PARALLEL ALGORITHM FOR SPHERICAL DELAUNAY TRIANGULATIONS AND SPHERICAL CENTROIDAL VORONOI TESSELLATIONS

Max Gunzburger

Department of Scientific Computing, Florida State University

Workshop on Weather and Climate Prediction on
Next Generation Supercomputers: Numerical and Computational Aspects

Exeter, October 22–25, 2012

Joint work with: Doug Jacobsen and Todd Ringler of the Los Alamos National Laboratory and John Burkardt and Janet Peterson of Florida State University

Supported by the US Department of Energy

BACKGROUND

- Voronoi diagrams and their dual Delaunay tessellations have become, in many settings, a natural choice for spatial gridding due to their ability to handle arbitrary boundaries and refinement well
  - however, creating such meshes can be time consuming, especially for high-quality, high-resolution grids

- Several groups have worked on creating parallel divide-and-conquer algorithms for the construction of Delaunay triangulations
  - the algorithms that do exist for the parallel construction of Delaunay triangulations are mostly for two-dimensional planar regions

- Unstructured meshes for spatial discretizations on the sphere are now becoming somewhat popular in climate modeling
  - there is a need for fast algorithms to generate high-resolution spatial meshes on the sphere
  - also of interest are variable-resolution grids with smooth transition regions

- With these needs in mind, we combine tools in computational geometry to allow for the fast generation of spherical Delaunay triangulations and spherical Voronoi meshes
  - these tools lead to a new algorithm that makes use of stereographic projections and a novel approach to the domain decomposition needed for parallelization
  - the new algorithm is designed to allow for the parallel computation of spherical meshes that tessellate the entire sphere or some subregion of interest

- With some further development, the algorithm is useful for the construction of spherical centroidal Voronoi tessellations (SCVTs)
  - among several desirable features, SCVT variable-resolution grids feature smooth transitions

PRELIMINARIES

SIMPLICES AND DELAUNAY TESSELLATIONS

- Let $d+1$ points in $\mathbb{R}^d$ be in general position
  - this means that, for $s = 1, \ldots, d$, no subset of $s+1$ points lies on an $(s-1)$-dimensional hyperplane
  - e.g., no subset of $s+1=3$ points lies on a line, no subset of $s+1=4$ points lies on a 2-dimensional plane, ... , the $d+1$ given points do not lie on an $(d-1)$-dimensional hyperplane

- A d-simplex is the d-dimensional polytope which is the convex hull of the $d+1$ points
  - a d-simplex is the smallest convex set containing the $d+1$ points
  - the given points are the vertices of the simplex

- a 1-simplex is a line segment, a 2-simplex is a triangle, and a 3-simplex is a tetrahedron
- in case you have always wondered, a 4-simplex is called a pentachoron
- a single point may be considered a 0-simplex

- A d-simplex has s-faces ($s \leq d$) corresponding to any $s+1$ distinct vertices of the d-simplex
  - an s-face is itself an s-simplex
  - e.g., a d-face is the simplex itself, . . . , a 3-face is a tetrahedron, a 2-face is a triangle, a 1-face is an edge, and a 0-face is a vertex

- Given a point set P in $\mathbb{R}^d$, the Delaunay tessellation $D(P)$ of the point set is the set of d-simplices such that:
  - a point $\mathbf{p} \in \mathbb{R}^d$ is a vertex of a simplex in $D(P)$ if and only if $\mathbf{p} \in P$
  - the intersection of two simplices in $D(P)$ is either the empty set or a common face
  - the interior of the circumscribing d-sphere through the $d+1$ vertices of any simplex in $D(P)$ contains no other points from the point set P

- If a circumscribing d-sphere has more than $d+1$ points lying on its perimeter, the Delaunay tessellation is not unique

- The Delaunay tessellation of a point set defined in $\mathbb{R}^d$ is related to the convex hull of the point set when projected onto a paraboloid in $\mathbb{R}^{d+1}$

- In 2D, and even in general, Delaunay tessellations are referred to as Delaunay triangulations

VORONOI TESSELLATIONS

- Given a set $P = \{\mathbf{x}_j\}$ of points $\mathbf{x}_j$ in $\mathbb{R}^d$, the Voronoi region or Voronoi cell associated with the point $\mathbf{x}_i$ is the subset $V_i \subset \mathbb{R}^d$ defined by

$$V_i = \{\mathbf{x} \in \mathbb{R}^d \ : \ |\mathbf{x} - \mathbf{x}_i| < |\mathbf{x} - \mathbf{x}_j| \qquad \forall j \neq i\}$$

  - V_i consists of all points in $\mathbb{R}^d$ that are closer to $\mathbf{x}_i$ than to any of the other points in P

- The set $V = \{V_j\}$ of Voronoi regions is referred to as a Voronoi tessellation or Voronoi diagram

  - the set of points P are referred to as the generators of the Voronoi tessellation

- Given a set P of points in $\mathbb{R}^d$, the corresponding Delaunay and Voronoi tessellations are dual tessellations
  - e.g., if you connect all pairs of generators whose Voronoi cells share a common face, you get the Delaunay tessellation

STEREOGRAPHIC PROJECTIONS

- Stereographic projections are special mappings between the surface of a sphere and a plane tangent to the sphere
  - stereographic projections are conformal mappings (angles are preserved)
  - stereographic projections preserve circularity (circles are preserved)
    - also, interior of circles are mapped to interior of circles

- Preserving circularity is a particularly important property

  $\Longrightarrow$ the Delaunay criteria is preserved by stereographic projections
  - because Delaunay triangle circumcircles (along with their interiors) are preserved, therefore Delaunay triangulations are preserved

- Stereographic projections can be used to compute a Delaunay triangulation of a portion of the sphere by allowing the triangulation to be constructed in the more convenient planar geometry

- In $\mathbb{R}^3$ we let

  $\mathbf{C}$ = the center of the sphere (typically the origin)

  $\mathbf{T}$ = the point of tangency (where the projection plane is tangent to the sphere)

  $\mathbf{F}$ = the focus point which is a reflection about $\mathbf{C}$ of $\mathbf{T}$, i.e., $\mathbf{F}$ is the antipode of $\mathbf{T}$

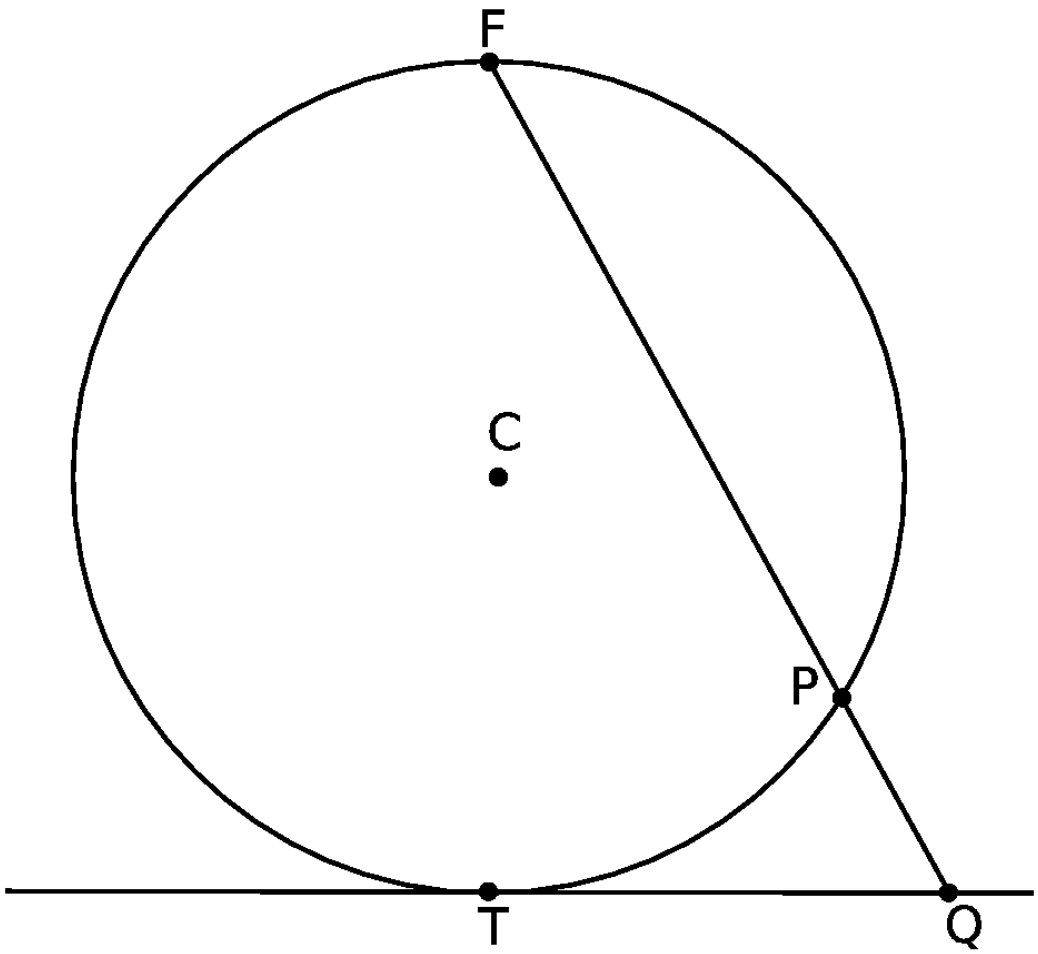
  $\mathbf{P}$ = the point on the surface of the sphere that is to be projected

