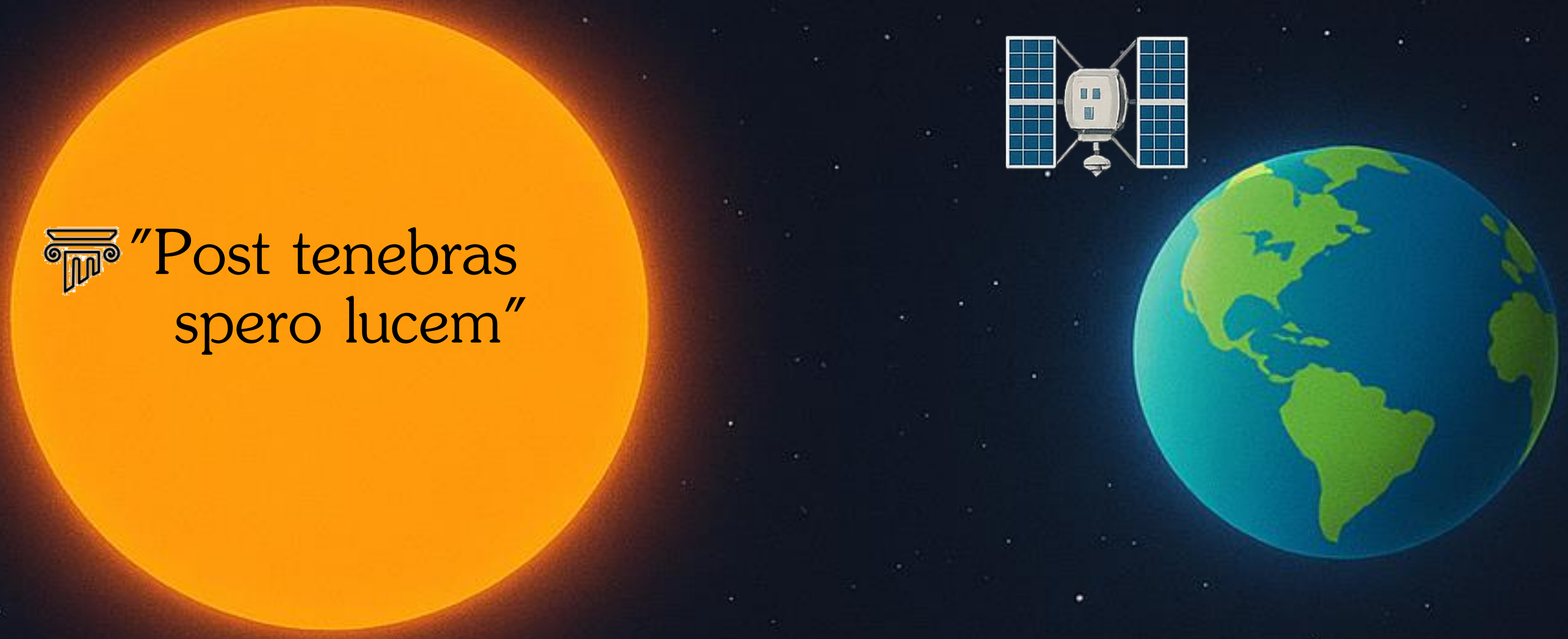
PercentShadow Algorithm UserFunction for GMAT

