Minimum-error world map projections defined by polydimensional meshes

Justin Kunimune[a]

[a]Olin College of Engineering, 1000 Olin Way, Needham, MA, US

ARTICLE HISTORY
Compiled November 8, 2019

ABSTRACT
In order to transform the curved surface of the earth onto the planar surface of a map, some distortion must be introduced. Conventional map projections are defined by mathematical formulas, and are thereby limited in their ability to control that distortion by the complexity of those formulas. Piecewise map projections based on interpolation onto unstructured meshes have no such constraints, and can therefore create maps that are better suited to many use cases. A method using multi-dimensional optimization to optimize such map projections is presented here, along with seven new map projections generated by it. Each new map projection is differentiated by its initial state, cost function, and weights, and is analyzed in terms of distortion criteria and qualitative properties. These map projections are presented as the Danseiji projections, along with some of their potential applications.

KEYWORDS
Map projections; world maps; optimization; minimum-error; finite element analysis

This article has been submitted for publication in the International Journal of Cartography.

1. Introduction

The challenge of accurately representing the surface of the earth on a flat map has plagued cartographers for millennia. Due to the mathematical differences between a sphere and a plane, it cannot be done without distorting some geographic properties in some parts of the map. Therefore, the goal of most map projections is to perform the transformation while faithfully representing some quantities of interest, usually at the expense of other, less important quantities or less important parts of the map (Snyder, 1993, p. 1).

In some cases, exactly one map projection can preserve the desired quantity exactly, and the formulas of that map projection can be found analytically—for example, the Mercator projection (Snyder, 1993, p. 46–47). In other cases, there exists no analytic solution, or the desired quantity is known only qualitatively. In these cases, map projections must be created by combining arbitrary formulas or existing map projections until one obtains a desirable result—for example, the Van der Grinten projection (van der Grinten, 1905).

CONTACT Justin Kunimune. Email: justin.kunimune@gmail.com

It is also possible, in the cases where the desired property cannot be preserved everywhere, to optimize the map projection to have the minimum possible distortion. In highly constrained cases, this can be done analytically using the calculus of variations (Airy, 1861). However, in general, such a task requires numerical approaches (Tobler, 1977). A numerical optimization algorithm cannot directly control the formula that defines a map projection; instead, it controls a finite number of parameters that describe it. These parameters can be standard parallels (Baselga, 2019), polynomial coefficients (Györffy, 2018; Snyder, 1985), or simply the coordinates of specific points on the globe among which the rest of the map can be interpolated (Gott, Mugnolo, & Colley, 2007; Tobler, 1977). While these minimum-error map projections often have much lower distortion criteria than conventionally derived map projections, they are all prevented from reaching the full potential of minimum-error map projections because of one or more limitations.

Most are limited by the number of parameters that are optimized. While most optimization algorithms work for an arbitrary number of parameters, Baselga and Györffy both keep to fewer than ten, while Gott et al. optimizes 6 000 degrees of freedom, but then fits them to a simple single-parameter formula. This restriction allows the resulting map projections to be expressed with simple equations (optionally with a small auxiliary table of coefficients), but limits the optimization algorithm's ability to reduce the distortion. Many are limited to specific topologies. Gott et al., Snyder, and Tobler do not allow for interruptions, and therefore yield map projections well-optimized for regional maps but poorly suited for world maps. All prior minimum-error projections also use as cost functions simple distortion criteria that treat all parts of the globe equally. In principal, there is no reason not to use cost functions that weigh some regions more than others, or are zero for some ideal state other than geographic reality.

To circumvent these limitations, in this study, an expansion on existing techniques is presented. It uses unstructured meshes of triangular elements, a common tool from finite element analysis (Wriggers, 2008, p. 103), to define world map projections. Each mesh has over 12 000 nodes with independent positions in the plane, which are optimized via a numerical algorithm to minimize elastic energy with geographically-defined weights, in order to produce new minimum-error world map projections.

2. Methods

The technique comprises three parts. First, a mesh of triangular elements is constructed. Then, the positions of that mesh's nodes as well as the positions of interruptions between them are optimized to minimize some cost function of the node positions. Finally, linear interpolation is performed in each of that mesh's cells to create a continuous map projection.

2.1. Mesh construction

The meshes used are simple arrays of N nodes at evenly-spaced latitudes and longitudes. At all latitudes besides those adjacent to the poles, the four-sided cells this creates are each divided diagonally into two three-sided elements, the direction of the slice alternating between adjacent cells. The northernmost and southernmost rows of nodes are each combined into a single pole node, causing all adjacent cells to become triangular elements without the need for diagonal dividers. The resulting mesh has $M \approx 2N$ elements. Figure 1 shows the edges of such a mesh.

To define the mesh's state, each node is assigned a position in the plane $\vec{\mathbf{x}}_i = \left[\begin{smallmatrix} x_i \\ y_i \end{smallmatrix}\right]$, and those positions are concatenated into a state vector of length $2N$: $\vec{\mathbf{y}} = \begin{bmatrix} x_1 & y_1 & x_2 & \dots & y_N \end{bmatrix}$. Some nodes are duplicated and given different planar positions for the same spherical location, and then attached to different elements. For example, nodes on the antimeridian in a conventionally-oriented map projection are duplicated for the two sides of the mesh: one of each pair serving as a vertex for the adjacent elements in the Eastern Hemisphere, and the other for those in the Western Hemisphere.

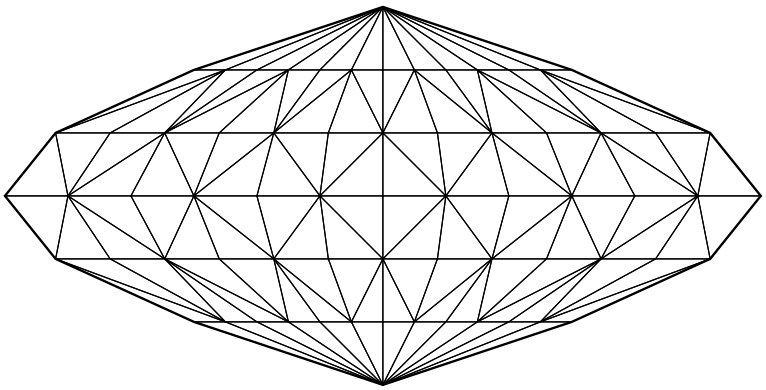

Figure 1. A mesh of $N = 67$ nodes and $M = 120$ elements spaced at 30° and arranged according to a sinusoidal projection. Note that pairs of nodes at the same latitude on opposite sides of the outer boundary represent the same spherical location.

Since each element has nodes for vertices, the elements' positions and shapes are completely defined by the nodes' positions.

2.2. Optimization

These meshes can approximately represent virtually any existing uninterrupted map projection. Their nodes must simply be positioned to match the projected positions of their corresponding spherical locations. This repositioning can be carried out by a human or simple set of mathematical formulas to meet aesthetic or mathematical constraints. Alternatively, by allowing an optimization algorithm such as L-BFGS to determine the node positions, one can ensure that the resulting map projection minimizes some predefined distortion criterion.

By duplicating nodes, giving duplicates independent planar positions, and setting them as vertices to some adjacent elements, interrupted map projections can also be approximately represented. This node splitting can also be performed by an algorithm, though true optimization is more difficult with such a discrete process. A heuristic algorithm for node splitting is presented and implemented in this study.

2.2.1. Initial state

The first step in optimization is the definition of the initial state of the mesh. In theory, so long as it is reasonably near the optimum, the initial state should not affect the results of an optimization algorithm. However, which nodes begin duplicated (i.e. what interruptions initially exist) does affect the results of the node splitting algorithm. Furthermore, if it is not constructed carefully, it is possible for an initial mesh state to have an infinite or undefined cost function, which will cause node position optimization to fail.

