rocedure for Simulating Occultation Observations

❶Preparation for study for new fly-by data

1. Download the ExPRES data about the occultation timing you want to examine from ExPRESdata→expres\_gll\_jupiter .. .cdf at <https://maser.lesia.obspm.fr/publications/doi/jovian-auroral-radio-source.html?lang=en>

グラフィカル ユーザー インターフェイス, アプリケーション

自動的に生成された説明

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション, メール

自動的に生成された説明

1. Copy cdf data to the directory named result\_for\_yasudaetal2022/expres\_cdf\_data/.

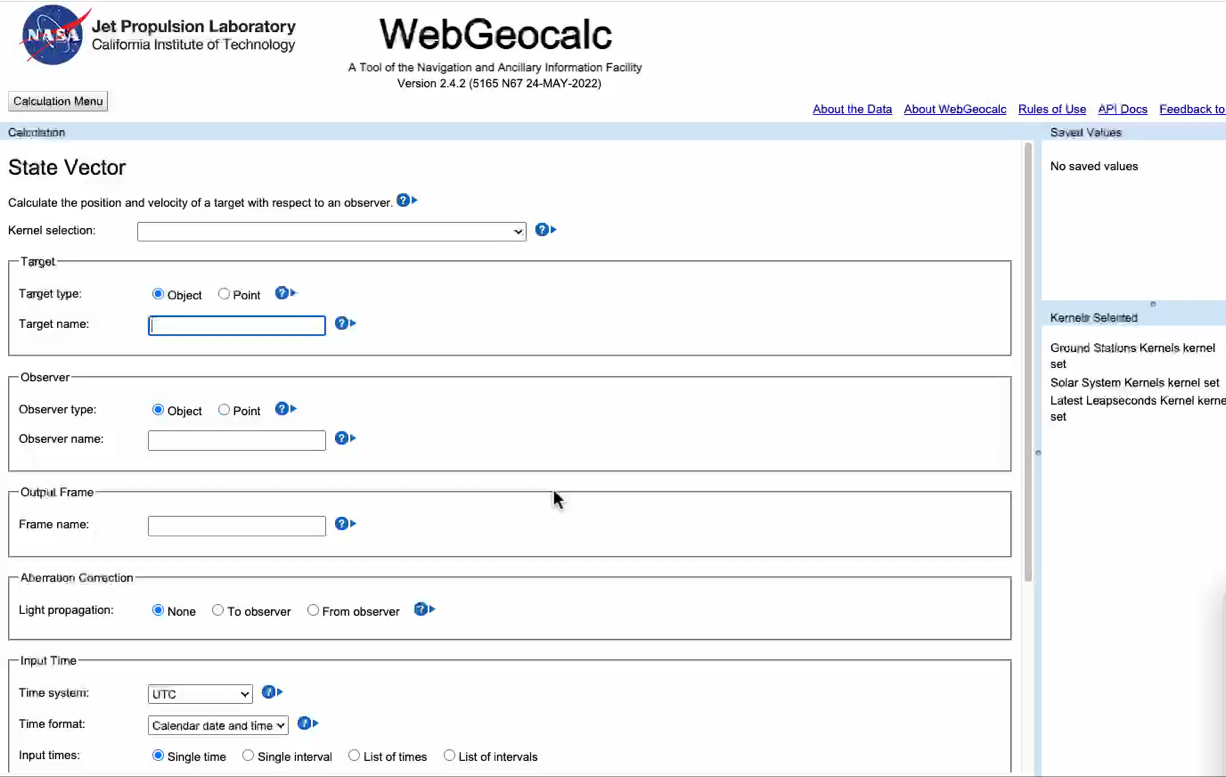
グラフィカル ユーザー インターフェイス, アプリケーション

自動的に生成された説明

1. Fill in the csv file named result\_for\_yasudaetal2022/occultation\_flyby\_list with the target celestial body, spacecraft name, number of flybys, cdf file name, data start and end dates used.



