JPL Publication 07-3

Some Algorithms for Polygons on a Sphere

Robert.G.Chamberlain
William.H.Duquette
Jet Propulsion Laboratory

Presented at the
Association of American Geographers Annual Meeting
San Francisco, California
17–21 April 2007

National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

June 2007

JPL Publication 07-3

Some Algorithms for Polygons on a Sphere

Robert.G.Chamberlain
William.H.Duquette
Jet Propulsion Laboratory

Presented at the
Association of American Geographers Annual Meeting
San Francisco, California
17–21 April 2007

National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

June 2007

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, and was sponsored by the United States Army National Simulation Center, Fort Leavenworth, Kansas, and the National Aeronautics and Space Administration.

Errata

- A communication from Mr. Krzysztof Czernek pointed out that the first cosine term on the right-hand side of the second equation on page 11 should have been a sine. The correct equation is

$$\tan\theta_i = \begin{cases} \dfrac{\sin(\lambda_{i+1}-\lambda_i)\cos\phi_{i+1}}{\cos\phi_i \sin\phi_{i+1} - \sin\phi_i \cos\phi_{i+1}\cos(\lambda_{i+1}-\lambda_i)} & \text{if } \lambda_i \neq \lambda_{i+1} \\ (\theta_i = 0) & \text{if } \lambda_i = \lambda_{i+1} \text{ and } \phi_i < \phi_{i+1} \\ (\theta_i = \pi) & \text{if } \lambda_i = \lambda_{i+1} \text{ and } \phi_i > \phi_{i+1} \end{cases}$$

- A communication from Charles Karney pointed out that the name of the mathematician l'Huilier was misspelled near the top of page 6.

- Charles Karney also offered an alternative formula for the exact area of a spherical triangle with an apex at the South Pole that was given at the top of page 6, but it is still far too complicated to be useful in our application. His formula, in our notation, is

$$A_{i,i+1} = 2\,R^2\left[(\lambda_{i+1}-\lambda_i) + \tan^{-1}\left(\frac{-\tan\left(\frac{\lambda_{i+1}-\lambda_i}{2}\right)\left(\tan\left(\frac{\phi_i}{2}+\frac{\pi}{4}\right)+\tan\left(\frac{\phi_{i+1}}{2}+\frac{\pi}{4}\right)\right)}{1+\tan\left(\frac{\phi_i}{2}+\frac{\pi}{4}\right)\tan\left(\frac{\phi_{i+1}}{2}+\frac{\pi}{4}\right)}\right)\right]$$

— Robert G. Chamberlain
— 28 May 2013

BLANK PAGE

Abstract

A limited search for polygon algorithms for use in a new military training simulation that interfaces with several others produced only planar algorithms. To avoid having to implement several different sophisticated map projections to guarantee compatibility with all the other simulations, we opted to develop algorithms that work directly on a sphere.

The first is an algorithm to compute the area of a polygon whose edges are segments of great circles.

Since our model represents certain object locations as mathematical points, the second topic is whether a specified point is inside a specified polygon. Possibly pathological cases are identified and eliminated.

When we realized that most political boundaries are actually rhumb lines, use of the Mercator projection equations seemed unavoidable. We then reasoned that if all the edges were short enough, lat-lon lines, great circle segments, and rhumb lines would be close enough to being identical that we could use whichever was most convenient. Thence, we looked at the relationship between the maximum distances between great circle segments and rhumb lines and between lat-lon lines and rhumb lines as functions of length, azimuth, and latitude.

The final algorithm finds the area overlapped by two polygons. Again, potentially pathological cases are identified and eliminated.

Contents

Introduction

A search for polygon algorithms for use in a new military training simulation that interfaces with several other simulations found only planar algorithms. To avoid having to implement a sophisticated map projection—or worse, several different sophisticated projections in an inherently doomed attempt to achieve compatibility with all the other simulations—we opted to develop algorithms that work directly on a sphere.

The first is an algorithm to compute the area of a polygon whose edges are segments of great circles.

Our model represents such things as small groups of people as being located at a mathematical point. The second algorithm addresses the issue of whether that point is located inside a specified polygon (representing the boundaries of a "neighborhood") or outside it.

A planar polygon's edges are straight lines. On a sphere, the edges can be defined as great circle segments, rhumb lines, or *lat-lon lines*[1]. We began with great circle segments; then we realized that most "straight line" political boundaries are actually rhumb lines rather than great circle segments. Rhumb lines are straight on a Mercator projection[2], but that fact does not help us avoid translating between projections. In fact, recasting the point-in-polygon line segment intersection tests to use rhumb line edges revealed the Mercator projection equations.

We then reasoned that we could use great circle segments, rhumb lines, or lat-lon lines for the polygons' edges *if they were not too long*. But how long is too long?

A preliminary investigation suggested that an edge could be over a hundred kilometers long before the difference between rhumb lines and great circle segments exceeds the size of a pixel on our lat-lon screen display. It also appeared that rhumb lines between two points are between the great circle segments and the lat-lon lines between the same points. Furthermore, the lat-lon lines are closer to the rhumb lines than the great circle segments are. The fourth section gives quantitative comparisons between the maximum distances between these three kinds of "straight line segments" as a function of length, azimuth, and latitude. Either the screen resolution, the real-world uncertainty as to the locations of the boundaries, or the real-world footprints of the things that are modeled as being located at a point can be used to choose a value for the maximum acceptable distance.

The final section presents an algorithm for computing the area of overlap between two polygons. There are a variety of potentially pathological situations that must be considered if the polygons are allowed to have gerrymandered shapes.

---

[1] This shorthand expression is meant to mean "straight line segments connecting points on a plot of latitude vs longitude, which is also known as an equidistant cylindrical, equirectangular, or Plate–Carée projection." See [Snyder 87] or [Snyder 89]. We will refer to this projection as the "lat-lon" projection.

[2] [Snyder 87] p 38.

Preliminaries

Assume we are dealing with a polygon with *N* vertices. The polygon is simply connected (a single piece), has no holes, no edge crosses or touches another, and neither pole is inside the polygon.

The polygon is described in a counterclockwise direction[3] by a succession of vertices, numbered from 0 to *N*-1. A vertex *N*, if given, is identical to vertex 0. Note that "inside" and "outside" are defined by the requirement that the polygon be counterclockwise and/or by the requirement that neither pole be inside.

The location of each vertex is given by its latitude and longitude. In the algorithms, latitude and longitude are expressed in radians. Latitude is zero at the equator; north is positive, south is negative. The latitude of point *i* is denoted by ϕ_i. Longitude is zero at Greenwich; east is positive, west is negative. The longitude of point *i* is denoted by λ_i.

The radius of the Earth is denoted by *R*. The area of the polygon is denoted by *A* and is expressed in the square of the units used for *R*.

Area of a Polygon on a Sphere

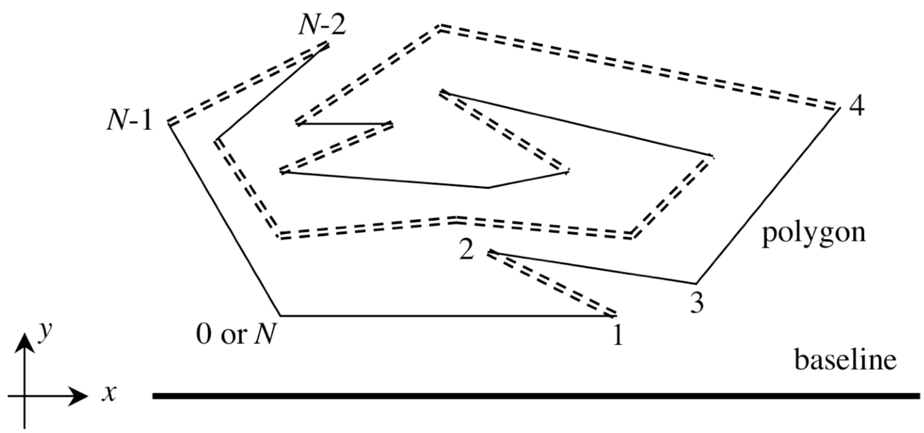
The Planar Case

Only a few changes are needed to extend the planar algorithm for use on a sphere.

