**ELLIPSE EVALUATOR v0.1**

**User Manual**

# Table of Contents

# Introduction

**Operation**

The program runs by placing user-defined ellipses at a set of user-defined positions, then evaluating what fraction of each ellipse meets the constraints set for each raster layer than the ellipse intersects with. There are two modes of operation.

***Grid mode***

The centroid of the ellipse is positioned at a grid of points defined in lat-lon space.

***Azimuth mode***

The centroid of the ellipse is fixed and the user defines minimum and maximum azimuth at which the ellipse should be evaluated, and a step that defines the interval that the ellipse is rotated between the limits.

**Layer Tab**

Adding layers

Displaying layers

Removing layers

**Ellipse Tab**

Creating an ellipse

Previewing an ellipse

**Evaluate Tab**

Setting the evaluation mode

Defining a grid for ellipse placement

Defining

Optimizing

Exporting results

**Raster Layers**

Ellipse Evaluator performs its calculations in the Lambert Equal Area Cylindrical map projection (Snyder, 1987). This is required so that areas are evaluated equivalently throughout the workspace. The projection parameters used are:

* R = 3396190 (equatorial radius in the IAU 2000 Mars shape model)
* lat1 = 0 (latitude of the standard parallel is the equator)
* lon0 = 0 (projection is centered at 0° longitude)

Each raster layer that is loaded must therefore be transformed to this projection in order for the calculation to be correct. Table 1 shows supported data types and their respective map-projections.

For Mars datasets, equidistant cylindrical is often used in equatorial regions, while polar stereographic is used for polar latitudes.

For PDS products with labels, this projection is denoted by the PDS keyword ‘map\_projection\_type’ having a value of ‘CYLINDRICAL EQUAL AREA’.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data type |  |  |  |  |
| IMG files |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Table 1. Supported layer data formats and map-projections.

**Known Issues/Future Developments**

***Fractional pixels***

Sometimes pixels could be large compared to ellipse size, e.g. MOLA 128ppd MEGDR’s are still ~468m/pixel at the equator. For a landing ellipse of a few km’s, whether partial pixels that are not wholly covered by the ellipse are included or not, has a large effect on the results. The effect increases with the ratio of pixel size to ellipse size.

Another effect is that fractions of the ellipse covered by a raster may not be equal to 1 even though the entire ellipse is covered by the raster. This is because the operation that decides whether a pixel is inside the ellipse or not uses the pixel centre. It can therefore be the case that a pixel centre resides inside the ellipse boundary, but some of the pixel does not. Therefore, more pixel area is included than actually is inside the ellipse, so that when the fraction of the ellipse covered by the raster is calculated, it is > 1. Currently a kluge is in place in *optimizeXYCore.m* to set fractional areas > 1, to 1.

However, there is no kluge for the opposite case: where part of a pixel is inside the ellipse, but the pixel centre is not, so it is not counted. This results in the fractional area being wrongly < 1.

Fix:

Instead of simply including or excluding a mask of areas, *optimizeXYCore.m* should calculate the fraction of each pixel that is inside the ellipse boundary and multiply this by the area covered by the pixel.

***Ellipse Azimuth***

Presently, the only ellipse parameters that can be varied in a single optimization are the ellipse centre coordinates. Ideally one would be able to also optimize within a range of ellipse azimuths.

Fix:

This would require a new core optimization function that operates in a slightly different way, e.g. *optimizeAzCore.m*, or that *optimizeXYCore.m* would need to be sufficiently modified. The main additions would be that the work area would be defined by drawing ellipses at all azimuths in the user-defined range (instead of drawing them placed at all XYs in the range), and that the ellipse loop would run over azimuth instead of XY. This would be a good opportunity to make the routine more universal, so that ellipses with any properties could be drawn and assessed.

***Actual Optimization***

While the name of the program suggests it optimizes variables, it currently just computes their values over a parameter space. The intention was always to have the program semi-analyse the data in some way to indicate the set of parameters that produce the optimal result.

Fix:

Analysis of the output arrays would have to be semi-informed. For example, perhaps there is a small area where an ellipse centre could be placed that would be far optimal than it’s surroundings, but it could be a very small area so the chances of actually landing a spacecraft specifically within an ellipse centred there are low. The program would need some contextual checks to avoid suggesting such situations. Perhaps it’s unnecessary, given that humans will always carefully interpret the results.

***Ellipse Shape***

It must be confirmed that the shape of n-sigma ellipse is commensurate with the shape of the ellipse produced by engineering studies.

Fix:

It is unclear how the boundary of the actual ellipse is calculated using the dispersion of possible trajectories. In any case, there should be precise agreement between engineering-produced ellipse *n*-sigma boundaries and the ellipse that is drawn. This may require developing capability to import ellipse vertices from data resulting from engineering studies.