- The stereographic projection of $\mathbf{P}$ into a point $\mathbf{Q}$ on the tangent plane is defined as

$$\mathbf{Q} = s\mathbf{P} + (1 - s)\mathbf{F}$$

where

$$s = 2\frac{(\mathbf{C} - \mathbf{F}) \cdot (\mathbf{C} - \mathbf{F})}{(\mathbf{C} - \mathbf{F}) \cdot (\mathbf{P} - \mathbf{F})}$$



Cross-sectional illustration of a stereographic projection of the point P on the surface of the sphere to the point Q on the tangent plane through the point T; $\mathbf{F}$ is the antipode of $\mathbf{T}$

A PARALLEL SPHERICAL DELAUNAY TRIANGULATION ALGORITHM

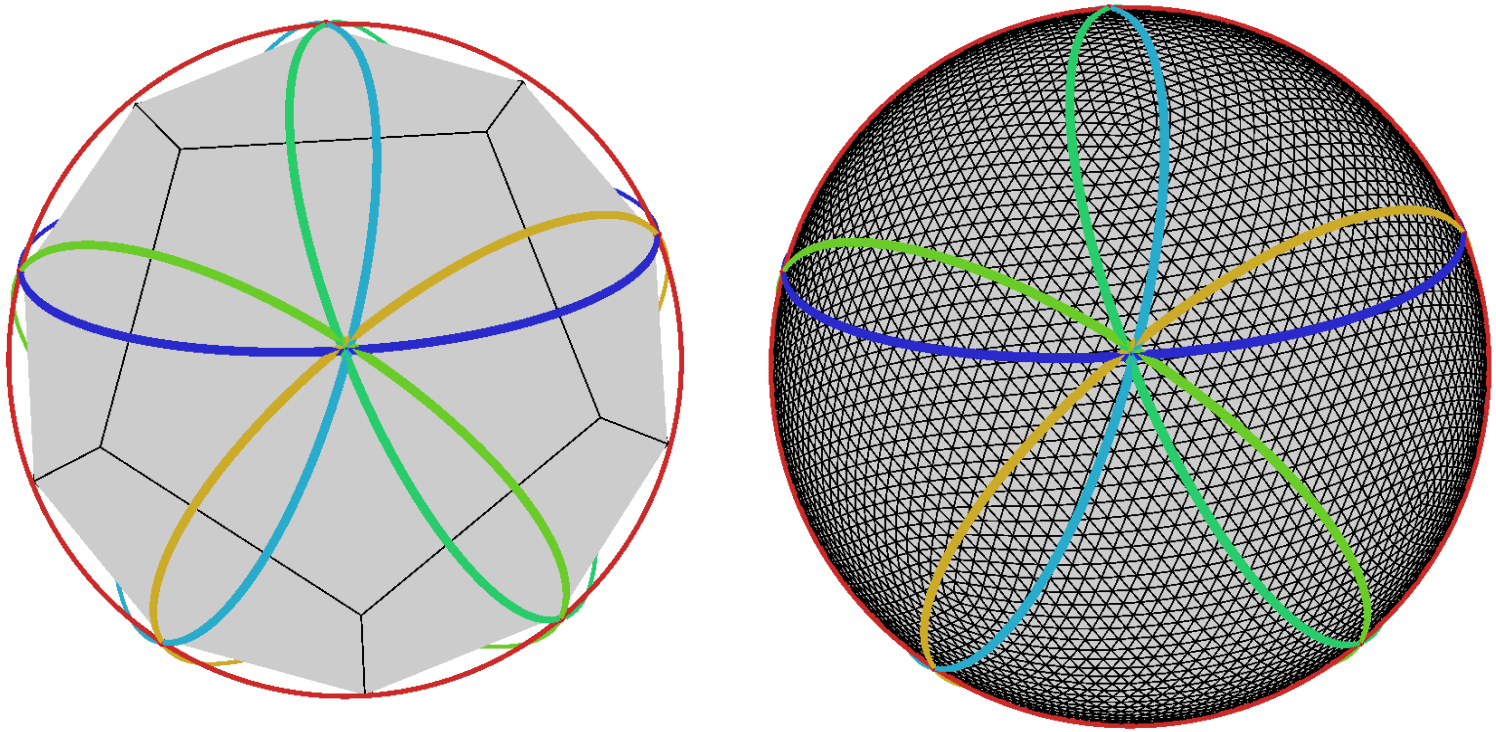
- Computing a planar Delaunay triangulation in parallel has been worked on for several years
  - typically, such algorithms divide the point set up into smaller sets that are triangulated independently from each other
  - the triangulations then need to be stitched together to form a global triangulation
  - the stitching, or merge step, is typically computed serially
  - the merge step is the main difference between most parallel algorithms

- We develop an overlapping domain decomposition approach to create spherical Delaunay triangulations in parallel
  - stereographic projections allow us to use planar Delaunay triangulators as a central step of the algorithm
  - overlapping allows for the transparent merging of subdomain triangulations

THE ALGORITHM

- Choose N tangency points $\{\mathbf{T}_k\}_{k=1}^N$ and N region radii $\{R_k\}_{k=1}^N$

- Divide the sphere into N overlapping coarse regions $\{Y_k\}_{k=1}^N$
  - for each k, Y_k is defined by the geodesic arc having radius R_k and the tangent plane at the the tangency point $\mathbf{T}_k$
  - each region is owned by an independent processor and has connectivity information, i.e., a list of neighbors, defined
  - the regions look like overlapping spherical caps
  - each region Y_k includes points $\mathbf{p}_i$ from the points set P that are inside of its region radius R_k, i.e., such that $\cos^{-1}(\mathbf{T}_k \cdot \mathbf{p}_i) \leq R_k$
  - this sorting may result in a point being in more than one of the regions $\{Y_k\}_{k=1}^N$



Domain decomposition illustration with $N = 12$
– Left: The points of tangency $\{\mathbf{T}_k\}_{k=1}^{12}$ *are the centers of the* 12 *spherical pentagons; the dark blue circle defines the edge of the spherical cap* Y_k *corresponding to the tangency point* $\mathbf{T}_k$ *at the north pole*
– Right: a $10,242$ *generator SCVT Delaunay triangulation illustrating how points (the vertices of the triangles) can belong to more than one spherical cap*

- Let $P_k \subset P$ denote the points in the point-set P that are in the spherical cap Y_k associated with the tangency point $\mathbf{T}_k$

- Let $\widetilde{P}_k$ denote the set of points determined by stereographically projecting the points P_k on the sphere onto the tangent plane corresponding to $\mathbf{T}_k$

- Let $\widetilde{T}_k$ denote the Delaunay triangulation of the planar point set $\widetilde{P}_k$

  – one can use one's favorite planar Delaunauy triangulation algorithm to determine $\widetilde{T}_k$

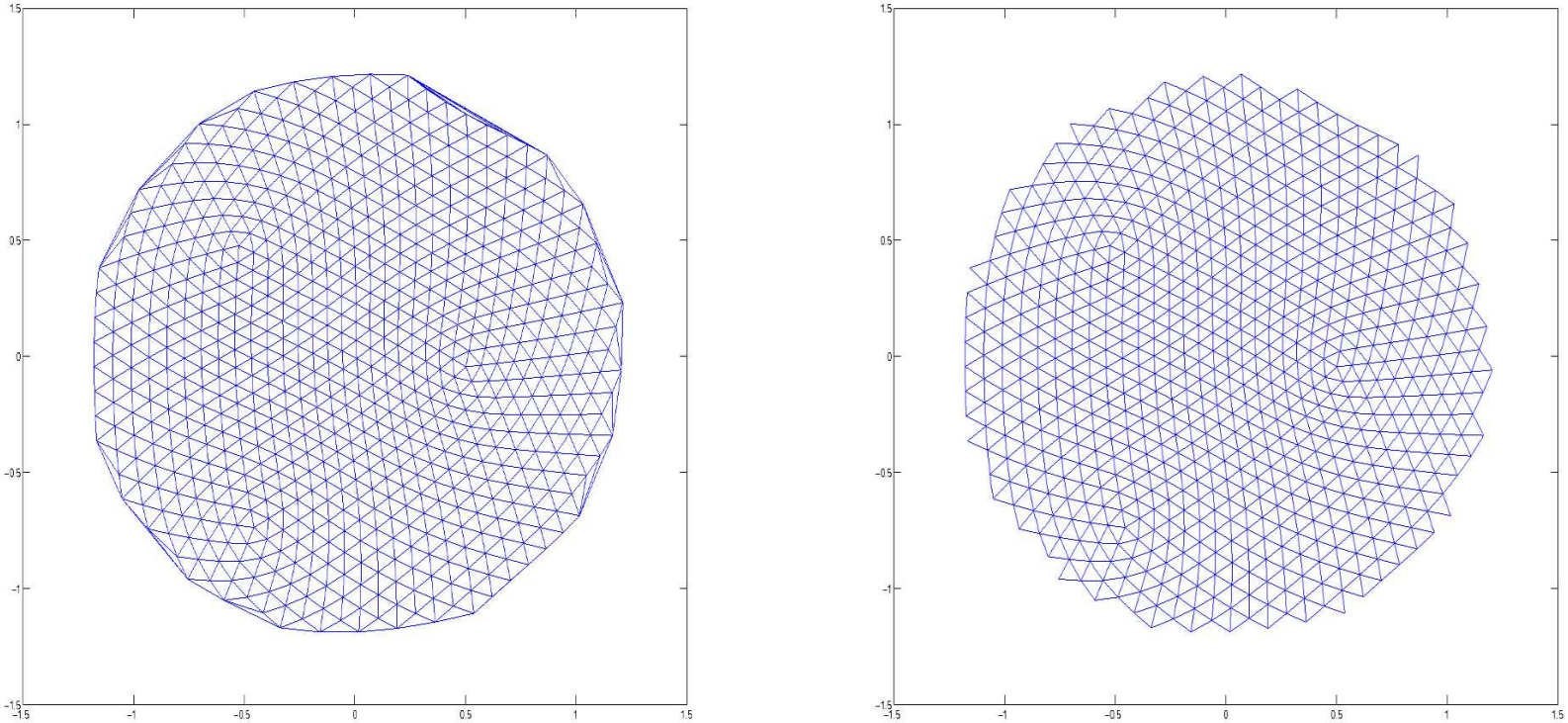
    - e.g., the Delaunay triangulator that comes with Shewchuk's Triangle package