for Satellite Eclipse Modeling

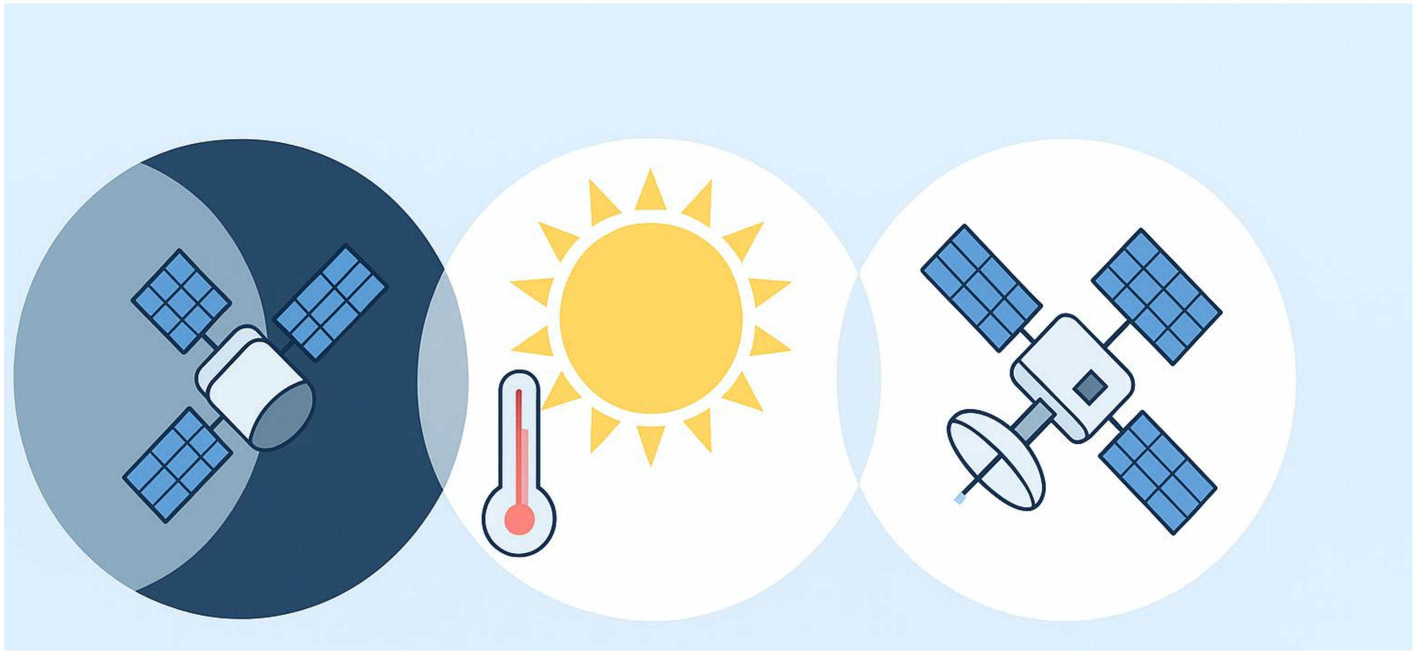


Based on Montenbruck & Gill (2000)

Adapted By Sacha J. Tholl

Significance

- Provides smooth shadow transitions for precise power modeling.
- Essential for solar panel sizing, thermal analysis.
- Standard method in satellite mission design.



Purpose of the PercentShadow Algorithm

- Compute how much of the Sun is visible to a spacecraft.
- Output is percent sunlight
  - 0% = full shadow, or „Umbra“ → no covarage
  - >0% - >100%: Penumbra, Atumbra → partial coverage
  - 100% = full sun → total coverage
- Usefull for eclipse modeling in mission analysis within Report computations and visualizations into XY-plots.

⚡ **Use as a GMAT-Userfunction**

👉 Allows for precise calculation of the overlap of two circles in the satellite's sky.

👉 Models exactly what we would see visually as a pilot inside of the satellite.

Purpose of the PercentShadow Algorithm

Imagine:

👉 You are the satellite.

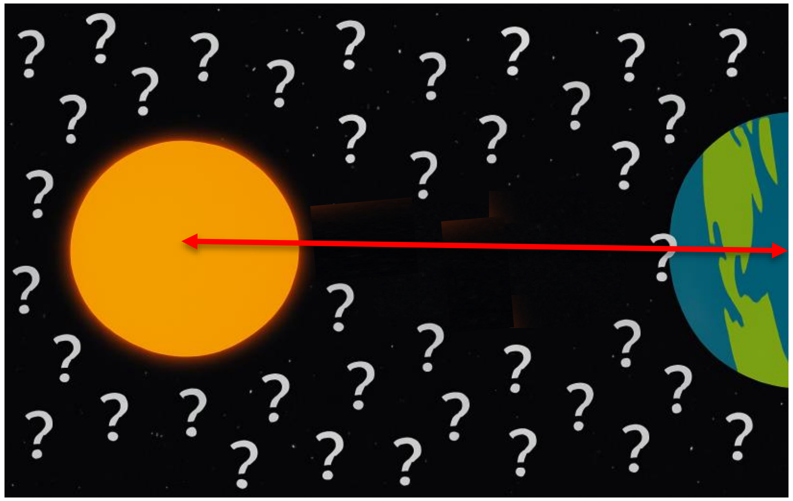
👉 In the sky, you see **two circles**:

One circle constitutes the sun

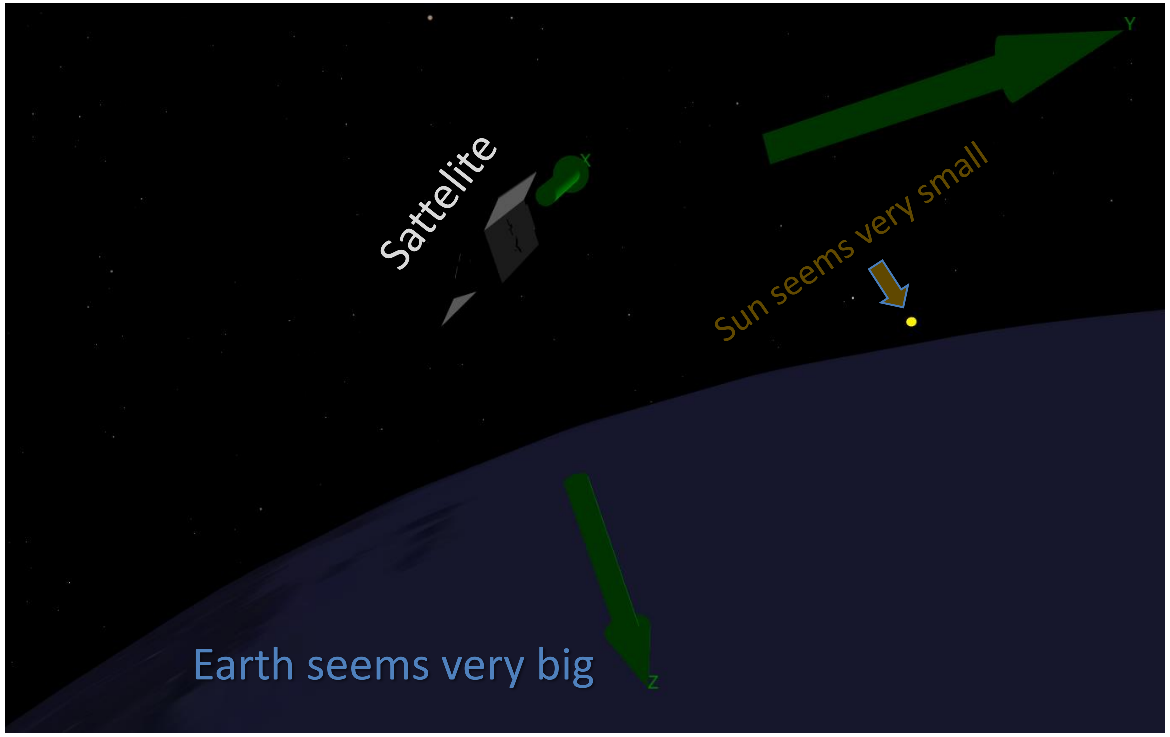
The other circle stands for the Earth (or another occulting body)