Two different initial states are used in this study. The first, shown in figure 2, is based on the Hammer projection (Snyder, 1993, p. 132–33).

$$x = \frac{2R\sqrt{2}\cos\varphi\sin\frac{\theta-\theta_0}{2}}{\sqrt{1+\cos\varphi\cos\frac{\theta-\theta_0}{2}}} \tag{1}$$

$$y = \frac{R\sqrt{2}\sin\varphi}{\sqrt{1+\cos\varphi\cos\frac{\theta-\theta_0}{2}}} \tag{2}$$

where θ_0 is the central meridian, set to either 0°E to match convention or 11.25°E to move the antimeridian into the Bering Strait.

The nodes along the antimeridian are split, creating a longitudinal interruption. This interruption, which is present in most common maps, gives rise to relatively conventional map projections.

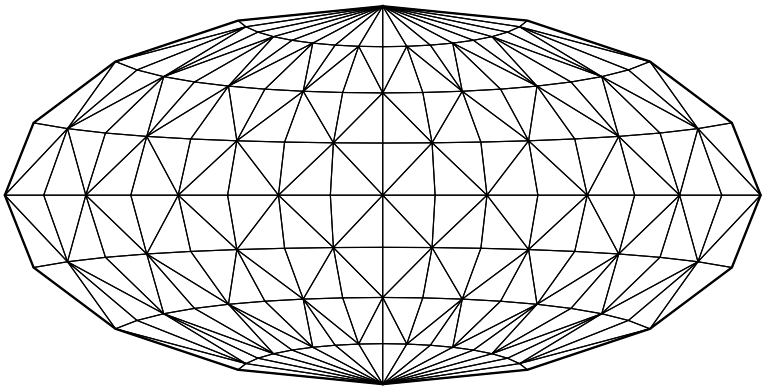

Figure 2. A mesh of $N = 121$ nodes arranged in the Hammer initial state with central meridian $\theta_0 = 0°$.

The second initial state, shown in figure 3, is based on the azimuthal equidistant projection (Snyder, 1987, p. 195).

$$x = R\left(\frac{\pi}{2} - \varphi'\right)\sin\theta' \tag{3}$$

$$y = -R\left(\frac{\pi}{2} - \varphi'\right)\cos\theta' \tag{4}$$

$$\varphi' = \arcsin(\sin\varphi_0\sin\varphi + \cos\varphi_0\cos\varphi\cos(\theta-\theta_0)) \tag{5}$$

$$\theta' = \arctan\left(\frac{\cos\varphi\sin(\theta-\theta_0)}{\cos\varphi_0\sin\varphi + \sin\varphi_0\cos\varphi\cos(\theta-\theta_0)}\right) \tag{6}$$

where φ_0 and θ_0 are the coordinates of the center of the map, set to 49.5°N, 54.0°E to place the antipode of the center near the oceanic pole of inaccessibility in the South Pacific Ocean.

Because it suffers from extreme distortion near the antipode, the azimuthal equidistant projection alone does not produce a usable initial state; many elements near the edge have negative areas, which can yield undefined cost functions. To remedy this, the resulting mesh state is adjusted. First, the node at the antipode is split into eight nodes, one for each adjacent element. Each is placed at an azimuth corresponding to the spherical center of its element. Then, the following procedure ensures that any nodes

that are vertices to elements with negative area are moved away from the origin.

In ascending order by spherical distance from $\left[\begin{smallmatrix}\varphi_0\\ \theta_0\end{smallmatrix}\right]$, each node is examined. All of the elements that have that node as a vertex and whose other vertices have already undergone this process are considered; if an element has other vertices that will be examined later, it is ignored for now. If all of the elements under consideration have positive area, the node is ignored. Otherwise, it is determined how far Δr the node would have to move radially to make all of the elements' areas become zero or positive. The node is then moved $\Delta r + \frac{1}{2}R\Delta\varphi$ away from the origin, where $R\Delta\varphi$ is the spherical distance between adjacent nodes with the same longitude.

After performing this process for every node on the mesh, all elements have positive area, and the mesh can be optimized. The resulting map projections show no bias for common conventions such as the horizontality of parallels or the presence of an interruption along the antimeridian. The only property this initial state imposes on map projections generated with it is the presence of some interruption running through the antipode.

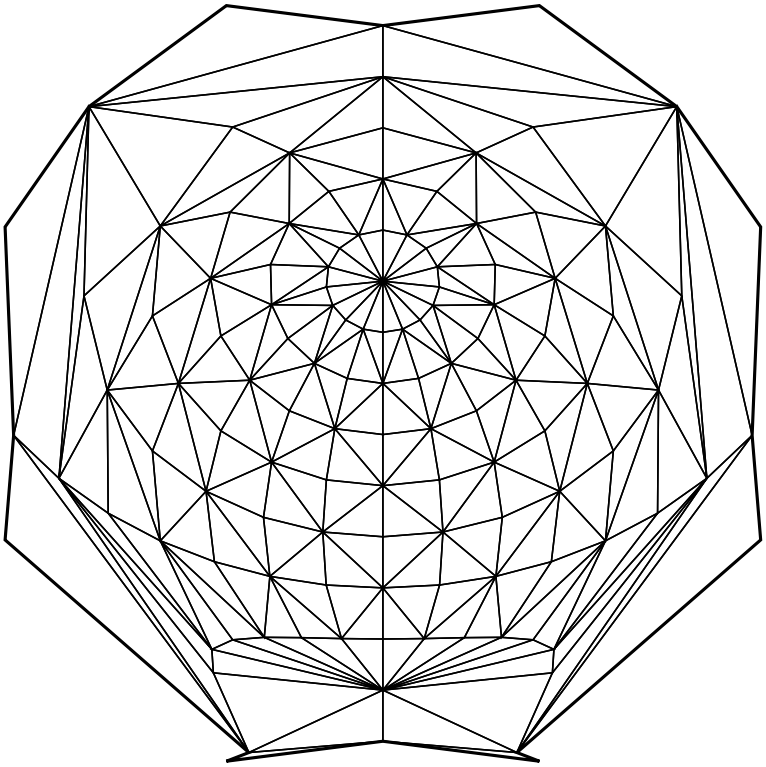

Figure 3. A mesh of $N = 121$ nodes arranged in the azimuthal initial state with center $\varphi_0 = 45°, \theta_0 = 45°$.

2.2.2. Cost function

With node positions fully defined, each element j can compute a deformation gradient (Wriggers, 2008, p. 115–16) that describes the deformed locations of its three vertices

$\vec{\mathbf{x}}_{j1}$, $\vec{\mathbf{x}}_{j2}$, and $\vec{\mathbf{x}}_{j3}$, ordered counterclockwise.

$$\vec{\mathbf{x}}_i = \mathbf{F}_j \vec{\mathbf{X}}_{ij} + \vec{\mathbf{b}}_j \tag{7}$$

$$\mathbf{F}_j = \frac{1}{2A_j} \begin{bmatrix} x_{j2} - x_{j1} & x_{j3} - x_{j1} \\ y_{j2} - y_{j1} & y_{j3} - y_{j1} \end{bmatrix} \begin{bmatrix} Y_{j3} - Y_{j1} & X_{j1} - X_{j3} \\ Y_{j1} - Y_{j2} & X_{j2} - X_{j1} \end{bmatrix} \tag{8}$$

$$A_j = \frac{1}{2} \begin{vmatrix} Y_{j3} - Y_{j1} & X_{j1} - X_{j3} \\ Y_{j1} - Y_{j2} & X_{j2} - X_{j1} \end{vmatrix} \tag{9}$$

This, however, requires that the nodes have undeformed positions $\vec{\mathbf{X}}_{ij} = \begin{bmatrix} X_{ij} \\ Y_{ij} \end{bmatrix}$ that can be deformed in the first place. To consider their three-dimensional undeformed positions on the globe would yield a 3×3 deformation gradient and require assumptions about the transformation of points not on the sphere. Instead, to keep all of the calculations two-dimensional, each node was given a different set of planar undeformed coordinates for each cell it bordered. The deformation gradient in each element was then computed using its containing cell's set of undeformed positions.

