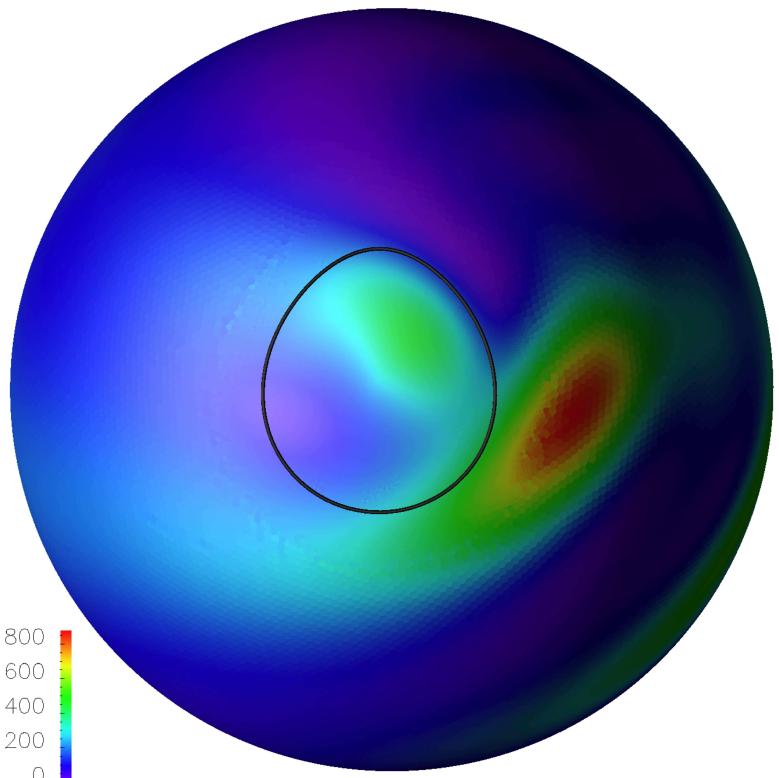
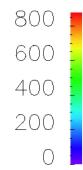
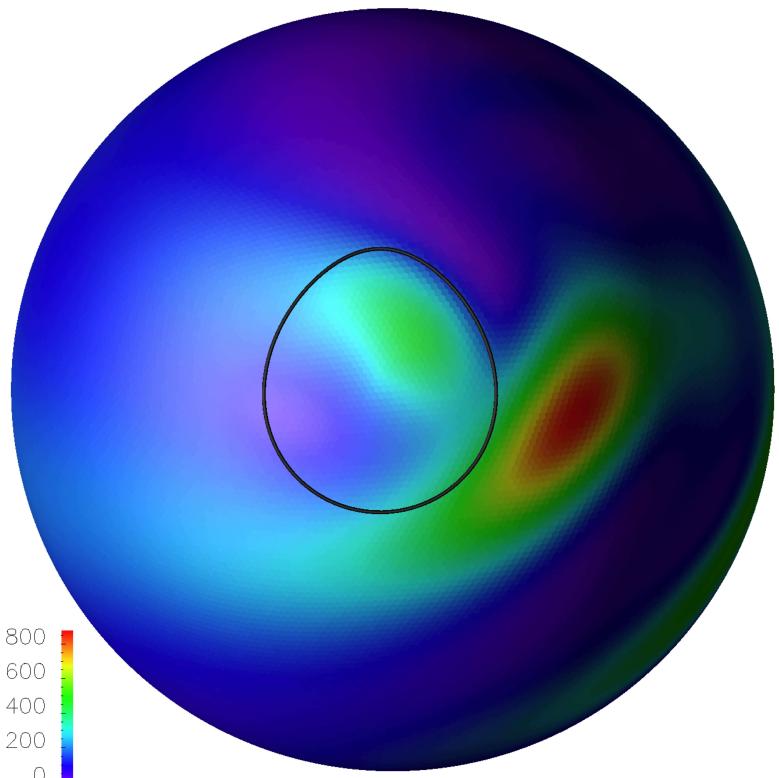
*Refined local grid*



*Match between global grid and refined local grid*



*Quasi-uniform (left) and variable resolution (right) grids on the sphere; the grid on the right is refined in the vicinity of an orographic feature (a mountain) that is the sole forcing in the simulation; the colors indicate a domain-decomposition strategy for efficient implementation on distributed memory systems – each block represents a different computational processor*



*Kinetic energy field at day 10 of simulation; left: simulation using quasi-uniform mesh; right: simulation using variable-resolution mesh*

## MESHLESS COMPUTING

- Meshless computing refers to numerical methods that do not involve meshes
- They can be used, e.g., for approximating
  - multivariate functions
  - multiple integrals
  - solutions of partial differential equations
- The hope is that since meshes are not required, problems posed on complicated and/or high-dimensional regions can be treated more easily
- Of course, meshless computing is nothing new, e.g., recall
  - Monte Carlo methods for numerical integration and the numerical solution of PDE's
  - particle methods for PDE's

- A typical efficient meshless computing method requires
  1. the selection of a set of points
  2. the selection of a support region associated with each point
  3. the selection of a basis function associated with each point having support over the region selected in step 2
- The implementation of a typical meshless method also requires
  4. knowledge about the overlap of the support regions associated with distinct basis functions
  5. the application of a discretization method, e.g., in the PDE setting, one can use any of
    - Galerkin, collocation, mixed, or least squares methodswhile in the function approximation setting, one can use, e.g.,
    - interpolation or least squares approximation
- One can also analyze meshless methods
  6. derive error estimates

- A great deal of attention has been devoted to steps 3, 5, and 6
  - lots of papers on deriving error estimates for specific choices of basis functions, assuming the point distribution is given and the support radii and the overlap information are known or at least are easily determined
- Some papers determine support radii and overlap information by using graph theoretic ideas

since graph = mesh, these papers do not address truly meshless methods
- Much less effort has been devoted to the efficient determination, in a truly meshless way, of
  - optimal point distributions
  - support radii
  - overlap information

## POINT PLACEMENT

- We use **the generators of a centroidal Voronoi tessellation** of the computational domain for the points on which the meshless method is based
  - using probabilistic algorithms, the CVT points can be determined in a **totally meshless** manner
- There are, of course many other methods for selecting point positions
  - for example, for uniform point distributions in simple domains, Halton sequences are a popular point selection method
- CVT point selection strategies offer substantially more flexibility than most existing methods in that they can be used for
  - general domains
  - nonuniform and anisotropic point distributions
  - some or all points can be placed on bounding surfaces

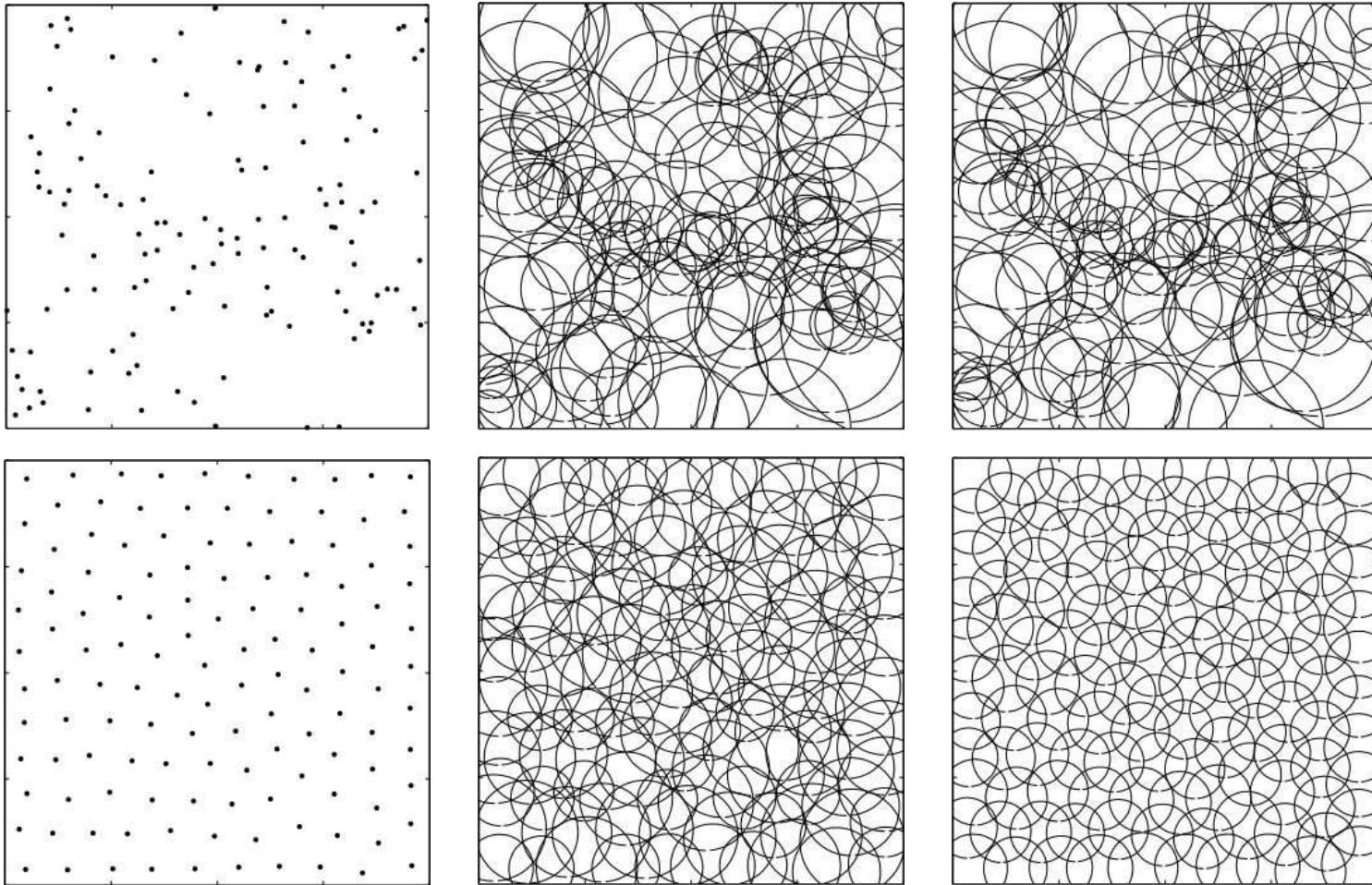
## SUPPORT RADII AND OVERLAP DETERMINATION

- Having chosen a set of  $K$  points  $\{x_i\}_{i=1}^K$  in the domain  $\overline{\Omega} \subset \mathbb{R}^N$ , we want to now choose, for each point  $x_i$ , an associated radius  $h_i$
- For each point  $x_i$ , we then define the sphere centered at  $x_i$  and having radius  $h_i$

$$S_i = \{ y \in \mathbb{R}^N : |y - x_i| \leq h_i \}$$

- other patch shapes associated with points can be used, e.g., ellipsoids, rectangles, or bricks
- the sphere (or other patch) associated with a point will determine the support region for basis function associated with the point
- The selection of the radii  $\{h_i\}_{i=1}^K$  should meet two requirements:
  - the union of the spheres  $\{S_i\}_{i=1}^K$  should cover  $\overline{\Omega}$  a specified number of times
  - if the intersection  $S_i \cap S_j$  of two distinct spheres is not empty, it should not be “too small” nor “too large”

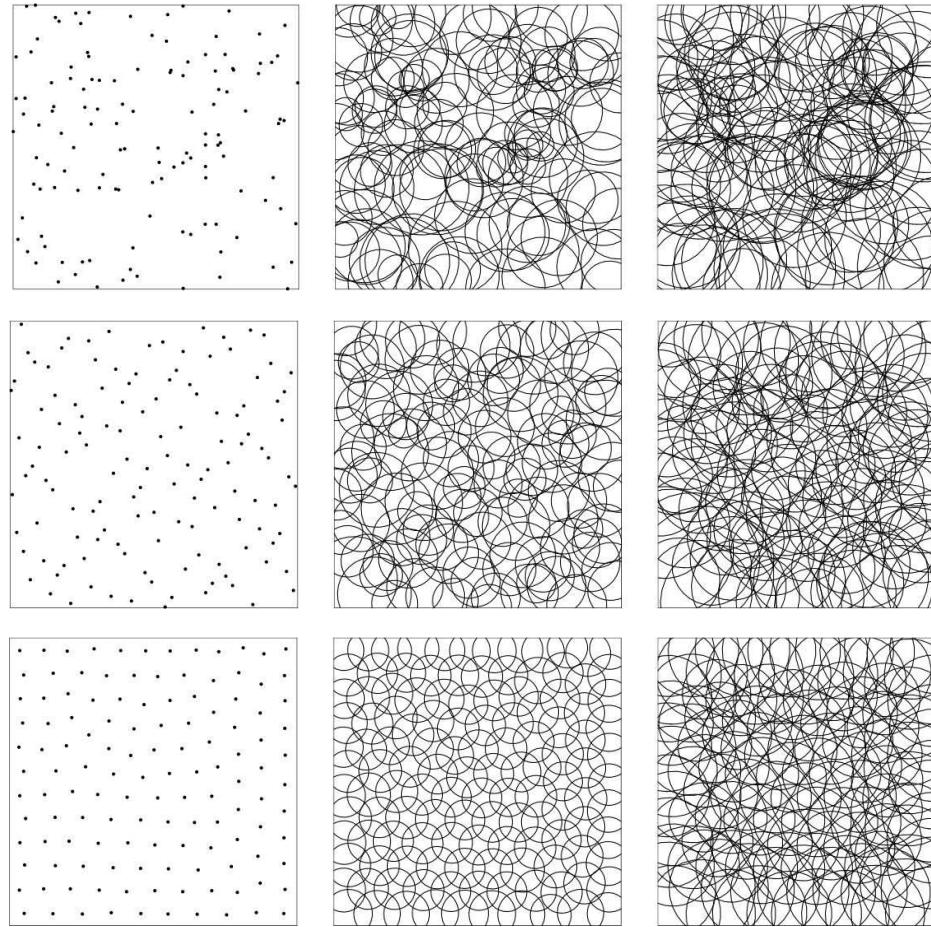
- Algorithms are known (notably due to C. Duarte and J. Oden and to M. Griebel and M. Schweitzer) for support radii determination
  - these algorithms involve a background **mesh**
  - they do not give good results for nonuniform point distributions
- A new, (**totally meshless**) algorithm for determining support radii has been developed that
  - guarantees, in a probabilistic sense, that the patches cover every point in the domain  $p$ -times, where  $p$  is an input integer
    - some meshless methods for PDE's require that each point in the region be covered several times by the support spheres associated with the point set
  - allows for control of the amount of overlap of the support regions
  - gives good results for nonuniform point distributions in general regions
- Although we illustrate using circular patches in 2D, there is no difficulty treating 3D and/or other shaped patches, e.g., rectangles or bricks



*Sets of 128 uniformly distributed Monte Carlo (top-left) and CVT (bottom-left) points in the square; the corresponding sphere coverings resulting from the methods due to C. Duarte and J. Oden and also M. Griebel and M. Schweitzer (middle) and due to the new algorithm (right)*

No. of points	Support region algorithm	Point selection method	
		Monte Carlo	CVT
128	old	13.33%	10.66%
	new	12.26%	5.21%
256	old	6.61%	4.87%
	new	6.24%	2.58%
512	old	3.10%	2.49%
	new	2.81%	1.25%

*Percentages of nonzero elements in the characteristic matrix for a uniform distribution of points in a square domain*

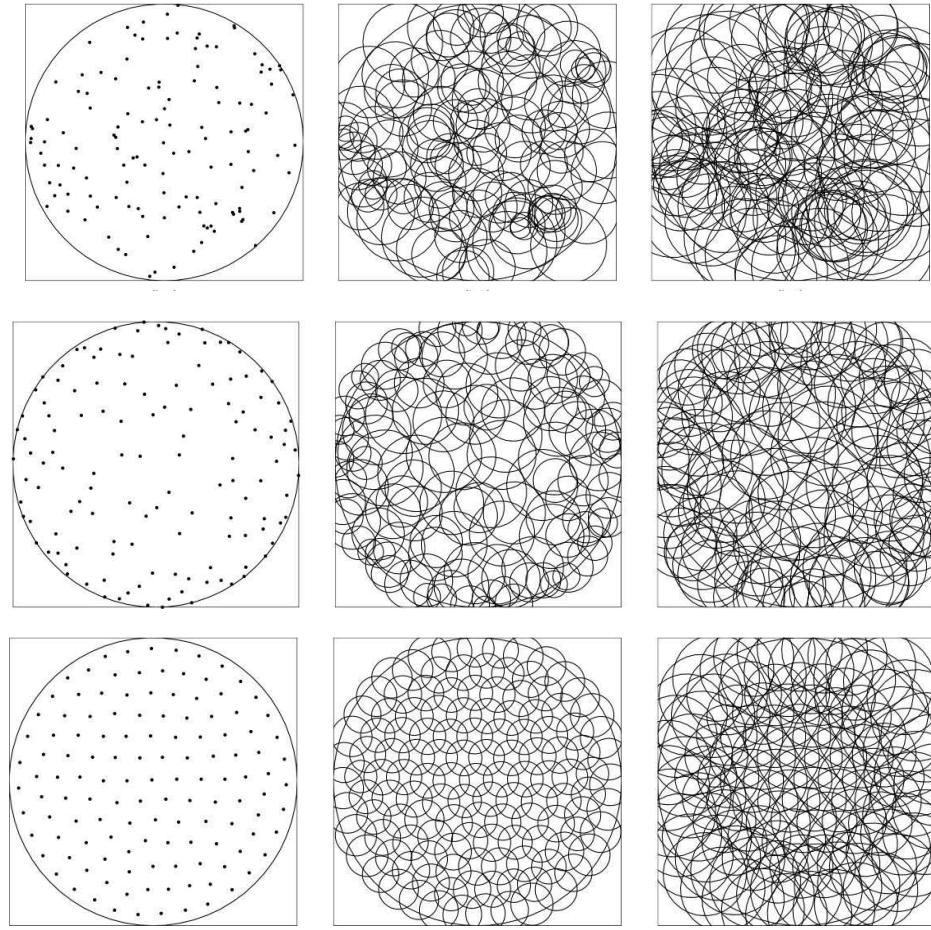


*Sets of 128 points in a square (left) and the associated spherical patches determined by the new algorithm for  $p = 1$  (middle) and  $p = 3$  (right) for the Monte Carlo (top), (2,3) Halton sequence (middle), and CVT (bottom) point selection methods for a uniform density function*