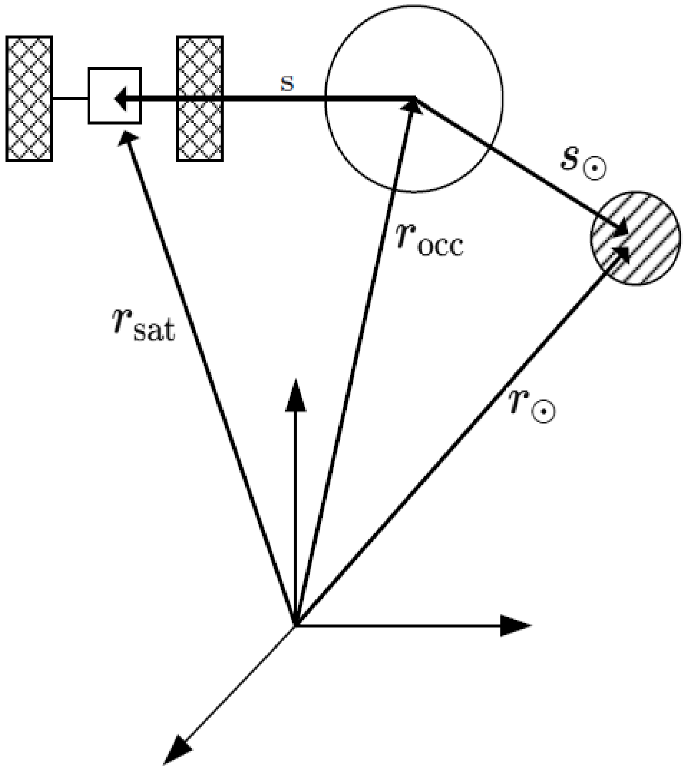
Both appear as “disks” in the satellite's sky because they are very far away → the question is:

→ How far apart are the centers of these two disks in the sky?



Geometry of the Shadow





🌞 **Apparent distance between two discs of apparent sizes**

(Note that when the occulting body is the same as the central body, $\mathbf{s} = \mathbf{r}$, and $\mathbf{s}_\odot = \mathbf{r}_\odot$)

1) Geometry: Vectors and Apparent Sizes

These vectors describe the relative geometry in space:

- satellite position relative to occulting body.

$$\vec{s} = \vec{r}_{\text{sat}} - \vec{r}_{\text{occ}}$$

→ Vector from the occulting body to the satellite

- Sun position relative to occulting body.

$$\vec{s}_{\odot} = \vec{r}_{\odot} - \vec{r}_{\text{occ}}$$

→ Vector from the occulting body to the Sun

2) Apparent Radii Between Disks

These radii are angular quantities that determine the apparent size of the Sun and the occulting body in the satellite's sky.

- Apparent radius of the Sun:

$$R'_\odot = \arcsin\left(\frac{R_\odot}{|\vec{r}_\odot - \vec{r}_{\text{sat}}|}\right)$$

→ Apparent angular radius of the Sun as seen from the satellite

- Apparent radius of occulting body:

$$R'_{\text{occ}} = \arcsin\left(\frac{R_{\text{occ}}}{|\vec{s}|}\right)$$

→ Apparent angular radius of the occulting body as seen from the satellite

3) Apparent Distance Between Disks

- Compute D' = angle between Sun and occulting body as seen by spacecraft.

$$D' = \arccos\left(\frac{\vec{s}\cdot(\vec{r}_\odot - \vec{r}_{\text{sat}})}{|\vec{s}|\,|\vec{r}_\odot - \vec{r}_{\text{sat}}|}\right)$$

→ This is the angular distance between the centers of the two discs in the sky of the satellite.