First, consider the planar polygon at the top of the next page. Assume that the coordinates of its vertices are given in sequence in an overall counterclockwise direction. The sequence numbers of some of the vertices are shown in the figure: 0, 1, 2, 3, 4, …, *N*-2, *N*-1, *N*. Think of each edge that goes to the right as being a bottom surface and each edge that goes to the left as being a top surface.

Then, the area of the polygon is simply the area that is below the top surfaces and above the bottom surfaces. Or, to put it another way, the area of the polygon is the area between the top surfaces and the indicated baseline *minus* the area between the bottom surfaces and the baseline. For simplicity, the baseline must be entirely below the polygon.

[3] If the polygon is clockwise, the absolute value of the area will be the same, but the sign will be reversed (that is, negative). The point-in-polygon algorithm will be unaffected.



Planimeter Algorithm. Without loss of generality, the polygon is assumed to be described in a counterclockwise direction. Edges that go from right to left are considered to be "top" edges, those that go from left to right are "bottom" edges. Then, the area of the polygon is the sum of the areas between the top edges and the baseline minus the sum of the areas between the bottom edges and the baseline. It does not matter how far the baseline is from the polygon, as its coordinates will cancel out of the final equation.

The area, then, is the sum of the areas under the edges, taking edges that go to the right as generating negative contributions to area, and those that go to the left as positive. Vertical edges will not contribute directly to the sum, but they will cause the top surfaces to be separated from the corresponding bottom surfaces—so they most definitely will affect the area. Denoting the signed area under the edge that goes from point *i* to point *i*+1 by $A_{i,\,i+1}$ gives the following expression.

$$A = A_{01} + A_{12} + A_{23} + \cdots + A_{N-2,\,N-1} + A_{N-1,\,N}$$

For convenience, let us define the *x* coordinate as increasing to the right with zero at any convenient location, *y* as increasing upward with zero at the baseline. Then the region under the edge from point *i* to point *i*+1 is a trapezoid and its area is given by the width times the average height:

$$A_{i,\;i+1} = \left(x_i - x_{i+1}\right) \cdot \frac{\left(y_i + y_{i+1}\right)}{2}$$

Skipping the steps of using this expression in the area equation for all the values of *i*, then collecting terms in y_i gives a formula for the **area of the planar polygon**.

$$-2 \cdot A = \left(x_1 - x_{N-1}\right) \cdot y_0 + \left(x_2 - x_0\right) \cdot y_1 + \cdots + \left(x_{N-1} - x_{N-3}\right) \cdot y_{N-2} + \left(x_0 - x_{N-2}\right) \cdot y_{N-1}$$

This formula can be rewritten as

$$A = -\frac{1}{2} \sum_{i=0}^{N-1} \left(x_{i+1} - x_{i-1}\right) \cdot y_i$$

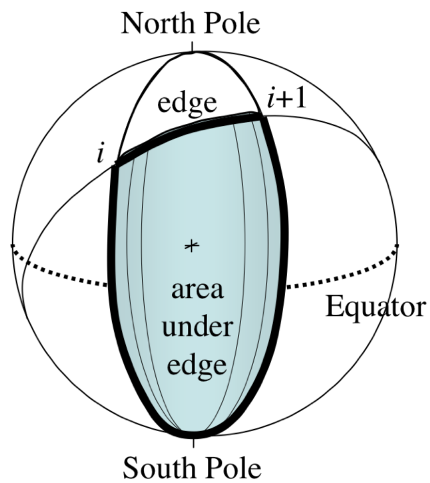
The Spherical Case — Exact Solution

When large polygons are drawn on a sphere, the area enclosed by the edges will be significantly more than would be the case if the edges were drawn on a plane. Suppose the edges are segments of great circles. The same approach can be used to compute the area as on the plane, but the baseline is replaced by the South Pole. The formula developed below will not apply to a polygon that contains either pole.

The area of the polygon can again be obtained by adding the signed areas south of each edge of the polygon:

$$A = A_{01} + A_{12} + A_{23} + \cdots + A_{N-2,\,N-1} + A_{N-1,\,N}$$

The regions whose areas are summed, however, are not trapezoids, but spherical triangles, with one vertex at the South Pole, the other two at the ends of the edge.



Area of a Polar Spherical Triangle. The spherical triangle of interest has a vertex at the South Pole, is bounded on two sides by meridians and on the other side by the polygon edge, which is a segment of a great circle.

Since one vertex is at the South Pole and two sides are along meridians, it seems like there should be a simple expression for the area. If there is, we were unable to find it. A complicated exact answer can be found by the following sequence of steps:

First, compute the great circle distance between the points, d, from the Haversine Formula, which is well conditioned for small distances.[4]

$$\sin^2 \frac{d}{2} = \sin^2 \frac{\phi_{i+1} - \phi_i}{2} + \cos\phi_i \cdot \cos\phi_{i+1} \cdot \sin^2 \frac{\lambda_{i+1} - \lambda_i}{2}$$

Thus, the three sides of the triangle have lengths d, $\frac{\pi}{2} + \phi_i$, and $\frac{\pi}{2} + \phi_{i+1}$. Compute the semi-perimeter, s, from

[4] [Chamberlain 01].

$$s = \frac{1}{2}\left(d + \left(\frac{\pi}{2} + \phi_i\right) + \left(\frac{\pi}{2} + \phi_{i+1}\right)\right) = \frac{d}{2} + \frac{\pi}{2} + \frac{\phi_i + \phi_{i+1}}{2}$$

Then, using l'Huiller's formula[5], compute the spherical excess, E, from

$$E = 4 \cdot \tan^{-1}\sqrt{\tan\frac{s}{2} \cdot \tan\left(\frac{s-d}{2}\right) \cdot \tan\left(\frac{s - \left(\frac{\pi}{2} + \phi_i\right)}{2}\right) \cdot \tan\left(\frac{s - \left(\frac{\pi}{2} + \phi_{i+1}\right)}{2}\right)}$$

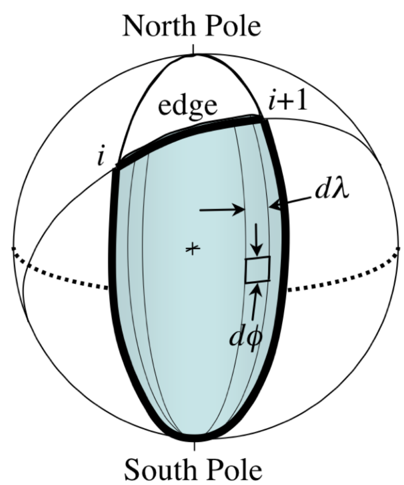
Finally, the area is given by

$$A_{i,\,i+1} = E \cdot R^2$$

This result is obviously unusable. Instead of using a looked-up formula, let us go back to the idea that there should be a simple solution that winds up looking something like the solution in the planar case.

The Spherical Case — Approximation

To get the area under an edge, let us integrate, first over latitude, then over longitude.



Area of a Polar Spherical Triangle. The integration element for computing the area of a spherical triangle with one vertex at the South Pole is bounded on two sides by meridians and on the other two sides by great circle segments through endpoints that are at the same latitude.