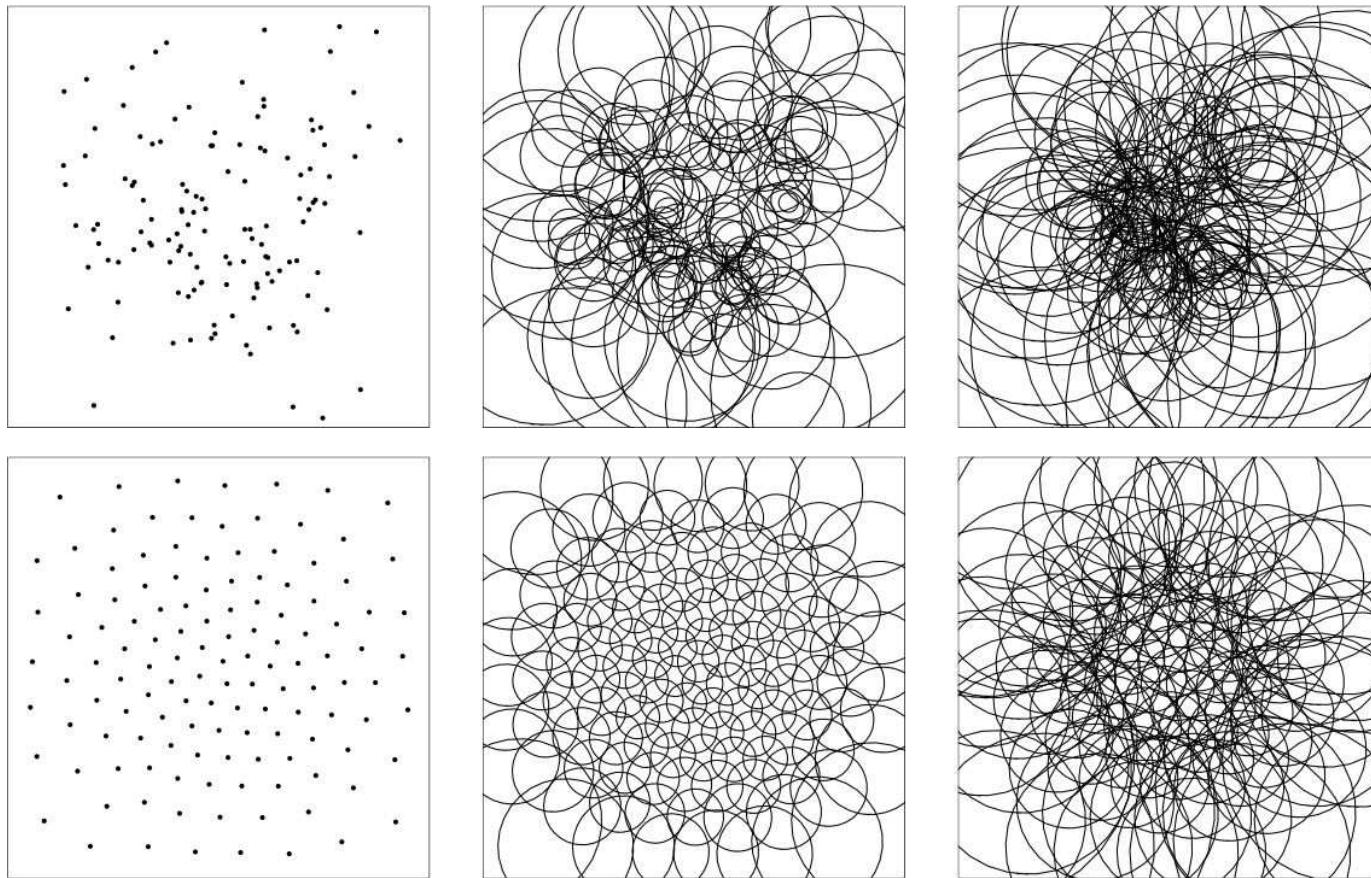
No. of points	$p$	Point selection method		
		Monte Carlo	Halton sequence	CVT
128	1	12.02%	10.04%	5.05 %
	3	25.41%	21.63%	18.68%
256	1	6.25%	4.95%	2.61%
	3	14.23%	10.97%	9.53%
512	1	3.29%	2.72%	1.33%
	3	7.30%	5.91%	4.72%

*Percentages of nonzero elements in the characteristic matrices resulting from the new support region determination algorithm for a uniform distribution of points in a square domain*

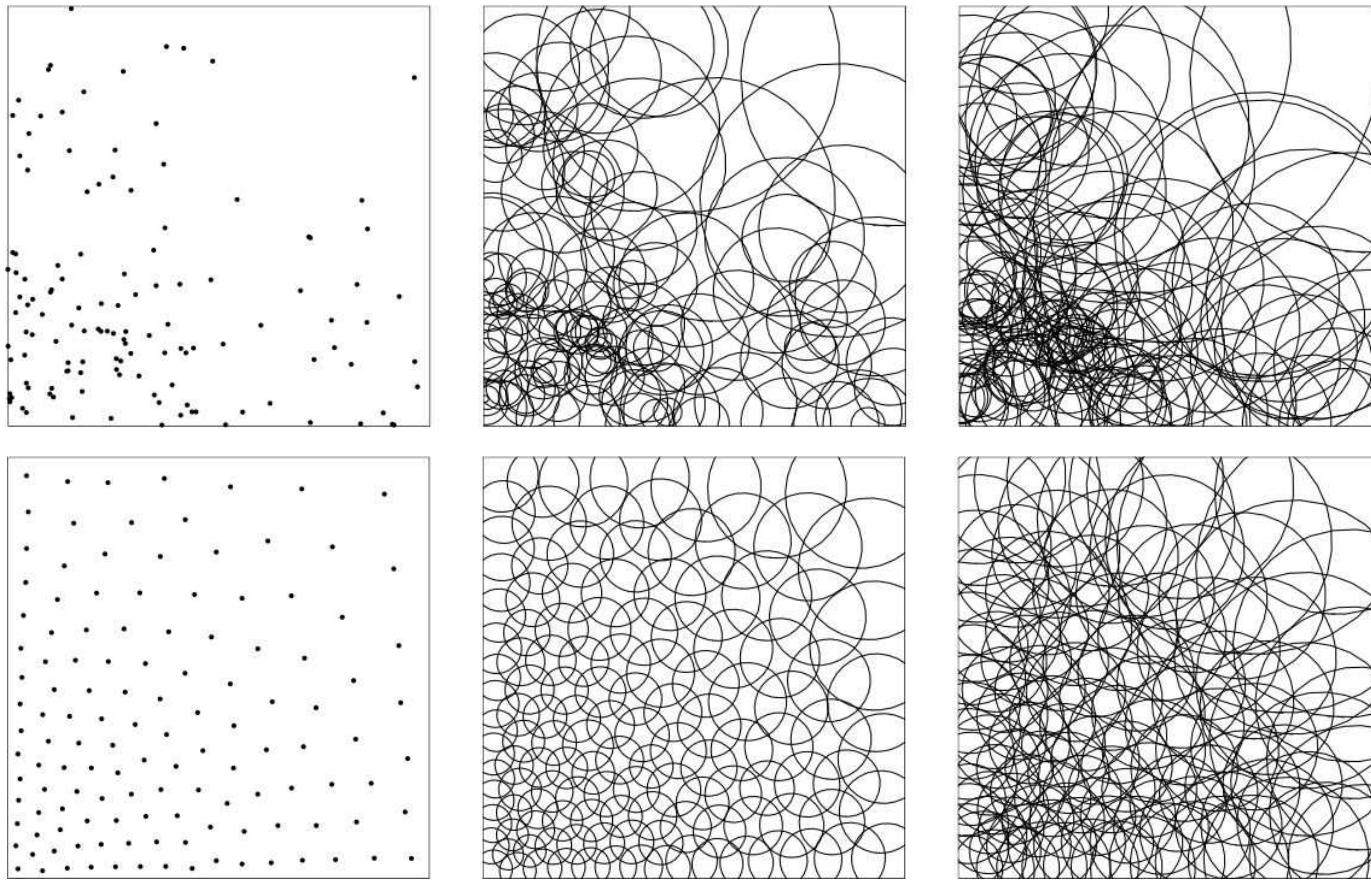
- The CVT numbers using the new support region determination algorithm are almost identical to what one obtains
  - for **linear finite element** discretizations in the case  $p = 1$  (a 1-covering of the domain)
  - for **quadratic finite element** discretizations in the case  $p = 3$  (a 3-covering of the domain)



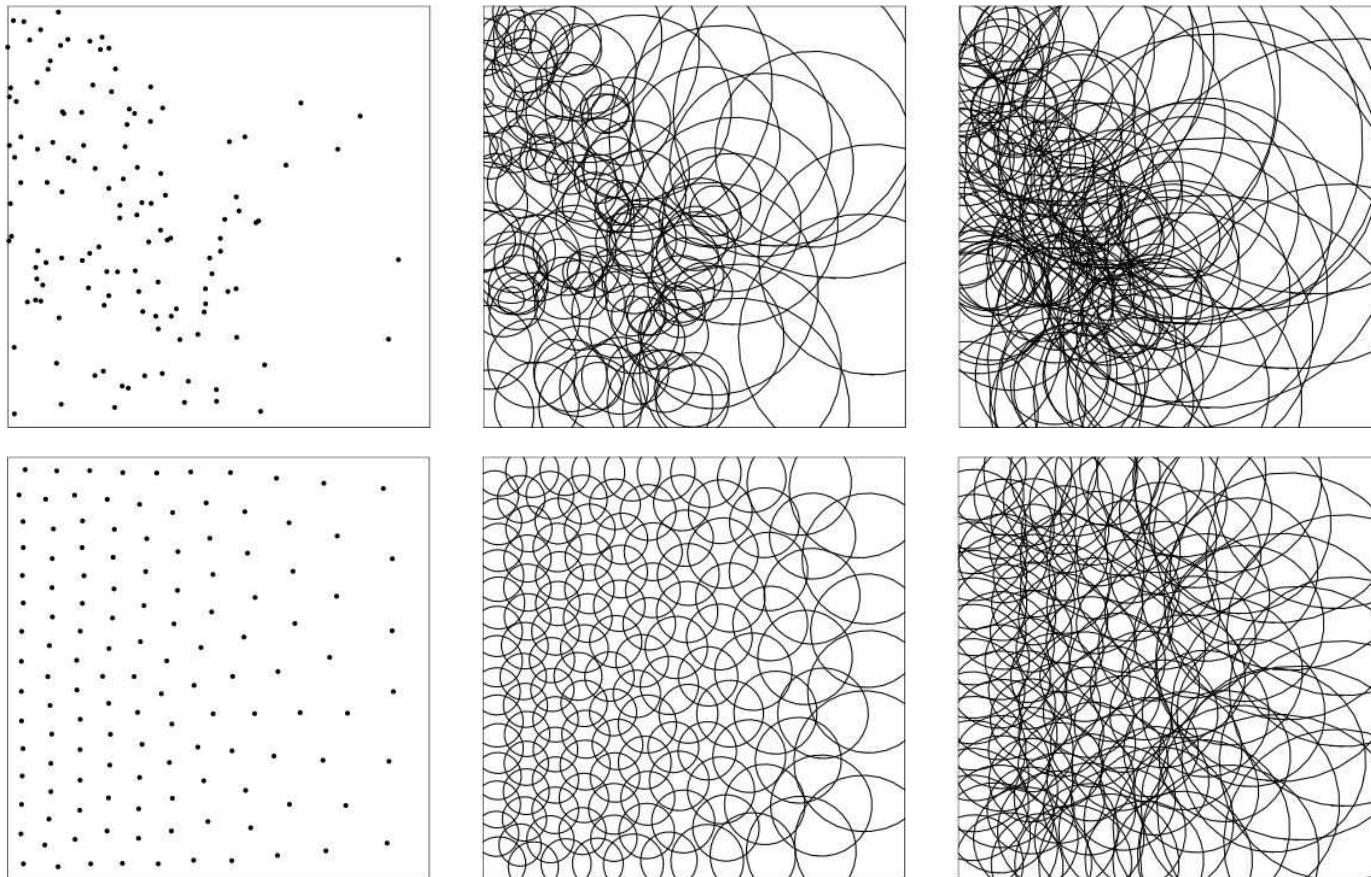
*Sets of 128 points in a circle (left) and the associated spherical patches determined by the new algorithm for  $p = 1$  (middle) and  $p = 3$  (right) for the Monte Carlo (top), (2,3) Halton sequence (middle), and CVT (bottom) point selection methods for a uniform density function*



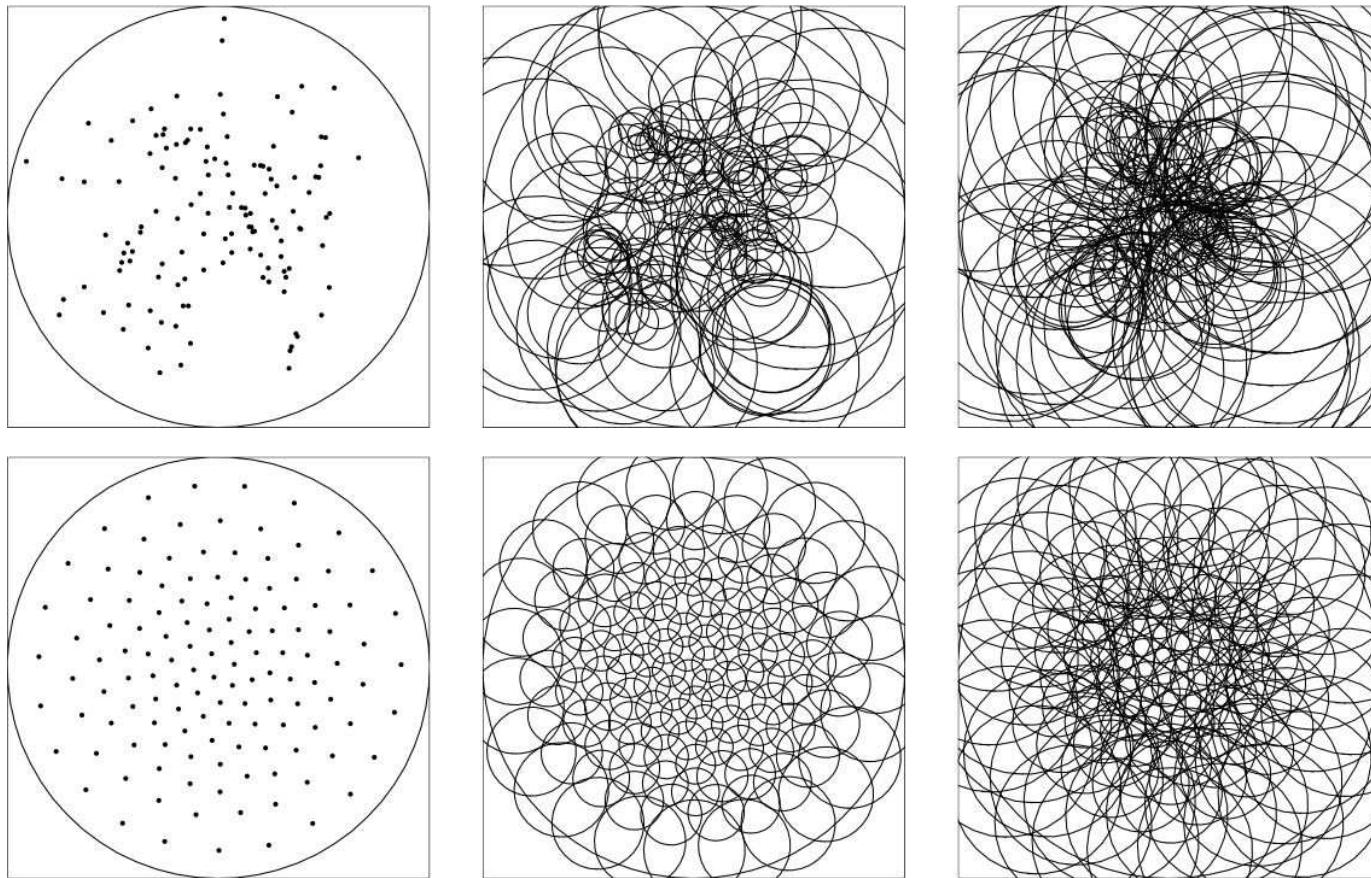
*The sets of 128 points in a square (left) and the associated spherical patches determined by the new algorithm for  $p = 1$  (middle) and  $p = 3$  (right) for the Monte Carlo (top) and CVT (bottom) point selection methods for the density function  $e^{-3(x^2+y^2)}$*