- The triangulation $\widetilde{T}_k$ of the point set $\widetilde{P}_k$ is by construction a Delaunay triangulation of $\widetilde{P}_k$
  - by the properties of stereographic projections, the corresponding spherical triangulation T_k of the point set $P_k \in Y_k$ is a spherical Delaunay triangulation
  - however, there are triangles in the triangulation T_k of P_k that may not be globally Delaunay
    - they may not be Delaunay with respect to the points in P that are not in P_k, i.e., points that are not in the spherical cap Y_k
  - only triangles whose circumcircles are completely contained inside of the region radius R_k are guaranteed to be globally Delaunay
  - so, we discard those triangles whose circumcircles are not completely contained inside of the region radius R_k

– the triangles to be discarded are identified as those that fail to satisfy

$$\cos^{-1} |\mathbf{T}_k - \mathbf{c}_j| + r_j < R_k$$

where $\mathbf{T}_k$ is the region center, R_k is the region radius, r_j is a triangle circumradius, and $\mathbf{c}_j$ is a triangle circumcenter



Triangulations in the plane after stereographic projection
– Left: before the circumcircle containment criteria is applied to identify triangles that are possibly not globally Delaunay
– Right: after the deletion of triangles that are possibly not globally Delaunay

- After this step is complete, each regional triangulation is now exactly a portion of the global Delaunay triangulation
  - the regional triangulations overlap
  - but they must coincide wherever they overlap

- Thus, the union of the regional Delaunay triangulations is the global Delaunay triangulation
  - by using an overlapping domain decomposition, the stitching of the regional Delaunay triangulations into a global one is transparent

- The radii $\{R_k\}_{k=1}^{N}$ should be chosen large enough so that there are no gaps between the regional Delaunay triangulations
  - a region radius corresponding to the maximum distance to any adjacent region center allows enough overlap, at least for quasi-uniform grids
  - this heuristic may not be optimal for variable resolution grids

QUASI-UNIFORM GRID RESULTS FOR DELAUNAY TRIANGULATIONS

- We first compare timings for our parallel spherical Delaunay and Renka's STRIPACK triangulator
  - STRIPACK is a commonly used ACM-TOMS serial FORTRAN 77 algorithm for constructing Delaunay triangulations and Voronoi diagrams on the sphere

Algorithm	Processors	Regions	Time (ms)	Speedup
STRIPACK	1	1	207529	–
parallel	1	2	9504	21
parallel	42	42	5663	37

Comparison of timings for STRIPACK and serial and parallel implementations of our algorithm for a single triangulation of a 163,842 *generator (*60*km global) quasi-uniform grid on the whole sphere; our algorithm needs at least 2 regions to run because the stereographic projection becomes singular if only one region is used on the whole sphere*

MORE PRELIMINARIES

CENTROIDAL VORONOI TESSELLATIONS

- In $\mathbb{R}^d$, given a non-negative function $\rho(\mathbf{x})$, referred to as the point-density function, the center of mass or centroid $\mathbf{x}_i^*$ of a Voronoi cell V_i with respect to the density function is defined as

$$\mathbf{x}_i^* = \frac{\displaystyle\int_{V_i} \mathbf{x}\rho(\mathbf{x})d\mathbf{x}}{\displaystyle\int_{V_i} \rho(\mathbf{x})d\mathbf{x}}$$

- In general, $\mathbf{x}_i \neq \mathbf{x}_i^*$
  - i.e., the centroids of the Voronoi cells do not coincide with the generators of the Voronoi tessellation

- If it just so happens that $\mathbf{x}_i = \mathbf{x}_i^*$ for every point in the point set P, we refer to the Voronoi tessellation as a centroidal Voronoi tessellation (CVT)

- The point-density function can be used to guide local grid spacings
  - if h_i denotes the linear dimension of the Voronoi cell V_i,

    i.e., h_i is the local grid size,

    then the h's for two Voronoi cells are roughly related to the density function in the cells by the formula

$$\frac{h_i}{h_j} \approx \left[\frac{\rho(\mathbf{x}_j)}{\rho(\mathbf{x}_i)}\right]^{\frac{1}{d+2}}$$

  - this density-grid size relation has not been rigorously proven, but has been observed countless times in computational results

  - thus, the density function can be used for the construction of variable resolution meshes
    - in $\mathbb{R}^2$ and $\mathbb{R}^3$, adaptive CVT grid generation algorithms that connect the point-density function ρ to a posteriori error indicators have been developed

- The key to the effectiveness of CVTs for variable-resolution meshing (and for the many other applications of CVT) is Gersho's conjecture (proven only in 2D and only for a constant point-density function)
  - as the number of points goes to infinity, CVTs become a locally uniform mesh made up of nice congruent cells
    - regular hexagons in 2D, truncated regular octahedra in 3D
- CVTs have to be constructed
  - the most commonly used construction method is Lloyd's method
    - start with "any" set of generators
    - compute the corresponding Voronoi tessellation
    - compute the centroids of the Voronoi cells
    - replace the generators with the centroids and start all over again
  - there are nice theorems about the convergence of Lloyd's method

SPHERICAL CENTROIDAL VORONOI TESSELLATIONS

- Replacing the discussion with an analogous discussion on the surface of the sphere leads to spherical centroidal Voronoi tessellations (SCVTs) and the corresponding spherical Delaunay tessellations

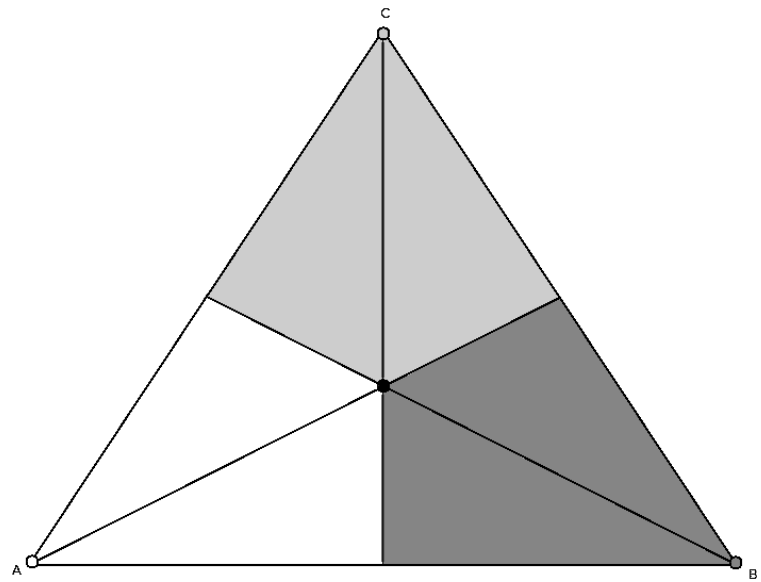
- There are a few issues that arise, such as the fact that the center of mass of a region on the sphere is not on the sphere, but all such issues are easily resolved

- Note that for the surface of the sphere, $d = 2$ for the purposes of scaling the density function

DETERMINING CENTROIDS FROM THE DELAUNAY TRIANGULATION

- In the second step of Lloyd's method, the centroids of the Voronoi cells have to be determined
  - this involves integrations over the Voronoi cells
  - so it seems that one has to construct the Voronoi diagram to do this
  - but one can effect this step using only the Delaunay triangulation
- Each triangle from a Delaunay triangulation contributes to the integration over three different Voronoi cells

- If the triangle is split into three kites, each made up of two edge midpoints, the triangles circumcenter, and a vertex of the triangle, each kite contributes to the Voronoi cell associated with the triangle vertex that is part of the kite



Triangle subdivision used for integrating over Voronoi cells using only the Delaunay triangulation without any adjacency information; kite sections contribute to the Voronoi cell whose generator is the vertex that is in the kite; triangular regions that are similarly shaded contribute to the same Voronoi region

- Integrating over the two triangles in each kite (using one's favorite triangle quadrature rule) updates a portion of integrals appearing in the centroid calculation

- This allows one to only use the Delaunay triangulation when computing centroids, so that no mesh connectivity needs to be computed during the Lloyd iteration

THE NEED FOR A FAST SCVT CONSTRUCTION ALGORITHM

- Why do we need a fast way to determine Voronoi diagrams and Delaunay triangulations on the sphere?