These planar undeformed coordinates were derived by approximating each cell as a trapezoid. They are depicted in figure 4, and their expressions are given thus:

$$X_{ij} = R\sqrt{s_j}(\theta_i - \Theta_j)\cos\varphi_i \tag{10}$$

$$Y_{ij} = R\sqrt{s_j}(\varphi_i - \Phi_j) \tag{11}$$

where ϕ_i and θ_i are the spherical locations of node i, s_j is a scaling factor, and Φ_j and Θ_j are the center of element j $\frac{1}{2}(\min(\varphi) + \max(\varphi))$ and $\frac{1}{2}(\min(\theta) + \max(\theta))$.

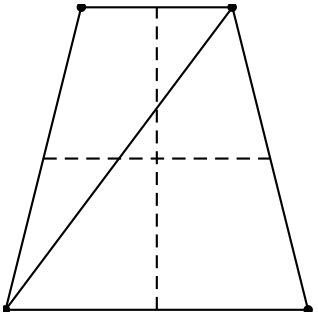

Figure 4. The planar undeformed locations of the nodes in a nonpolar cell. In polar cells, two of the nodes become coincident.

When s_j is uniform, the deformation gradient is smallest for mesh configurations that approximate geographic reality, and thus characterizes the distortion in the map projection that it represents. The scaling factor can also be set to vary with geographic attributes such as altitude or population density to favor map projections that represent quantities other than geographic area.

For example, a scaling function that is proportional to population density, as that of figure 5, creates map projections that enlarge populous areas and shrink unpopulated ones. Alternatively, a scaling function that is high over land and small over water, as that of figure 6, creates map projections that enlarge continents and shrink oceans.

With this deformation gradient, a variety of distortion criteria can be computed for the cost function $U(\vec{\mathbf{y}})$. The obvious choice might be the Airy–Kavrayskiy criterion or some other logarithmic RMS criterion (Krisztián, 2019). This is usually evaluated with

Figure 5. A geographic function that is higher for areas that are more densely populated, with a blur applied to reduce map artifacts. Derived from data from the Center for International Earth Science Information Network – CIESIN – Columbia University (2018).

Figure 6. A geographic function that is higher over land and lower over sea, with a blur applied to mark coastal waters important, and regions south of 60°S dimmed to mark Antarctica unimportant. Derived from Natural Earth data.

respect to a and b, the minimum and maximum values of linear distortion at a point, which can be obtained from $\mathbf{F}$ (Wriggers, 2008, p. 26). While the Airy–Kavrayskiy criterion is a popular choice for optimization of polynomial map projections (Krisztián, 2019), it is unsuitable for optimization of mesh map projections. This is because, evaluated locally with constant b, it goes as $(\ln a)^2$, which is concave-down for $a > 2.72$. This diminishing slope incentivizes meshes to choose thin strips of elements to distort in the extreme, in order to minimize the distortion in adjacent elements. This leads to undesirable sharp jumps in the projection's derivatives that approximate interruptions, which can be seen in figure 7.

For the purposes of this study, a two-dimensional form of the compressible neo-Hookean strain energy density function (Pence & Gou, 2015) is implemented. This was chosen because it is physically intuitive, it is easy to compute locally given a deformation gradient, and it has multiple elasticity coefficients that can be independently adjusted: the Lamé parameters μ and λ (while μ penalizes map projections that distort length in any direction and thereby penalizes distortion of both angle and area, λ specifically penalizes map projections that distort area). In addition, unlike the Airy–Kavrayskiy criterion, the neo-Hookean strain energy density function is always concave-up in both a and b. Thus, the meshes that optimize it always have smooth derivatives, as evidenced by figure 8.

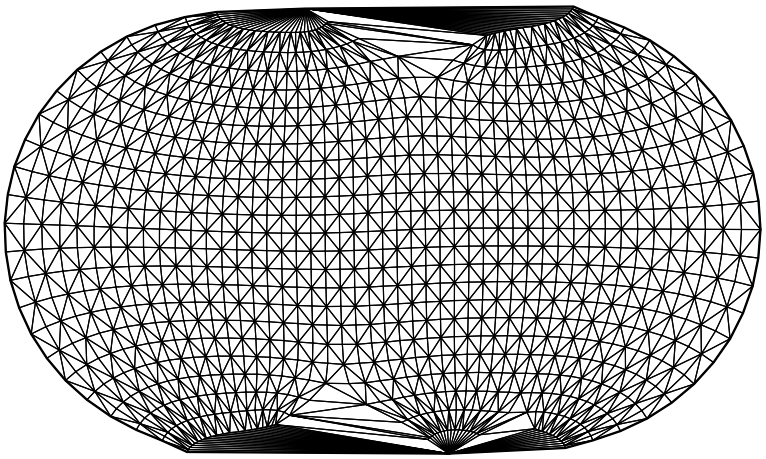

Figure 7. A mesh of $N = 1\ 129$ nodes that minimizes the global Airy–Kavrayskiy criterion. Large discontinuities in derivative are visible.

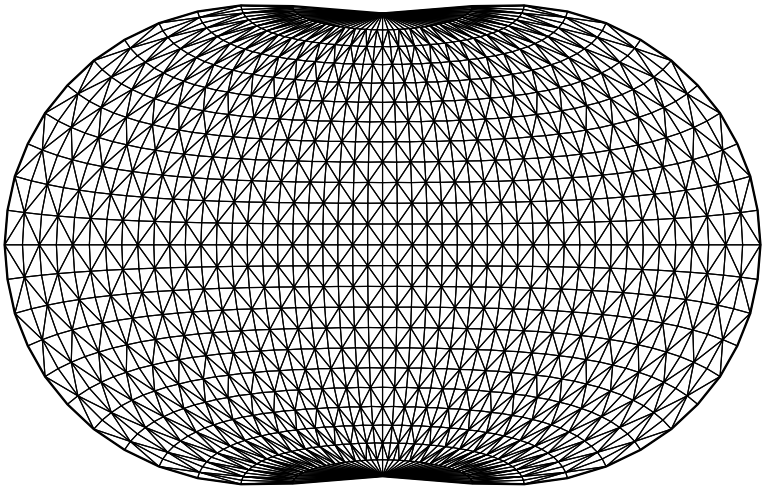

Figure 8. A mesh of $N = 1\ 129$ nodes that minimizes the total neo-Hookean elastic energy. The mesh's derivatives are smooth everywhere.

Total energy is found by evaluating the strain energy density function in each element, multiplying by that element's undeformed area, and summing up over the mesh.

$$U(\vec{\mathbf{y}}) = \sum_{j=1}^{M} w_j u_j A_j \tag{12}$$

$$u_j = \frac{1}{2}\mu(I_j - 2 - 2\ln J_j) + \frac{1}{2}\lambda(\ln J_j)^2 \tag{13}$$

$$I_j = \operatorname{tr}\left(\mathbf{F}_j \mathbf{F}_j^{\mathsf{T}}\right) = a_j^2 + b_j^2 \tag{14}$$

$$J_j = \det(\mathbf{F}_j) = a_j b_j \tag{15}$$

where w_j is the weight given to the containing cell of element j (Pence & Gou, 2015).