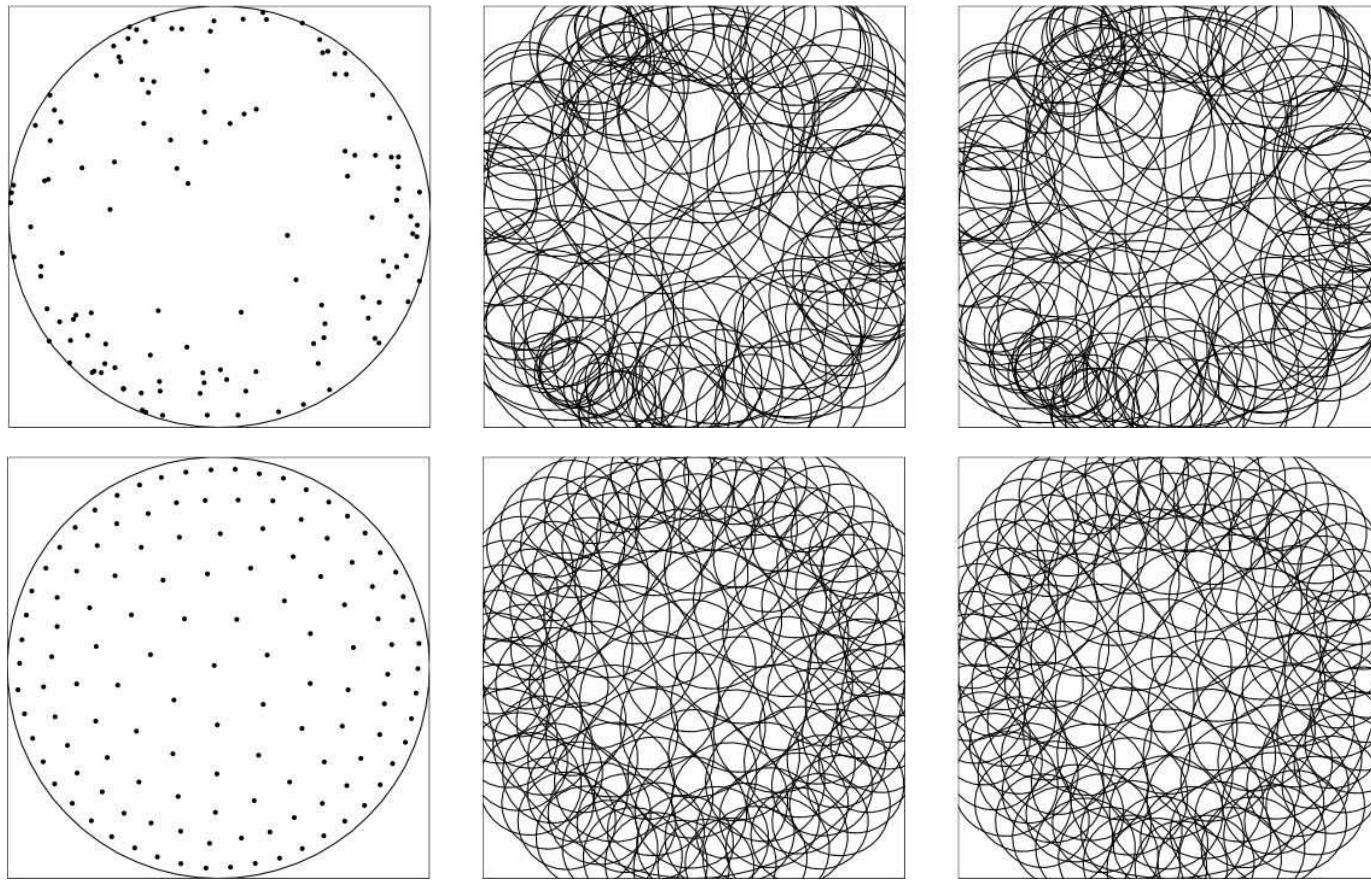
*The sets of 128 points in a square (left) and the associated spherical patches determined by the new algorithm for  $p = 1$  (middle) and  $p = 3$  (right) for the Monte Carlo (top) and CVT (bottom) point selection methods for the density function  $e^{-1.5(2+x+y)}$*



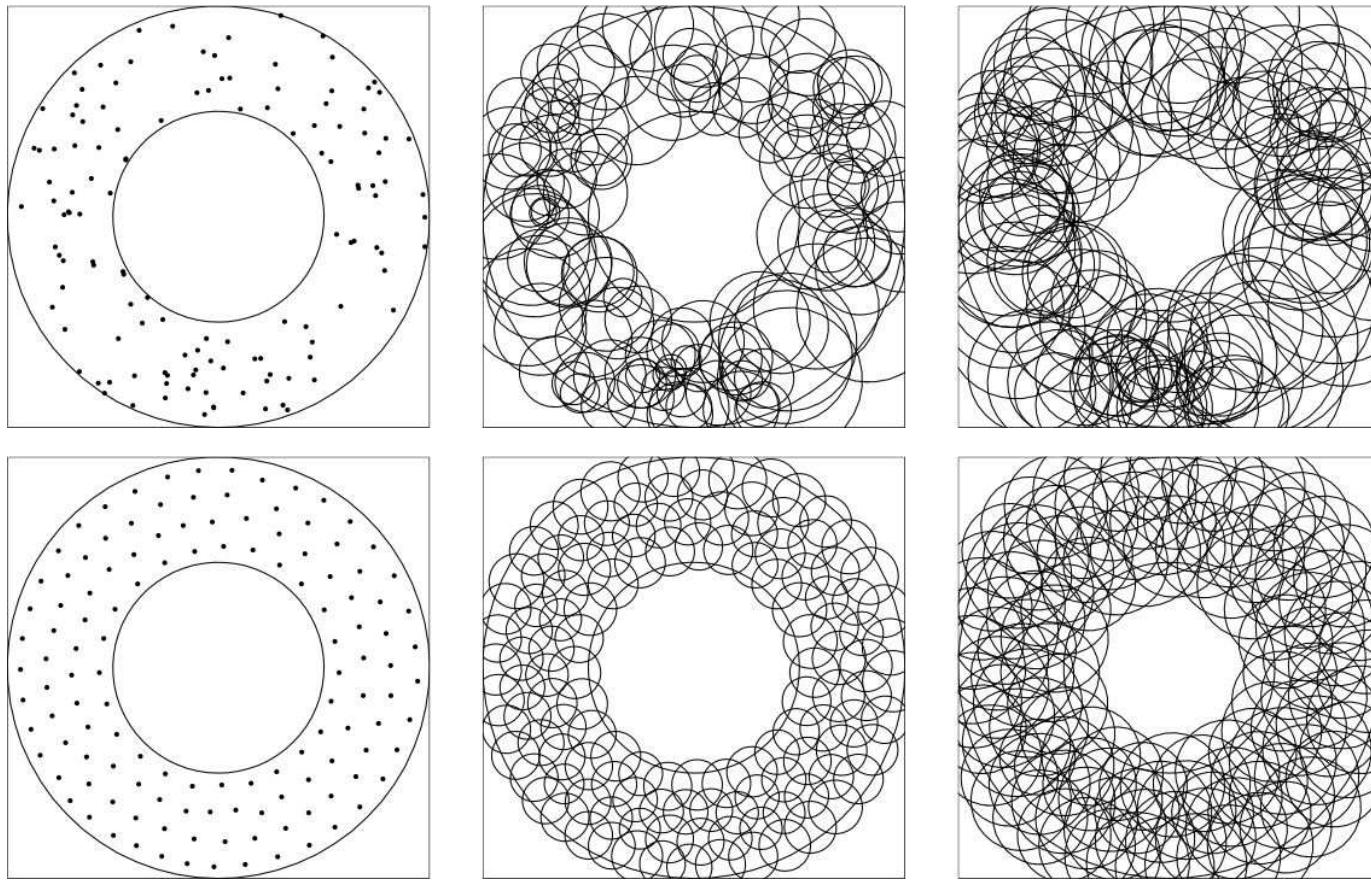
*The sets of 128 points in a square (left) and the associated spherical patches determined by the new algorithm for  $p = 1$  (middle) and  $p = 3$  (right) for the Monte Carlo (top) and CVT (bottom) point selection methods for the density function  $e^{-(1+x)^2}$*



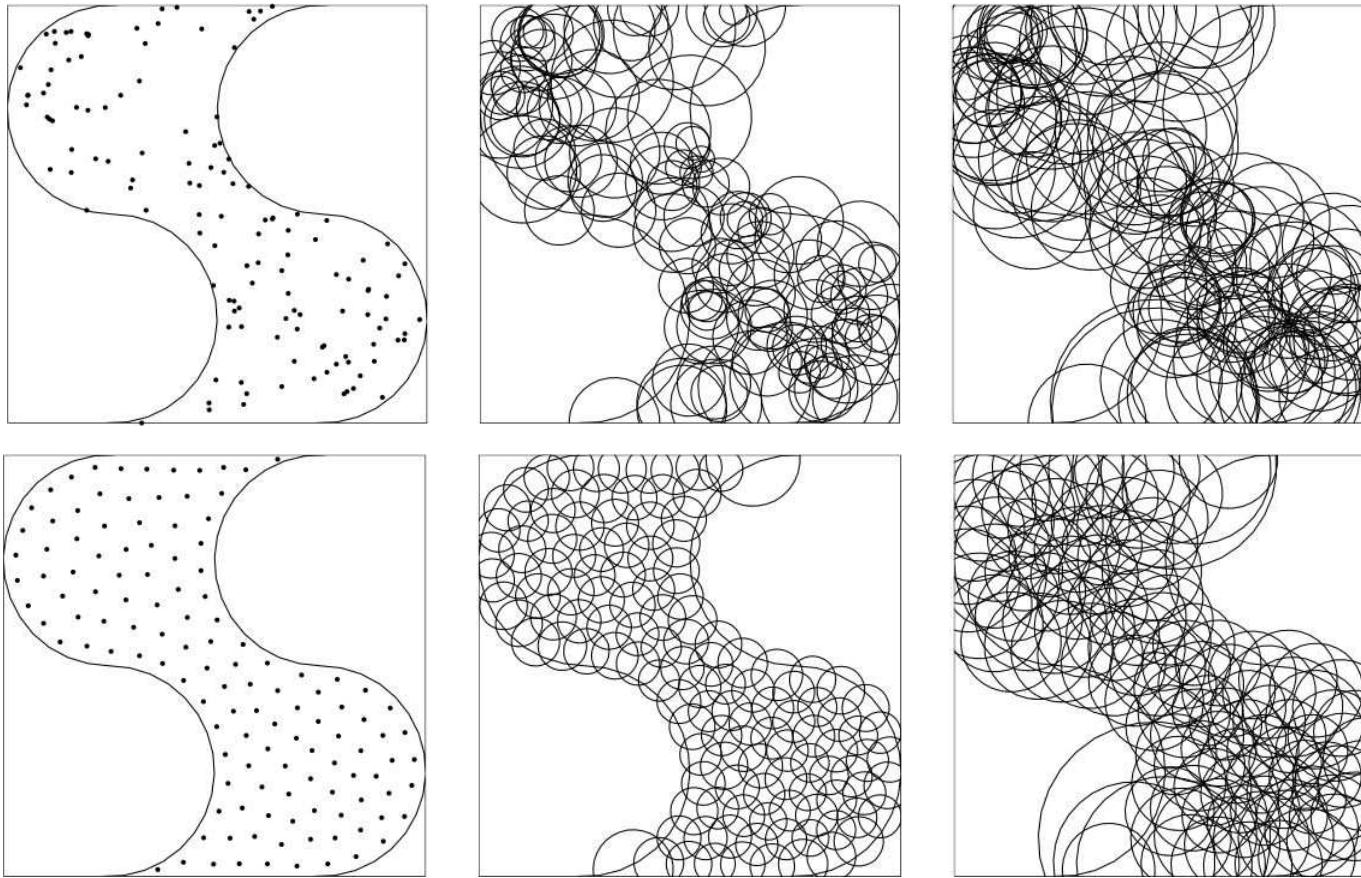
*The sets of 128 points in a circle (left) and the associated spherical patches determined by the new algorithm for  $p = 1$  (middle) and  $p = 3$  (right) for the Monte Carlo (top) and CVT (bottom) point selection methods for the density function  $e^{-4(x^2+y^2)}$*



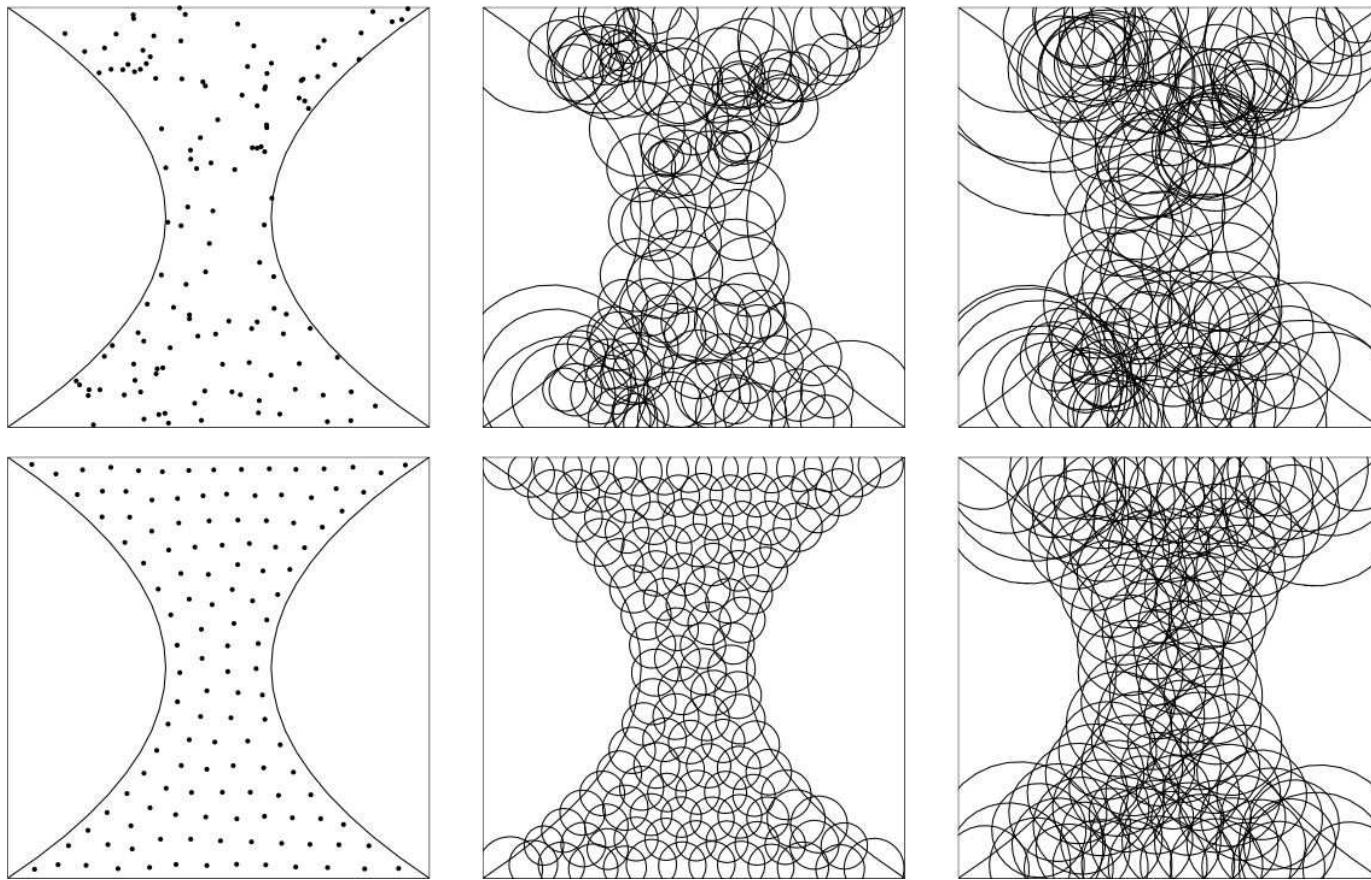
*The sets of 128 points in a circle (left) and the associated spherical patches determined by the new algorithm for  $p = 1$  (middle) and  $p = 3$  (right) for the Monte Carlo (top) and CVT (bottom) point selection methods for the density function  $e^{-3(1-x^2-y^2)}$*



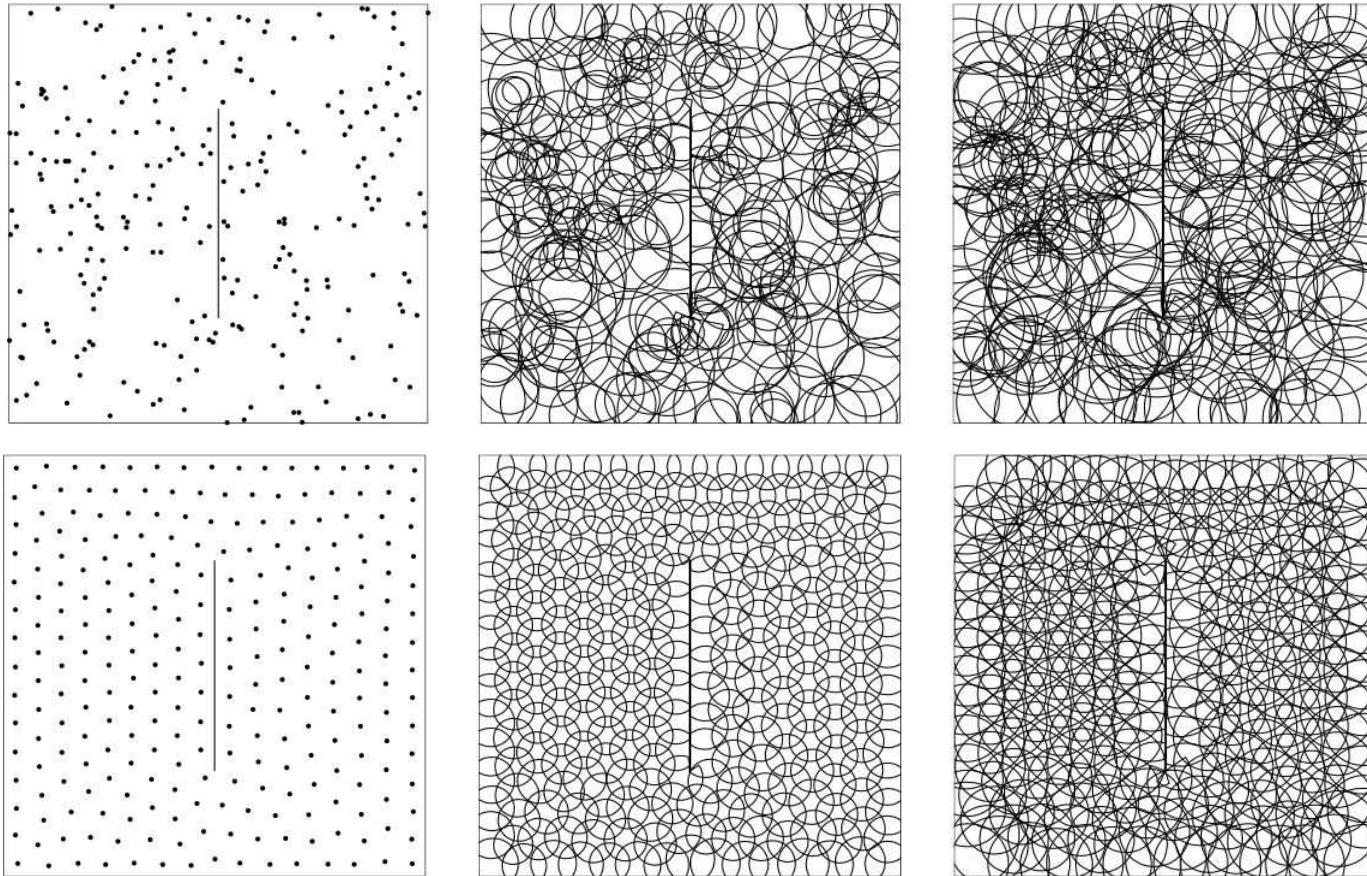
*The sets of 128 points in a non-convex domain (left) and the associated spherical patches determined by the new algorithm for  $p = 1$  (middle) and  $p = 3$  (right) for the Monte Carlo (top) and CVT (bottom) point selection methods for a uniform density function*