The N-S sides of the element of area are along a meridian, so their length is $R \cdot d\phi$. The end points of the E-W sides have the same latitude, but they are connected by a segment of a great circle, not by a line of constant latitude. As the width of the wedge approaches zero, the great circle approaches the chord, and the length of that chord is $R \cdot \cos\phi \cdot d\lambda$. Thus, the element of area is $R^2 \cdot \cos\phi \cdot d\phi \cdot d\lambda$.

[5] [Williams 06], "Some general spherical triangle formulae."

Integrating over the latitude from the South Pole to the point (φ, λ) on the edge from point *i* to point *i*+1 gives the following for the area of the *d*λ-wide slice.

$$dA = \int_{\phi=-\pi/2}^{\phi(\lambda)} R^2 \cdot \cos\phi \cdot d\phi \; d\lambda$$

Note that the upper limit is the great circle segment that connects the two points. Extreme rigor would have demanded use of a different symbol than ϕ for the variable of integration than for the boundary. Keeping that in mind, the integration over latitude yields the following.

$$dA = R^2 \cdot \sin\phi \Big|_{-\pi/2}^{\phi(\lambda)} \cdot d\lambda = R^2 \cdot \left(1 + \sin\phi(\lambda)\right) \cdot d\lambda$$

Approximating $\sin\phi(\lambda)$ by its average value throughout the longitude interval from λ_i to λ_{i+1}, then integrating, gives

$$A_{i,i+1} = \frac{R^2}{2} \cdot \left(\lambda_{i+1} - \lambda_i\right) \cdot \left(2 + \sin\phi_i + \sin\phi_{i+1}\right)$$

for the signed area under the edge that goes from point *i* to point *i*+1.

Summing the signed areas under all the edges gives the following.

$$\begin{aligned} -A \cdot \frac{2}{R^2} = {} & \left(\lambda_1 - \lambda_0\right)\left(2 + \sin\phi_1 + \sin\phi_0\right) \\ & + \left(\lambda_2 - \lambda_1\right)\left(2 + \sin\phi_2 + \sin\phi_1\right) \\ & + \left(\lambda_3 - \lambda_2\right)\left(2 + \sin\phi_3 + \sin\phi_2\right) \\ & + \cdots \\ & + \left(\lambda_N - \lambda_{N-1}\right)\left(2 + \sin\phi_N + \sin\phi_{N-1}\right) \end{aligned}$$

Many terms appear with both plus and minus signs, the polygon is closed, and point *N* is point 0. Simplifying and collecting terms by latitude gives the **area of a polygon on a sphere**:

$$A = -\frac{R^2}{2} \sum_{i=0}^{N-1} \left(\lambda_{i+1} - \lambda_{i-1}\right) \cdot \sin\phi_i$$

The similarity of this formula to the planar formula is startling.

Point in Polygon

This section discusses whether a point *Q* with coordinates $\left(\lambda_Q, \phi_Q\right)$ is inside a specified polygon. If the point is exactly on an edge or at a vertex, it is defined to be inside the polygon, though little change would be needed with other definitions.

The algorithm is based on the familiar planar algorithm[6] of constructing a test ray from the point in question to a point known to be outside the polygon (the North Pole is used here), followed by counting how many edges the ray crosses. An odd number indicates that the point is inside the polygon. Since each edge is considered in turn, vertices that have the same longitude

[6] See, for example, [Bourke 87] or [Finley 06].

as the test ray are potentially pathological. To avoid miscounting, such points (provided they are north of the test point) are treated as if they were east of the ray. Thus, those edges whose other end is westward will be counted, while those whose other end is *not* westward will not.

First, avoid performing the computationally expensive spherical trigonometry computations when possible by checking the test ray against a pre-computed bounding box before looking at any of the polygon's edges. If the test ray does intersect the bounding box, Q is checked against each of the polygon's vertices. There is no point in testing whether Q is on an edge at this point, because that will be revealed when the intersection tests are performed and the rest of the edges can be skipped at that time.

If Q is not on a vertex, the ray is compared to each edge of the polygon so the number of edge crossings can be counted.[7] North-south edges are ignored because they add an even number of crossings (either zero or two). If the longitude of the ray is not between the longitudes of the ends of the edge, there is no intersection. If both ends of the edge are in the northern hemisphere and the test point is south of the chord (on a lat-lon projection) between the end points, it intersects the edge. Only if the test point is north of that chord is it necessary to compute the latitude of the edge at the test point's longitude and compare it to the latitude of Q.

"Bullet-proof" implementation of this algorithm allows for the fact that computers have finite precision when making equality tests between real numbers and when computing the arguments for inverse trigonometric functions.

Three Kinds of Straight Lines

While the point-in-polygon algorithm is topological in nature, the bounding box and the crossing tests assume a particular shape for the edges between successive vertices. On a plane, the edges are straight-line segments, but what are they on the surface of a sphere?

The shortest distance between two points is a segment of a great circle. A line with a constant direction is a rhumb line.[8] A straight line on a lat-lon projection is neither of these. If both points are in the northern (or southern) hemisphere, the rhumb line lies between the great circle segment and the lat-lon line.

Various implications of each of these kinds of "straight lines" are discussed below. We have drawn heavily on Ed Williams' extremely useful collection of solutions of problems in spherical trigonometry.[9]

---

[7] An actual count is not needed; a Boolean variable can be flipped back and forth between *even* and *odd*.

[8] Most "straight line" political boundaries are actually rhumb lines. The oblique portion of the California–Nevada border, for example, bears southeast from a defined latitude and longitude in Lake Tahoe. See [Supreme Court 80].

[9] See [Williams 06]. While that reference contains general solutions, some of the equations used here require that the lines are no more than $\pi/2$ radians (about 6214 miles) in length and do not cross the ±180° meridian.

Lat-Lon Lines

Latitude as a Function of Longitude on a Lat-Lon Line

The expression for latitude as a function of longitude on a lat-lon line that goes through two specified points is the familiar equation of a straight line:

$$\phi = \phi_1 + \frac{\lambda - \lambda_1}{\lambda_2 - \lambda_1}(\phi_2 - \phi_1)$$

Bounding Box for Lat-Lon Edges

If the point-in-polygon test is going to be made repeatedly, it will generally be advantageous to pre-compute the maximum and minimum latitudes and longitudes for the polygon: $\phi_N, \lambda_E, \phi_S, \lambda_W$ for the northern, eastern, southern, and western limits.

The coordinates of the bounding box for polygons whose edges are lat-lon lines are simply the maxima and minima if the coordinates of the vertices.

$$\begin{aligned}
\phi_N &= \max \phi_i \\
\phi_S &= \min \phi_i \\
\lambda_W &= \max \lambda_i \\
\lambda_E &= \min \lambda_i
\end{aligned}$$

Intersection Test for Lat-Lon Edges

The purpose of the intersection test is to count the number of edges crossed by a (northerly) test ray from point Q, which might be in the polygon, to a point that is known to be outside. Each edge is examined one at a time, so if the test ray goes through a vertex, it could be counted twice, once for each of the edges that meet at the vertex. To ensure that a test ray that goes through a vertex is counted once if the other ends of the two edges are on opposite sides of the test ray and either zero or two times if the other ends are on the same side, vertices that are exactly on the test ray are treated as being to the east of the test ray.

The latitude of the crossing point, X, is as follows:

$$\phi_X = \phi_i + \left(\frac{\phi_{i+1} - \phi_i}{\lambda_{i+1} - \lambda_i}\right)(\lambda_Q - \lambda_i)$$

Interpolates in a Lat-Lon Line

Suppose an edge with endpoints A and B, with coordinates (λ_A, ϕ_A) and (λ_B, ϕ_B), is assumed to be a lat-lon line. If the lat-lon line is to be used as an approximation of a great circle segment or rhumb line, it must not be too long—as is discussed in a later section. If the maximum acceptable length is L_{max}, where should the line be split? That is, what are the lat-lon coordinates of suitable interpolated points on the edge?