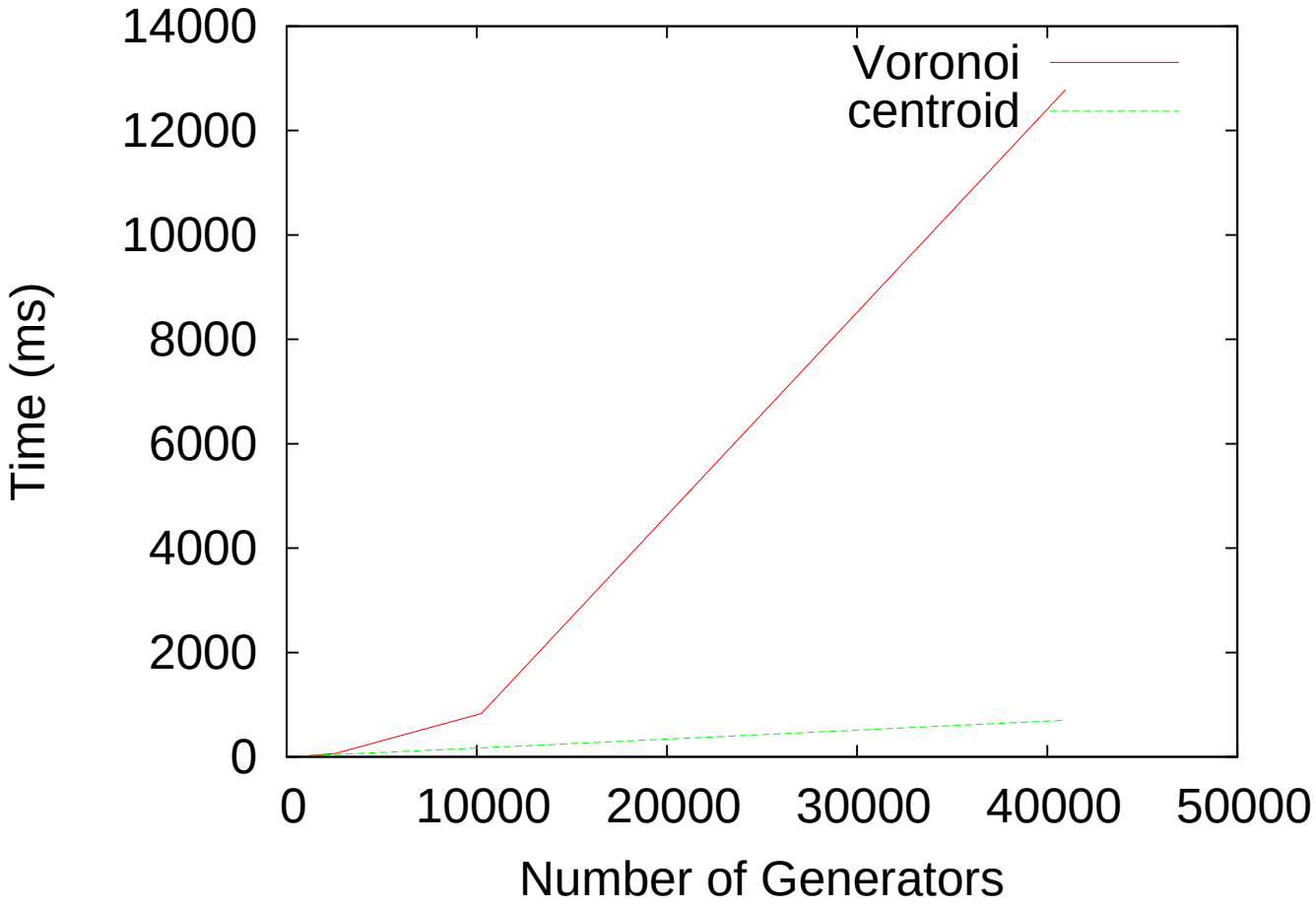
  - we could use Renka's STRIPACK package to determine the Voronoi diagram

  - let's compare the time spent doing the two steps of Lloyd's method, i.e,

    Voronoi construction

    and

    centroid determination



Timings for the Voronoi construction step (using STRIPACK) and the centroid determination step of Lloyd's method

THE PARALLEL SCVT/DELAUNAY ALGORITHM

- The parallel algorithm for constructing an SCVT follows Lloyd's method so there are two key steps in each iteration
  - the first step is the construction of the Voronoi diagram of the current set of generators
  - the second step is the determination of the centroids that involves integrations over the Voronoi cells
  - so it seems that at each step of Lloyd's method we have to construct the Voronoi diagram

- However, Lloyd's method can be implemented so that Voronoi diagrams are avoided and instead only Delaunay triangulations are needed
  - only after the Lloyd iteration terminates, do we construct the spherical Voronoi diagram, if that is needed, corresponding to the converged Delaunay triangulation

THE ALGORITHM

- Given the current positions of the generators, the first step is to determine, in parallel, the N overlapping regional Delaunay triangulations
  - we have already seen how to do this
  - within each Lloyd iteration, we do not merge the N regional Delaunay triangulations into a single global one

- The second step is the determination of the centroids of the Voronoi cells
  - although this involves integrations over the Voronoi cells, we have seen how those integrations can be carried out using the Delaunay triangulation
  - because we have not assembled the regional Delaunay triangulations into a single global one and because the regional Delaunay triangulations overlap, a triangle may appear in more than one regional Delaunay triangulation

- To make the the algorithm parallel
  - one has to make sure that each generator is only updated by one region
    - so that each triangle only contributes once to a centroid calculation
- To make sure one ends up with a global SCVT grid
  - some communication between processors is needed

- Making sure each generator is only updated once can be done using any of a variety of domain decomposition methods
  - we do this by using a coarse Voronoi tessellation $\{W_k\}_{k=1}^N$ of the tangency points $\{\mathbf{T}_k\}_{k=1}^N$
  - the points in the point set P are each updated by the processor corresponding to the coarse Voronoi region W_k in which they are located
  - because the Voronoi cells W_k do not overlap, this makes sure each point is only updated once
- Once each of the generators is updated, each region W_k needs to transfer its newly updated points only to its adjacent neighbors, not to all of the active processors
  - this limits each processor's communications to roughly 6 sends and receives, regardless of the total number of processors used

- After this step is over, the convergence of the grid is checked, and the iterations continue or stop depending on the result

- We have used the L_2 and L_∞ norms of generator movement as convergence criteria

  - if the norm goes below some tolerance, the iteration process is deemed to have converged

  - the two norms result in similar convergence paths when plotted against the iteration number

  - other grid metrics can be used, such as the clustering energy, but in practice they tends to be less strict and more computationally expensive, when compared with generator movement

- A variety of initial generator positions can be used for SCVT construction
  - the obvious choice is Monte Carlo sampling
  - if one wants a quasi-uniform grid, the sampling can be done uniformly
  - if one wants a variable resolution grid, then sampling using the point-density function usually leads to a reduction in the number of iterations required for convergence
- Alternately, one can use bisection grids as initial conditions for SCVT construction algorithm

RESULTS

- We compare the use of STRIPACK and the new Delaunay triangulation as a component of the SCVT construction method
  - we average the triangulation step over 2,000 Lloyd iterations

Algorithm	Processors	Regions	Time (ms)	Speedup
STRIPACK	1	1	207529	–
new method serial	1	2	3623	57
new method parallel	42	42	50.66	4092

Comparison of timings using STRIPACK and serial and parallel versions of our algorithm for one step of the Lloyd iteration for constructing a 163,842 *generator (*60*km global) quasi-uniform SCVT on the whole sphere; results are averaged over* 2,000 *iterations of Lloyd's method*

QUASI-UNIFORM GRID RESULTS FOR SCVT CONSTRUCTION

- The initial generator placement can greatly affect the convergence behavior of the Lloyd iteration

- We consider two methods for quasi-uniform placement of the initial generators
  - uniformly sampled Monte Carlo
  - bisection grids