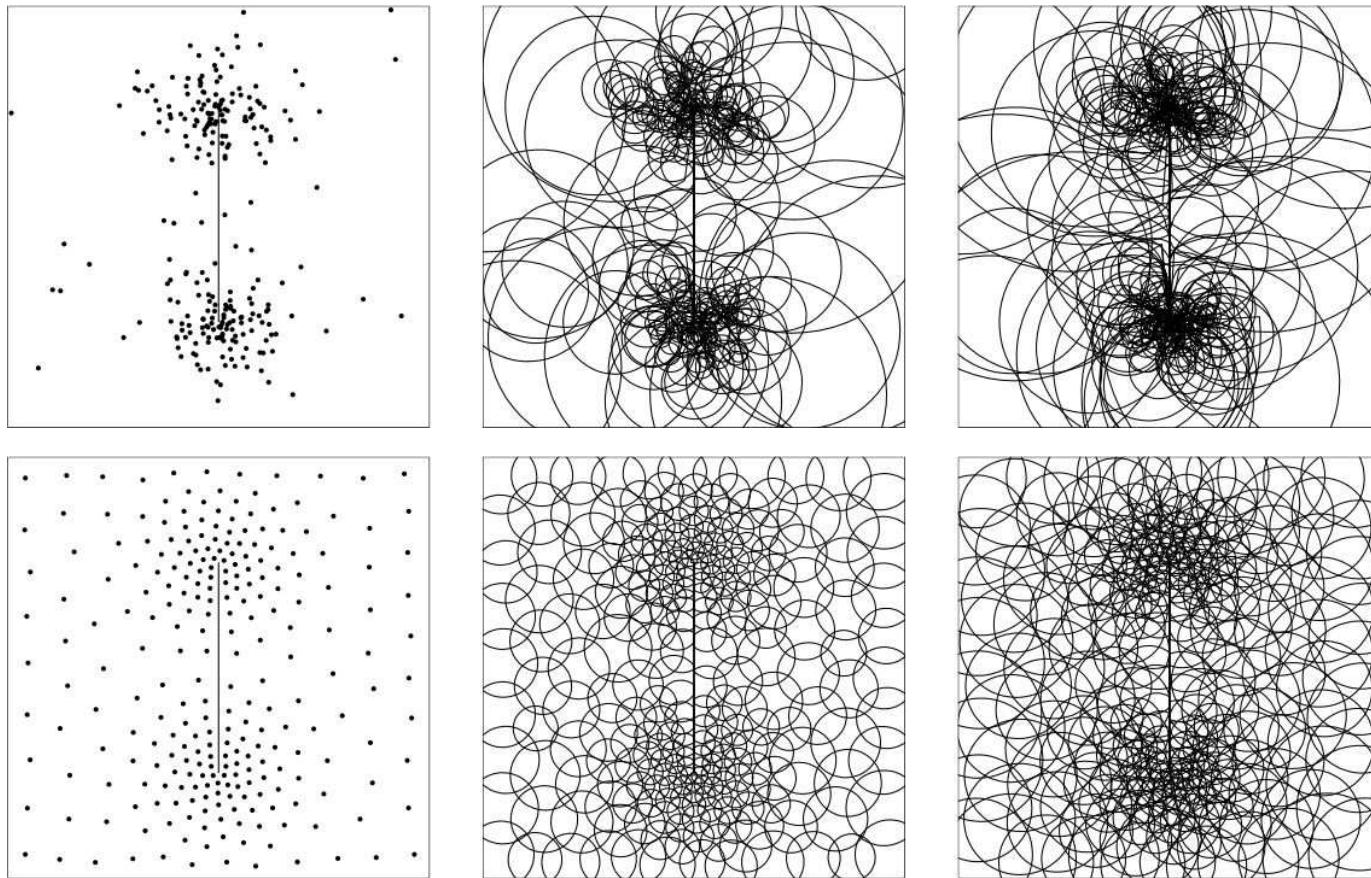
*The sets of 128 points in a non-convex domain (left) and the associated spherical patches determined by the new algorithm for  $p = 1$  (middle) and  $p = 3$  (right) for the Monte Carlo (top) and CVT (bottom) point selection methods for a uniform density function*



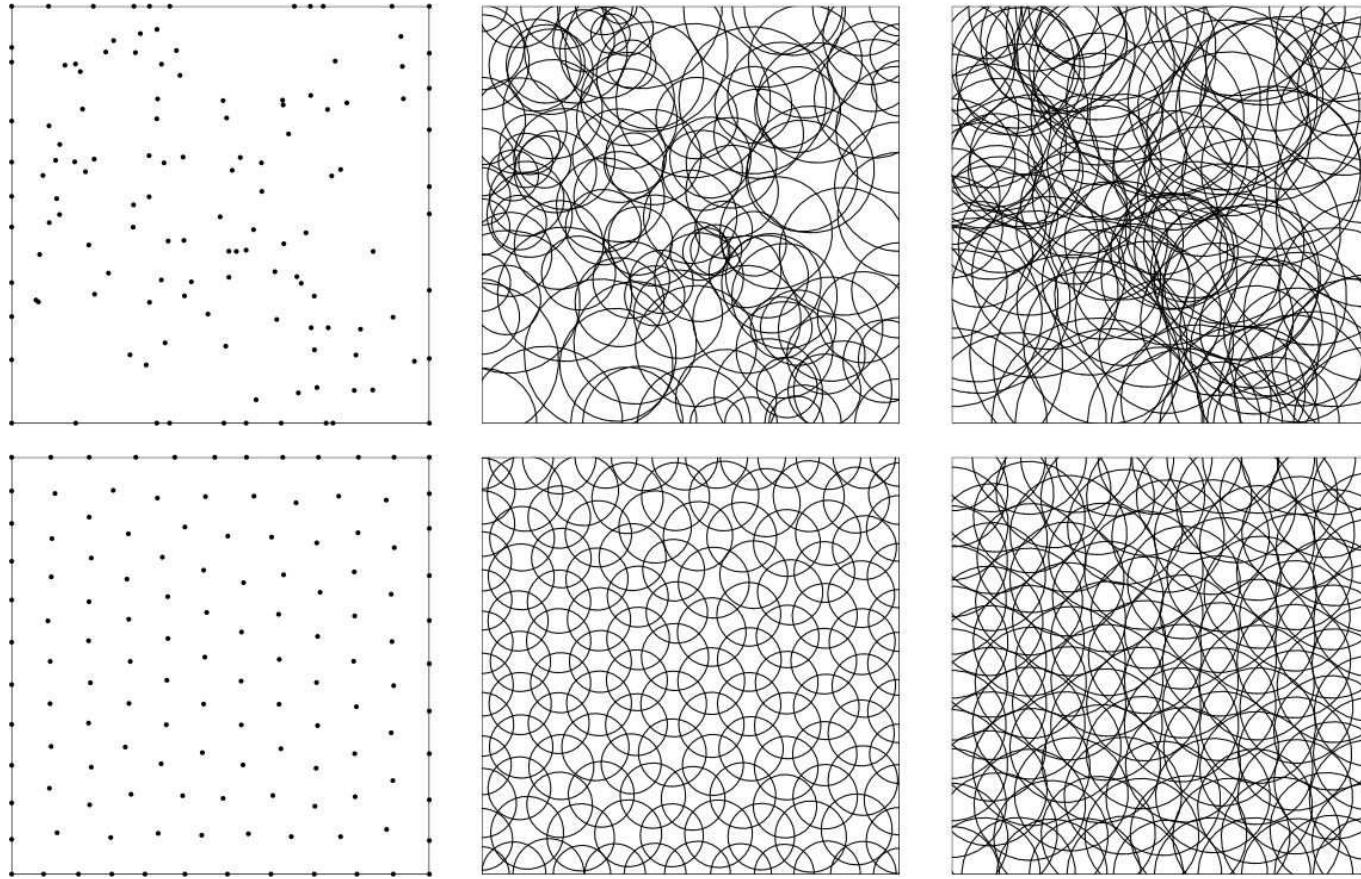
*The sets of 128 points in a non-convex domain (left) and the associated spherical patches determined by the new algorithm for  $p = 1$  (middle) and  $p = 3$  (right) for the Monte Carlo (top) and CVT (bottom) point selection methods for a uniform density function*



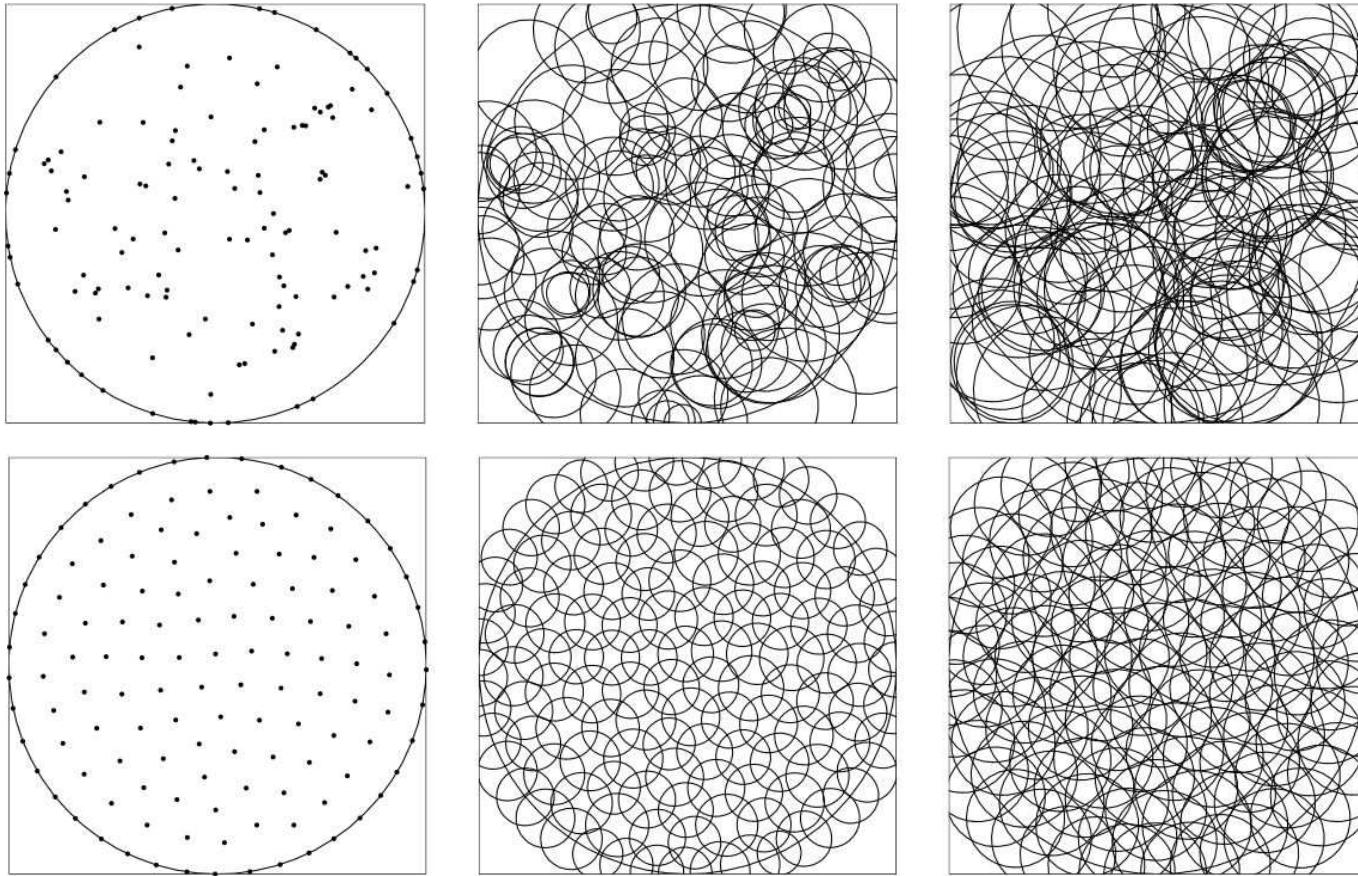
*The sets of 256 points in a domain with a slit (left) and the associated spherical patches determined by the new algorithm for  $p = 1$  (middle) and  $p = 3$  (right) for the Monte Carlo (top) and CVT (bottom) point selection methods for a uniform density function*



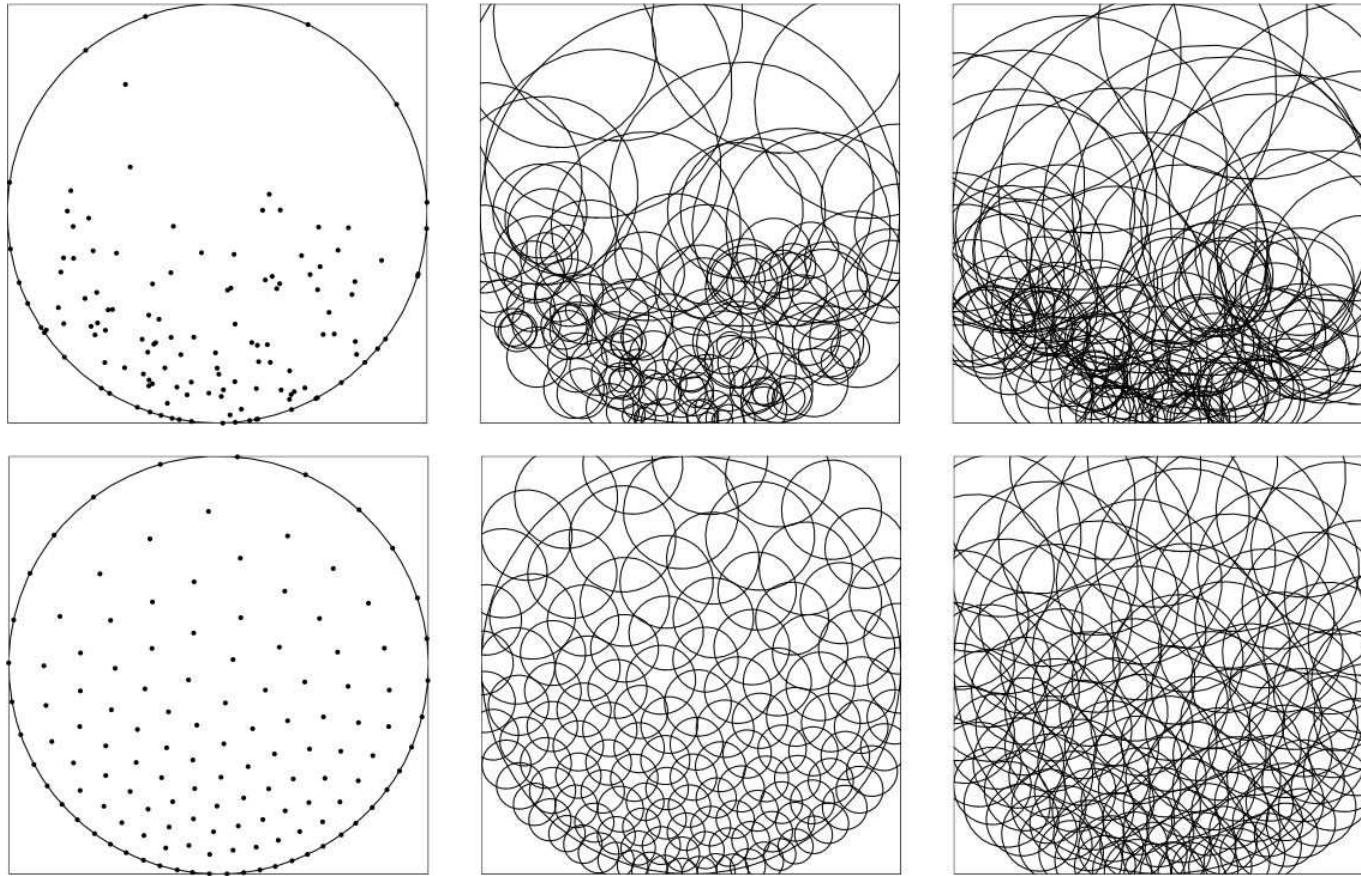
*The sets of 256 points in a domain with a slit (left) and the associated spherical patches determined by the new algorithm for  $p = 1$  (middle) and  $p = 3$  (right) for the Monte Carlo (top) and CVT (bottom) point selection methods for a nonuniform density function*