1. グラフィカル ユーザー インターフェイス, テキスト, アプリケーション, メール

   自動的に生成された説明Get position data of the spacecraft at the time of the target flyby (see step 1)
2. copy orbit data to result\_for\_yasudaetal2022/spacecraft\_ephemeris/.
3. write the file name of the orbit data in 　result\_for\_yasudaetal2022/occultation\_flyby\_list
4. Obtain location data of the moon (satellite) at the time of the target flyby from <https://wgc2.jpl.nasa.gov:8443/webgeocalc/#StateVector>. WebGeocalc input is set for “the spacecraft at the time of the target flyby" and the time you want to look up.
5. Move the orbital data of celestial objects to result\_for\_yasudaetal2022/moon\_ephemeris/.
6. Write the orbital data name in result\_for\_yasudaetal2022/occultation\_flyby\_list
7. enter the name of the flyby object, the name of the spacecraft, and the number of flybys in python\_Jovian\_radio\_occultation /preparing\_position\_data.py and run
   * 1. The information for all detectable radio is created in result\_for\_yasudaetal2022/expres\_detectable\_radio\_data\_of\_each\_flyby as following: 0 year, 1 month, 2 days, 3 hours, 4 minutes, 5 seconds, ◯, ◯, ◯, ◯, 9 Frequency (MHz), 10 Longitude of magnetic field line (0~360) or (-1000) for Io, 11 Poles (North:1, South:-1), 12 x-coordinate, 13 y-coordinate, 14 z-coordinate, like “All\_galileo\_ganymede\_1\_Radio\_data.txt”.
     2. The interpolated time resolution to 1 second is also saved in the same folder with name like “lnterpolated\_all\_galileo\_ganymede\_1\_Radio\_data.txt”
     3. The position data of the probe transformed by coordinates using the radio source results are stored in result\_for\_yasudaetal2022/calculated\_expres\_detectable\_radio\_data\_of\_each\_flyby with name like interpolated\_calculated\_all\_galileo\_ganymede\_1\_Radio\_data.txt. The following figure indicates the coordinate transformation method.

ダイアグラム

自動的に生成された説明

1. Write the date and time (day, hour, and minute) that are completely occulted into result\_for\_yasudaetal2022/occultation\_flyby\_list based on the previous study. Chart

   Description automatically generatedGraphical user interface, application

   Description automatically generated
2. Get Galileo radio data from <http://das2.org/browse/uiowa/galileo/pws/survey_electric>. Fill in the time range of the radio data (same as the figure in the previous study) and use "get from Jupiter" to get the data. Move the radio data to tools/result\_for\_yasudaetal2022/galileo\_radio\_data/. Add the name of the radio data to the radio\_data\_txt field of result\_for\_yasudaetal2022/occultation\_flyby\_list.

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application

Description automatically generated

❷Procedure when performing calculations with the new electron density distribution

1. Change the density parameter of /Users/yasudarikuto/research/icymoon\_raytracing/raytrace.tohoku/src/rtc/model/test\_plasma\_model.cpp

Below picture is for the Europa hydrostatic equilibrium model with a surface density of 400(/cc) and a scale height of 600 km.

double plasma::europa\_nonplume::getDensity(const vector &point) const ///////////////新しいプラズマモデル（z軸方向にexpで減少）

{

const double

r = std::sqrt((pow(point(0), 2.0)) + (pow(point(1), 2.0)) + (pow(point(2) + 1.601e6, 2.0)));

const double

rxy = std::sqrt((pow(point(0), 2.0)) + (pow(point(1), 2.0)));

const double

t = std::fabs(4e8 \* exp(-(r - 1.601e6) / 6e5)); //////////////エウロパ静水圧平行モデル 地表面で4.0\*10^2(/cc) スケールハイト600km

;

return t;