When w_j is uniform, the cost function values all parts of the map equally. Alternatively, a weight function that is higher in some regions of the map than in others, as

those of figures 5 and 6, values those regions more and pushes distortion in the final map projection out of them into regions that are weighted less.

In addition to the cost function itself, the gradient of the energy with respect to the node positions can be computed for later use. The gradient of the energy with respect to the position of a particular node is found by summing the gradients of the energies of all elements that have that node as a vertex.

$$\boldsymbol{\nabla}_i U(\vec{\mathbf{y}}) = \sum_{j=1}^{M} w_j \boldsymbol{\nabla}_i u_j A_j [i \text{ is a vertex of } j] \tag{16}$$

$$\boldsymbol{\nabla}_i u = \mu\left(\frac{1}{2}\boldsymbol{\nabla}_i I_j - \frac{1}{J_j}\boldsymbol{\nabla}_i J_j\right) + \lambda\left(\frac{\ln J_j}{J_j}\boldsymbol{\nabla}_i J_j\right) \tag{17}$$

$$\boldsymbol{\nabla}_i I_j = \begin{bmatrix} 2\,\mathrm{tr}\left(\frac{\partial \mathbf{F}_j}{\partial x_i}\mathbf{F}_j^{\mathsf{T}}\right) \\ 2\,\mathrm{tr}\left(\frac{\partial \mathbf{F}_j}{\partial y_i}\mathbf{F}_j^{\mathsf{T}}\right) \end{bmatrix} \tag{18}$$

$$\boldsymbol{\nabla}_i J_j = \begin{bmatrix} \mathrm{tr}\left(\mathrm{adj}(\mathbf{F}_j)\frac{\partial \mathbf{F}_j}{\partial x_i}\right) \\ \mathrm{tr}\left(\mathrm{adj}(\mathbf{F}_j)\frac{\partial \mathbf{F}_j}{\partial y_i}\right) \end{bmatrix} \tag{19}$$

$$\frac{\partial \mathbf{F}_j}{\partial x_i} = \frac{1}{2A_j}\begin{bmatrix} Y_{j2} - Y_{j3} & X_{j3} - X_{j2} \\ 0 & 0 \end{bmatrix} \tag{20}$$

$$\frac{\partial \mathbf{F}_j}{\partial y_i} = \frac{1}{2A_j}\begin{bmatrix} 0 & 0 \\ Y_{j2} - Y_{j3} & X_{j3} - X_{j2} \end{bmatrix} \tag{21}$$

where $\boldsymbol{\nabla}_i$ is the gradient with respect to the deformed coordinates of node i, and $j2$ and $j3$ are the vertices of element j other than node i.

The gradient of the cost function with respect to the state vector can then be defined by concatenating the gradients with respect to the vertices: $\boldsymbol{\nabla}_{\vec{\mathbf{y}}} U(\vec{\mathbf{y}}) = \begin{bmatrix}\boldsymbol{\nabla}_1 U(\vec{\mathbf{y}})^{\mathsf{T}} & \boldsymbol{\nabla}_2 U(\vec{\mathbf{y}})^{\mathsf{T}} & \dots & \boldsymbol{\nabla}_N U(\vec{\mathbf{y}})^{\mathsf{T}}\end{bmatrix}$.

2.2.3. Node positioning

With initial state $\vec{\mathbf{y}}_0$, cost function $U(\vec{\mathbf{y}})$, and gradient $\boldsymbol{\nabla}_{\vec{\mathbf{y}}} U(\vec{\mathbf{y}})$ in hand, any $2N$-dimensional optimization algorithm can be used to solve for the best possible position of all nodes. For this study, the limited-memory BFGS (L-BFGS) algorithm described by Nocedal (1980) with the backtracking line search described by Armijo (1966) is chosen for its speed and memory efficiency.

2.2.4. Node splitting and recombining

Because the interruptions in a map projection are at least as important as the formulas that describe its interior, the complete optimization algorithm must split and recombine nodes in addition to repositioning them. Thus, each time the node positioning optimization algorithm converges, either one node is split and an interruption is created or lengthened, or two nodes are recombined and an interruption shortens. In both cases, the position optimization algorithm then begins anew. Whether interruption is added or removed depends on the total undeformed length of all interruptions currently in the mesh, including the interruption of length πR initially present in the Hammer initial state.

(1) If the total interruption length is less than a given desired interruption length,

then a node is split to create or lengthen an interruption;

(2) if the total interruption length is greater than or equal to the desired length, then two nodes that occupy the same spherical location are re-combined, shortening or removing an interruption.

In case 1, the node to be split must first be selected. This is done by examining each node on the boundary, considering each edge that connects that node to a node not on the boundary, and computing the tension across that edge. The weighted tension on node i across edge k is defined as

$$\sigma'_{ik} = \frac{1 - w_k}{w_k} \sigma_{ik} \tag{22}$$

$$\sigma_{ik} = \frac{1}{l_k} \sum_{j=1}^{M} \vec{\mathbf{f}}_{ij} \cdot \widehat{\mathbf{n}}_{jk} [i \text{ is a vertex of } j] \tag{23}$$

where w_k is the weight of edge k, equal to the average of the weights of the two elements that border it; l_k is the undeformed length of edge k; $\widehat{\mathbf{n}}_{jk}$ is the unit vector perpendicular to edge k and pointing toward element j; and $\vec{\mathbf{f}}_{ij}$ is the force that element j exerts on node i. The latter is given by

$$\vec{\mathbf{f}}_{ij} = -w_j \boldsymbol{\nabla}_i u_j A_j \tag{24}$$

This metric favors edges where the map is heavily dilated in the direction perpendicular to that edge, and thus estimates where an interruption could relieve the most distortion. It also favors edges adjacent to less heavily weighted elements, and thus prefers to place interruptions in regions with low weight.

The node-edge pair with the highest weighted tension is then split. The node is replaced with two nodes of the same spherical and planar positions. One is set as a vertex of all adjacent elements on one side of the edge, and the other is set as a vertex to all those on the other. The undeformed length of the edge is then added to the total interruption length, and L-BFGS is restarted. When the node positioning optimization resumes, the two nodes move to different positions, creating an interruption along the selected edge. Figure 9 depicts the process.

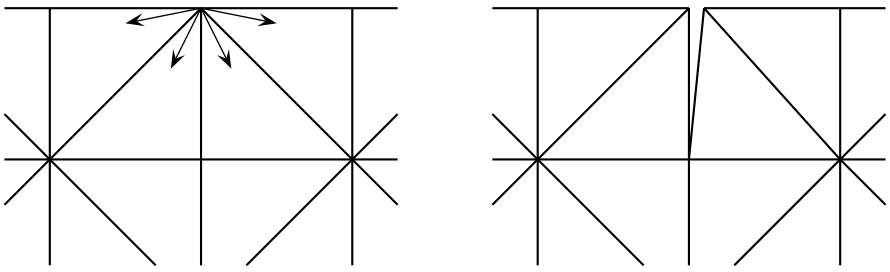

Figure 9. A section of the boundary of a mesh before and after splitting the node at the top, with force vectors determining tension indicated. The two nodes split from it are depicted in slightly different planar positions for clarity, but in fact, until node position optimization resumes, they are coincident.

In case 2, the nodes to be recombined must be selected. First of all, to prohibit the formation of gaps wholly confined by the mesh, only spherically coincident pairs of nodes that are both connected to a single third node are considered. Furthermore, spherically coincident nodes on the antimeridian that were split by the Hammer initial state are

disqualified, to prevent the loss of recognizability that this would cause in meshes with that initial state. Finally, to create a termination condition, all pairs of nodes that have already been recombined and then re-split are disqualified.