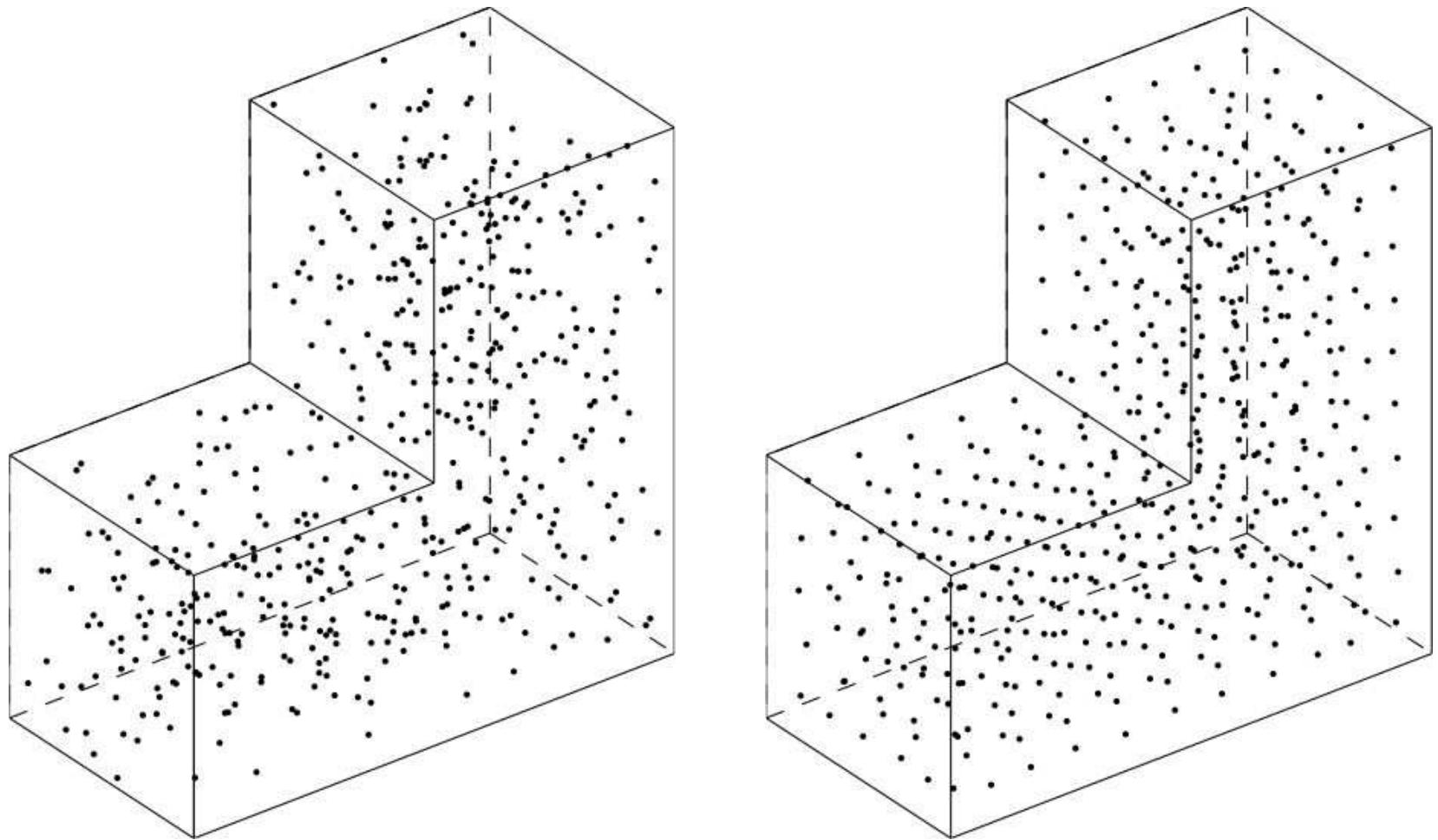
*The sets of 128 points in a square (left) under constraint and the associated spherical patches determined by the new algorithm for  $p = 1$  (middle) and  $p = 3$  (right) for the Monte Carlo (top) and CVT (bottom) point selection methods for a uniform density function*



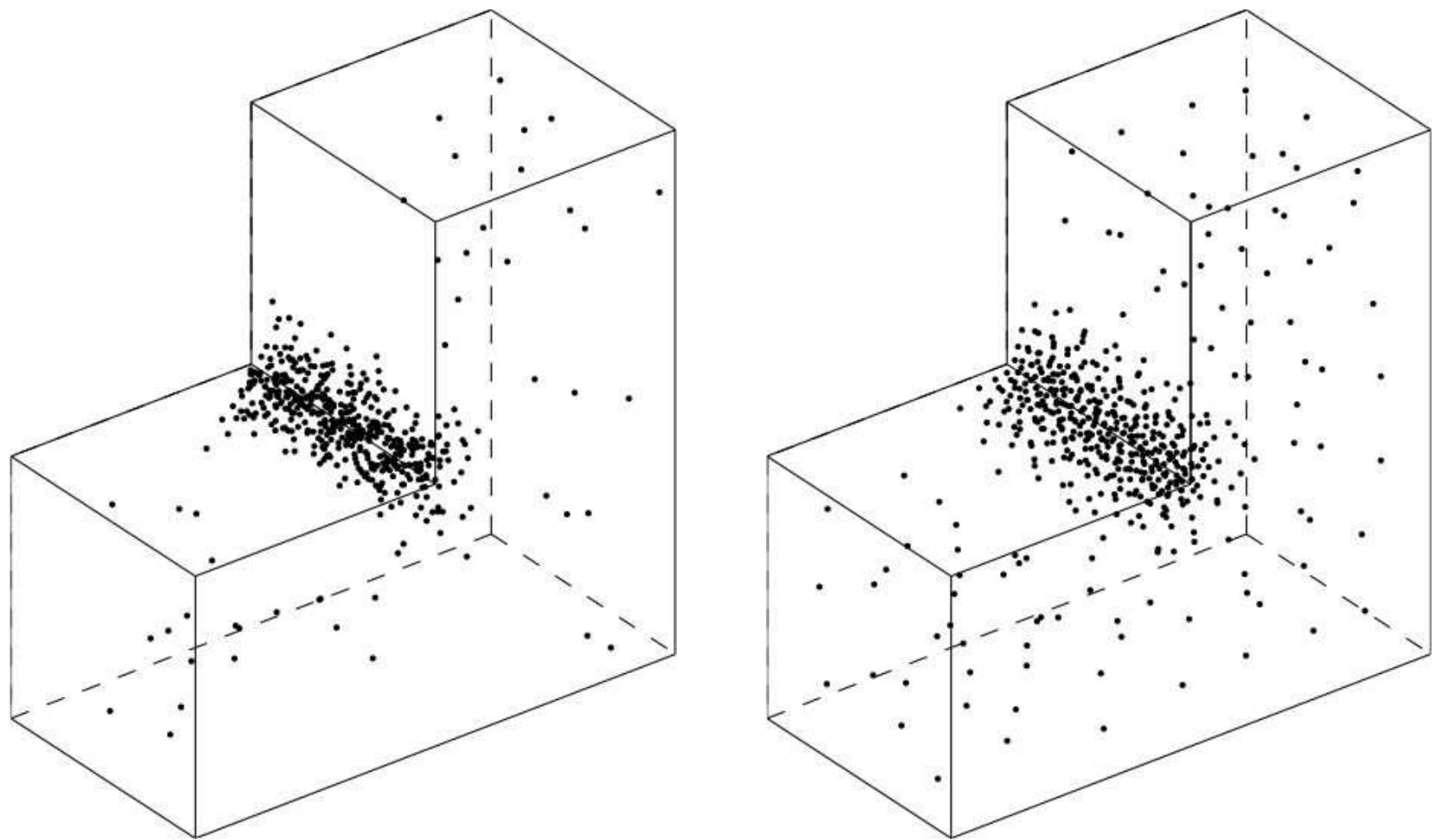
*The sets of 128 points in a circle (left) under constraint and the associated spherical patches determined by the new algorithm for  $p = 1$  (middle) and  $p = 3$  (right) for the Monte Carlo (top) and CVT (bottom) point selection methods for a uniform density function*



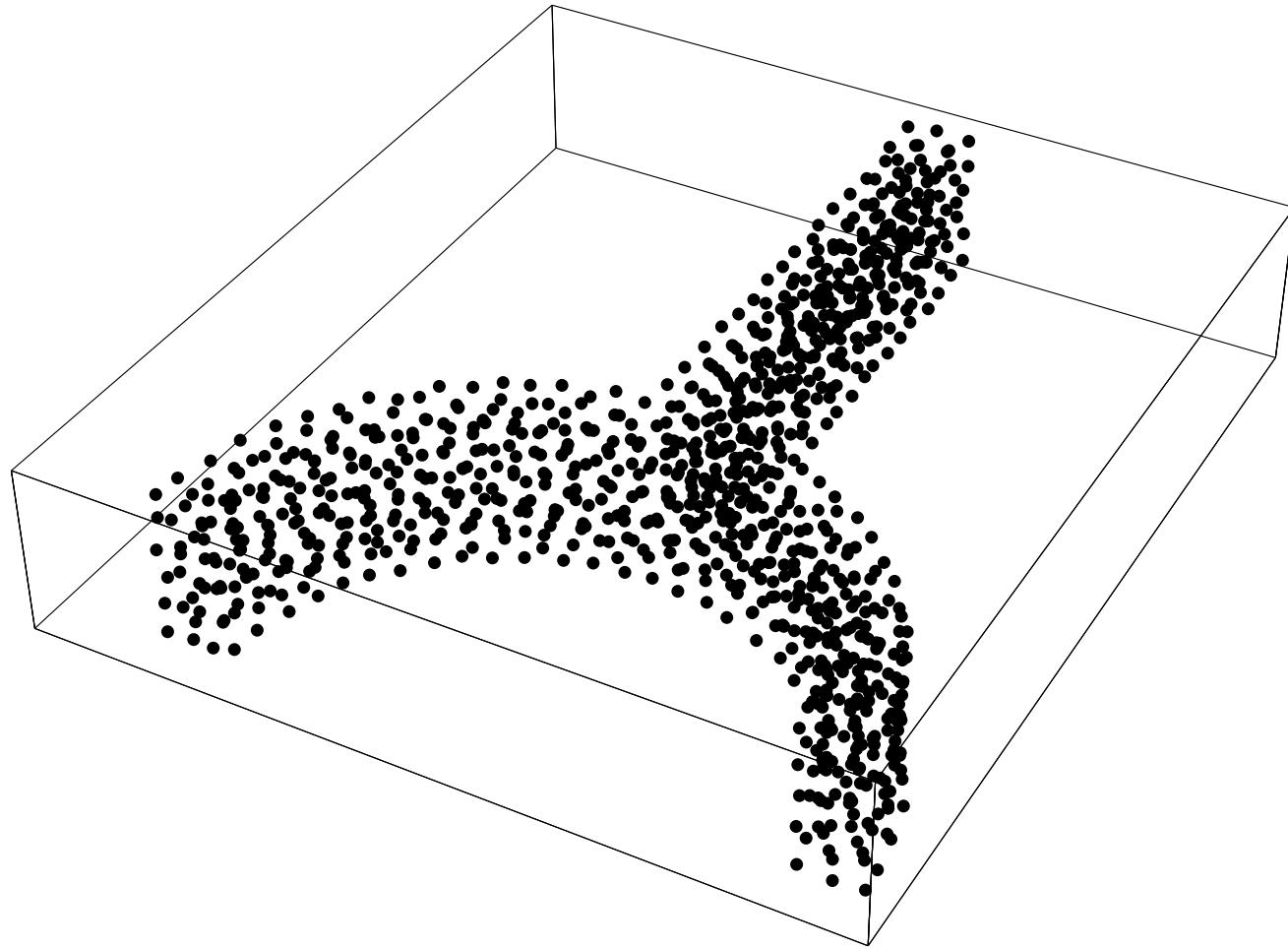
*The sets of 128 points in a circle (left) under constraint and the associated spherical patches determined by the new algorithm for  $p = 1$  (middle) and  $p = 3$  (right) for the Monte Carlo (top) and CVT (bottom) point selection methods for the density function  $\rho(x, y) = e^{-5.3(1+y)}$*



*The sets of 512 points in a three-dimensional non-convex domain for the Monte Carlo (left) and CVT (right) point selection methods for a uniform density function*



*The sets of 512 points in a three-dimensional non-convex domain for the Monte Carlo (left) and CVT (right) point selection methods for a nonuniform density function*



*Uniformly distributed CVT points in a “tuning fork” region*

## REDUCED-ORDER MODELING

- Solutions of (nonlinear) complex systems are expensive with respect to both storage and CPU costs
- As a result, it is difficult if not impossible to deal with a number of situations such as
  - continuation or homotopy methods for computing state solutions
  - parametric studies of state solutions
  - optimization and control problems (multiple state solutions)
  - feedback control settings (real-time state solutions)
- Not surprisingly, a lot of attention has been paid to reducing the costs of the nonlinear state solutions by using reduced-order models for the state
  - these are low-dimensional approximations to the state

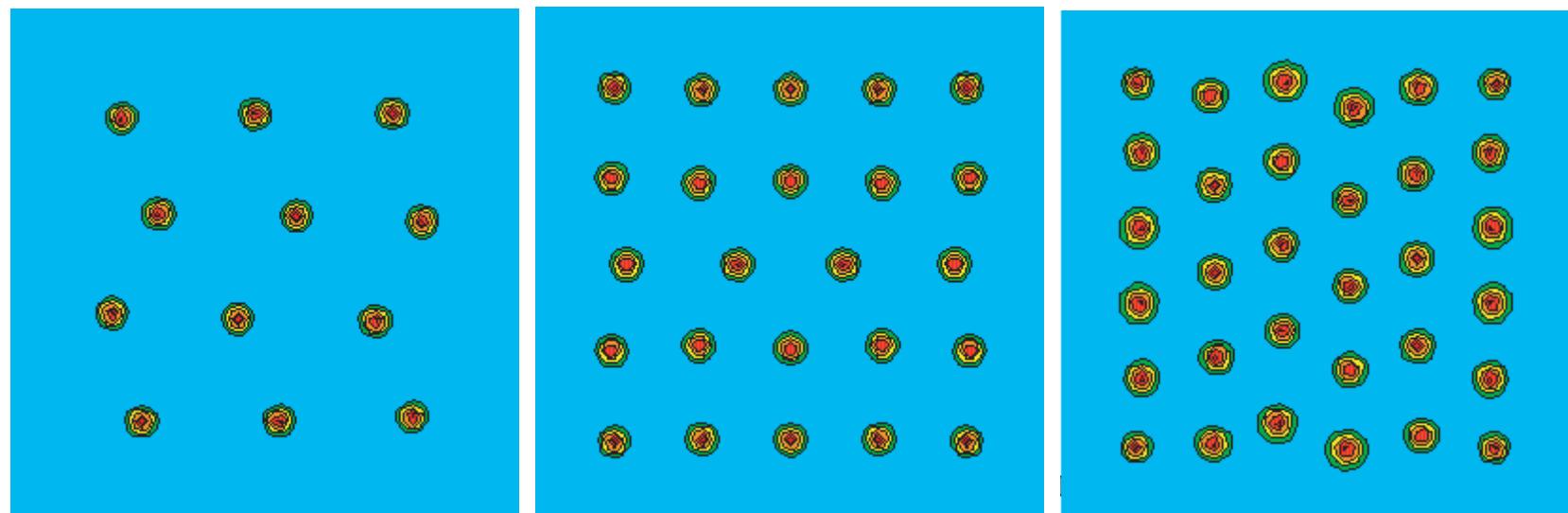
- Reduced-order modeling has been and remains a very active research direction in many seemingly disparate fields, e.g., to name five:
  - **linear algebra**: singular value decomposition (SVD), Hankel SVD
  - **statistics**: Karhunen-Loève analysis, clustering
  - **information science**: representation, interpolation, reconstruction
  - **boundary layers**: e.g., in a fluids setting, replacing the Navier-Stokes equations with the simpler Prandtl boundary layer equations
  - **turbulence modeling**: e.g., in a fluids setting, replacing the Navier-Stokes equations by another complex system, e.g., a  $k-\epsilon$  or LES model, that is “easier” to approximate
- In fact, many of us already do the ultimate in reduced-order modeling
  - any computational approximation to a complex system governed by partial differential equations constitutes an attempt at reduced-order modeling, i.e., **reducing an infinite-dimensional problem to a finite-dimensional one**

## A TYPE OF REDUCED-ORDER MODELING

- For a **state simulation**, a reduced-order method would proceed as follows
  - one chooses a **reduced basis**  $\mathbf{u}_i$ ,  $i = 1, \dots, d$   
 $d$  is hopefully very small compared to the usual number of functions used in a finite element approximation or the number of grid points used in a finite difference approximation
  - next, one seeks an approximation  $\tilde{\mathbf{u}}$  to the state of the form
$$\tilde{\mathbf{u}} = \sum_{i=1}^d c_i \mathbf{u}_i \in V \equiv \text{span}\{\mathbf{u}_1, \dots, \mathbf{u}_d\}$$
  - then, one determines the coefficients  $c_i$ ,  $i = 1, \dots, d$ , by solving the state equations in the set  $V$   
e.g., one could find a Galerkin solution of the state equations in a standard way, using  $V$  for the space of approximations
  - the cost of such a computation would be very small if  $d$  is small (**ignoring**

the cost of the **off-line** determination of the reduced basis  $\{\mathbf{u}_1, \dots, \mathbf{u}_d\}$ )

- Does reduced-order modeling work?
  - it is clear that reduced-order methods should work in an **interpolatory setting**
  - What happens in an **extrapolatory setting** is not so clear



*Superconducting vortices for three different values of the applied magnetic field  
Reduced-order modeling would be difficult to do in this setting*

## SNAPSHOT SETS

- The state of a complex system is determined by **parameters** that appear in the specification of a mathematical model for the system
- Of course, the state of a complex system also depends on the **independent variables** appearing in the model
- Snapshot sets consist of (expensive computational or, in principle, even experimental) state solutions corresponding to several parameter values and/or evaluated at several values of one or more of the dependent variables
  - steady-state solutions corresponding to **several sets of design parameters**
  - a time-dependent state solution for a fixed set of design parameter values **evaluated at several time instants** during the evolution process
  - several state solutions corresponding to different sets of parameter values evaluated at several time instants during the evolution process

- For practical, time-dependent problems, one needs to accumulate hundreds or even thousands of snapshots
  - however, snapshot sets often contain lots of redundant information
  - we will examine two means for **removing the redundant information** so that one obtains reduced-order bases of small dimension
    - POD (proper orthogonal decomposition) ⇐ based on projecting snapshots
    - CVT ⇐ based on clustering snapshots

## POD-BASED REDUCED-ORDER MODELING

- Given  $n$  snapshots  $\tilde{\mathbf{x}}_j \in \mathbb{R}^N$ ,  $j = 1, \dots, n$ , set

$$\mathbf{x}_j = \tilde{\mathbf{x}}_j - \tilde{\boldsymbol{\mu}}, \quad j = 1, \dots, n, \quad \text{where} \quad \tilde{\boldsymbol{\mu}} = \frac{1}{n} \sum_{j=1}^n \tilde{\mathbf{x}}_j$$

the set  $\{\mathbf{x}_j\}_{j=1}^n$  are the modified snapshots

- Let  $A$  denote the snapshot matrix, i.e., the  $N \times n$  matrix whose columns are the modified snapshots  $\mathbf{x}_j$ , i.e.,

$$A = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n) = (\tilde{\mathbf{x}}_1 - \tilde{\boldsymbol{\mu}}, \tilde{\mathbf{x}}_2 - \tilde{\boldsymbol{\mu}}, \dots, \tilde{\mathbf{x}}_n - \tilde{\boldsymbol{\mu}})$$