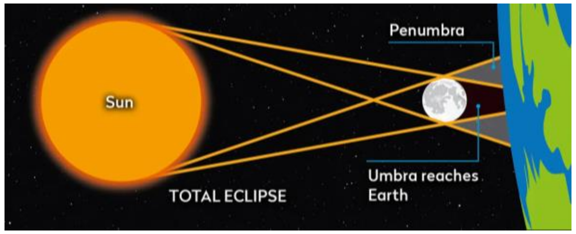
4) Decision Logic

- If $D' \geq R'_\odot + R'_{\text{occ}}$:

  → 100% sunlight.

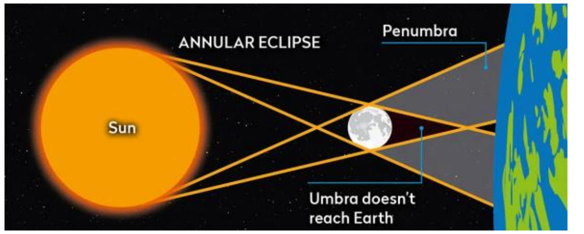
- If $D' \leq |R'_\odot - R'_{\text{occ}}|$:
  - If $R'_{\text{occ}} \geq R'_\odot$:

    → p = 0% sunlight (Umbra).



  - If $R'_{\text{occ}} < R'_\odot$:

    → p = 100% sunlight (annular eclipse).



- Else

  → partial shadow, compute overlap (Penumbra)

  → Compute the overlapping areas between the two circles

  →See next page!!

4) Overlap Area Between Sun and Occulting Body

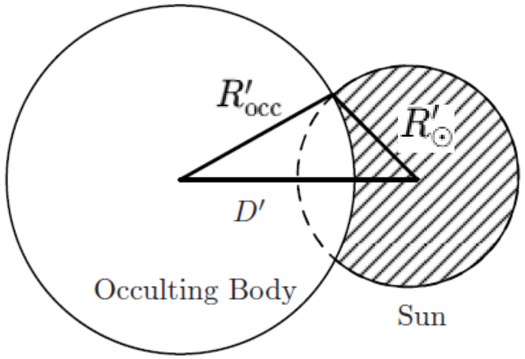
The A-Term calculates the area where the Sun and the occulting body (e.g. Earth, Moon) overlap as seen from the spacecraft. This is essential for determining the percent of sunlight received.

$$A = R'^2_\odot \arccos\left(\frac{D'^2 + R'^2_\odot - R'^2_{\text{occ}}}{2D'R'_\odot}\right) + R'^2_{\text{occ}} \arccos\left(\frac{D'^2 + R'^2_{\text{occ}} - R'^2_\odot}{2D'R'_{\text{occ}}}\right)$$

$$- \frac{1}{2}\sqrt{(-D' + R'_\odot + R'_{\text{occ}})(D' + R'_\odot - R'_{\text{occ}})(D' - R'_\odot + R'_{\text{occ}})(D' + R'_\odot + R'_{\text{occ}})}$$

Where:

R'_Sun	= apparent radius of the Sun (radians)
R'_Occ	= apparent radius of the occulting body (radians)
D'	= apparent angular distance between centers (radians)



Notes:

- The first two terms compute the segment areas of each circle.
- The last term computes the area of the lens-shaped overlap region.
- This is an exact solution for circle overlap geometry.
- Suitable for all overlap configurations (small overlap, annular eclipse, full umbra).

Result: The total overlap area A is equal to the sum of the arc areas minus the chord quadrilateral area.

Partial Shadow: Overlap Area

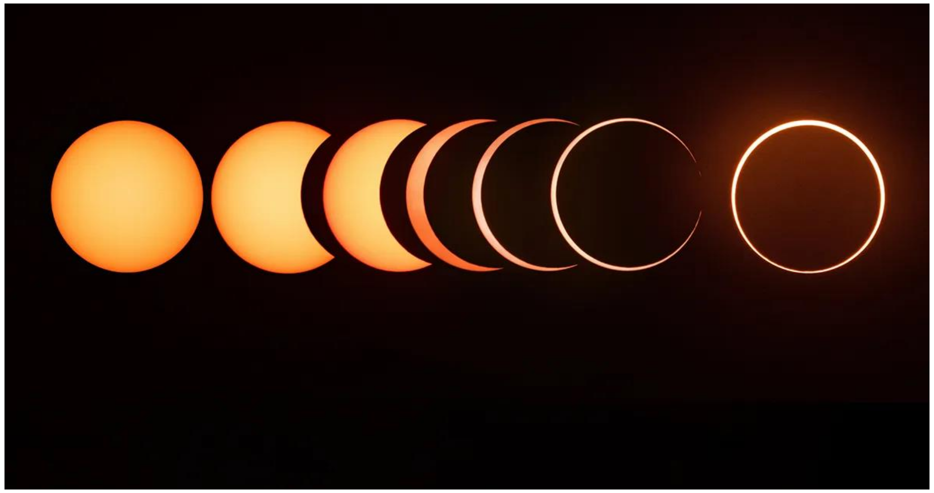
- Use geometry formula combining arcs and chord area:

The percent shadow can be calculated using

$$p = 100\frac{A}{\pi R'^2_\odot}$$

If the condition $|R'_\odot - R'^2_{\text{occ}}| < D' < R'_\odot + R'_{\text{occ}}$ is not satisfied, then the eclipse is annular, and we use

$$p = 100\frac{R'^2_{\text{occ}}}{R'^2_\odot}$$



Before coding - remember:

- Simplify the equations into meaningful portions.
- Never use equations inline **If** statements,
- Only compare variables against one other variable
- Inside the gmf file, you must also use the BeginMissionSequence;
- Pay attention not to swap the position of your variables when using more than one.
- Don't forget to "Create Variable" also for your output variable:

```
function [output1, output2] = FunctionName(Input1, input2, inputN)
```

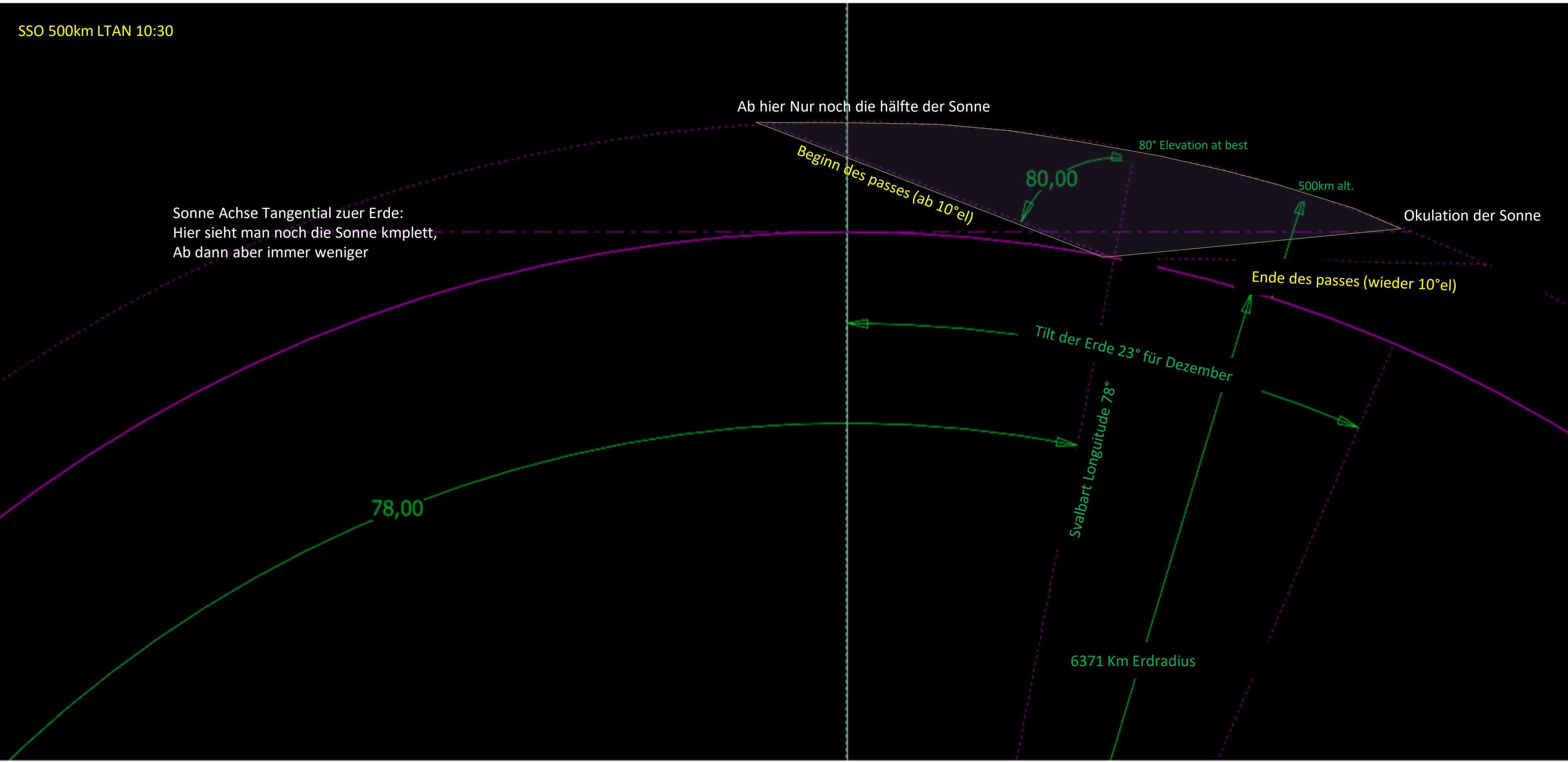
Code of the gmf-function

```
function [IsSunLit, PercentShadow] = IsSunLitGMF(r_sat_x, r_sat_y, r_sat_z, r_sun_x, r_sun_y, r_sun_z,     r_occ_x, r_occ_y, r_occ_z, R_occ)

Create Variable s_x s_y s_z;
Create Variable s_mag;
Create Variable s_sun_x s_sun_y s_sun_z;
Create Variable s_sun_mag;
Create Variable R_sun;
Create Variable R_sun_apparent R_occ_apparent;
Create Variable D_apparent dotp;
Create Variable IsSunLit PercentShadow;
Create Variable sum_R_apparent diff_R_apparent;
Create Variable min_R_apparent;

R_sun = 695700.0; % km - physical radius of the Sun

BeginMissionSequence;

% Compute satellite position relative to occulting body
s_x = r_sat_x - r_occ_x;
s_y = r_sat_y - r_occ_y;
s_z = r_sat_z - r_occ_z;
s_mag = sqrt(s_x^2 + s_y^2 + s_z^2);

% Compute Sun position relative to occulting body
s_sun_x = r_sun_x - r_occ_x;
s_sun_y = r_sun_y - r_occ_y;
s_sun_z = r_sun_z - r_occ_z;
s_sun_mag = sqrt(s_sun_x^2 + s_sun_y^2 + s_sun_z^2);

% Compute apparent angular radius of Sun
R_sun_apparent = asin(R_sun / sqrt((r_sun_x - r_sat_x)^2 + (r_sun_y - r_sat_y)^2 + (r_sun_z - r_sat_z)^2));

% Compute apparent angular radius of occulting body
R_occ_apparent = asin(R_occ / s_mag);

% Compute apparent angular distance between disk centers
dotp = (s_x * (r_sun_x - r_sat_x) + s_y * (r_sun_y - r_sat_y) + s_z * (r_sun_z - r_sat_z)) / (s_mag * sqrt((r_sun_x - r_sat_x)^2 + (r_sun_y - r_sat_y)^2
+ (r_sun_z - r_sat_z)^2));

D_apparent = acos(dotp);

% Precompute sum and difference of apparent radii
sum_R_apparent = R_sun_apparent + R_occ_apparent;
diff_R_apparent = abs(R_occ_apparent - R_sun_apparent);

% Determine smaller apparent radius
If R_sun_apparent <= R_occ_apparent
   min_R_apparent = R_sun_apparent;
Else
   min_R_apparent = R_occ_apparent;
EndIf;

% Determine illumination state based on angular geometry
If D_apparent <= sum_R_apparent
   IsSunLit = 1; % Full illumination
Else
   If D_apparent >= diff_R_apparent
      If R_occ_apparent >= R_sun_apparent
         IsSunLit = 0;  % Full shadow (Umbra)
      Else
         IsSunLit = 1;  % Sun larger than occulter -> full illumination
      EndIf;
   Else
      % Partial shadow (Penumbra), compute linear transition
      IsSunLit = 1 * (D_apparent - diff_R_apparent) / (2 * min_R_apparent);
      If IsSunLit < 0
         IsSunLit = 0;
      EndIf;
      If IsSunLit > 1
         IsSunLit = 1;
      EndIf;
   EndIf;
EndIf;

% Compute percent shadow for reporting or plotting
PercentShadow = 100 * (sum_R_apparent - D_apparent) / (2 * min_R_apparent);
If PercentShadow < 0
   PercentShadow = 0;
EndIf;
If PercentShadow > 100
   PercentShadow = 100;
EndIf;
```