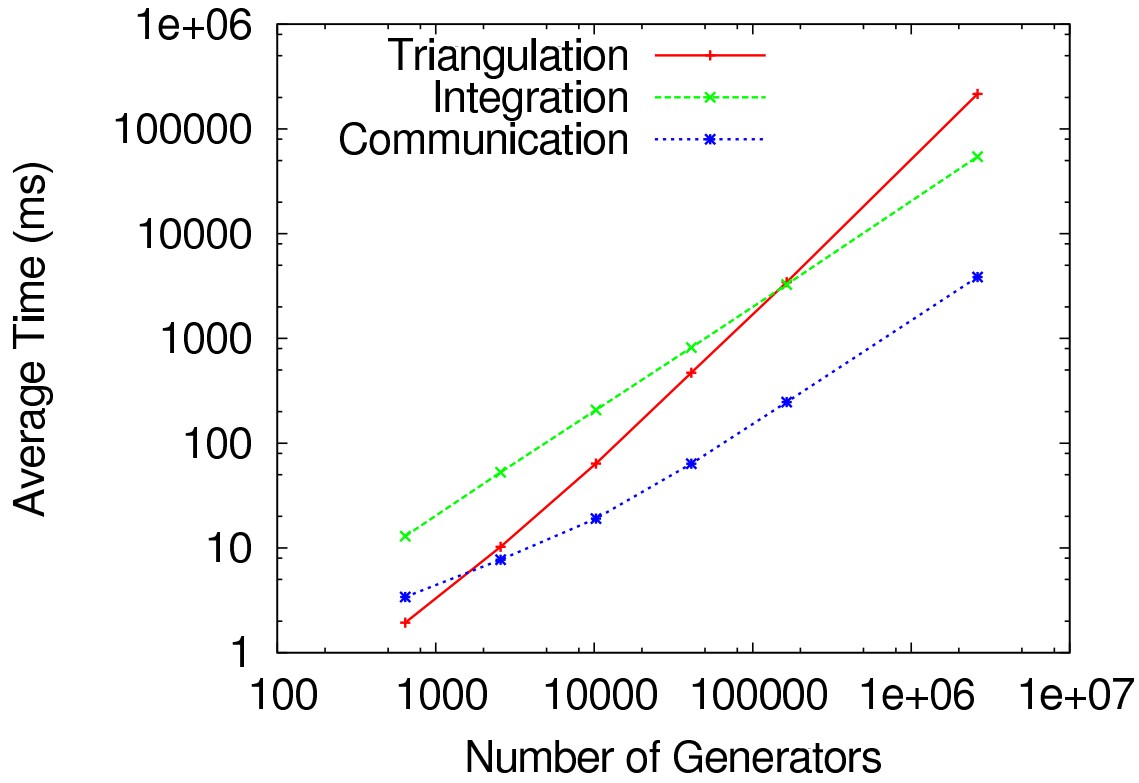
- The convergence tolerance for the Lloyd iteration is set to 10^{-6}
  - this is a good as you can get using Monte Carlo initialization
  - bisection initialization can get you convergence to much smaller tolerances

- We provide timing results for the components of the SCVT construction algorithm for a 2,621,442 (15km) global grid

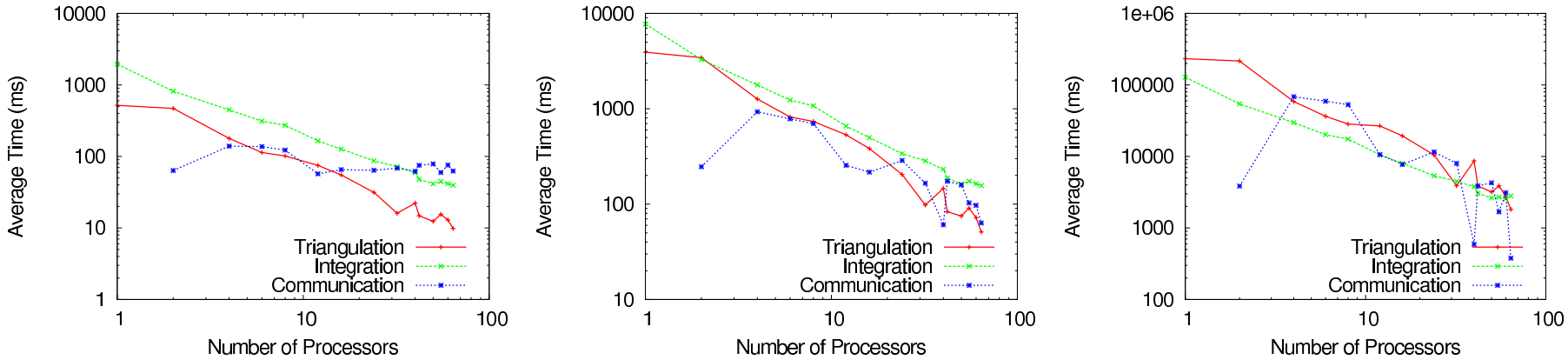
timed portion	initial generator placement choice		speedup
	bisection (B)	Monte Carlo (MC)	MC/B
triangulation time (ms)	73,684	21,164,512	287.23
centroid time (ms)	235,016	12,211,376	51.95
communication time (ms)	3,152,376	33,713,473	10.69
total time (ms)	3,526,041	70,581,300	20.01

Timing results for the parallel SCVT construction algorithm using the bisection and Monte Carlo choices for the initial position of the generators for a 2,621,442 *(*15*km) global quasi-uniform grid*

– *Note that the timings for the bisection initial positions include the time to obtain the converged* 2,621,442*-node mesh,* including *all coarser meshes starting with the coarsest* 12*-point mesh*



Timings vs. problem size for various portions of parallel SCVT method using 2 *processors and* 2 *regions; the triangulation does not become more expensive than the integration until after roughly* 163,842 *generators*

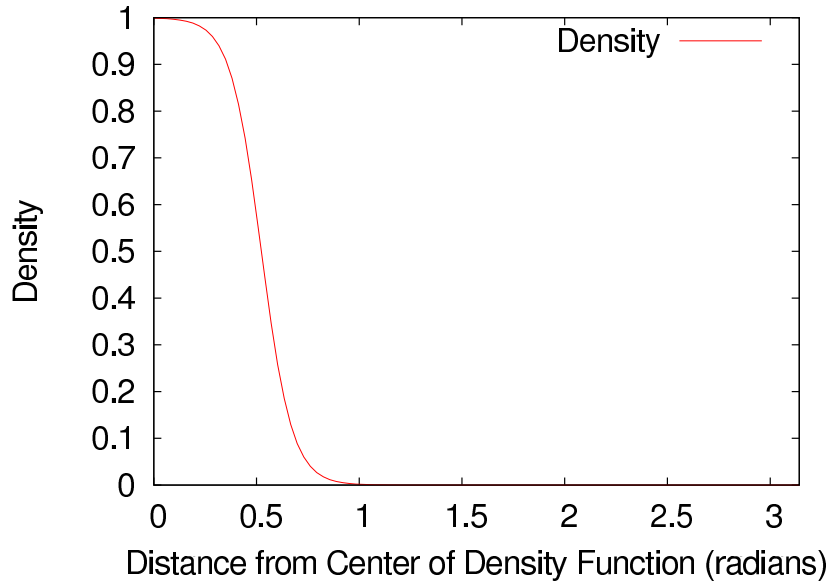


Timing vs. number of processors for the parallel SCVT method for three different problem sizes

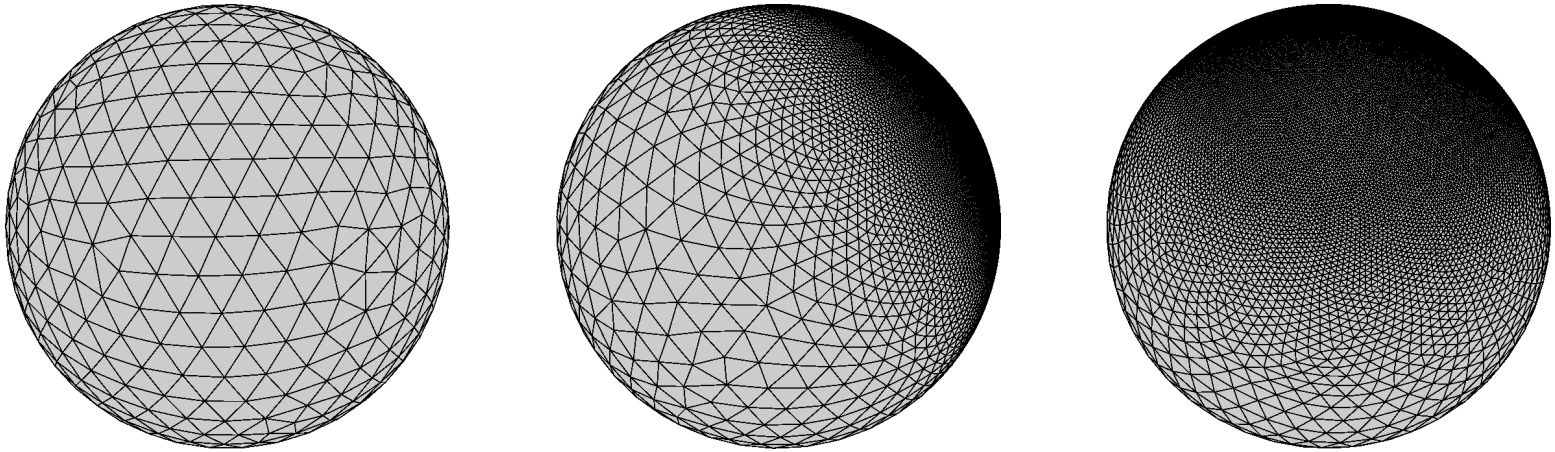
- Todd Ringler has pushed things quite a bit further
  - a 10 million node mesh with sub-2km resolution in a domain about the size of the North Atlantic using 2,562 processors