- A POD basis of dimension  $d$  consists of the first  $d$  left singular vectors of the snapshot matrix  $A$

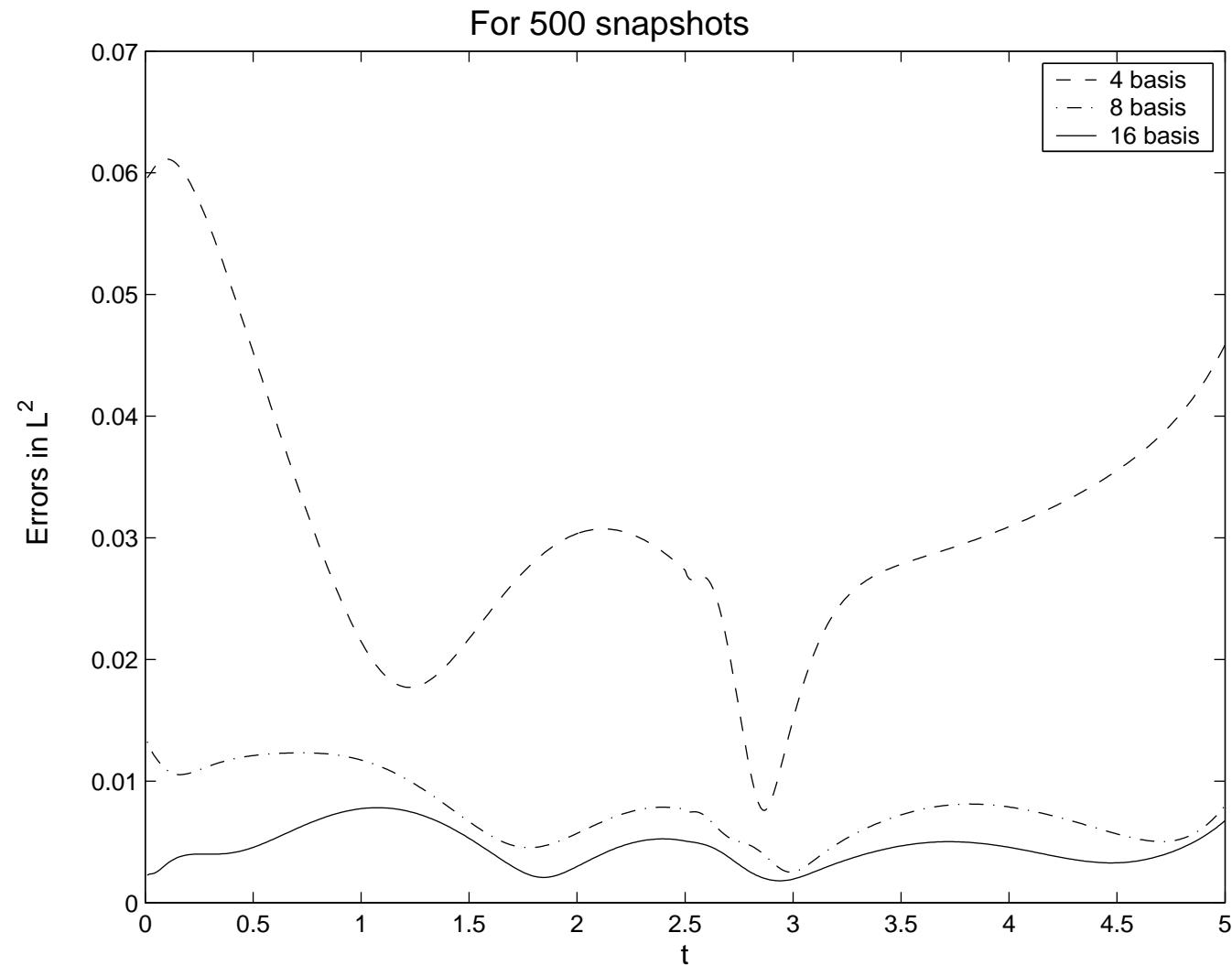
- POD is closely related to the statistical methods known as Karhunen-Loève analysis or the method of empirical orthogonal eigenfunctions or principal component analysis
- The POD basis satisfies an optimality property
- POD is the most popular reduced-order modeling technique
- Many variations on POD have been proposed

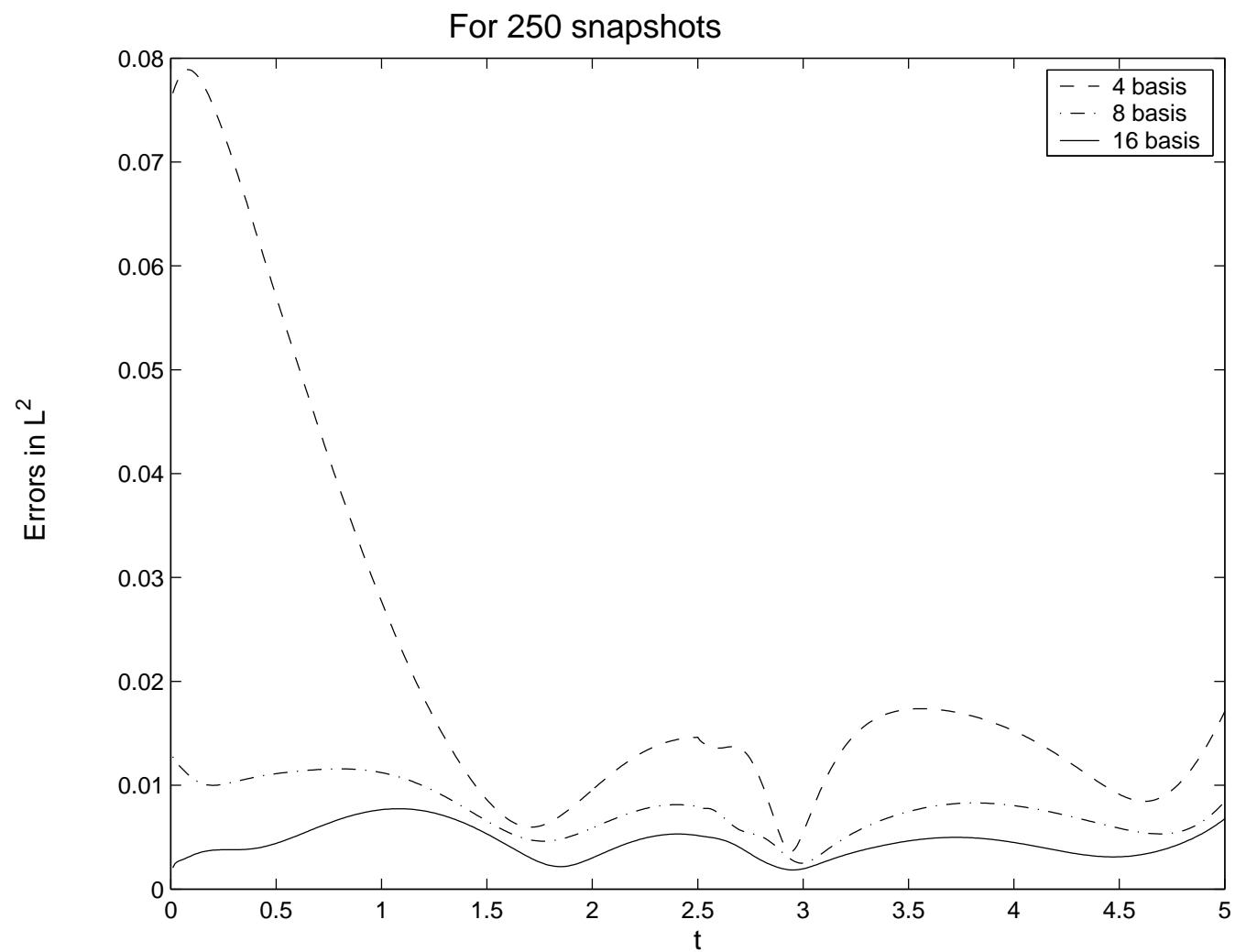
## CVT'S AND REDUCED-ORDER MODELING

- CVT's have been successfully used in data compression; one particular application was to image reconstruction
  - therefore, it is natural to examine CVT's in another data compression setting, namely reduced-order modeling
- The idea, just as it is in the POD setting, is to **extract**, from a given set of (modified) snapshots  $\{\mathbf{x}_j\}_{j=1}^n$  of vectors in  $\mathbb{R}^N$ , **a smaller set of vectors** also belonging to  $\mathbb{R}^N$ 
  - in the POD setting, the reduced set of vectors was the  $d$ -dimensional set of POD vectors  $\{\phi_j\}_{j=1}^d$
  - in the CVT setting, the reduced set of vectors is the  $k$ -dimensional set of vectors  $\{\mathbf{z}_k\}_{k=1}^k$  that are the generators of a centroidal Voronoi tessellation of the set of modified snapshots

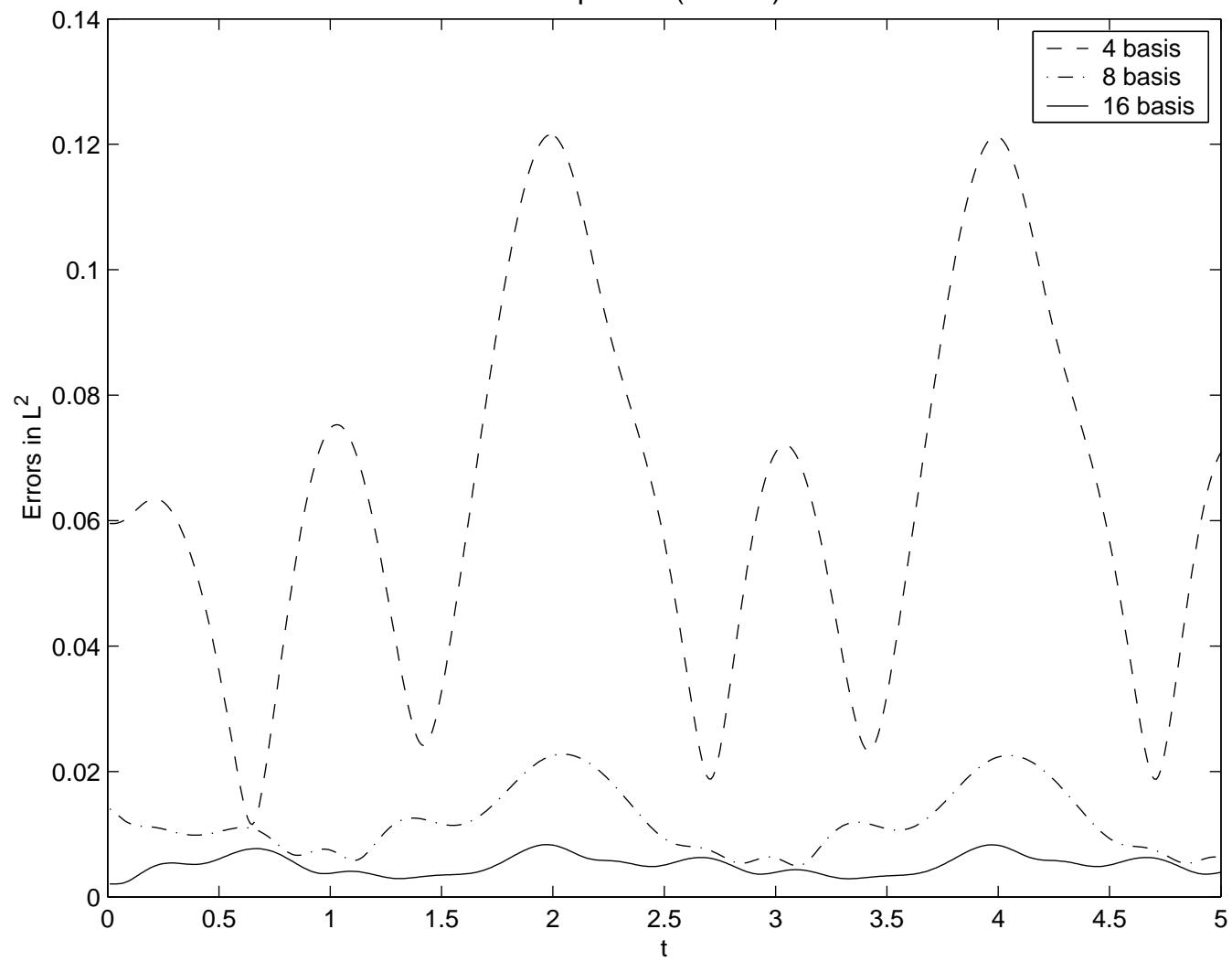
- Just as POD produces an optimal reduced basis in some sense, CVT produces an optimal reduced basis in the sense that the CVT “energy” is minimized
- One can, in principle, determine the dimension  $d$  of an effective POD basis, e.g., using the singular values of the snapshot matrix
  - similarly, using the elbowing effect, one can determine the dimension  $k$  of an effective CVT basis by examining the (computable) CVT “energy”

# COMPUTATIONAL RESULTS FOR CVT MODEL REDUCTION

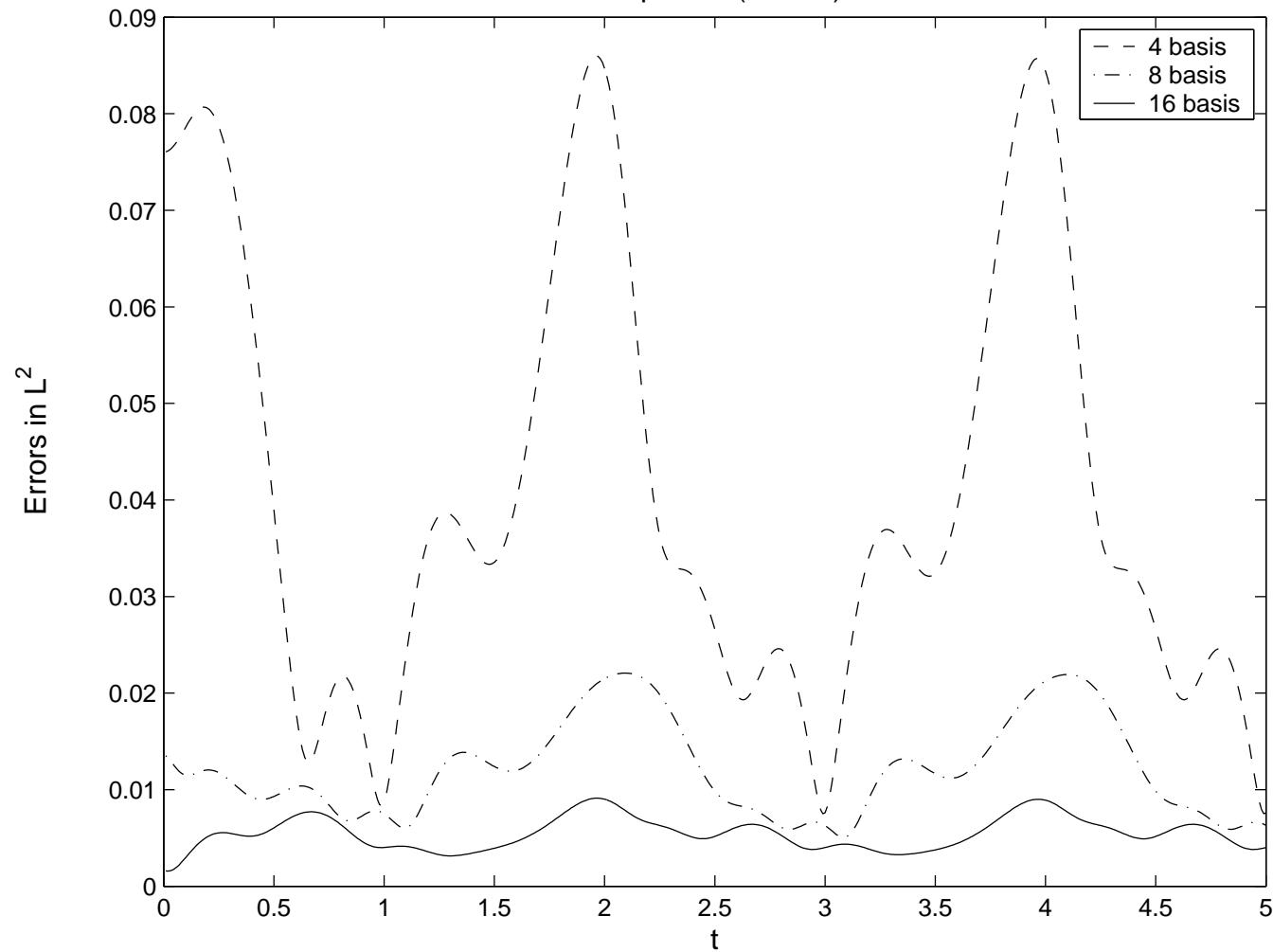




For 500 snapshots (case 2)



For the 250 snapshots (case 2)



## CVT VS. POD

- Question: why should one use CVT instead of POD?
  - although justifications have to be substantiated through analyses and extensive further numerical experiments, one can make some arguments
- CVT naturally introduces the concept of clustering into the construction of the reduced basis
- CVT is “cheaper” than POD
  - POD involves the solution of an  $n \times n$  eigenproblem, where  $n$  is the number of snapshots
  - CVT requires no eigenproblem solution
    - CVT can handle many more snapshots for the same cost
    - adaptively changing the reduced basis is less expensive with CVT
- Another potentially interesting feature of CVT is that it avoids the over-crowding of the reduced basis into a few dominant modes

## CVT COMBINED WITH POD (CVOD)

- We have already mentioned that the concept of centroidal Voronoi tessellations can be extended to more general notions of distance
  - this allows us to combine POD and CVT to (hopefully) take advantage of the best features of both approaches.
- Why should one use CVOD instead of POD or CVT?
  - CVOD offers the possibility of taking advantage of the best features of both POD and CVT
  - CVOD is cheaper than POD since it requires the solution of several smaller eigenproblems instead of one large one