The IsSunLitGMF function is designed for seamless integration into a high-level GMAT mission script, providing real-time evaluation of a spacecraft's illumination status during simulation. Within the main script, the function is typically called at each propagation step, receiving as input the current position vectors of the spacecraft, the Sun, and the occulting body (e.g., Earth), as well as the physical radius of the occulting body. These inputs are usually extracted from GMAT's built-in parameters, such as Spacecraft1.EarthMJ2000Eq.X for position components.
The function returns two outputs: IsSunLit, a normalized value between 0 (full shadow) and 1 (full sunlight), and PercentShadow, representing the percentage of the Sun's disk obscured by the occulting body. These outputs can be written to a report file or visualized in an XYPlot alongside other mission parameters, such as elevation angle or elapsed time.
By embedding this function into the main mission sequence, users can track eclipse transitions with high resolution and use the data for power, thermal, or attitude analysis. The function's modular structure allows it to be adapted to different occulting bodies simply by adjusting the radius parameter. This flexibility makes IsSunLitGMF an essential tool for precise mission planning.

Example real world problems:

Can we or not perform sun sensor and Nadir sensor calibration over the Svalbard ground station during the autumn or Winter period?


SSO 500km LTAN 10:30
Ab hier Nur noch die hälfte der Sonne
Beginn des passes (ab 10°el)
80,00
80° Elevation at best
500km alt.
Okulation der Sonne
Sonne Achse Tangential zuer Erde:
Hier sieht man noch die Sonne kmplett,
Ab dann aber immer weniger
Ende des passes (wieder 10°el)
Tilt der Erde 23° für Dezember
Svalbart Longuitude 78°
78,00
6371 Km Erdradius