The remaining pairs of nodes are then examined, if any still qualify, and the angle they form with the adjacent third is computed. The pair with the smallest angle is then recombined. The node clockwise of the third node is removed, and the node counter-clockwise of the third node is set as the vertex of any elements that were previously adjacent to the one that was removed.

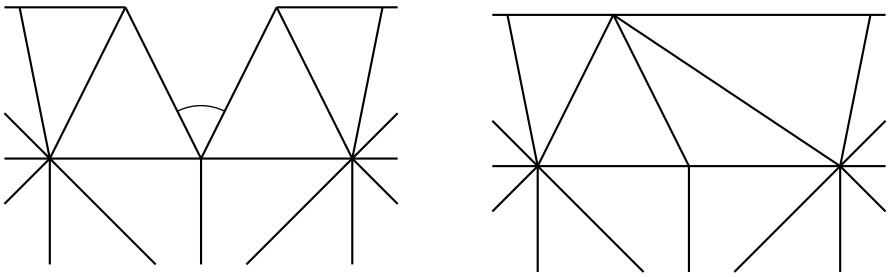

Figure 10. A section of the boundary of a mesh before and after recombining the two nodes at the top, with the angle used to select the pair indicated.

Because adding or removing new nodes changes N and thus the number of degrees of freedom in the mesh state, L-BFGS must be restarted after every node split or recombination with saved values of step length and gradient change discarded. When the interruption length is greater than or equal to the desired length but no pairs of nodes qualify for recombination, the process ends. The optimal mesh state is thus reached, and is presented as a map projection.

2.3. Interpolation

To use the mesh as a map projection, points on the globe are mapped to the plane via interpolation among the vertices of the element they occupy. The quality of the map projection depends somewhat on the interpolation method used. However, the resolution of the mesh can be increased arbitrarily, and as it approaches infinity, the interpolation method used ceases to matter.

Maps shown in this paper use bilinear interpolation:

$$x = \frac{1}{(\varphi_2 - \varphi_1)(\theta_2 - \theta_1)} \begin{bmatrix} \varphi_2 - \varphi & \varphi - \varphi_1 \end{bmatrix} \begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{bmatrix} \begin{bmatrix} \theta_2 - \theta \\ \theta - \theta_1 \end{bmatrix} \tag{25}$$

$$y = \frac{1}{(\varphi_2 - \varphi_1)(\theta_2 - \theta_1)} \begin{bmatrix} \varphi_2 - \varphi & \varphi - \varphi_1 \end{bmatrix} \begin{bmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{bmatrix} \begin{bmatrix} \theta_2 - \theta \\ \theta - \theta_1 \end{bmatrix} \tag{26}$$

where $\left[\begin{smallmatrix} \varphi_1 \\ \theta_1 \end{smallmatrix}\right]$, $\left[\begin{smallmatrix} \varphi_1 \\ \theta_2 \end{smallmatrix}\right]$, $\left[\begin{smallmatrix} \varphi_2 \\ \theta_1 \end{smallmatrix}\right]$, $\left[\begin{smallmatrix} \varphi_2 \\ \theta_2 \end{smallmatrix}\right]$ are the spherical locations of the four vertices of the element containing $\left[\begin{smallmatrix} \varphi \\ \theta \end{smallmatrix}\right]$, and $\vec{\mathbf{x}}_{11}$, $\vec{\mathbf{x}}_{12}$, $\vec{\mathbf{x}}_{21}$, $\vec{\mathbf{x}}_{22}$ are their respective planar positions. Since each element only actually has three vertices, the fourth is either obtained by counting the polar vertex twice if the element is adjacent to a pole, or extrapolated from the existing three to make a parallelogram otherwise.

This interpolation method is depicted in figure 11, and then demonstrated in figure 12 by projecting the outlines of the continents onto the mesh of figure 1.

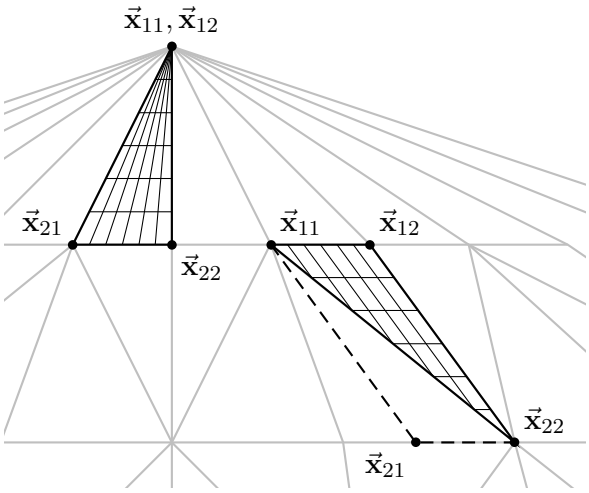

Figure 11. Graticules with 5° spacing interpolated into a polar element and a nonpolar element of a mesh, with vertices labelled.

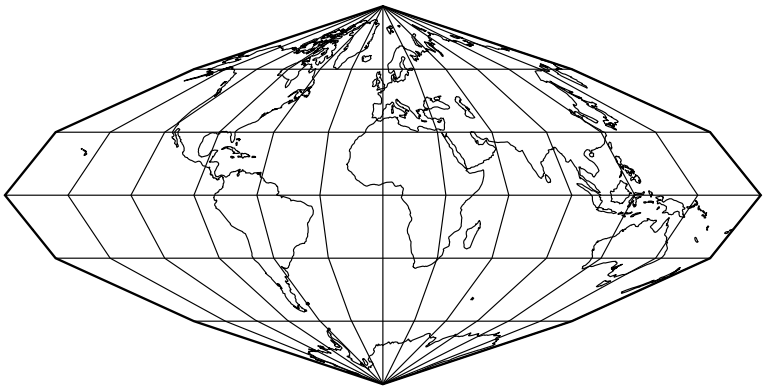

Figure 12. Shorelines and a 30° graticule, projected onto the mesh shown in figure 1, creating an approximation of a sinusoidal projection.

3. Results

Seven world map projections were created using the process outlined above. They are titled the Danseiji projections, from the Japanese words 弾性 (dansei, "elasticity") and 地 (chi, "earth").

Each map projection is presented along with the parameters that produced it. For comparison with previous analyses, logarithmic RMS areal and angular distortion criteria (Krisztián, 2019) are also estimated using the deformation gradient of each element to obtain a and b:

$$E_p = \sqrt{\frac{1}{4\pi R^2} \sum_{j=1}^{M} \left(\ln\left(a'_j b'_j\right)\right)^2 \frac{A_j}{s_j}} \tag{27}$$

$$E_a = \sqrt{\frac{1}{4\pi R^2} \sum_{j=1}^{M} \left(\ln\left(\frac{a'_j}{b'_j}\right)\right)^2 \frac{A_j}{s_j}} \tag{28}$$

where a' and b' are uniformly scaled to minimize E_p.

Unless otherwise stated, each presented map projection is based on an 80×160 cell mesh.

3.1. Danseiji N

This map projection was created from the Hammer initial state, with central meridian $\theta_0 = 0°$, Lamé parameters $\mu = 1$, $\lambda = 1$, uniform scaling, and uniform weighting. No node splitting or recombining was allowed. It has distortion criteria $E_p = .209$, $E_a = .503$. It is shown in figure 13.