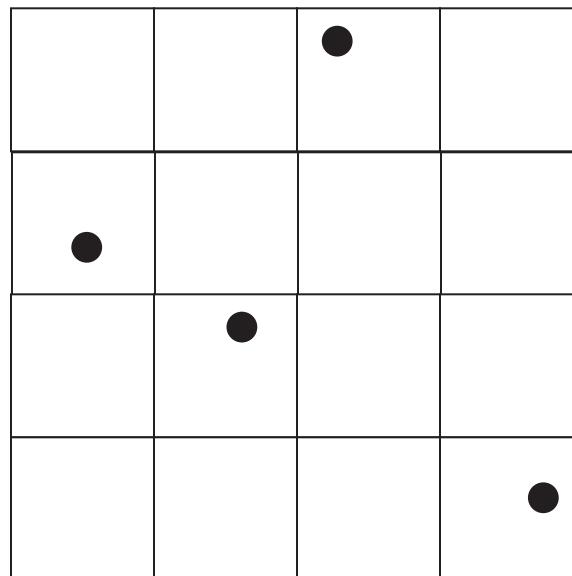
## CVT AND SNAPSHOT GENERATION

- CVT can play another important role in reduced-order modeling, regardless of how the reduced-order basis is determined
- Any reduced-order basis cannot be better than the snapshot set from which it is generated
  - if it ain't in the snapshot set, it ain't in the reduced-order basis
- Thus, the intelligent, effective, and efficient generation of snapshot vectors is crucial to the success of any reduced-order model
  - very little attention has been paid to this aspect of reduced-order modeling
  - ad-hoc techniques are used to generate snapshots
- In a real sense, snapshot generation is an exercise in the design of experiments
  - CVT point sampling in parameter space can be effective in this regard

## IMPROVED HYPERCUBE POINT SAMPLING

---

- A set of  $N$  points in a  $d$ -dimensional hypercube is said to have the [Latin-hypercube sample \(LHS\)](#) property if one subdivides the hypercube into  $N^d$  little cubes, then all  $(d - 1)$ -dimensional “hyperplane slices” of little cubes contain exactly one point

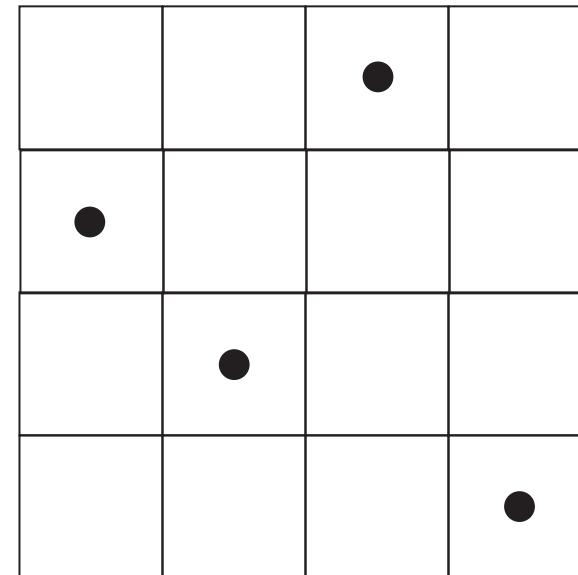
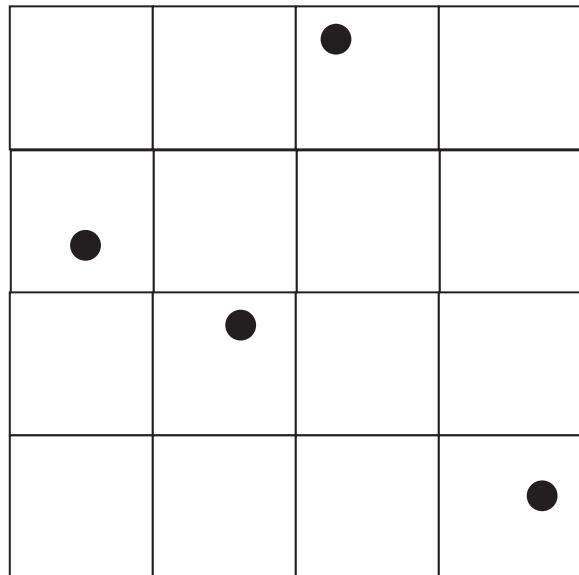


*A 4-point LHS in the square*

- Whenever one constructs an LHS point set, the points may be
  - randomly located within a little cube

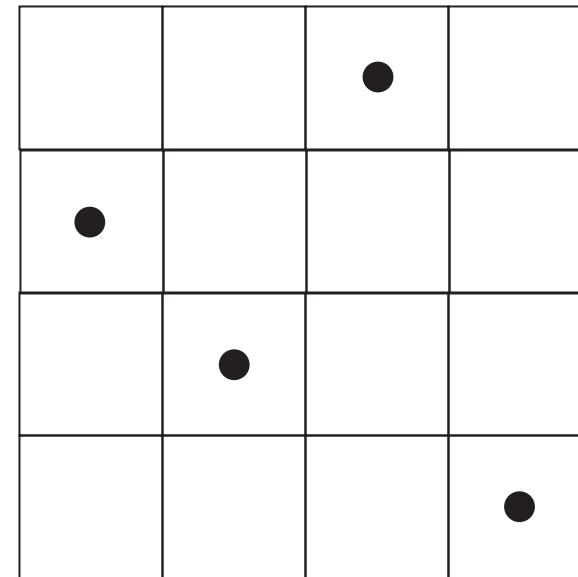
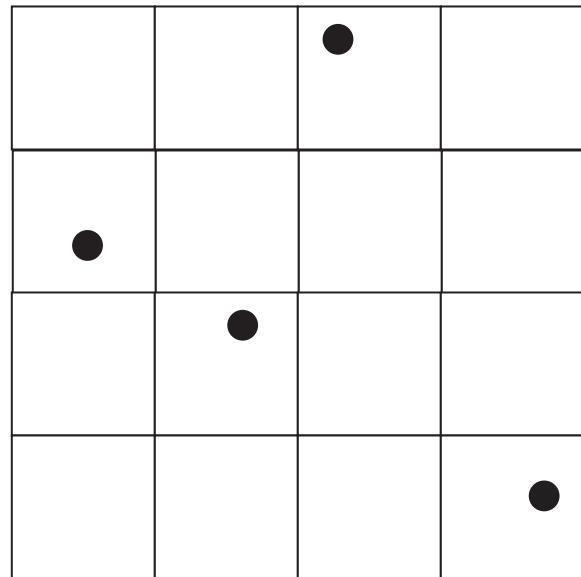
or may be

- located at the center of the little cube



*Random (left) and centered (right) 4-point LHS point sets in the square*

- The most direct way to construct LHS point sets is
  - to choose  $(d - 1)$  random permutations of the numbers 1 through  $N$
  - then place a point randomly or at the center of the  $N$  little cubes determined by the permutations
    - points are numbered lexicographically according to their first coordinate



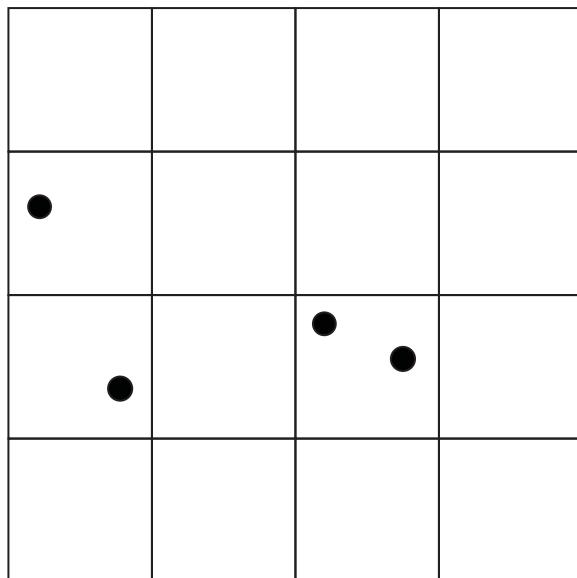
*Random (left) and centered (right) 4-point LHS point sets in the square determined by the permutation  $\{3, 2, 4, 1\}$  for the second coordinate*

- For many applications, the LHS property is very desirable
  - however, LHS point sets constructed in the direct way are often found to have poor “other” properties
- For this reason, there have been several attempts made at “improving” LHS point sampling
  - with the goal of improving other measures of quality for point sets
- Thus, we ask the related questions
  - can any point set be transformed into an LHS point set?
  - what happens to the quality of points sets after they are transformed into LHS point sets?
  - can one obtain improved LHS point sets by transforming other sets into LHS point sets?

## LATINIZATION OF POINT SETS

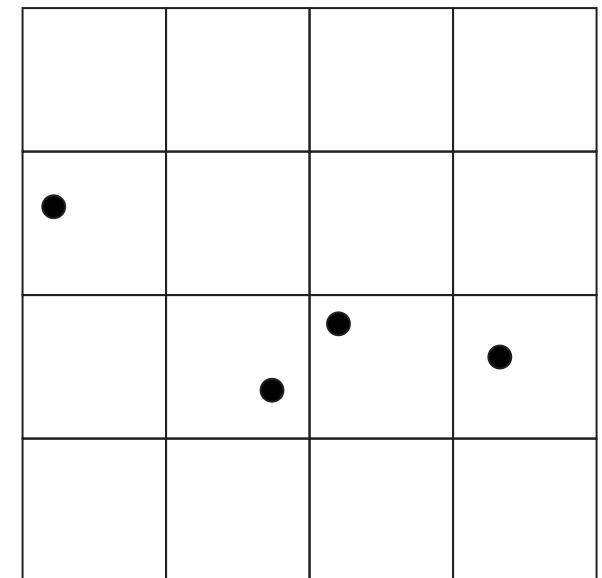
- Any point set can be transformed into an LHS point set through a simple [Latinization](#) process
- The process consists of, sequentially for each coordinate,
  - ordering the points according to their coordinate value
  - shifting the points so that they are in separate “hyperplane slices” for that coordinate

- Starting with a non-LHS point set, first order and shift the points according to the first coordinate
  - the other coordinates of the points remain fixed



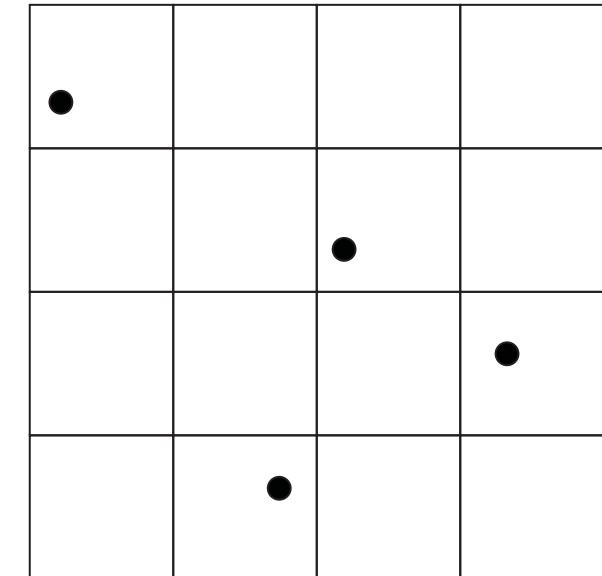
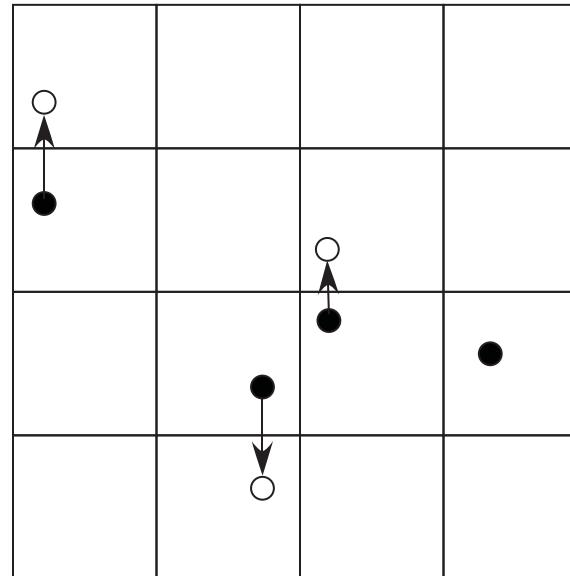
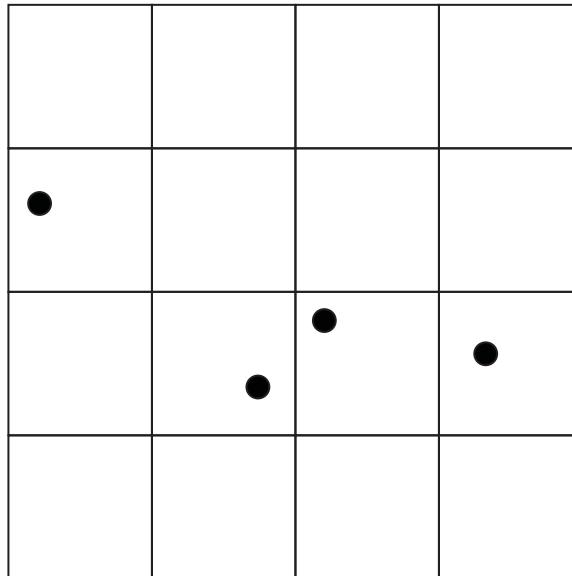
*A non-LHS point set*

*order the x-coordinates  
shift into separate x-slices*



*there is now one  
point in each x-slice*

- Keep on ordering and shifting, one coordinate at a time



*the points after the x-coordinate is taken care of*

*order the y-coordinates shift into separate y-slices*

*there is now one point in each x and y slice*

↑  
*results in an LHS point set*

- What happens to the star discrepancy of a point set after it is Latinized?
  - we can show that under certain hypotheses, the star-discrepancy of the Latinized point set has the same asymptotic bound as that of the original point set

- Specifically, if

if the star discrepancy of the original point set  $\leq f(N)$

and

$$f(N) \geq \frac{C}{N} \text{ for some } C > 0,$$

then

the star discrepancy after Latinization is  $O(f(N))$ ,

where the constant in the big-O notation depends on  $d$

- What happens to the CVT energy of a point set after it is Latinized?
  - we can show that under certain hypotheses, the CVT energy of the Latinized point set has the same asymptotic bound as that of the original point set
- Specifically,
  - if the original point satisfies
    - the volumes  $V_i$  of the Voronoi cells corresponding to the points satisfy, for some  $0 \leq \gamma < 1$
$$\frac{1}{1 + \gamma N} \leq V_i \leq \frac{1}{1 - \gamma N}$$
  - the maximum difference in the coordinates of any point and the points in its Voronoi cell is bounded by  $R$  for some  $R > 0$
  - then, the difference between the CVT energy of the original and Latinized point sets is bounded by

$$C \frac{(R + \gamma)^2}{N}$$

- This result holds for any point set
- The CVT energy of a uniform CVT point set is bounded by  $\frac{R^2}{N}$ 
  - thus, Latinization changes the CVT energy of a CVT point set by the same order as the CVT energy of the original point set

# MEASURES OF POINT SET QUALITY

## The star discrepancy

- To define the star discrepancy  $D^*$  of a point set  $\{\mathbf{z}_i\}_{i=1}^N$  in the unit hypercube  $[0, 1]^d$  in  $d$  dimensions,

- choose a box  $B \subset [0, 1]^d$  having one corner at the origin

- compute the ratio

$$r_B = \frac{\text{number of points from } \{\mathbf{z}_i\}_{i=1}^N \text{ that are in the box } B}{N, \text{ the total number of points in the point set}}$$

- set

$$D^* = \sup_{B \subset [0,1]^d} |\text{volume}(B) - r_B|$$

- The star-discrepancy is of interest because of the Koksma-Hlawka theorem:

$$\left| \int_{[0,1]^d} f(x) dx - \frac{1}{N} \sum_{i=1}^N f(\mathbf{z}_i) \right| \leq D^* V(f)$$

where  $V(f)$  denotes the “variation of  $f$  in the sense of Hardy and Krause”

- This results motivates the popular activity of developing for algorithms that produce low-discrepancy point sets.

- The best known bound for  $D^*$  is given by

$$D^* \leq C_d \frac{\ln(N)^{d-1}}{N}$$

- For large  $d$  and not too large  $N$ , these types of bounds are not very useful
  - we have that  $\ln(N)^{d-1} N^{-1}$  is an increasing function for  $N \leq e^{d-1}$
  - often,  $C_d$  grows super-exponentially in  $d$

## Measures based on the Voronoi tessellation of the points sets

### The point distribution norm

- Given a Voronoi tessellation  $\mathcal{V} = \{\mathbf{z}_i, V_i\}_{i=1}^N$ , let

$$h = \max_{i=1,\dots,N} h_i \quad \text{where} \quad h_i = \max_{\mathbf{y} \in V_i} |\mathbf{z}_i - \mathbf{y}|$$

- Thus,

- $h_i$  gives the maximum distance between the particular generator  $\mathbf{z}_i$  and the points in its associated cell  $V_i$
  - $h$  gives the maximum distance between any generator and the points in its associated cell
- For an ideal tessellation of  $\Omega$  into congruent regular hexagons,  $h_i = h = (\sqrt{12}|\Omega|/9N)^{1/2} \approx 0.6204(|\Omega|/N)^{1/2}$ , where  $|\Omega|$  denotes the area of  $\Omega$ 
  - thus, the smaller  $h$  is, the more uniform is the mesh.