First, compute the edge length,[10]

$$L = R \cdot \sqrt{(\lambda_B - \lambda_A)^2 \cos^2\left(\frac{\phi_A + \phi_B}{2}\right) + (\phi_B - \phi_A)^2}$$

where R is the radius of the Earth.

If the edge is too long, the number of points that have to be inserted, M, is the truncated quotient of L divided by L_{max}. Then the coordinates of the r-th interpolated point are:

$$\lambda_r = \lambda_A + \frac{r}{M+1} \cdot (\lambda_B - \lambda_A)$$

$$\phi_r = \phi_A + \frac{r}{M+1} \cdot (\phi_B - \phi_A)$$

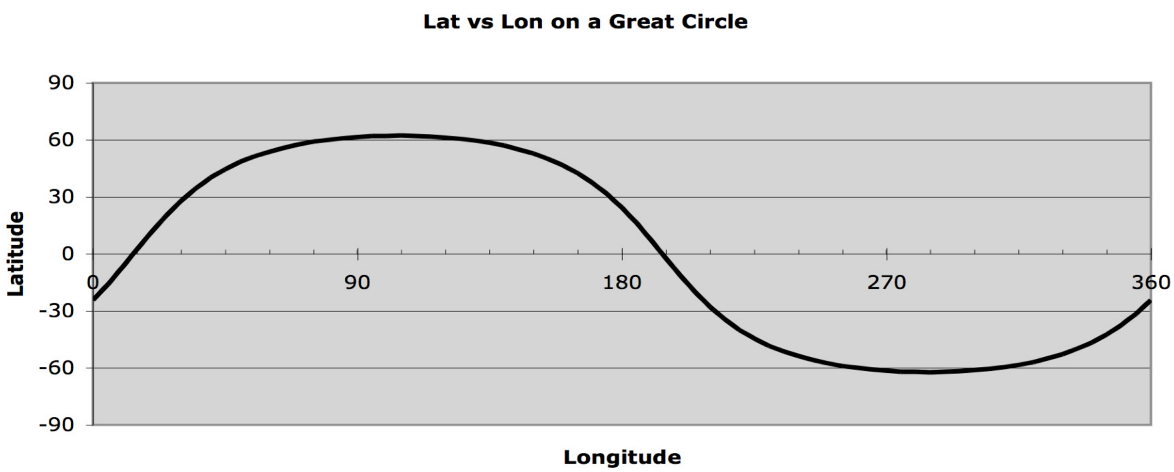
Great Circle Segments

Latitude as a Function of Longitude on a Great Circle

The formula for latitude as a function of longitude on a great circle that goes through two specified points is given by:[11]

$$\tan\phi = (\tan\phi_1)\frac{\sin(\lambda - \lambda_2)}{\sin(\lambda_1 - \lambda_2)} - (\tan\phi_2)\frac{\sin(\lambda - \lambda_1)}{\sin(\lambda_1 - \lambda_2)}$$

The figure below shows a great circle on a lat-lon projection. The equator and the meridians are the only great circles that appear as straight lines in this projection.



Latitude vs Longitude on a Great Circle. When plotted on a lat-lon projection, great circles look somewhat like blunted sine curves, as illustrated here.

---

[10] Note that the accuracy of this calculation is not too important, so there is no harm in using the average latitude. Also, keep in mind that latitudes and longitudes are assumed to be expressed in radians in all computations in this paper—though degrees are sometimes used in figures.

[11] [Williams 06], "Latitude of point on GC."

Bounding Box for Great Circle Edges

The northern and southern edges of the bounding box are defined by lines of constant latitude, and are not generally great circles. The eastern and western edges are defined by lines of constant longitude, which are great circles.

Thus, the limiting longitudes are the maximum and minimum longitudes of the vertices.

$$\lambda_W = \min_i \lambda_i \quad \text{and} \quad \lambda_E = \max_i \lambda_i$$

If the polygon is entirely in the northern hemisphere, the latitude of the southern edge of the bounding box is the minimum of the latitudes of the vertices. That is: $\phi_S = \min_i \phi_i$. If the polygon is entirely in the southern hemisphere, the northern edge is given by the maximum vertex latitude: $\phi_N = \max_i \phi_i$.

The other latitudinal boundary, however, is not so obvious, as the extreme latitude might not be at the end point of a segment. Again drawing upon Ed Williams' formulas, we have a formula for θ, the azimuth ("initial course"[12]) of the great circle segment from P_i toward P_{i+1} at P_i:

$$\tan\theta_i = \begin{cases} \dfrac{\sin(\lambda_{i+1}-\lambda_i)\cos\phi_{i+1}}{\cos\phi_i\,\cos\phi_{i+1} - \sin\phi_i\,\cos\phi_{i+1}\,\cos(\lambda_{i+1}-\lambda_i)} & \text{if } \lambda_i \neq \lambda_{i+1} \\ (\theta_i = 0) & \text{if } \lambda_i = \lambda_{i+1} \text{ and } \phi_{\text{i}} < \phi_{i+1} \\ (\theta_i = \pi) & \text{if } \lambda_i = \lambda_{i+1} \text{ and } \phi_{\text{i}} > \phi_{i+1} \end{cases}$$

If $\lambda_i = \lambda_{i+1}$ and $\phi_i = \phi_{i+1}$, the azimuth is undefined, but then the two points are identical, a case that should be removed before further processing.

Then, the maximum (and minimum) latitude on the great circle segment from point *i* to point *i* + 1, which is usually (but not necessarily) at one of the endpoints, is given by

$$\phi_{mx\,i} = \max\begin{cases} \arccos\left(\left|\sin\theta_i\,\cos\phi_i\right|\right) \\ \phi_i \\ \phi_{i+1} \end{cases}$$

Intersection Test for Great Circle Edges

The latitude of the point at which the northerly ray from Q intersects the great circle segment that goes through the end points of E_i is given by extracting ϕ_X from the following equation:

$$\tan\phi_X = (\tan\phi_i)\frac{\sin(\lambda_Q - \lambda_{i+1})}{\sin(\lambda_i - \lambda_{i+1})} - (\tan\phi_{i+1})\frac{\sin(\lambda_Q - \lambda_i)}{\sin(\lambda_i - \lambda_{i+1})}$$

[12] [Williams 06] "Course between points."

Interpolates in a Segment of a Great Circle

Suppose an edge with endpoints A and B, with coordinates (λ_A, ϕ_A) and (λ_B, ϕ_B), is assumed to be a segment of a great circle. If it is longer than the maximum acceptable length, L_{max}, what are the lat-lon coordinates of suitable interpolated points on the edge?

First, compute the edge length,

$$L = 2 \cdot R \cdot \arcsin\left(\sqrt{\sin^2\left(\frac{\phi_1 - \phi_2}{2}\right) + \cos\phi_1 \cdot \cos\phi_2 \cdot \sin^2\left(\frac{\lambda_1 - \lambda_2}{2}\right)}\right)$$

If the edge is too long, the number of points that have to be inserted, M, is the truncated quotient of L divided by L_{max}.