}

1. Provide the "output file name" of icymoon\_raytracing/raytrace.tohoku/src/rtc/testing/mkray3.sh. Ex）ray-Peuropa\_nonplume\_4e2\_6e2-Mtest\_simple-benchmark-LO-Z-2-FR3.984813988208770752e5
2. Change the value of the for statement of HIG in icymoon\_raytracing/raytrace.tohoku/src/rtc/testing/mkray3.sh. If it is less than or equal to 0, it is 100 every other time, and if it is greater than 0, it is 500,600,1000,1100, etc. +500 +100 repeated. Ex) for HIG in -300 -200 -100 0 100 500 600 1000 1100 1500 1600 2000 2100 2500 2600
3. Compile with make command in icymoon\_raytracing/raytrace.tohoku/src/rtcicymo on\_raytracing/raytrace.tohoku/src/rtc
4. Compile with make command in icymoon\_raytracing/raytrace.tohoku/src/rtcicymo on\_raytracing/raytrace.tohoku/src/rtc/testing
5. Run bash mkray3.sh in icymoon\_raytracing/raytrace.tohoku/src/rtc/testing
6. After completing the calculation, write the target object, spacecraft name, number of flybys, maximum electron density, scale height, and maximum and minimum values of H I G specified in 3. in tools/python\_Jovian\_radio\_occultation/occultaion\_range\_calc.py and run.

object\_name = 'europa' # europa/ganymde/callisto

spacecraft\_name = "galileo" # galileo/JUICE(?)

time\_of\_flybies = 14 # ..th flyby

highest\_plasma = '4e2' # 単位は(/cc) 2e2/4e2/16e2

plasma\_scaleheight = '6e2' # 単位は(km) 1.5e2/3e2/6e2

raytrace\_lowest\_altitude = -300 # レイトレーシングの下端の初期高度(km) 100の倍数で

raytrace\_highest\_altitude = 2600 # レイトレーシング上端の初期高度(km) 500の倍数+100で

1. If the message "Start position need to be far from moon" is displayed, the starting position of the ray tracing is too close to the celestial body, so decrease the value of SX and rerun.
2. The highest and lowest altitudes at each frequency are output, so refer to the output results and specify the range of HIG and FREQ for statements in icymoon\_raytracing/raytrace.tohoku/src/rtc/testing/mkray3.sh and run it. . / python\_for\_yasudaetal2022 /tracing\_range\_'spacecraft\_name'\_'celestial\_name'\_'flyby\_number'\_flybys/para\_'maximum\_electron\_density'\_'scale\_height'.csv' outputs the altitude range for which the calculation is required.
3. Write the target object, probe name, number of flybys, maximum electron density, and scale height in tools/python\_Jovian\_radio\_occultation/detectable\_radio\_check.py and run it. The result returns result\_for\_yasudaetal2022/raytracing\_'object\_name'\_results/'object\_name'\_'maximum\_electron\_density'\_'scale\_height'/Interpolated\_'object\_name'\_'probe\_name'\_'number\_of\_flybys'\_'maximum\_electron\_density'\_' Scale height'\_dectable\_radio\_data.txt'.

object\_name = 'ganymede' # ganydeme/europa/calisto``

spacecraft\_name = "galileo" # galileo/JUICE(?)

time\_of\_flybies = 1 # ..th flyby

highest\_plasma = '4e2' # 単位は(/cc) 2e2/4e2/16e22

plasma\_scaleheight = '6e2' # 単位は(km) 1.5e2/3e2/6e2

1. Write the target object, spacecraft name, number of flybys, maximum electron density, scale height, and how many times the background noise should be thresholded in the tools/ python\_Jovian\_radio\_occultation/plot\_f-t\_2.py and run.A computer screen shot of a code