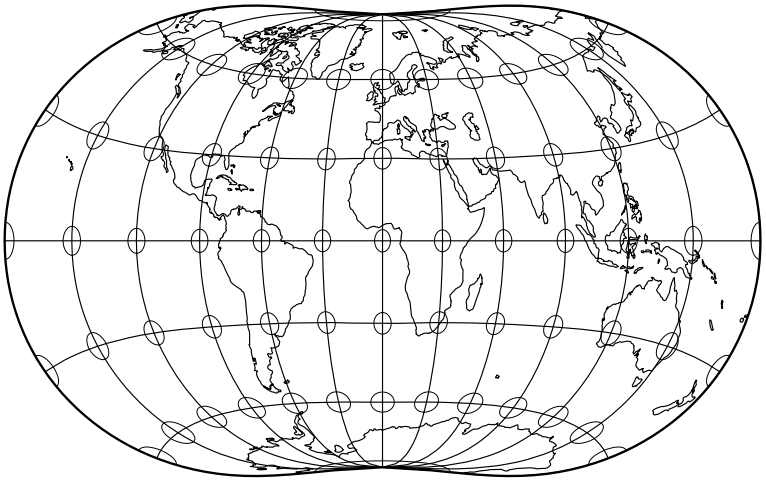

Figure 13. A Danseiji N projection of shorelines, a 30° graticule, and Tissot's indicatrix of distortion.

3.2. Danseiji I

This map projection was created from the Hammer initial state, with central meridian $\theta_0 = 0°$, Lamé parameters $\mu = 1.0$, $\lambda = 100$, uniform scaling, and uniform weighting. No node splitting or recombining was allowed. It is similar to the Danseiji N projection in all respects but λ, which is increased to place added emphasis on maintaining area equality. It has distortion criteria $E_p = .008$, $E_a = .692$. It is shown in figure 14.

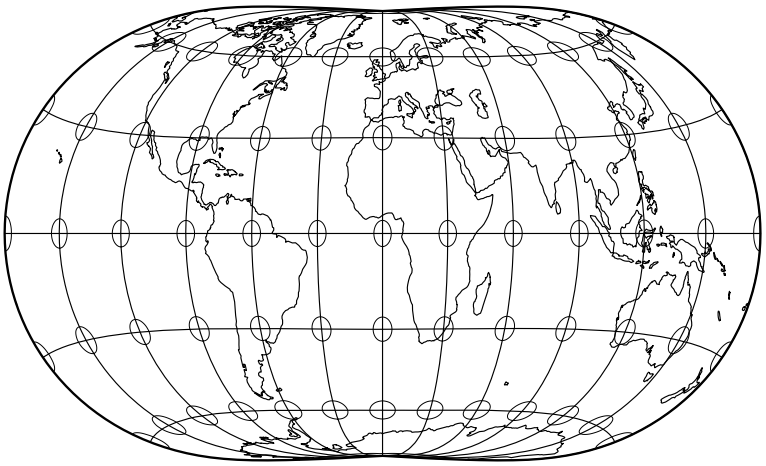

Figure 14. A Danseiji I projection of shorelines, a 30° graticule, and Tissot's indicatrix of distortion.

3.3. Danseiji II

This map projection was created from the Hammer initial state, with central meridian $\theta_0 = 0°$, Lamé parameters $\mu = 1.0$, $\lambda = 0$, uniform scaling, and uniform weighting. No node splitting or recombining was allowed. It is similar to the Danseiji N projection in all respects but λ, which is set to zero to place maximal emphasis on preserving lengths and angles. It has distortion criteria $E_p = .347$, $E_a = .387$. It is shown in figure 15.

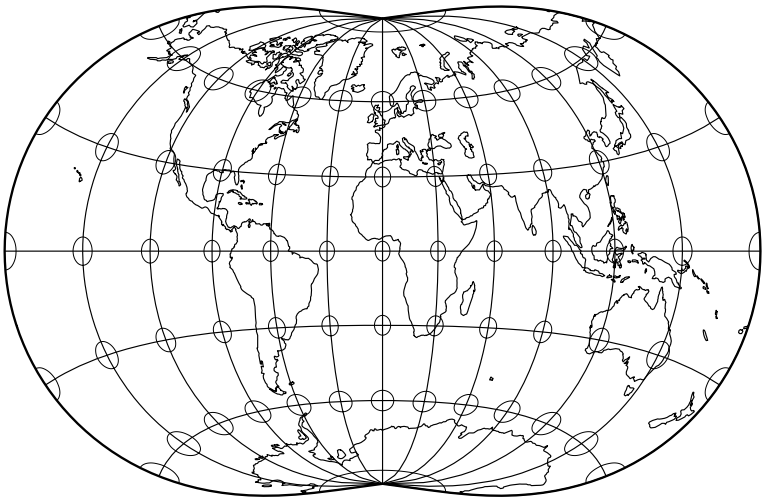

Figure 15. A Danseiji II projection of shorelines, a 30° graticule, and Tissot's indicatrix of distortion.

3.4. Danseiji III

This map projection was created from the Hammer initial state, with central meridian $\theta_0 = 11.25°$, Lamé parameters $\mu = 1.0$, $\lambda = 1.0$, uniform scaling, and weighting given by the function of figure 6 linearly mapped to the range $[0.17, 1.00]$. It was allowed to tear up to a desired interruption length of $3.926R$. It has distortion criteria $E_p = .165$, $E_a = .431$. If one sums over only land elements, those values drop to $E_p = .078$,

$E_a = .178$. It is shown in figure 16.

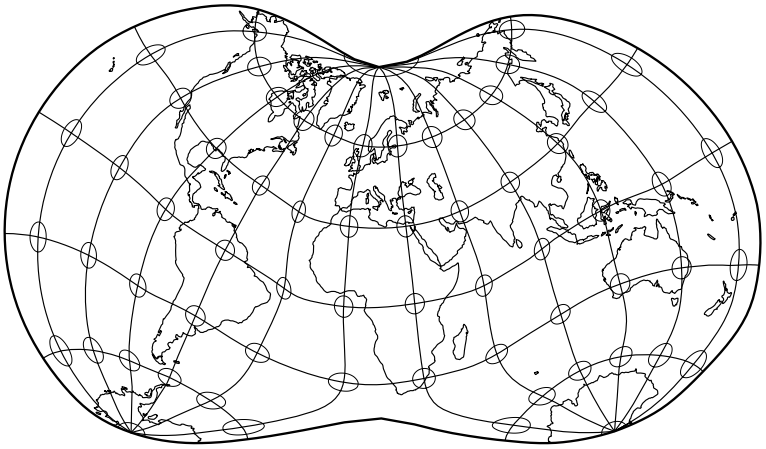

Figure 16. A Danseiji III projection of shorelines, a 30° graticule, and Tissot's indicatrix of distortion.

3.5. Danseiji IV

This map projection was created from the azimuthal initial state, with center $\varphi_0 = 49.5°$, $\theta_0 = 54.0°$, Lamé parameters $\mu = 1.0$, $\lambda = 1.0$, uniform scaling, and weighting given by the function of figure 6 linearly mapped to the range $[0.17, 1.00]$. It was allowed to tear up to a desired interruption length of $6.283R$. It has distortion criteria $E_p = .121$, $E_a = .315$. If one integrates over only land, those values drop to $E_p = .056$, $E_a = .126$. It is shown in figure 17.