VARIABLE-RESOLUTION GRID RESULTS FOR SCVT CONSTRUCTION

- We use the point-density function depicted in the figure



The point-density function raised creates a grid with resolutions that differ by a factor of 16 *between the coarse and the fine region; the maximum value of the density function is* 1*, whereas the minimum value is* $(1/16)^4$



coarse region transition region fine region

A variable resolution Delaunay grid corresponding to a SCVT of the whole sphere; all three figures are of the same grid, only the viewing perspective is changed

- The criteria mentioned earlier involving the region radii $\{R_k\}_{k=1}^{N}$ for sorting points into spherical caps does not work well for variable resolution meshes
  - very poor load balancing results
- To resolve this issue, a new algorithm for determining the regions for domain decomposition was developed
- The new algorithm begins by sorting points into a coarse Voronoi tessellation of the tangency points
- Then, instead of the spherical cap, the region corresponding to a particular tangency point is its Voronoi cell and all neighboring Voronoi cells
- This sorting method has greater overlap than the maximum distance method for quasi-uniform grids, but less overlap for variable-resolution grids
  - the better load balancing reduces idle computing time coming from processors that have small loads

– one may also combine the two criteria

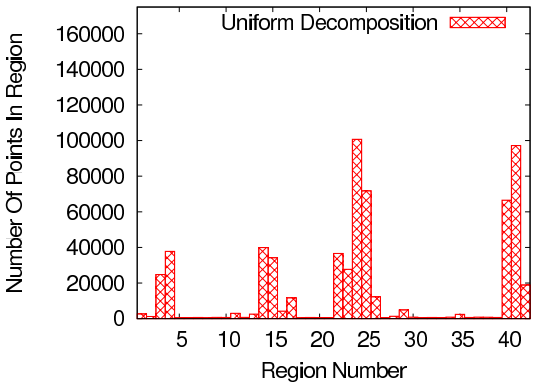
- first sort into Voronoi regions
- then apply the maximum distance criteria

- The table and figure in the next two slides were determined using the exact same initial conditions, which was a converged 16 to 1 nonuniform grid with 163,842 generators, and they all used 42 processors and 42 regions, and timings are averages over 3000 iterations

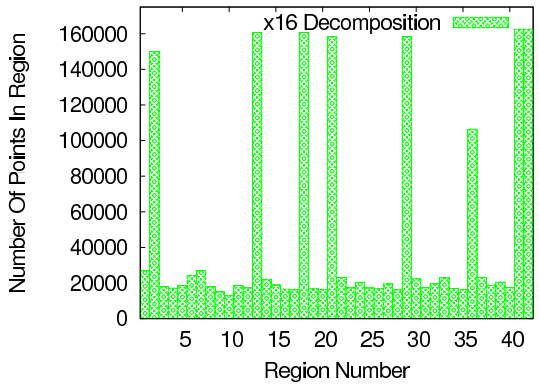
decomposition	triangulation	integration	communication	iteration	speedup
uniform max distance	14.9779	39.3149	2556.971	2611.35	–
nonuniform max distance	104.793	276.681	1560.71	1965.56	1.32
nonuniform Voronoi	98.5482	249.77	288.694	640.472	4.07

Timings based on the domain decomposition method used

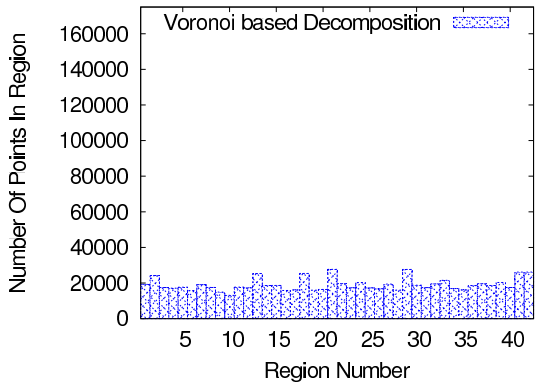
– Uniform *means a coarse quasi-uniform SCVT is used to assign points to processors*
– Nonuniform *means a coarse* 16 *to* 1 *ratio nonuniform SCVT is used to assign points to processors*
– Max distance *means the maximum distance to neighboring generators criteria is used to define the subdomain radius for a point of tangency*
– Voronoi *means that the neighboring coarse SCVT Voronoi regions are included in the regions for a point of tangency*



uniform
max-distance



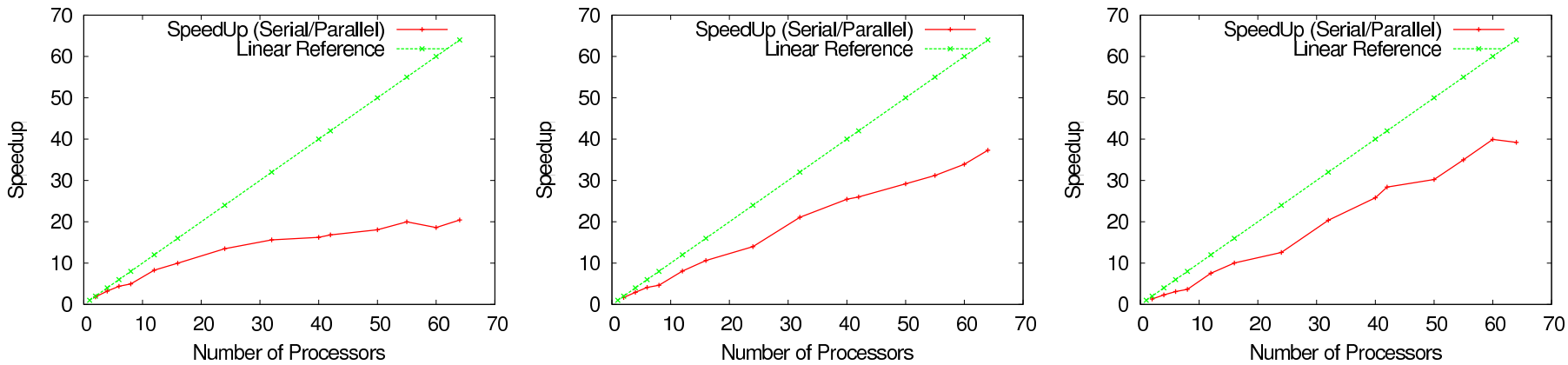
non-uniform
max-distance



non-uniform
Voronoi

Number of points each processor has to triangulate

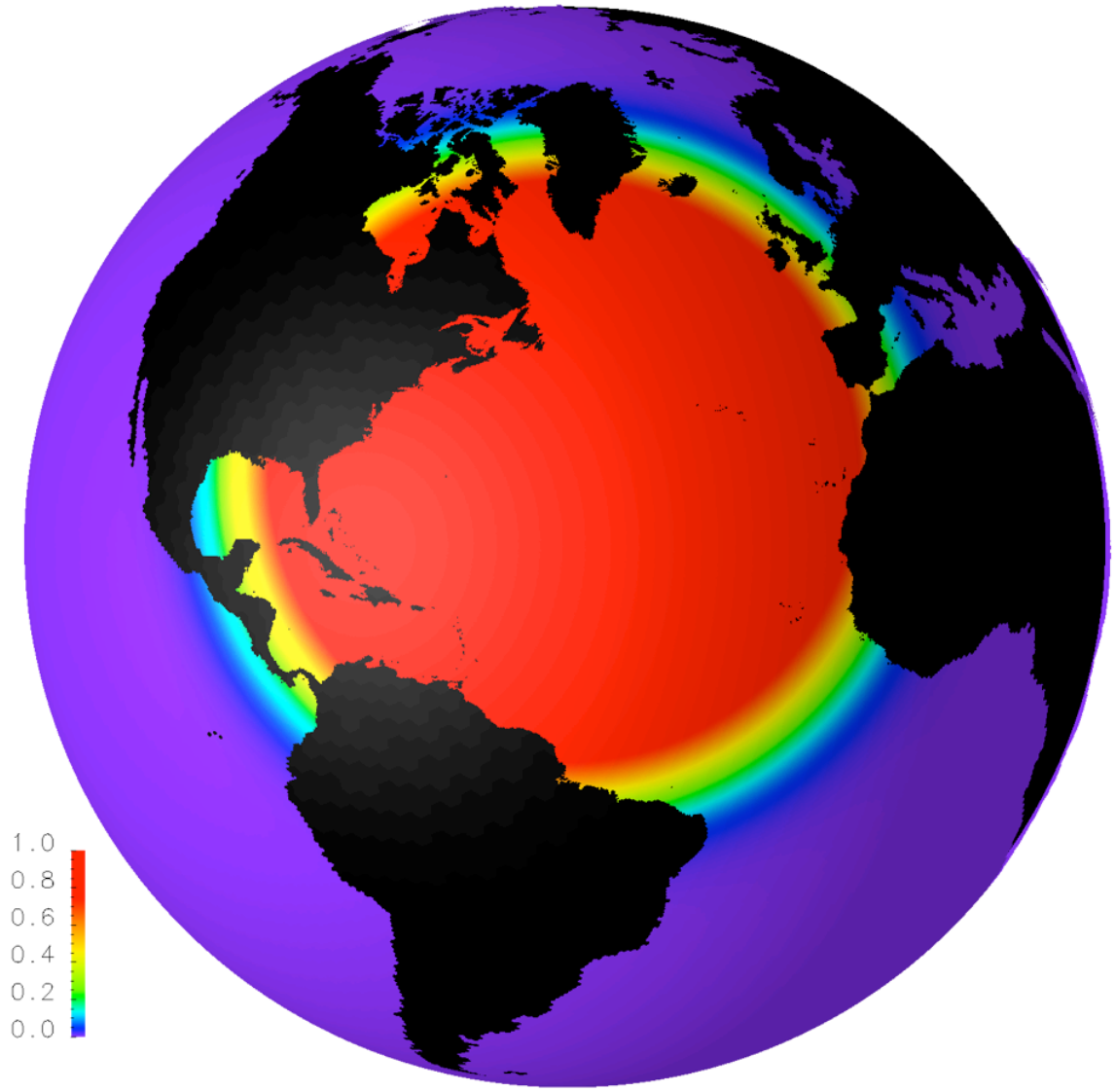
- To assess the overall performance of this algorithm, some scalability results are now presented



