Thrusters modelling, propellant choice and plume expansion: openPlumeEP capabilities

IEPC-2019-211

Presented at the 36th International Electric Propulsion Conference
University of Vienna • Vienna, Austria
September 15-20, 2019

B. Zitouni[1], J. Laube, N. Massaccesi, V. Sonneveld, M. Peukert
OHB System AG, Universitätsallee 27-29. D-28359 Bremen, Germany

Abstract: Electric propulsion has made the expected progress last years to be fully deployed on every kind of spacecraft: on single or constellations setups, for telecommunication or scientific missions, on geostationary or low orbits. OHB, as a major European system integrator, has the task to accommodate the thrusters into its satellite platforms. Plume impingement is one of the driving parameter to correctly accommodate the developed thrusters. For performing the electric propulsion plume analysis, openPlumeEP has been developed and intensively validated with in-orbit data. As a family own company, OHB offers the necessary flexibility to monitor the promising technology. Therefore, the numerical tool need to be versatile and offer capabilities to answer the questions generated by the new thrusters' families. It has to be noted that openPlume is distributed under a GNU-GPL license with the source code available to the user.

Introduction

openPlumeEP is able to perform plume analysis by injecting the ions density and energy at the thruster exit plan and extrapolated in the far field following the $1/r^2$ rule. Using a 2D structured or unstructured mesh, results are generated in form of current density flux, erosion yields and forces and torques on each impacted surface.

In this case, the nature of the thrusters are not an obstacle since the HET, HEMPT, RIT or other thrusters can be easily modelled in the far field with the correct assumption coming from on-ground tests or in-orbit observations. This article presents examples of how different thrusters are modelled and therefore, in which way a trade-off can be made between different thrusters according to their impact on the spacecraft. By providing this information at early stages of a project, mitigations solutions can be imagined and the correct thruster for the right use is ensured.
In the last conferences, a constant effort is done by the EP community to choose another propellant than the Xenon. openPlume, by being able to load another type of data coming from a Particle-In-Cell [1] and to adapt the used gas, offer the possibility to compare Xenon and Krypton. The main result could then help the EP community in terms of not only cost advantages but also in terms of spacecraft accommodation. A simple example of a thruster impinging a solar panel hints at the best propellant for the thruster application from the spacecraft point of view.

Finally, the plume expansion in the far field is performed thanks to the $1/r^2$ extrapolation. However, this approach is limited and cannot capture the distribution of the ions when different effects have to be taken into account. As demonstrated in a previous paper[2], conical flow description has to be updated by considering the plume as a "Self Similar Model" after a certain distance from the thruster. This paper will present the implementation of this «SSM model» and its advantages compared to the classical approach. The ""SSM model"" is perfectly compatible with openPlume approach since it contains several equations able to model different type of thrusters (those with an adiabatic expansion, those with an isothermal expansion ...). The validation of this model is done by comparing the simulation outcome to the in-orbit OHB geostationary satellite "GeoSat" forces and torques data.

[1] Propulsion and Environment Analysis Expert, OHB Propulsion department, bayrem.zitouni@ohb.de

I. openPlumeEP structure

openPlumeEP provides Fast and simplified calculations using the ray tracing approach; ions that are emitted from the thruster remain to travel in the same direction and at the same velocity throughout the entire simulation domain. The structure of openPlumeEP is based on first injecting the correct thruster parameters, then these parameters are extrapolated to the far field thanks to the $1/r^2$ approximation. The ions are accelerated to a velocity according to the thruster potential. Meanwhile, the complete spacecraft can be modelled with a 2D structured or unstructured mesh. The ray-tracing is performed essentially through shadow calculations.

A. Algorithm: code and logic

To develop openPlumeEP, a choice of python language have been made due to its high adaptability and flexibility when it comes to modifications. No compilation is necessary and the python scripts allow quick modification of the code. The openPlumeEP application is available in a zipped folder containing an executable and the source code (in this case python scripts). openPlumeEP is intended to be used by two types of users profiles: the coder and the analysist. In addition to the User Manual important for the second, a special care have been made to have a simple and a clean code. The analysist can quickly launch the application through the executable whereas the coder can make his modifications and run the scripts in a standalone mode.
The code is written in Python version 3.4.3 and was programmed within the PyCharm IDE, as it is a powerful tool to generate robust code. Some python packages were included in openPlumeEP (such as Tkinter, used to create the GUI). The openPlumeEP application is governed by the Trace module that makes the communication between the other modules.