Next, compute, θ_A, the azimuth of the great circle at point A by the formula on the previous page. Then the coordinates of the r-th interpolated point are:[13]

$$\phi_r = \arcsin\left(\sin\phi_A \cdot \cos\left(\frac{r}{M+1} \cdot L\right) + \cos\phi_A \cdot \sin\left(\frac{r}{M+1} \cdot L\right) \cdot \cos\theta_A\right)$$

$$\lambda_r = \operatorname{mod}\left(\left[\lambda_A - \arcsin\left(\sin\theta_A \cdot \sin\left(\frac{r}{M+1} \cdot L\right) \Big/ \cos\phi_r\right) + \pi\right], 2\pi\right) - \pi$$

Rhumb Lines

Latitude as a Function of Longitude on a Rhumb Line

Ed Williams also gives formulas that relate latitude and longitude on a rhumb line through two specified points. First, calculate the constant azimuth of the shortest rhumb line between the two points.[14] [15]

$$A = \ln\left(\frac{\tan\left(\frac{\phi_2}{2} + \frac{\pi}{4}\right)}{\tan\left(\frac{\phi_1}{2} + \frac{\pi}{4}\right)}\right)$$

$$B = (\lambda_2 - \lambda_1)$$

$$\theta_{RL} = \operatorname{mod}\left\{\left[\operatorname{atan2}(B, A)\right], 2\pi\right\}$$

---

[13] [Williams 06], “Lat/lon given radial and distance.”

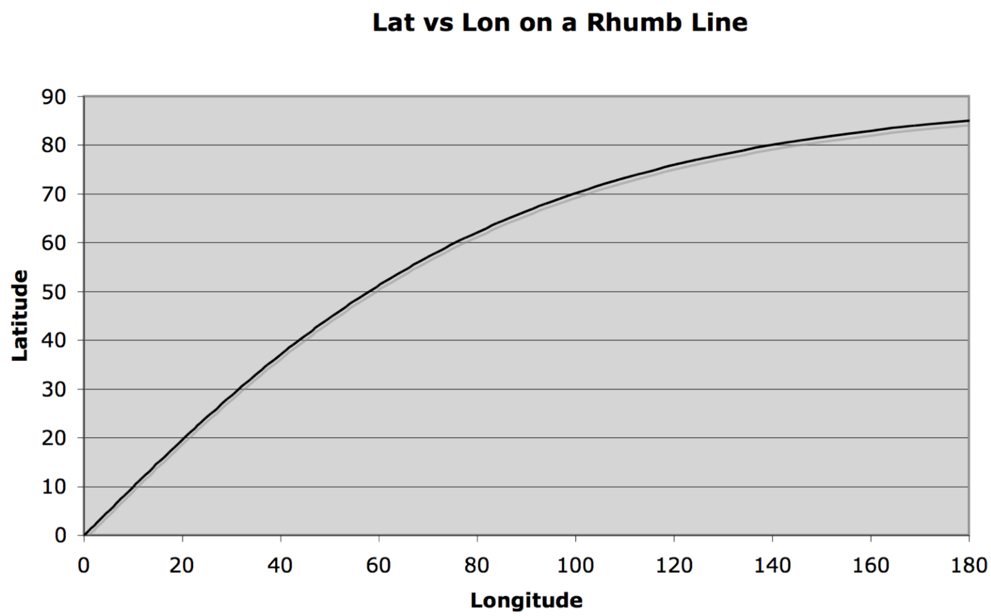
[14] As in [Williams 06], “Rhumb Line Navigation”, from which this was taken, the atan2 function “has the conventional (C) ordering of arguments, namely atan2(y,x)”, which differs from that of Microsoft Excel.

[15] Implementation note: If α is the angle between the rhumb line and due north or south given by the first quadrant result of $\alpha = \arctan(B/A)$, the atan function puts the result in the correct quadrant, then the mod function puts it in the range 0 to 2π. When the tangent of the rhumb line’s azimuth is needed, that is simply $\tan\theta_{RL} = B/A$.

Then, the formula that relates latitude to longitude can be expressed in terms of that azimuth as follows:

$$\tan\left(\frac{\phi}{2}+\frac{\pi}{4}\right)=\tan\left(\frac{\phi_1}{2}+\frac{\pi}{4}\right)\cdot\exp\left(\frac{\lambda-\lambda_1}{\tan\theta_{RL}}\right)$$

Meridians and parallels of latitude are the only rhumb lines that appear as straight lines in a lat-lon projection.



Latitude vs Longitude on a Rhumb LIne. When plotted on a lat-lon projection, rhumb lines that do not follow a meridian directly to the poles approach them asymptotically, as illustrated here.

Bounding Box for Rhumb Line Edges

Since rhumb lines are pieces of monotonically northerly-southerly spirals about both poles, the bounding box can be determined by looking only at the end points of the edges, just as with lat-lon lines: $\phi_N = \max \phi_i,\ \phi_S = \min \phi_i,\ \lambda_W = \max \lambda_i,\ \lambda_E = \min \lambda_i$

Intersection Test for Rhumb Line Edges

The latitude of the point at which the northerly ray from Q intersects the rhumb line that goes through the end points of edge E_i is given by extracting ϕ_X from the following equation:

$$\tan\left(\frac{\phi_X}{2}+\frac{\pi}{4}\right)=\tan\left(\frac{\phi_i}{2}+\frac{\pi}{4}\right)\cdot\exp\left(\frac{\lambda_Q-\lambda_i}{\tan\theta_{RL}}\right)$$

Interpolates in a Rhumb Line

Suppose an edge with endpoints A and B, with coordinates (λ_A,ϕ_A) and (λ_B,ϕ_B), is assumed to be a rhumb line. If it is longer than the maximum acceptable length, L_{max}, what are the lat-lon coordinates of suitable interpolated points on the edge?

First, compute the edge length,

$$L = R \cdot \begin{cases} \sqrt{(\phi_{\mathrm{B}} - \phi_A)^2 + \cos^2 \phi_A \cdot (\lambda_{\mathrm{B}} - \lambda_A)^2} & \text{if } \phi_A = \phi_B \\ (\phi_{\mathrm{B}} - \phi_A) \cdot \sqrt{1 + \dfrac{(\lambda_{\mathrm{B}} - \lambda_A)^2}{\ln\left(\tan\left(\dfrac{\phi_B}{2} + \dfrac{\pi}{4}\right) \Big/ \tan\left(\dfrac{\phi_A}{2} + \dfrac{\pi}{4}\right)\right)}} & \text{otherwise} \end{cases}$$

If the edge is too long, the number of points that have to be inserted, *M*, is the truncated quotient of *L* over L_{max}. Then the coordinates of the *r*-th interpolated point are:

$$\phi_r = \phi_A + \frac{r}{M+1} \cdot \cos \theta_{RL}$$

$$q = \begin{cases} \cos \phi_{\mathrm{A}} & \text{if } \phi_A = \phi_B \\ (\phi_r - \phi_A) \Big/ \ln\left(\tan\left(\dfrac{\phi_B}{2} + \dfrac{\pi}{4}\right) \Big/ \tan\left(\dfrac{\phi_A}{2} + \dfrac{\pi}{4}\right)\right) & \text{otherwise} \end{cases}$$

$$\lambda_r = \lambda_A + \frac{r}{M+1} \cdot \frac{\sin \theta_{RL}}{q}$$

How Long is Too Long?

Under what circumstances does it matter which kind of "straight line" we use?

Implied differences in edge length are small,[16] not particularly important in our context, and can be computed, if needed, from the formulas given in this paper.

The area of a polygon has only a second-order dependence on the exact shape of the edges unless the polygon is *extremely* gerrymandered. That is, a little fuzziness along the edges has only a small effect on the bulk of the area. In fact, if the polygon is entirely in either the northern or the southern hemisphere, the differences between the different definitions considered here is systematic. Consequently, errors near the southern edges will tend to cancel the errors near northern edges.

Only when the edge location is inspected closely—as in the point-in-polygon algorithm—does the exact shape of the edge matter. Consequently, we examined the maximum separation between the three kinds of "straight line" segments.