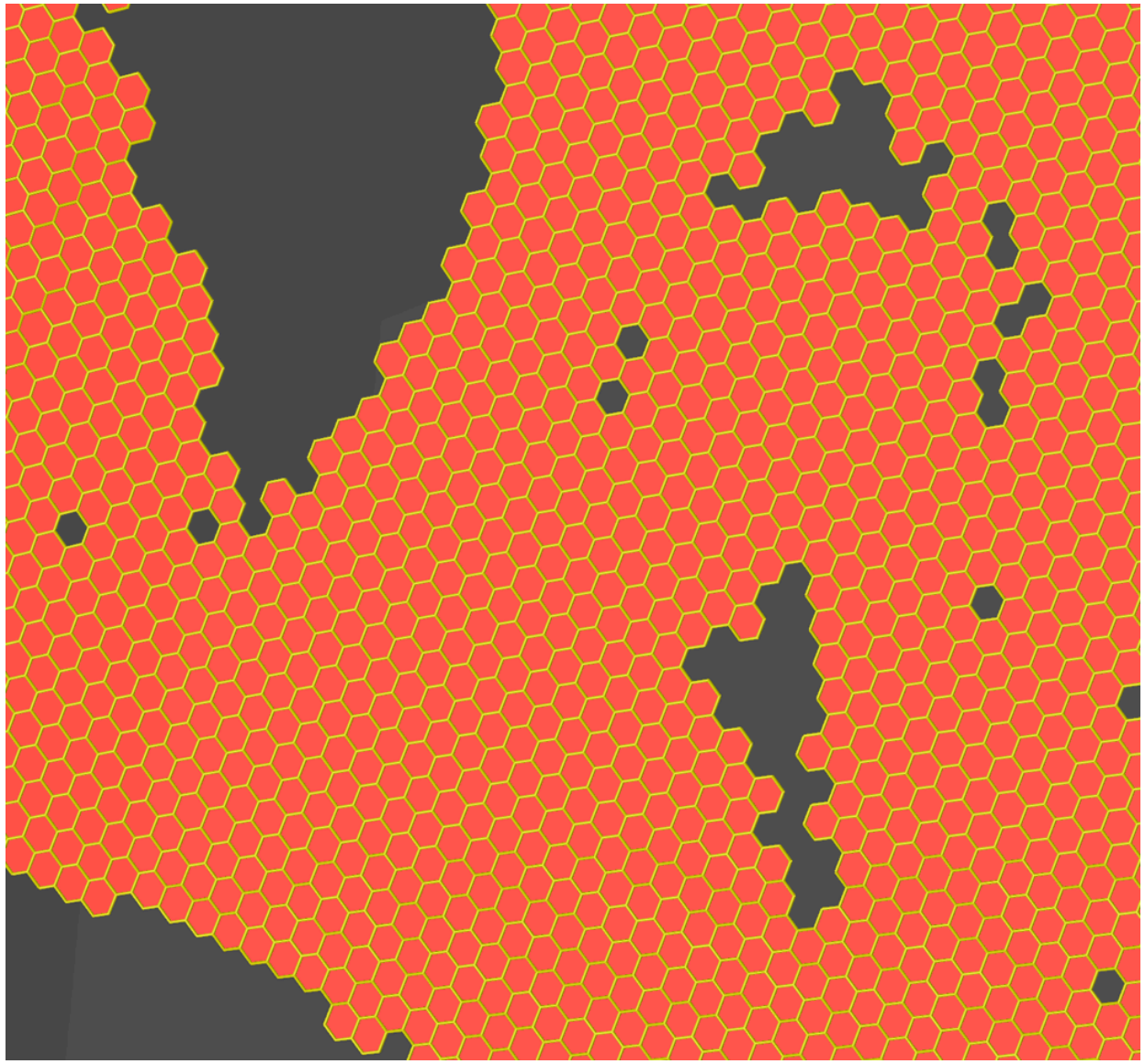
40,962 generators 163,842 generators 2,621,442 generators

Speedup vs. number of processors for the parallel SCVT method for three different problem sizes