Example code snippet of a high-level script

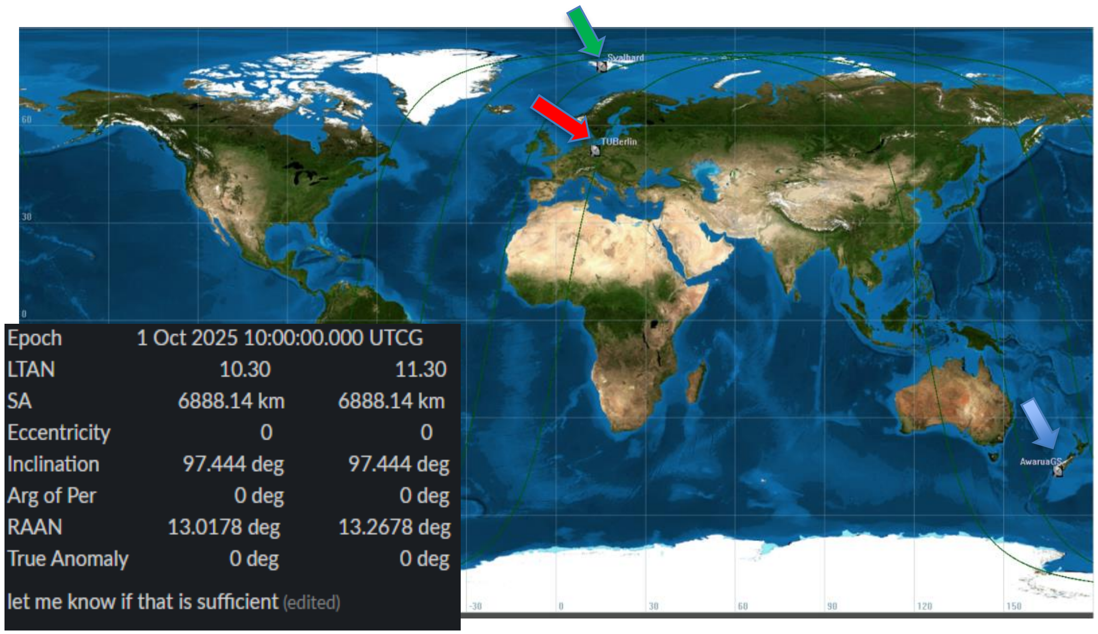
```
.
.
%-----------------------------------------
%---------- Functions
%-----------------------------------------
Create GmatFunction IsSunLitGMF;
IsSunLitGMF.FunctionPath = 'C:\GMAT\scripts\IsSunLitGMF.gmf';

%-----------------------------------------
%---------- Arrays, Variables, Strings
%-----------------------------------------
Create Variable SunElevation CSTUZ CSTUXY AzimuthTU ElevationTU AzimuthSV ElevationSV CSSVZ CSSVXY IsSunLit;
Create Variable PercentShadow R_Earth CSAWXY CSAWZ AzimuthAW ElevationAW;
Create String now format;
R_Earth = 6378.137; % km
.
%-----------------------------------------
%---------- Mission Sequence
%-----------------------------------------
BeginMissionSequence;

While OTTER.ElapsedDays < 93
   Propagate CubeSatProp(OTTER);

   CSTUZ = OTTER.BerlinTopo.Z;
   CSSVZ = OTTER.SvalbardTopo.Z;
   CSAWZ = OTTER.AwaruaTopo.Z;

   CSTUXY = Sqrt(OTTER.BerlinTopo.X^2 + OTTER.BerlinTopo.Y^2);
   CSSVXY = Sqrt(OTTER.SvalbardTopo.X^2 + OTTER.SvalbardTopo.Y^2);
   CSAWXY = Sqrt(OTTER.AwaruaTopo.X^2 + OTTER.AwaruaTopo.Y^2);
   AzimuthSV = Rad2Deg(Atan2(OTTER.SvalbardTopo.Y, -OTTER.SvalbardTopo.X));
   AzimuthTU = Rad2Deg(Atan2(OTTER.BerlinTopo.Y, -OTTER.BerlinTopo.X));
   AzimuthAW = Rad2Deg(Atan2(OTTER.AwaruaTopo.Y, -OTTER.AwaruaTopo.X));

   If AzimuthTU < 0.0
      AzimuthTU = 360.0 + AzimuthTU;
   EndIf;
   If AzimuthSV < 0.0
      AzimuthSV = 360.0 + AzimuthSV;
   EndIf;
   If AzimuthAW < 0.0
      AzimuthAW = 360.0 + AzimuthAW;
   EndIf;

   ElevationTU = Rad2Deg(Atan2(CSTUZ, CSTUXY));
   ElevationSV =  Rad2Deg(Atan2(CSSVZ, CSSVXY));
   ElevationAW = Rad2Deg(Atan2(CSAWZ, CSAWXY));

   [IsSunLit, PercentShadow] = IsSunLitGMF(OTTER.EarthMJ2000Eq.X, OTTER.EarthMJ2000Eq.Y, OTTER.EarthMJ2000Eq.Z, Sun.EarthMJ2000Eq.X, Sun.EarthMJ2000Eq.Y, Sun.EarthMJ2000Eq.Z, Earth.EarthMJ2000Eq.X,
                                Earth.EarthMJ2000Eq.Y, Earth.EarthMJ2000Eq.Z, R_Earth);

   Report SunReport OTTER.UTCGregorian OTTER.ElapsedSecs OTTER.Earth.Latitude OTTER.Earth.Longitude ElevationSV ElevationTU ElevationAW PercentShadow;

 EndWhile;
```