Closed-form expressions for those distances promised to be so complicated that they would provide no insight. So, we took the alternative approach of varying length, azimuth, and latitude,

[16] [Williams 06] notes that the distance between LAX (Los Angeles) and JFK (New York) on a rhumb line is only 1% greater than the great circle distance (2164.6 nm versus 2144 nm). [Alexander 04] notes on pages 355–356 that the worst possible ratio of rhumb line distance to great circle distance is π/2, or 57% too long. Lat-lon lines are a little worse.

then computing the distance between points for many fractions of the way along each line, then recording the maximum separation. We used this information to produce the following tables.[17]

The contents of the cells in the tables answer the question that titles this section: **How long an edge is too long?** Specifically,

1. Select a table for M_{SL}, the maximum acceptable difference in location of a point on the edge if the edge were to be defined as a rhumb line or as a lat-lon line; *or* select a table for M_{GC} (starting p. 19), the maximum acceptable difference between a great circle and a lat-lon line.
   - Find the cell for an edge for which the absolute value of the latitude is between the numbers given along the left side of the table, and
   - The absolute value of the azimuth (the angle with the meridian) is between the numbers along the bottom of the table.
2. The number in the cell is the critical edge length. That is, there would be a point somewhere along a longer edge whose location, depending on which definition is used, would differ by more than the maximum acceptable value.

Max_{SL}, Acceptable Distance between Rhumb Line & Lat-Lon Line = 1 meter
Cells contain Maximum Edge Length in kilometers

Latitude in ° \ Azimuth in °	0–10	10–20	20–30	30–40	40–50	50–60	60–70	70–80	80–90
70–80	8	6	5	5	5	5	4	4	4
60–70	11	8	7	7	7	7	7	8	8
50–60	14	10	9	8	8	8	9	10	12
40–50	16	12	10	10	10	10	10	12	16
30–40	19	14	12	12	12	12	12	14	19
20–30	23	17	15	14	14	14	15	17	23
10–20	29	21	18	17	17	17	18	21	29
0–10	41	30	26	25	25	25	26	30	42

[17] It should be noted that these tables are the conceptual equivalent of a function with three arguments: the latitude of the more southwesterly end point, the azimuth toward the other end point, and the maximum acceptable distance between the rhumb line and the other kind of "straight" line segment. This function produces—that is, the content of the tables is—a maximum edge length that satisfies those conditions. All three of the arguments must be known to use these tables.

Max$_{SL}$, Acceptable Distance between Rhumb Line & Lat-Lon Line = 5 meters
Cells contain Maximum Edge Length in kilometers

Latitude in ° \ Azimuth in °	0–10	10–20	20–30	30–40	40–50	50–60	60–70	70–80	80–90
70–80	17	12	11	10	10	10	11	12	17
60–70	24	17	15	14	14	14	15	17	24
50–60	30	22	19	18	18	18	19	22	30
40–50	36	26	23	21	21	21	23	26	36
30–40	42	31	27	25	25	25	27	31	43
20–30	51	37	32	30	30	30	32	38	51
10–20	64	47	41	38	38	38	41	47	64
0–10	91	67	58	54	54	54	58	67	92

Max$_{SL}$, Acceptable Distance between Rhumb Line & Lat-Lon Line = 10 meters
Cells contain Maximum Edge Length in kilometers

Latitude in ° \ Azimuth in °	0–10	10–20	20–30	30–40	40–50	50–60	60–70	70–80	80–90
70–80	23	17	15	14	14	14	15	17	23
60–70	33	24	21	20	20	20	21	24	33
50–60	42	31	27	25	25	25	27	31	42
40–50	50	37	32	30	30	30	32	37	50
30–40	60	44	38	36	36	36	38	44	60
20–30	72	53	45	43	43	43	46	53	72
10–20	90	66	57	54	54	54	57	66	91
0–10	127	93	81	76	76	76	81	95	130

Max_{SL}, Acceptable Distance between Rhumb Line & Lat-Lon Line = 20 meters
Cells contain Maximum Edge Length in kilometers

Latitude in ° \ Azimuth in °	0–10	10–20	20–30	30–40	40–50	50–60	60–70	70–80	80–90
70–80	33	24	21	20	20	20	21	24	33
60–70	47	34	30	28	28	28	30	34	47
50–60	59	43	37	35	35	35	37	43	59
40–50	71	52	45	42	42	42	45	52	71
30–40	84	62	53	50	50	50	53	62	85
20–30	101	74	64	60	60	60	64	74	102
10–20	127	93	80	75	75	75	81	93	128
0–10	177	131	113	107	107	107	115	133	183

Max_{SL}, Acceptable Distance between Rhumb Line & Lat-Lon Line = 50 meters
Cells contain Maximum Edge Length in kilometers

Latitude in ° \ Azimuth in °	0–10	10–20	20–30	30–40	40–50	50–60	60–70	70–80	80–90
70–80	51	38	32	31	31	31	33	38	52
60–70	73	54	47	44	44	44	47	54	74
50–60	92	68	59	55	55	55	59	68	93
40–50	111	82	70	66	66	66	71	82	112
30–40	132	97	84	79	79	79	84	97	133
20–30	159	117	101	95	95	95	101	117	161
10–20	198	146	126	118	118	119	127	147	202
0–10	275	204	177	167	157	168	179	209	288

Max_{SL}, Acceptable Distance between Rhumb Line & Lat-Lon Line = 100 meters
Cells contain Maximum Edge Length in kilometers

Latitude in ° \ Azimuth in °	0–10	10–20	20–30	30–40	40–50	50–60	60–70	70–80	80–90
70–80	72	53	46	43	43	43	46	53	73
60–70	103	76	65	62	62	62	66	76	104
50–60	130	95	82	77	77	77	83	96	131
40–50	157	115	99	93	93	93	99	115	158
30–40	186	137	118	111	111	111	118	137	188
20–30	223	164	142	133	133	133	142	165	227
10–20	277	204	177	167	167	167	178	207	285
0–10	380	283	247	233	233	235	252	294	405

Max_{SL}, Acceptable Distance between Rhumb Line & Lat-Lon Line = 200 meters
Cells contain Maximum Edge Length in kilometers

Latitude in ° \ Azimuth in °	0–10	10–20	20–30	30–40	40–50	50–60	60–70	70–80	80–90
70–80	101	74	64	60	60	60	64	75	103
60–70	145	107	92	87	87	87	93	107	147
50–60	183	134	116	109	109	109	116	135	185
40–50	220	162	140	131	131	131	140	163	223
30–40	262	192	166	156	156	156	167	194	266
20–30	313	230	199	187	187	188	201	233	320
10–20	387	286	248	234	234	235	251	292	402
0–10	523	392	343	325	325	328	352	411	569

Max$_{SL}$, Acceptable Distance between Rhumb Line & Lat-Lon Line = 500 meters

Cells contain Maximum Edge Length in kilometers

Latitude in ° \ Azimuth in °	0–10	10–20	20–30	30–40	40–50	50–60	60–70	70–80	80–90
70–80	157	116	100	94	94	95	101	118	161
60–70	227	167	145	136	136	136	146	169	232
50–60	286	211	182	171	171	172	183	213	292
40–50	344	253	219	206	206	207	221	257	352
30–40	408	301	261	245	245	246	262	305	419
20–30	487	360	312	294	294	295	315	367	504
10–20	598	445	387	365	365	367	393	458	632
0–10	787	597	525	500	500	505	545	639	889

Max$_{GC}$, Acceptable Distance between Rhumb Line & Great Circle = 1 meter

Cells contain Maximum Edge Length in kilometers

Latitude in ° \ Azimuth in °	0–10	10–20	20–30	30–40	40–50	50–60	60–70	70–80	80–90
70–80	8	6	5	4	4	4	4	3	3
60–70	11	8	7	6	5	5	5	5	5
50–60	13	10	8	7	7	6	6	6	6
40–50	16	12	10	9	8	8	7	7	7
30–40	19	14	12	10	9	9	9	8	8
20–30	23	17	14	12	11	11	10	10	10
10–20	29	21	17	15	14	13	13	12	12
0–10	41	29	24	22	20	19	18	18	17