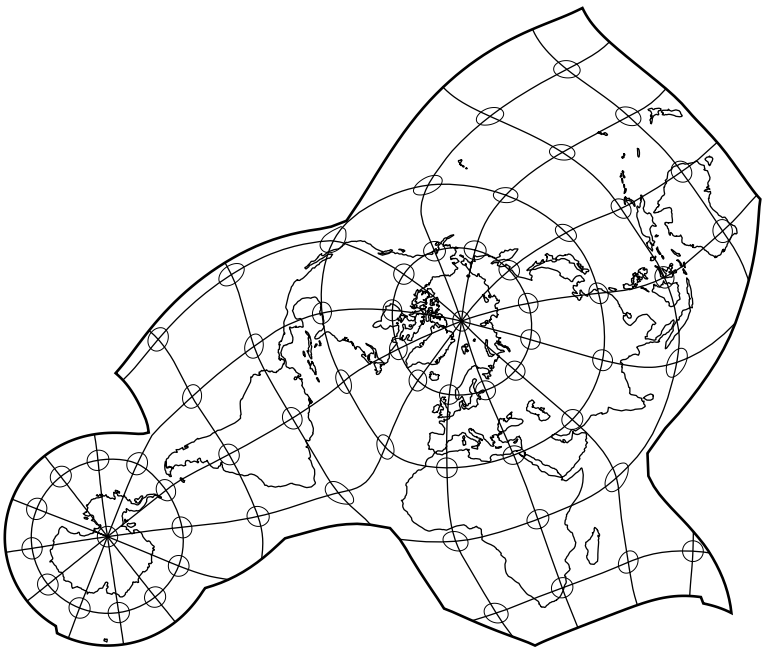

Figure 17. A Danseiji IV projection of shorelines, a 30° graticule, and Tissot's indicatrix of distortion.

3.6. Danseiji V

This map projection was created from the Hammer initial state, with central meridian $\theta_0 = 11.25°$, Lamé parameters $\mu = 1.0$, $\lambda = 0.5$, scaling given by the function of figure 6 linearly mapped to the range $[0.69, 1.38]$, and weighting given by that same function mapped to the range $[0.17, 1.00]$. No node splitting or recombining was allowed. It has distortion criteria $E_p = .563$, $E_a = .736$. It is shown in figure 18.

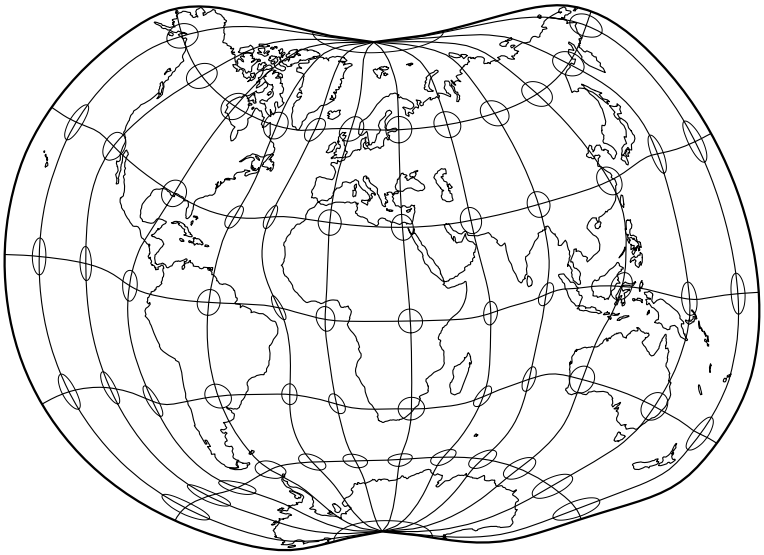

Figure 18. A Danseiji V projection of shorelines, a 30° graticule, and Tissot's indicatrix of distortion.

3.7. Danseiji VI

This map projection was created from the Hammer initial state, with central meridian $\theta_0 = 11.25°$, Lamé parameters $\mu = 1.0$, $\lambda = 1.0$, scaling given by the function of figure 5 linearly mapped to the range $[0.80, 25.60]$, and weighting given by the function of figure 6 linearly mapped to the range $[0.17, 1.00]$. It was allowed to tear up to a desired interruption length of $3.926R$. It has distortion criteria $E_p = .713$, $E_a = .756$. It is shown in figure 19.

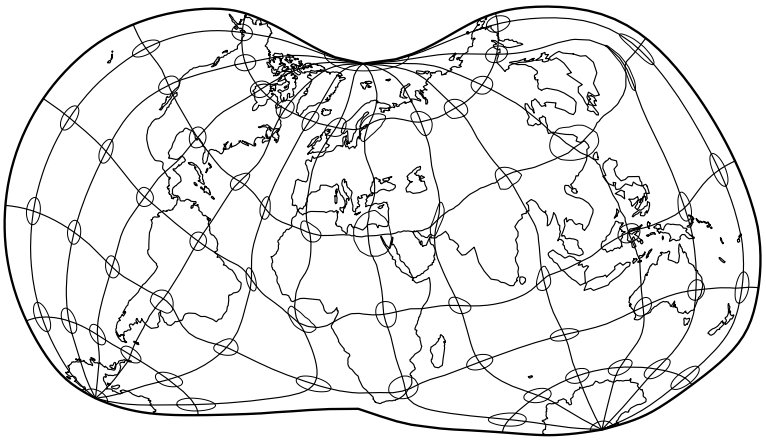

Figure 19. A Danseiji VI projection of shorelines, a 30° graticule, and Tissot's indicatrix of distortion.

4. Discussion

The Danseiji projections have several desirable properties. Foremost, each is the most accurate map projection possible, given its respective cost function as the definition of accuracy. They can all also be computed efficiently in the forward direction, and can be inverted efficiently when a precomputed inverse table is included with the mesh data, as it is in the given data files.

Being based on meshes rather than formulas also bears disadvantages, though. Most notably, the large data files required to reproduce the Danseiji projections may be infeasible to store and read in some contexts. The linear interpolation also creates discontinuities in the first derivatives, which give rise to aesthetically displeasing artifacts, visible upon close inspection. The latter effect can potentially be reduced through higher-order interpolation.

The Danseiji N, I, and II projections are the most similar to conventional map projections, as they are uninterrupted and symmetric, and are therefore the least novel of the seven. Despite this, the shapes of their parallels, relatively flat in the center and curving poleward near the edges, are unique and may be aesthetically valuable to some users. Each gives different weight to areal and angular distortion: Danseiji N has distortion criteria akin to those of Robinson or Winkel tripel; Danseiji I is approximately equal area, making it more similar to Eckert IV; and Danseiji II gives more weight to angles, placing it nearer Kavrayskiy VII (Krisztián, 2019).

The Danseiji III and IV projections are far less conventional in their dependence on Earth's geography. They push distortion out of the continents and into the oceans by adopting organic, asymmetric graticules, which would be nearly impossible to construct with simple formulas or low-order polynomials. They are similar in this respect to optimized local projections such as GS50 (Snyder, 1985). They also introduce interruptions into the oceans, though both use less total interruption than more common interrupted world map projections like Goode's homolosine (Goode, 1925) or Fuller's Dymaxion ("Life Presents R. Buckminster Fuller's Dymaxion World", 1943).

The Danseiji V and VI projections are novel in both form and function. Rather than representing geographic reality, they intentionally introduce areal distortion to emphasize more important parts of the map. Danseiji V takes the continents to be more important than the oceans, which they are on most political maps. Danseiji VI takes

importance to be proportional to population density, as it might be on some statistical maps, or to cartographers concerned about fairness. It is similar to a population cartogram, but differs in that it compromises between controlling areas and preserving angles.

These are only a few examples of map projections that can be created with the methods described above. The basic principles applied here can be modified in many ways. By changing the weight function, one could systematically optimize map projections for more specific regions and thus create mesh-based regional map projections. One could place limits on the values of x and y to force the map to conform to a certain external shape. One could impose constraints, to force true area equivalence or conformality. The latter option would create map projections of vastly different style than those presented here. The possibilities for mesh-based map projections are as rich or richer than those for formula-based ones.

There also exist many potential improvements to the methods described above. Most notably, the node splitting and recombining algorithm used in this study is fickle and highly susceptible to small changes in initial state. More advanced methods could produce more optimal placements of interruptions than those featured here.