   Description automatically generated
   1. Three f-t diagrams (full, ingress. Egress) are stored in '... /result\_for\_yasudaetal2022/raytracing\_'+cosm name+'\_results/' +cosm name+'\_'+maximum electron density+'\_'+scale height+'/ and '. /result\_for\_yasudaetal2022/f-t\_plot\_'+probe\_name+'\_'+cosm name+'\_'+number of flybys+'\_flyby/radio\_boundary\_intensity\_="magnitude of noise to threshold "dB
   2. An array examining the timing of exceeding the threshold value from the radio data at each of ingress and egress, as in '.../result\_for\_yasudaetal2022/radio\_data\_occultation\_timing\_' + spacecraft name + '\_'+ celestial name + '\_'+ number of flybys '\_flyby/'+ celestial name + '\_'+ "threshold noise factor" dB + '\_ingress\_time \_data.txt', is stored for each frequency.
   3. The estimated occultation start and end times in ray tracing are stored for each frequency in '... /result\_for\_yasudaetal2022/raytracing\_occultation\_timing\_' + spacecraft name +'\_'+ celestial name +'\_'+ flyby number +'\_flyby/interpolated\_' + spacecraft name +'\_'+ celestial name +'\_'+ flyby number+'\_'+maximum electron density+'\_'+scale height+'\_ingress\_time\_list.txt'
   4. The difference between the occultation start and end times of ray tracings and observations is stored in '... /result\_for\_yasudaetal2022/radio\_raytracing\_occultation\_timing\_def\_' + spacecraft name +'\_'+ celestial name +'\_'+ flyby number +'\_flyby\_radioint\_"threshold noise multiplier "dB +'/ interpolated\_'+source\_name+'\_'+max\_electron\_density+'\_'+scale\_height+'\_ingress\_defference\_time\_'+radio\_type\_(A~D)+'\_' multiplier\_of\_noise\_to\_threshold Magnification "dB.txt', calculated for each frequency.
2. Write the name of the object, the name of the spacecraft, the number of flybys, the frequency range to be examined, the radio wave strength at the threshold, the start or end of occultation (ingress or egress), and the type of radio wave (A~D) in tools/ python\_Jovian\_radio\_occultation /evaluate\_ft\_2.py and run the simulation.
3. Based on the ray tracing results so far, which electron density distribution fits the observation well, the results are output as a figure and text. The output will be sent to '... /result\_for\_yasudaetal2022/evaluate\_f-t\_diagram\_plot\_' + spacecraft name + '\_' + object name + '\_' + flyby number + '\_flyby\_radioint\_' + "noise factor to threshold" dB

####################################################

object\_name = 'ganymede' # ganydeme/europa/calisto`

spacecraft\_name = "galileo" # galileo/JUICE(?)

time\_of\_flybies = 1 # ..th flyby

"""lowest frequency and highest\_freqency(MHz)"""

using\_frequency\_range = [5.5e-1, 6] # egress

boundary\_intensity\_str = '7e-16' # '7e-16' '1e-15'

occultaion\_type = 'egress' # 'ingress' or 'egress'

radio\_type = 'A' # 'A' or 'B' or 'C' or 'D'

❸Method for plotting the locations of occultation observations

1. Obtain position data of the spacecraft and the Moon as seen from the center of Jupiter in the lunar system coordinates during the time period used in the observation.（from <https://wgc2.jpl.nasa.gov:8443/webgeocalc/#StateVector>）

Application, table

Description automatically generated

Time ranges and time steps are based on the simulation.（/Users/yasudarikuto/research/icymoon\_raytracing/tools/result\_for\_yasudaetal2022/occultation\_flyby\_list.csv）

1. Stored in /Users/yasudarikuto/research/icymoon\_raytracing/tools/result\_for\_yasudaetal2022/ephemeris\_for\_coordinate\_transformation/moon　or spacecraft
2. Write the file name in /Users/yasudarikuto/research/icymoon\_raytracing/tools/result\_for\_yasudaetal2022/occultation\_flyby\_list.csv

Table

Description automatically generated

1. Write the object name, spacecraft name, number of flybys, occultation start & end times at the relevant flyby, and radio source in /Users/yasudarikuto/research/icymoon\_raytracing/tools/python\_for\_yasudaetal2022/tangential\_position\_revised.py and execute it.
2. The information including the observation points of occultation observation is output to like /Users/yasudarikuto/research/icymoon\_raytracing/tools/result\_for\_yasudaetal2022/calculated\_expres\_detectable\_radio\_data\_of\_each\_flyby/calculated\_all\_galileo\_callisto\_9\_tangential\_point\_revised.txt.
3. The maximum, minimum, and average latitude and longitude are output on the command.