Max_{GC}, Acceptable Distance between Rhumb Line & Great Circle = 5 meters
Cells contain Maximum Edge Length in kilometers

Latitude in ° \ Azimuth in °	0–10	10–20	20–30	30–40	40–50	50–60	60–70	70–80	80–90
70–80	17	12	10	9	8	8	7	7	7
60–70	24	17	14	12	11	11	10	10	10
50–60	30	21	18	16	14	14	13	13	13
40–50	35	25	21	19	17	16	16	15	15
30–40	42	30	25	22	20	19	18	18	18
20–30	51	36	30	27	24	23	22	22	22
10–20	64	45	38	33	31	29	28	27	27
0–10	90	65	54	48	44	41	40	39	39

Max_{GC}, Acceptable Distance between Rhumb Line & Great Circle = 10 meters
Cells contain Maximum Edge Length in kilometers

Latitude in ° \ Azimuth in °	0–10	10–20	20–30	30–40	40–50	50–60	60–70	70–80	80–90
70–80	23	17	14	12	11	11	10	10	10
60–70	33	24	20	17	16	15	15	14	14
50–60	42	30	25	22	20	19	18	18	18
40–50	50	36	30	26	24	23	22	21	21
30–40	59	42	35	31	29	27	26	25	25
20–30	71	51	42	37	34	32	31	30	30
10–20	89	64	53	47	43	41	39	38	38
0–10	126	91	75	67	61	58	56	55	54

Max_{GC}, Acceptable Distance between Rhumb Line & Great Circle = 20 meters
Cells contain Maximum Edge Length in kilometers

Latitude in ° \ Azimuth in °	0–10	10–20	20–30	30–40	40–50	50–60	60–70	70–80	80–90
70–80	32	23	19	17	16	15	14	14	14
60–70	46	33	28	24	22	21	20	20	20
50–60	58	42	35	31	28	27	25	25	25
40–50	70	50	42	37	34	32	31	30	30
30–40	84	60	50	44	40	38	36	36	35
20–30	100	72	60	53	48	46	44	43	43
10–20	126	90	75	66	61	57	55	54	53
0–10	176	127	106	94	86	81	78	77	77

Max_{GC}, Acceptable Distance between Rhumb Line & Great Circle = 50 meters
Cells contain Maximum Edge Length in kilometers

Latitude in ° \ Azimuth in °	0–10	10–20	20–30	30–40	40–50	50–60	60–70	70–80	80–90
70–80	51	36	30	27	25	23	22	22	22
60–70	73	52	43	38	35	33	32	31	31
50–60	92	66	54	48	44	42	40	39	39
40–50	110	79	66	58	53	50	48	47	47
30–40	131	94	78	69	63	60	57	56	56
20–30	158	113	94	83	76	72	69	67	67
10–20	197	141	117	104	95	90	86	85	84
0–10	273	198	165	147	135	128	123	121	121

Max_{GC}, Acceptable Distance between Rhumb Line & Great Circle = 100 meters
Cells contain Maximum Edge Length in kilometers

Latitude in ° \ Azimuth in °	0–10	10–20	20–30	30–40	40–50	50–60	60–70	70–80	80–90
70–80	71	51	43	38	35	33	31	31	30
60–70	103	74	61	54	50	47	45	44	44
50–60	129	92	77	68	62	59	56	55	55
40–50	155	111	92	82	75	71	68	66	66
30–40	185	132	110	97	89	84	81	79	78
20–30	222	159	132	117	107	101	97	95	94
10–20	275	198	165	146	134	127	122	119	119
0–10	377	275	231	205	189	179	173	171	170

Max_{GC}, Acceptable Distance between Rhumb Line & Great Circle = 200 meters
Cells contain Maximum Edge Length in kilometers

Latitude in ° \ Azimuth in °	0–10	10–20	20–30	30–40	40–50	50–60	60–70	70–80	80–90
70–80	100	72	60	53	49	46	44	43	43
60–70	144	103	86	76	70	66	63	62	61
50–60	182	130	108	96	88	83	79	78	77
40–50	219	157	130	115	106	99	96	94	93
30–40	260	186	155	137	126	118	114	111	111
20–30	311	223	186	164	151	142	137	134	133
10–20	384	278	231	205	189	178	172	168	168
0–10	519	381	320	286	265	252	244	240	240

Max$_{GC}$, Acceptable Distance between Rhumb Line & Great Circle = 500 meters

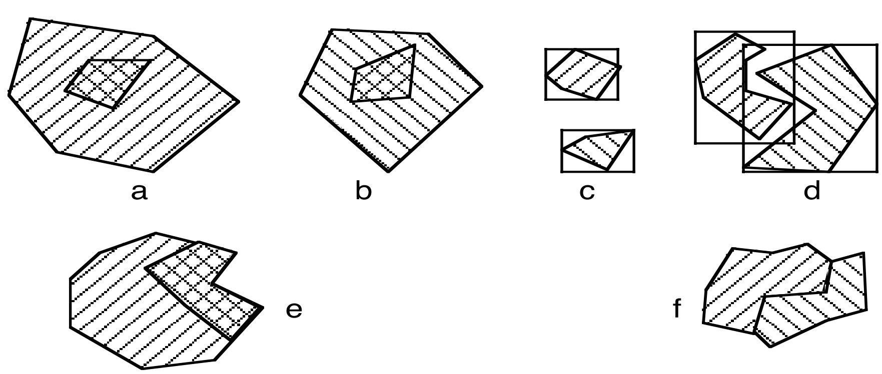
Cells contain Maximum Edge Length in kilometers

Latitude in ° \ Azimuth in °	0–10	10–20	20–30	30–40	40–50	50–60	60–70	70–80	80–90
70–80	156	112	93	83	76	72	69	68	68
60–70	225	162	135	119	110	103	99	97	97
50–60	284	204	170	150	138	130	125	122	122
40–50	342	246	204	181	166	157	151	148	147
30–40	405	292	243	215	198	186	179	176	175
20–30	483	349	291	258	237	224	216	211	211
10–20	593	432	361	321	296	280	270	266	265
0–10	782	581	491	441	410	392	381	378	378

Area Overlapped by Two Polygons

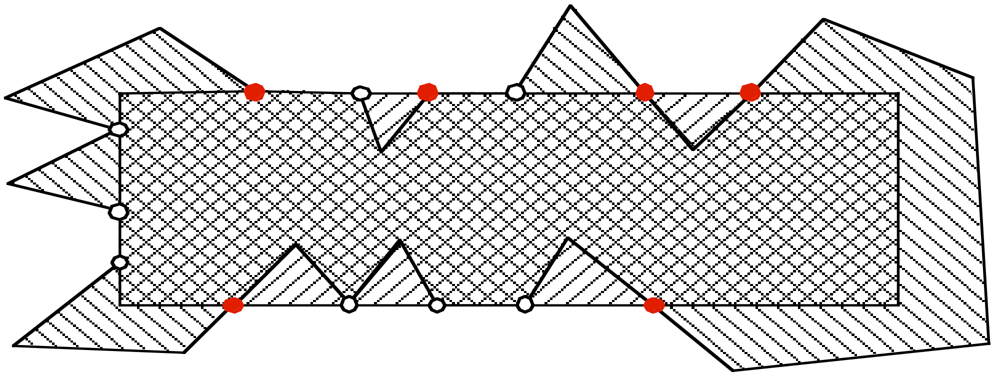
Suppose we have two polygons, say *P* and *T*, that may have arbitrarily gerrymandered shapes. The issue is to determine the area shared by both polygons.

To find that area, first check for simple cases, such as non-overlapping bounding boxes and one polygon being completely inside the other.