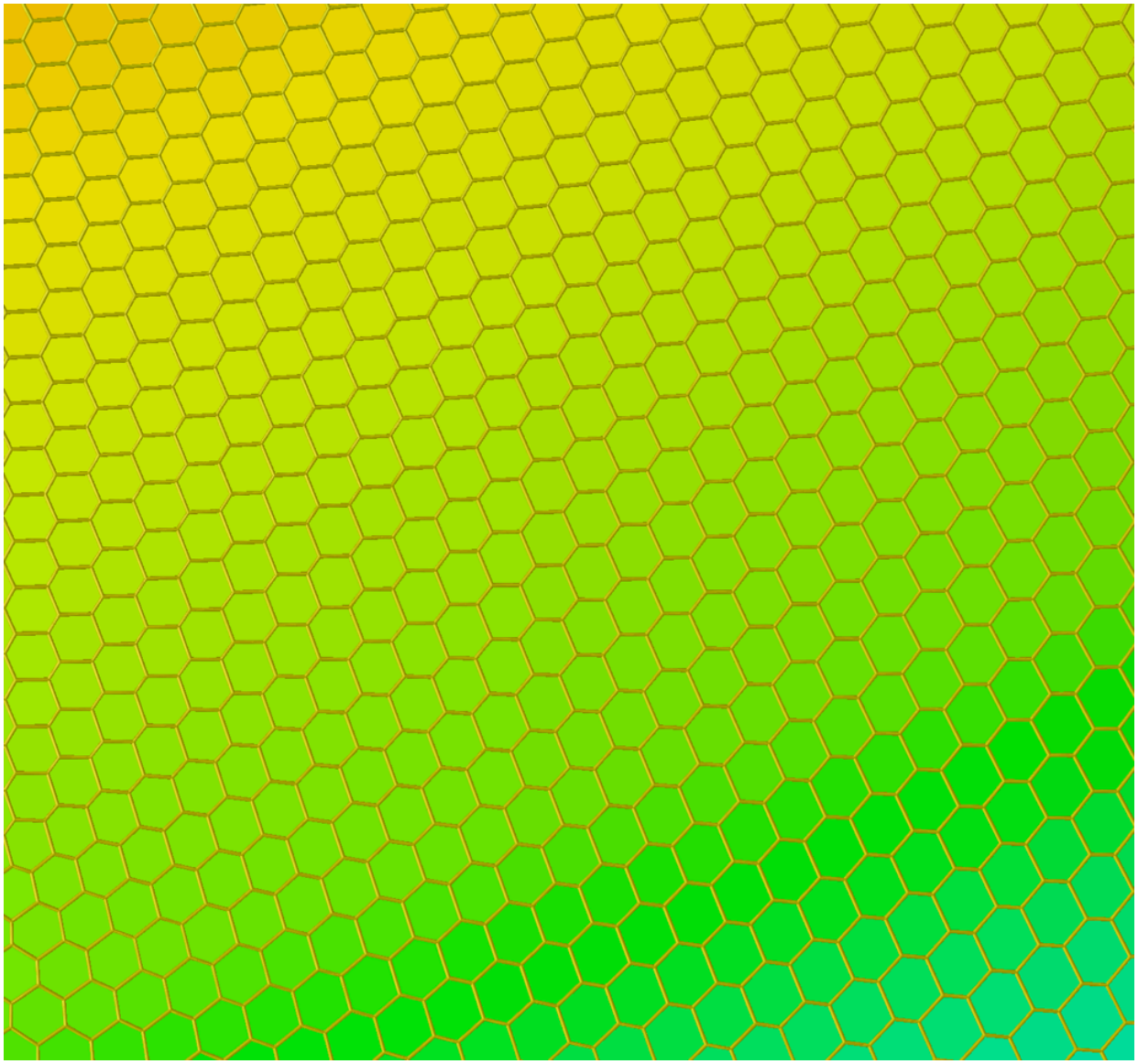
SAMPLE GRIDS



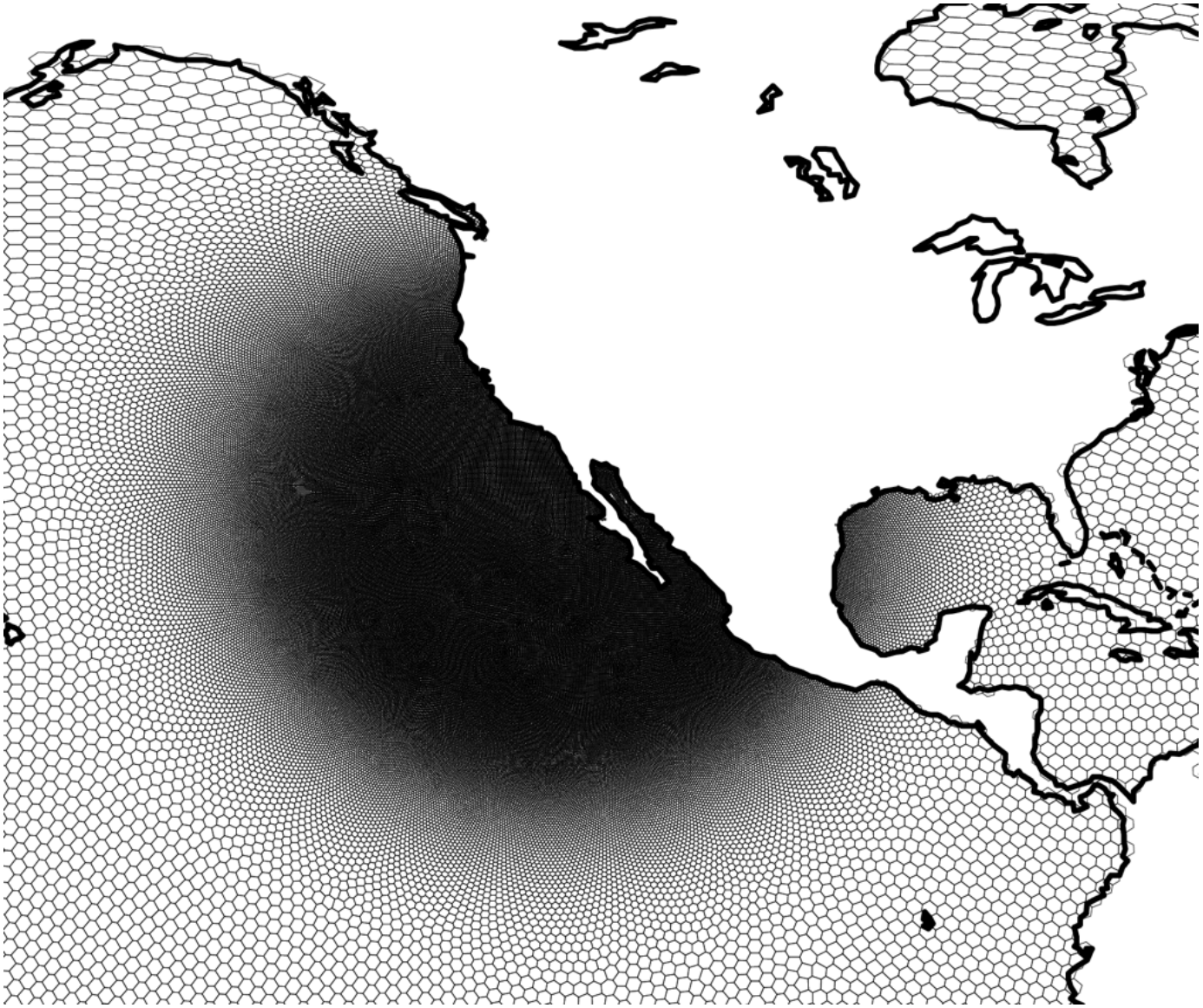
Density function for North Atlantic regional modeling



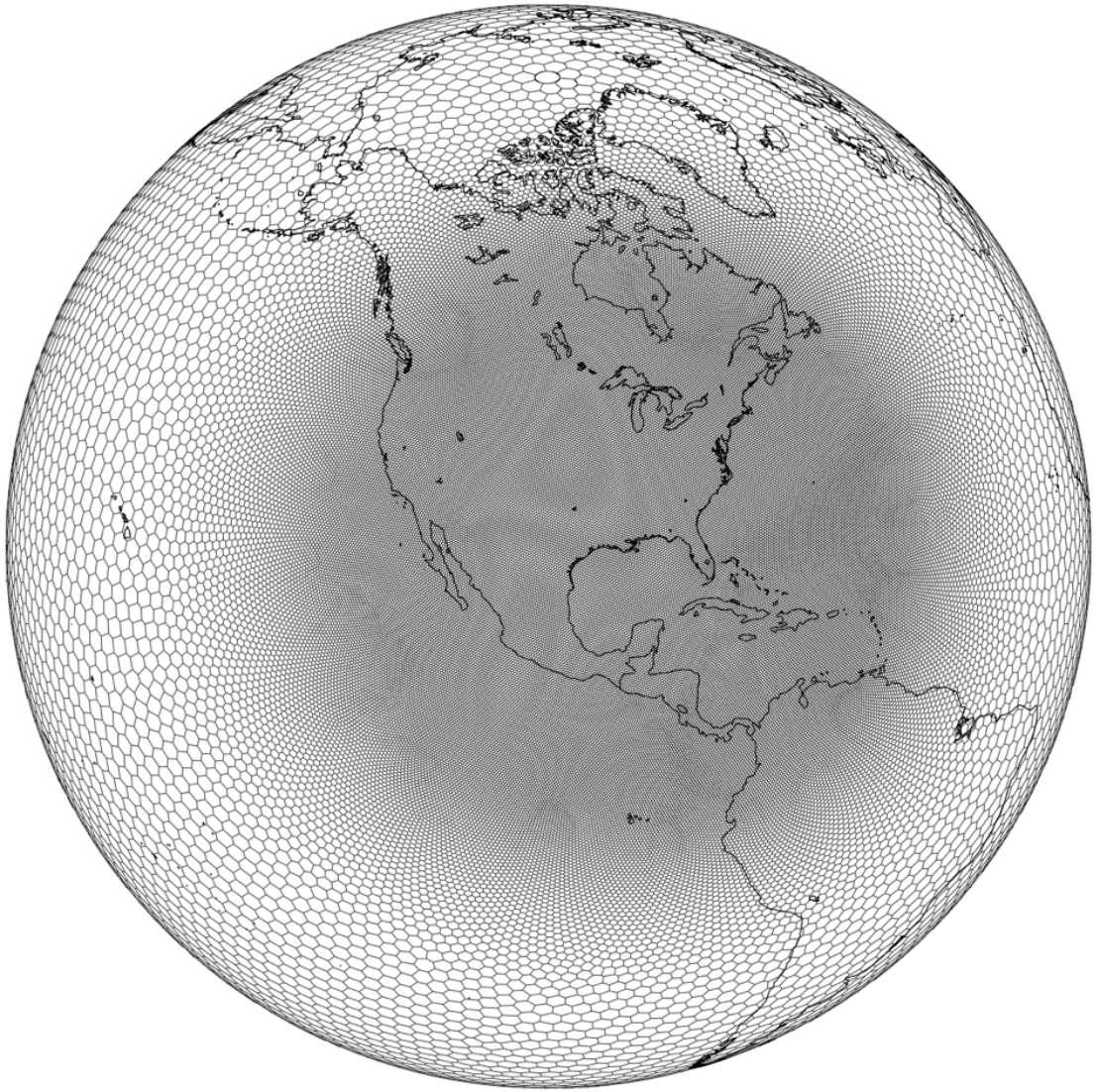
Zoom-in near Florida



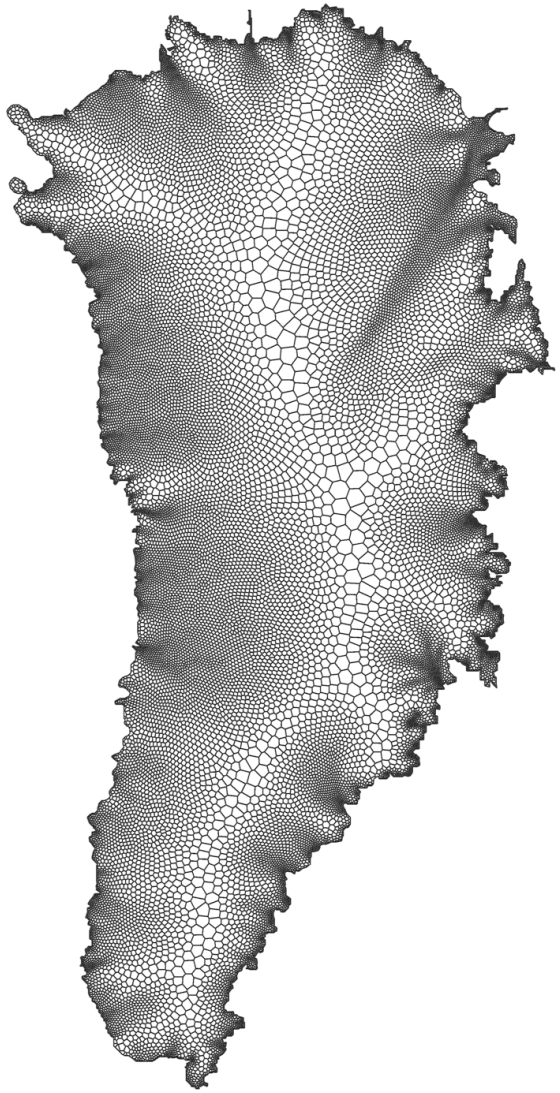
Zoom-in of grid transition region



Grid for California current regional modeling



Grid for North America atmospheric regional modeling



Greenland