## The regularity measure

- Given a Voronoi mesh  $\mathcal{V} = \{\mathbf{z}_i, V_i\}_{i=1}^N$ , let

$$\chi = \left( \max_{i=1,\dots,N} \chi_i \right) \quad \text{where} \quad \chi_i = \frac{\sqrt{3}h_i}{\gamma_i}$$

- For an ideal uniform hexagonal mesh,  $h_i = \tilde{h}$  and  $\gamma_k = \sqrt{3}\tilde{h}$  so that  $\chi_i = 1$  for all  $i$  and then  $\chi = 1$ 
  - thus, the smaller  $\chi$  is, the more uniform is the mesh.
- In addition, the value of  $\chi$  provides a measure of the mesh regularity, i.e., the **local uniformity** of a mesh
  - if a mesh is locally uniform in the sense that the cells in a neighborhood of any cell are nearly congruent to that cell, then the value of  $\chi$  will again be small

## The second moment trace measure

- Given a Voronoi mesh  $\mathcal{V} = \{\mathbf{z}_i, V_i\}_{i=1}^N$ , let  $t_i$  denote the trace of the second moment tensor (about the region generator)

$$\mathbb{M}_i = \frac{1}{|V_i|} \int_{V_i} (\mathbf{x} - \bar{\mathbf{z}})(\mathbf{x} - \bar{\mathbf{z}}_i)^T d\mathbf{x}$$

associated with each Voronoi region  $V_i$

- Let  $\bar{t} = \frac{1}{N} \sum_{i=1}^N t_i$  denote the average of the traces

- Then, let

$$\tau = \max_{i=1,\dots,n} |t_i - \bar{t}|$$

- For a perfectly uniform point distribution,  $t_1 = t_2 = \dots = t_N = \bar{t}$  so that  $\tau = 0$ 
  - thus, the smaller  $\tau$  is, the more uniform is the mesh

## The second moment determinant measure

- Given a Voronoi mesh  $\mathcal{V} = \{\mathbf{z}_i, V_i\}_{i=1}^N$ , let  $d_i$  denote the determinant of the deviatoric tensor

$$\mathbb{D}_i = \mathbb{M}_i - \frac{1}{N} t_i \mathbb{I}$$

associated with each Voronoi region  $V_i$

- Then, let

$$d = \max_{i=1,\dots,n} |d_i|$$

- For a perfectly uniform point distribution,  $d_1 = d_2 = \dots = d_N = 0$  so that  $d = 0$ 
  - thus, the smaller  $d$  is, the more uniform is the mesh

## COMPUTATIONAL EXPERIMENTS

- Ten methods are used

HAL Halton

HAM Hammersley

CVT centroidal Voronoi tessellation

CVTP periodic CVT

LHS Latin hypercube

IHS improved Latin hypercube

LHAL Latinized Halton

LHAM Latinized Hammersley

LCVT Latinized CVT

LCVTP Latinized CVTP

B. BEACHKOFSKI AND R. GRANDHI, Improved distributed hypercube sampling, AIAA Paper 2002-1274, AIAA, Washington, 2002

- 100 and 1000 points are sampled in 2, 3, and 7-dimensional hypercubes

- The star discrepancy of each point set was approximately determined by the method of E. Thiémard
  - E. THIEMARD, An algorithm to compute bounds for star discrepancy, *J. Complexity* **17** 2001, 850-868
    - the results in 7 dimensions are less accurate than the corresponding results in 2 and 3 dimensions
- Four Voronoi-based measures and the CVT energy were determined using intense sampling to evaluate integrals
  - again, the results in 7 dimensions are less accurate than the corresponding results in 2 and 3 dimensions

Measure	CVT	LCVT	HAL	LHAL	HAM	LHAM	LHS	IHS	CVTP	LCVTP
$D^* \times 10^2$	8.89	2.74	5.05	2.91	3.80	2.78	6.06	3.71	5.86	3.65
$h$	.072	.104	.125	.126	.140	.134	.165	.118	.109	.108
$\chi$	1.56	7.19	5.18	5.87	3.91	4.19	15.7	4.51	2.17	2.46
$\tau \times 10^3$	.31	.59	1.64	1.18	1.20	1.13	2.83	1.14	.81	.78
$\gamma \times 10^7$	.26	2.50	6.63	6.22	4.79	4.75	22.3	4.45	1.10	1.02
$\mathcal{E} \times 10^3$	1.64	1.84	2.22	2.25	1.96	1.96	3.03	1.97	1.73	1.81

100 points in 2D

Measure	CVT	LCVT	HAL	LHAL	HAM	LHAM	LHS	IHS	CVTP	LCVTP
$D^* \times 10^3$	31.1	1.5	7.29	5.64	4.81	3.71	19.2	8.87	16.3	8.42
$h \times 10^2$	2.29	3.10	3.60	3.69	4.08	4.07	5.93	3.72	3.34	3.35
$\chi$	1.52	3.59	7.31	7.82	11.4	13.0	45.1	4.82	2.42	2.35
$\tau \times 10^4$	.63	.67	1.21	1.18	1.34	1.35	4.06	1.28	.97	1.12
$\gamma \times 10^9$	1.14	2.18	9.56	11.1	3.98	3.71	29.9	5.06	3.25	3.68
$\mathcal{E} \times 10^3$	.163	.171	.221	.221	.183	.184	.315	.182	.168	1.69

1000 points in 2D

Measure	CVT	LCVT	HAL	LHAL	HAM	LHAM	LHS	IHS	CVTP	LCVTP
$D^* \times 10^2$	24.3	6.95	6.76	5.89	6.43	4.98	9.18	6.83	1.5	6.02
$h$	.200	.265	.312	.321	.269	.268	.350	.281	.281	.276
$\chi$	1.94	15.2	9.64	9.69	8.00	7.33	13.7	5.29	2.53	2.85
$\tau \times 10^3$	1.40	4.58	5.57	5.37	6.11	5.24	1.6	5.57	4.67	4.16
$\gamma \times 10^8$	.19	1.24	3.88	3.98	1.19	1.49	11.7	4.31	1.70	3.13
$\mathcal{E} \times 10^2$	1.14	1.37	1.55	1.55	1.41	1.41	1.83	1.41	1.31	1.31

100 points in 3D

Measure	CVT	LCVT	HAL	LHAL	HAM	LHAM	LHS	IHS	CVTP	LCVTP
$D^* \times 10^2$	11.8	1.97	1.64	1.48	1.33	1.17	3.28	2.00	2.53	1.83
$h$	.082	.114	.136	.136	.130	.129	.167	.129	.116	.123
$\chi$	1.83	4.74	1.4	1.3	6.43	6.52	31.8	6.19	2.32	2.36
$\tau \times 10^3$	.63	1.43	1.49	1.63	1.28	1.24	3.00	1.63	1.83	2.02
$\gamma \times 10^{10}$	.38	6.43	5.22	6.93	2.25	2.99	17.0	3.59	2.99	2.60
$\mathcal{E} \times 10^3$	2.42	2.63	3.13	3.14	2.84	2.84	3.70	2.73	2.60	2.60

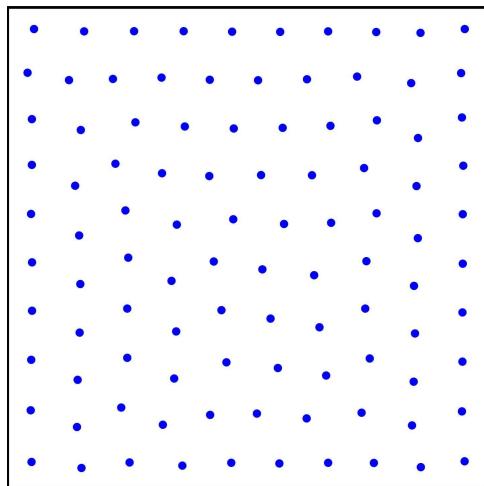
1000 points in 3D

Measure	CVT	LCVT	HAL	LHAL	HAM	LHAM	LHS	IHS	CVTP	LCVTP
$D^*$	.870	.250	.248	.203	.237	.200	.228	.234	.222	.210
$h$	.718	.817	.956	.919	.863	.863	.908	.904	.964	.948
$\chi$	2.63	5.18	4.05	4.03	4.69	4.73	6.56	4.05	3.26	3.23
$\tau \times 10^2$	.91	2.94	3.53	3.43	2.94	3.14	5.24	5.46	5.25	4.65
$\gamma \times 10^{14}$	.000108	1.79	6.15	6.36	1.8	11.3	14.3	17.8	2.26	1.89
$\mathcal{E}$	.156	.211	.212	.212	.206	.206	.215	.215	.203	.203

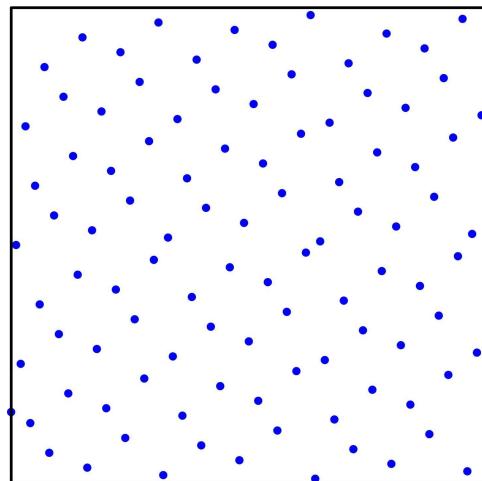
100 points in 7D

Measure	CVT	LCVT	HAL	LHAL	HAM	LHAM	LHS	IHS	CVTP	LCVTP
$D^*$	.741	.173	.132	.128	.125	.120	.139	.139	.137	.133
$h$	.552	.582	.600	.600	.587	.585	.622	.632	.569	.573
$\chi$	2.80	4.41	5.85	5.86	4.36	4.34	7.55	3.78	2.77	2.74
$\tau \times 10^2$	1.23	2.64	2.84	2.70	4.69	4.69	3.16	3.39	3.03	2.68
$\gamma \times 10^{15}$	.012	.96	4.80	4.94	3.00	2.99	2.3	6.53	3.19	3.44
$\mathcal{E} \times 10^2$	7.86	9.50	9.80	9.81	9.83	9.83	1.1	9.87	9.38	9.36

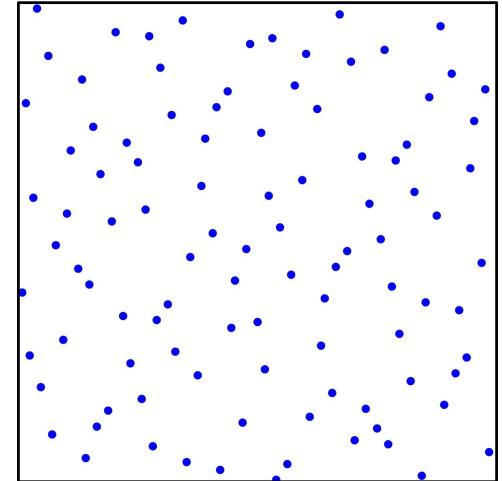
1000 points in 7D



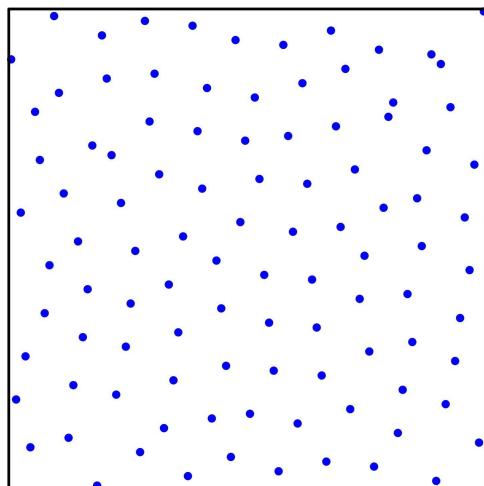
CVT



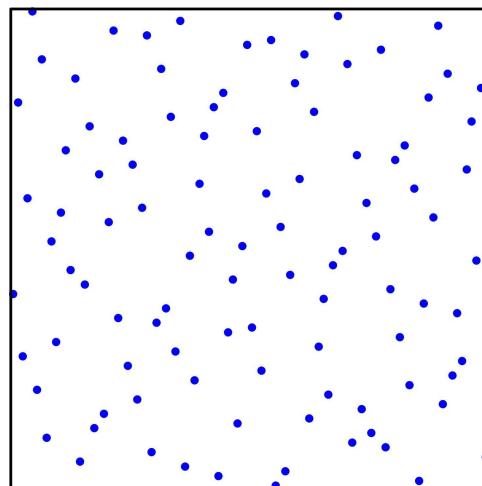
Halton



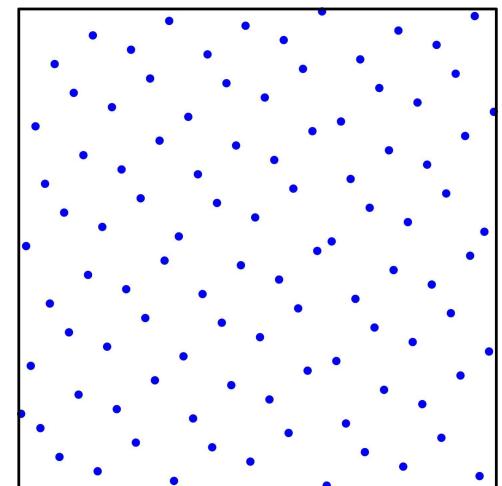
Hammersley



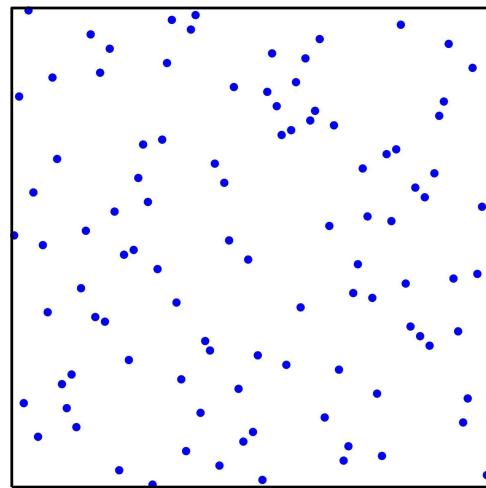
Latinized CVT



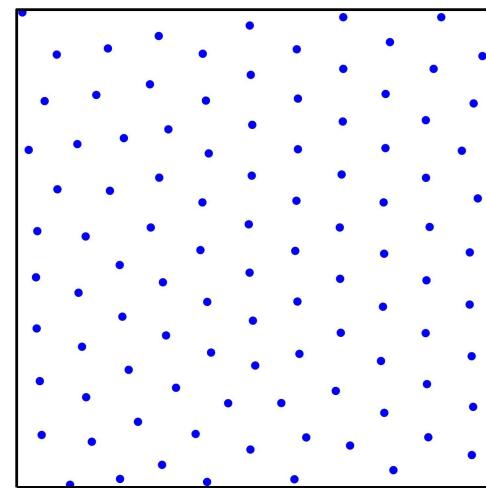
Latinized Halton



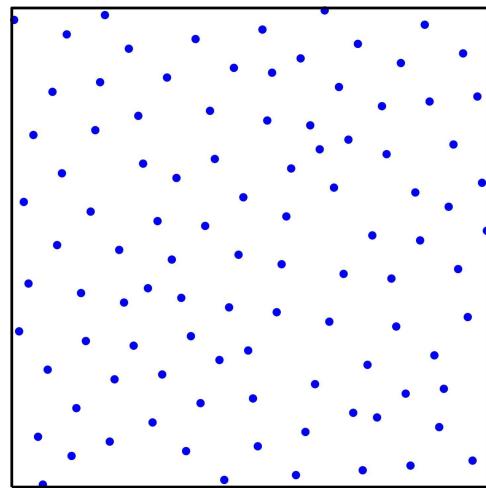
Latinized Hammersley



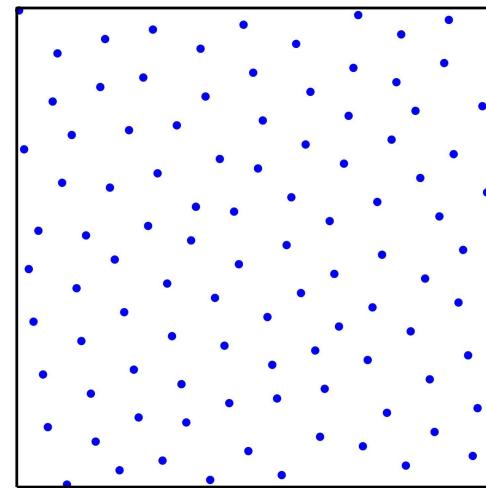
LHS



CVTP



IHS



Latinized CVTP

## INFERENCES DRAWN FROM THE EXPERIMENTS

- Of course, we present a very limited number of test runs so that drawing definitive conclusions with regard to the relative merits of the different point sets is not possible
  - however, the inferences drawn from the results presented are also consistent with the results of other tests we have performed

- Inferences drawn from the star discrepancies
  - Latinization reduces the star discrepancy of every CVT, Halton, and Hammersley point set
  - Latinized Hammersley point sets consistently have the smallest star discrepancy
  - CVT point sets have the largest star discrepancy
    - CVTP point sets have lower star discrepancies than do CVT point sets
  - Latinization of CVT point sets greatly reduces the star discrepancy to a value comparable to that for some of the other point sets
  - Latinized CVTP point sets have lower values of the star discrepancy than do the LHS and IHS points sets
    - in this respect, they begin to rival Halton and Hammersley

- Inferences drawn from the four Voronoi diagram-based measures
  - CVT point sets are consistently the best
  - LHS point sets are consistently the worst
  - IHS point sets are much better than LHS point sets
  - Latinization raises the measures for the CVT point sets, but LCVT point sets are still consistently better than any of the other points sets
  - Latinization mostly but not always raises these measures for the Halton and Hammersley point sets
  - for the most part, CVTP and LCVTP points sets are superior to Halton, Hammersley, LHS, and IHS and are only outdone by the CVT point sets

- Inferences drawn from the CVT energies

- naturally, CVT point sets have the lowest value of the CVT energy since they are, by design, minimizers of that energy
- a Latinized CVT point set has higher CVT energy than that of its parent CVT point set, but generally lower energy than that of the other point sets
- Latinization seems to have little effect on the CVT energy of Halton and Hammersley point sets

- **Summary**

- if the star discrepancy measure is of most importance, then Latinized Hammersley point sets seem to be best
- if the four Voronoi diagram-based measures are of most importance, then CVT point sets seem to be best
- if both measures are of interest, Latinized CVT and CVTP point sets seem to provide the best compromise
- LHS and IHS point sets are not competitive in either category

## CONCLUSIONS

- CVT and periodic CVT point sampling seems to be useful in some settings
- The Latinization of point sets such as Hammersley seems to produce very much improved LHS point sets
- Clearly, more tests and theoretical studies need to be done
  - especially in higher dimensions and for larger point sets

## FUTURE OF CVT

- Improved construction algorithms
- More theory
- Further development of existing applications
- New applications