The No-Crossing Cases. Either polygon can be completely inside the other (as in a or b), even if they share some vertices or edges (as in e). They can be so thoroughly separated that their bounding boxes do not intersect (as in c). Their bounding boxes can intersect, but the polygons themselves do not (as in d), even though they may share some vertices or edges (as in f).

If the situation is not one of these simple cases, some preparation of the polygons is in order. In particular, find all of the places where edges of *P* and *T* intersect each other, including end points of edges that lie on top of each other, and insert those additional vertices into both polygons as needed.



Crossings—Add Points As Needed. Every edge in one polygon is compared to every edge in the other polygon to identify all of the intersection points and regions of overlap. Not all vertices that are on both polygons are crossing points, so it is necessary to inspect them further. Following the perimeter of one of the polygons (in a counterclockwise direction), the final shared vertex before the perimeter changes from being inside to being outside (or *vice versa*) is marked as a crossing point (shown in the figure as solid circles; some of the other shared vertices are shown as hollow circles).

When done, both polygons contain all of the points they share. Then, follow the circumference of one of the polygons, say *P*, marking all of the crossings. This is a little tricky, because *P* might "bounce off" *T* instead of crossing it. Only the actual crossings are marked. For use in the extraction of sub-polygons, a record of the vertex number in the other polygon can be kept, either with the crossing points themselves[18] or in a separate table.

Then, the overlapped regions can be extracted by tracing along successive vertices of one of the polygons, say *P*, always in sequence, which is counterclockwise, until arriving at a vertex marked as a crossing. Then, an overlap polygon, *S*, is extracted by adding vertices from *P* to *S* until the next crossing is found. Then follow the other polygon, *T*, adding points from *T* to *S*, until the next crossing is found. Then follow *P*, then *T*, and so on, until back at the first of the points in *P* that is in *S*. While constructing *S*, unmark vertices as they are used to change from *P* to *T* or *vice versa*.

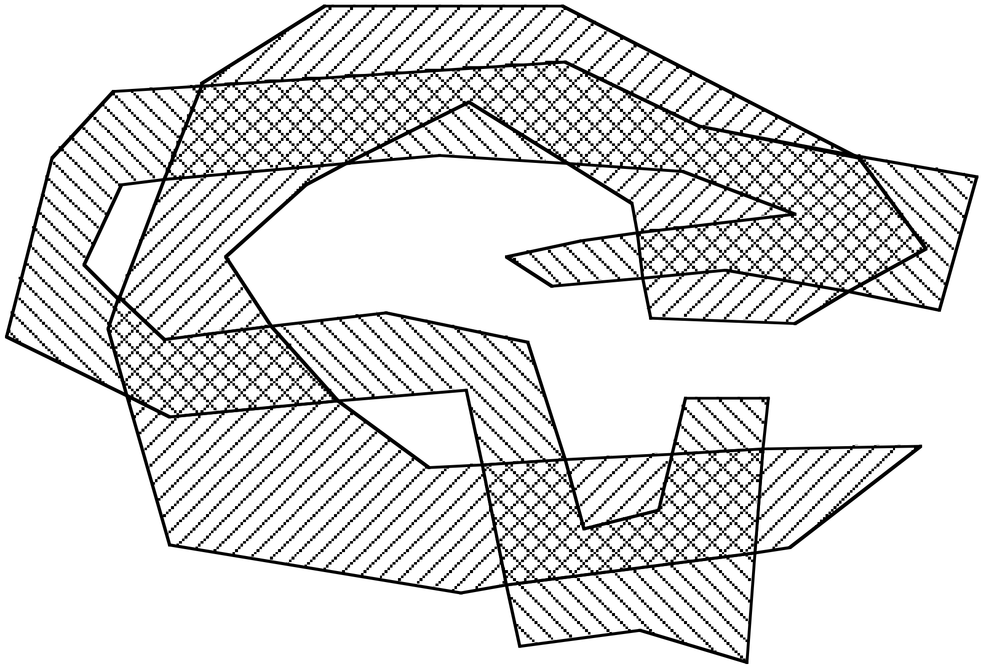
Once *S* has been completed, its area can be computed by the algorithm described earlier[19] and added to an accumulating total. Then *S* can be emptied and used again.

---

[18] In fact, the mark could be that sequence number itself, with a null value indicating that a vertex is not a crossing.

[19] The area computation requires an assumption about the shape of the edges. The relative error caused by assuming the wrong shape of the edges, however, is small. Consider the total area of the lens-shaped regions between each pair of vertices bounded by a great circle segment and a rhumb line. The relative error caused by using the great circle segments is the area of all the lenses divided by the area of the polygon itself. If the edges are short enough, the lenses are very slender, so the relative error is very small.

Having disposed of an overlap polygon and unmarked all of the vertices on *P* corresponding to the overlap polygon, proceed along *P*, repeating the process at the next marked vertex, extracting another overlap polygon *S* (if there are any), until back at the original starting point on *P*.



Area of Overlapping Polygons. Starting outside the other polygon, follow the perimeter of one of the polygons to a crossing point. Then extract the shared sub-polygon by alternating between polygons, changing polygons at each crossing point. Unmark each crossing point as it is used so that the subpolygons will only be extracted once each. Accumulate the areas of the subpolygons as they are extracted. The polygon must be followed in the same direction (e.g., counterclockwise) as was used for marking crossing points to deal properly with the shared vertices indcated by hollow circles in the previous figure.

References

[Alexander 04] James Alexander, "Loxodromes: A Rhumb Way to Go," Mathematics Magazine, Vol. 77, No. 5, December 2004, as downloaded on 10 November 2006 from <http://www.case.edu/artsci/math/alexander/mathmag349-356.pdf>.

[Bourke 87] Paul Bourke, "Determining if a Point Lies on the Interior of a Polygon," November 1987, as downloaded on 31 August 2006 from <http://local.wasp.uwa.edu.au/~pbourke/geometry/insidepoly/>.

[Chamberlain 01] Robert G. Chamberlain, "Q5.1: What is the best way to calculate the distance between 2 points?" February 2001, as downloaded on 2 November 2006 from <http://www.usenet-replayer.com/faq/comp.infosystems.gis.html>.

[Finley 06] Darel Rex Finley, "Point-in-Polygon Algorithm—Determining Whether a Point Is Inside a Complex Polygon", as downloaded on 31 August 2006 from <http://alienryderflex.com/polygon/>, 2006.

[Snyder 87] John P. Snyder, *Map Projections—A Working Manual*, United States Geological Survey Professional Paper 1395, Superintendent of Documents, United States Government Printing Office, Washington, D. C. 20402, 1987.

[Snyder 89] John P. Snyder and Philip M. Voxland, *An Album of Map Projections*, United States Geological Survey Professional Paper 1453, United States Government Printing Office, Washington, D. C. (for sale by the Books and Open-File Reports Section, United States Geological Survey, Federal Center, Box 25425, Denver, CO 80225.), 1989.

[Supreme Court 80] U.S. Supreme Court, "CALIFORNIA v. NEVADA, 447 U.S. 125 (1980) 447 U.S. 125 CALIFORNIA v. NEVADA. ON EXCEPTIONS TO REPORT OF SPECIAL MASTER. No. 73, originally argued 14 April 1980, decided 10 June 1980, as downloaded on 2 November 2006 from http://caselaw.lp.findlaw.com/scripts/getcase.pl?court=us&vol=447&invol=125. 1980.

[Wikipedia 06] Wikipedia Definition of "Loxodrome," as downloaded on 7 November 2006 from <http://en.wikipedia.org/wiki/Rhumb_lines>.

[Williams 06] Ed Williams, "Aviation Formulary V1.43," as downloaded on 31 August 2006 from <http://williams.best.vwh.net/avform.htm>, 2006.