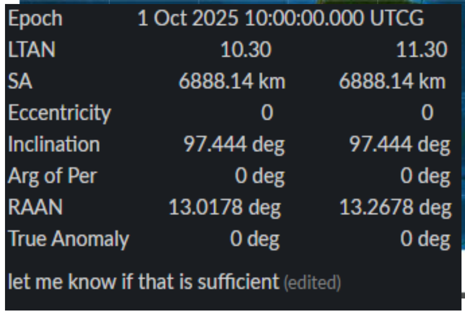
In this high-level GMAT mission script, the custom IsSunLitGMF function is seamlessly integrated into the mission sequence to compute the illumination conditions of the spacecraft during propagation. The script propagates the spacecraft (OTTER) over time and continuously evaluates geometric parameters, such as the satellite's elevation angles relative to multiple ground stations (Berlin, Svalbard, Awarua) and azimuth angles for situational awareness.

At each propagation step within the While loop, IsSunLitGMF is called using the current positions of the spacecraft, the Sun, and the occulting body (Earth). The function returns two key outputs: IsSunLit, a normalized indicator (0 or 1) of whether the spacecraft is fully illuminated or in full shadow, and PercentShadow, which provides a precise percentage (0% to 100%) of sunlight reaching the spacecraft, accounting for partial shadow conditions (penumbra).

These outputs are recorded alongside other mission data using the Report command, enabling post-processing or real-time analysis of eclipse events. The inclusion of IsSunLitGMF enhances the mission analysis by providing detailed illumination profiles critical for power management, thermal modeling, and communication link planning. This integration demonstrates how custom functions can extend GMAT's core capabilities for complex mission scenarios.

Application

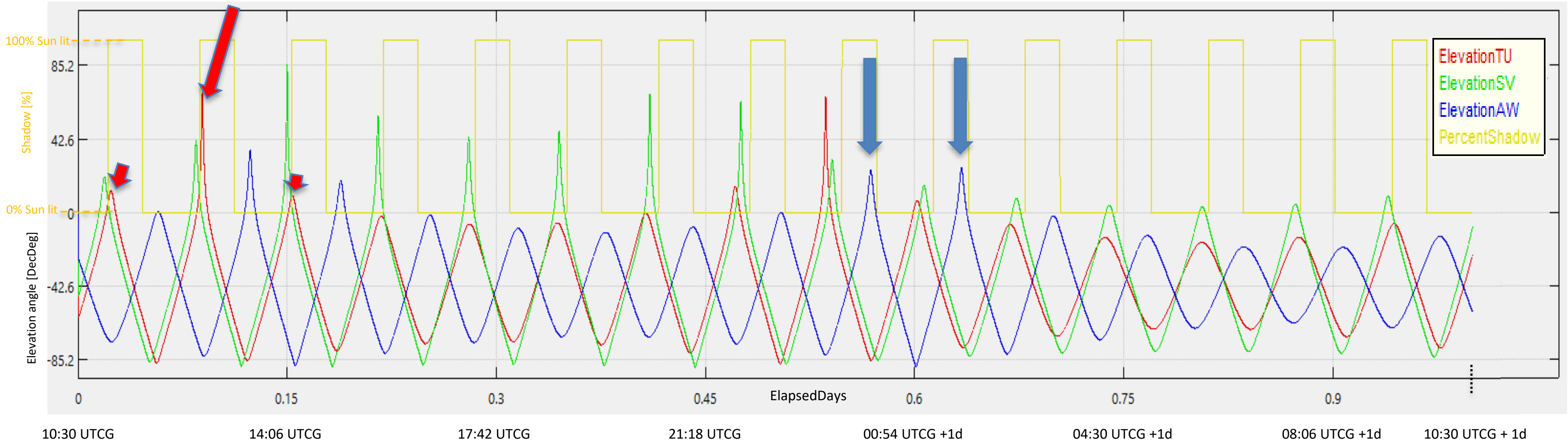




Epoch	1 Oct 2025 10:00:00.000 UTCG	
LTAN	10.30	11.30
SA	6888.14 km	6888.14 km
Eccentricity	0	0
Inclination	97.444 deg	97.444 deg
Arg of Per	0 deg	0 deg
RAAN	13.0178 deg	13.2678 deg
True Anomaly	0 deg	0 deg

let me know if that is sufficient (edited)

The yellow curve represents the percentage of shadow visible to the satellite as it orbits the Earth on October 1, 2025. As you can see, all of the satellite's passes over Svalbard (green elevation curve), occur in permanent darkness. The satellite is never lit by the sun. When passing over Berlin (red elevation curve), the satellite is barely lit by the sun during the day on three occasions and completely in the dark at night. In contrast, the satellite is well lit when passing over Awarua station in New Zealand (blue elevation curve) at two opportunities over the day. ***Question: Over which Ground-Station(s) would you perform Sun-sensor calibration on that day?***



Elapsed Day (1.00=24h)

Questions?

RSC³ web:

Youtube:

sacha.tholl@dlr.de