5. Conclusion

Here is presented a new method of creating minimum-distortion map projections. They are defined by unstructured meshes rather than formulas, enabling them to take any form regardless of symmetry or differentiability. The meshes are optimized with L-BFGS to minimize neo-Hookean elastic energy, and map projections are formed by interpolating geographic features onto them. Seven examples of world map projections created with this method are presented, together covering a wide variety of use cases. However, the potential for more such map projections is vast.

These mesh-based map projections have limitations—most notably the large data files that define them. Furthermore, map projections with very specific properties such as the Mercator and azimuthal equidistant projections will always need to be mathematically derived. However, for cases where the only goal is to faithfully represent certain quantities in some parts of the map at the expense of others, mesh-based map projections are superior to ones limited by formulas.

6. Data availability statement

The source code for this study is openly available on GitHub at `https://github.com/jkunimune15/Rubber-Earth/`. The data that support the findings of this study are openly available in Mendeley Data at `http://doi.org/10.17632/x3zjcdpt7b.1`.

7. Acknowledgements

I thank Maximilian Schommer for early conceptual work that lead to this study, and Daniel Strebe and John Savard for providing technical feedback on this manuscript.

Nomenclature

a	Major semi-axis of Tissot's ellipse
b	Minor semi-axis of Tissot's ellipse
R	Radius of the globe
x	Horizontal coordinate in the plane
y	Vertical coordinate in the plane
λ	Lamé's first parameter
μ	Lamé's second parameter
φ	Latitude on the globe
θ	Longitude on the globe

References

Airy, G. B. (1861). Explanation of a projection by balance of errors for maps applying to a very large extent of the earth's surface; and comparison of this projection with other projections. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, *22*(149), 409–421. Retrieved from https://doi.org/10.1080/14786446108643179

Armijo, L. (1966). Minimization of functions having Lipschitz continuous first partial derivatives. *Pacific J. Math.*, *16*(1), 1–3. Retrieved from https://projecteuclid.org:443/euclid.pjm/1102995080

Baselga, S. (2019). TestGrids: Evaluating and optimizing map projections. *Journal of Surveying Engineering*, *145*. Retrieved from https://doi.org/10.1061/(ASCE)SU.1943-5428.0000279

Center for International Earth Science Information Network – CIESIN – Columbia University. (2018). *Gridded population of the world, version 4 (gpwv4): Population density, revision 11.* NASA Socioeconomic Data and Applications Centre (SEDAC). Retrieved from https://doi.org/10.7927/H49C6VHW

Goode, J. P. (1925). The homolosine projection: A new device for portraying the earth's surface entire. *Annals of the Association of American Geographers*, *15*(3), 119–125. Retrieved from https://www.tandfonline.com/doi/abs/10.1080/00045602509356949

Gott, J. R., Mugnolo, C., & Colley, W. N. (2007). Map projections minimizing distance errors. *Cartographica*, *42*(3), 219–234. Retrieved from https://www.utpjournals.press/doi/abs/10.3138/carto.42.3.219

Györffy, J. (2018). Minimum distortion pointed-polar projections for world maps by applying graticule transformation. *International Journal of Cartography*, *4*(2), 224–240. Retrieved from https://doi.org/10.1080/23729333.2018.1455263

Krisztián, K. (2019). Comparing finite and infinitesimal map distortion measures. *International Journal of Cartography*, *5*(1), 3–22. Retrieved from https://doi.org/10.1080/23729333.2018.1500255

Life presents R. Buckminster Fuller's Dymaxion world. (1943, March 1). *Life*, *14*(9), 41–55.

Nocedal, J. (1980). Updating quasi-Newton matrices with limited storage. *Mathematics of Computation*, *35*(151), 773–782. Retrieved from http://www.jstor.org/stable/2006193

Pence, T. J., & Gou, K. (2015). On compressible versions of the incompressible neo-Hookean material. *Mathematics and Mechanics of Solids*, *20*(2), 157–182. Retrieved from https://doi.org/10.1177/1081286514544258

Snyder, J. P. (1985). *Computer-assisted map projection research* (Bulletin No. 1629). U.S. Geological Survey. Retrieved from http://pubs.er.usgs.gov/publication/b1629

Snyder, J. P. (1987). *Map projections—a working manual* (Professional Paper No. 1395). U.S. Geological Survey. Retrieved from https://pubs.er.usgs.gov/publication/pp1395

Snyder, J. P. (1993). *Flattening the earth: Two thousand years of map projections.* The University of Chicago Press.

Tobler, W. R. (1977). Numerical approaches to map projections. *Studies in Theoretical cartography*, 51–64.
van der Grinten, A. J. (1905). New circular projection of the whole earth's surface. *American Journal of Science*, *19*, 357–366.
Wriggers, P. (2008). *Nonlinear finite element methods*. Springer, Berlin, Heidelberg.

Appendix A. Data files

Files containing the mesh states needed to generate maps in each of the seven presented map projections are available as comma-separated value files with variable row lengths. Each can be interpreted as follows.

The first row has eight values:

(1) N the number of nodes,
(2) m_1 the number of cells at each longitude,
(3) m_2 the number of cells at each latitude,
(4) l, the number of nodes in the boundary,
(5) o the width of the interpolation table,
(6) p the height of the interpolation table,
(7) w the width of the mesh in units of R, and
(8) h the height of the mesh in units of R.

Following that are N rows representing the nodes in ascending order of their indices (the first vertex has index 0). Each has two values:

(1) the node's x position in units of R, and
(2) the node's y position in units of R.

Next are $m_1 m_2$ rows representing the cells, ordered west to east, then north to south. The spherical size and coordinates of each cell can be infered from the fact that they are evenly spaced in latitude and longitude. Each has either five or seven values. In both cases, the first number indicates the type of cell.

An initial 0 indicates a polar cell—one that is adjacent to the pole and therefore did not need to be diagonally divided. Rows starting with 0 have five values:

(1) the number 0,
(2) the index of the northeastern vertex,
(3) the index of the northwestern vertex,
(4) the index of the southwestern vertex, and
(5) the index of the southeastern vertex.

Two of the four vertices will be the same, depending on whether the cell is adjacent to the north pole or the south pole.

An initial 1 indicates a nonpolar cell that was divided into a northwestern element and a southeastern element. Rows starting with 1 have seven values:

(1) the number 1,
(2) the index of the eastern vertex of the northwestern element,
(3) the index of the northwestern vertex of the northwestern element,
(4) the index of the southern vertex of the northwestern element,
(5) the index of the western vertex of the southeastern element,
(6) the index of the southeastern vertex of the southeastern element, and

(7) the index of the northern vertex of the southeastern element.

An initial -1 indicates a nonpolar cell that was divided into a northeastern element and a southwestern element. Rows starting with -1 have seven values:

(1) the number -1,
(2) the index of the northeastern vertex of the northeastern element,
(3) the index of the western vertex of the northeastern element,
(4) the index of the northern vertex of the southwestern element,
(5) the index of the southwestern vertex of the southwestern element,
(6) the index of the eastern vertex of the southwestern element, and
(7) the index of the southern vertex of the northeastern element.

Following that are l rows representing the boundary nodes, ordered counterclockwise. Each has one value:

(1) the index of the node.

Finally, the remaining *op* rows are a redundant interpolation table to aid in inverse implementations. Each row represents one point on the map, ordered left to right, then top to bottom. Their exact planar coordinates can be inferred from the fact that they are evenly spaced and include points coincident with the edge of the map. Each has two values:

(1) the point's corresponding φ in radians, and
(2) the point's corresponding θ in radians.

Points not on the mesh have extrapolated spherical coordinates.

A Python script demonstrating proper parsing of these comma-separated value files is included in the source code and named `src/example/read_danseiji.py`.

List of Figures