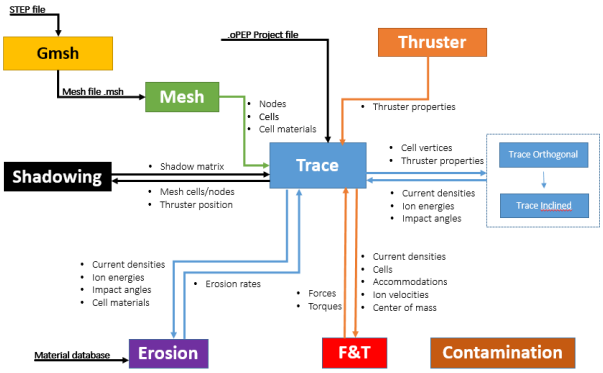

Figure 1: openPlumeEP structure

The Trace module is the core of the openPlumeEP algorithm: it is constituted from the TraceInclined and the TraceOrthogonal. In TraceOrthogonal, the current density of the cell is calculated, but only as if the cell was oriented orthogonally towards the thruster point. In TraceInclined, this current density is corrected for the impingement angle on the cell. The trace module is fed by the two inputs modules: the meshing and the thruster modules.
The meshing module is able to load gmsh native output files that contain the meshed spacecraft. This module contains information and vertices of each cell. The cell is the finite element of the mesh constituted from its geometrical coordinates that are available from a step file.
The thruster module injects the ions distribution into the simulation and can be interfaced with different types of data (experiment, PIC, probabilistic).
Optionally, when the shadowing module is activated, it checks if cells are shadowed by other cells and adjusts the calculated cell properties accordingly.
Finally, the trace module transfers to the erosion module the required current densities impinging the cell. The erosion module converts it then into an eroded thickness. The material database is necessary to interpolate the impact by taking into account different erosion models.
Furthermore, the forces on each cell are calculated together with the torques they produce around the center of mass. It has to be noticed that the forces and torques module and the contamination module are still underdevelopment.

B. Geometrical modelling: meshing, shadowing and ray-tracing

openPlumeEP uses purely 2D meshes. In Gmsh these 2D meshes have to be defined on 2D surfaces. There are two types of 2D cells that openPlumeEP can currently handle: triangles and quads. It is possible to use both of them inside one mesh.

In order to mesh geometries, Gmsh can load different formats such as STEP that could contain the whole spacecraft. During the development of openPlumeEP, generic geo files are created. These geo files are scripts that helps creating and meshing the geometrical elements (same approach as in **Error! Reference source not found.**).

```
Merge "mystepfile.stp"
Transfinite Line Transfinite Line {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12} = 10
Using Progression 1;
Transfinite Surface {14, 16, 18, 20, 22, 24};
Physical Surface(21) = {14, 16, 18, 20, 22, 24};
```

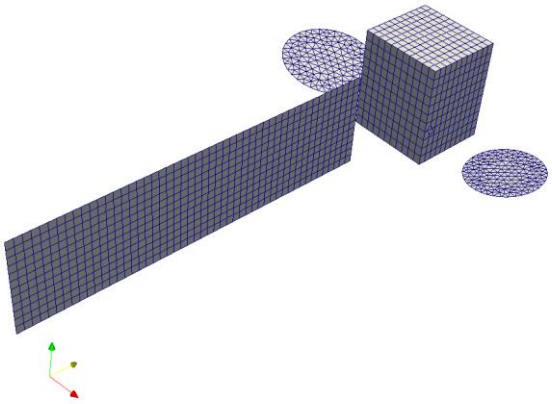

Figure 2: Spacecraft meshed with gmsh

The ray-tracing approach consists also in checking if some mesh cells are hidden with regard to the thrusters. To assess which cells of the mesh are being shadowed from the thruster by other cells, it is evaluated for every cell whether there is any other cell located between the thruster and the impinged cell.
To perform this check between two cells (cell A and cell B for example), a certain number of operations are performed. First, a line between the thruster and the centroid of cell A is "traced". The intersection point between this line, called Trace Vector, and the plane going through cell B is computed. If this point does not lie on the positive side of trace vector A, then the cell A is not shadowed by cell B.

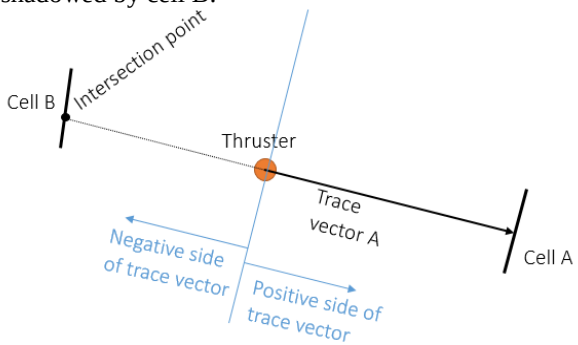

Figure 3: Trace vector concept

If this point lie on the positive side of trace vector A, then several cases can happen. The intersection point can still be outside cell B (the point belong to the plane but not to the cell). Thanks to a mathematical formula, the point can be assessed if it is belonging to the cell or not.

C. Thrusters modelling

The thruster is always modelled as a point source. In openPlumeEP it is possible to characterize the thruster in three different ways, using:

- Experimental data
- Particle In Cell (PIC) simulation data
- Custom method(curve fitting)

The ion current density is extrapolated into the far field as a function of the radius r from the thruster, according to:

$$cd(r,\theta) = cd_{ref}(\theta) \cdot \frac{r_{ref}^2}{r^2}$$

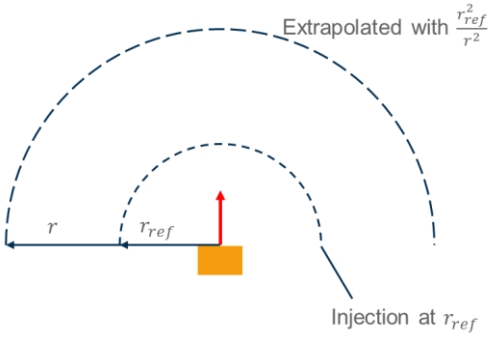

Figure 4: Source flow method

The parameters and settings for each of the methods are stored in an openPlumeEP project file (.oPEP). This file contains both settings that apply to all kinds of thrusters, such as the position, and settings, which are specific for the used method.

- *Experimental*

In the point source method, the experimental data can be injected at a reference distance r_{ref} from the thruster exit plane. Two measured data are required:

- Current Density as a function of angle(θ)
- Ion Energy as a function of angle(θ)

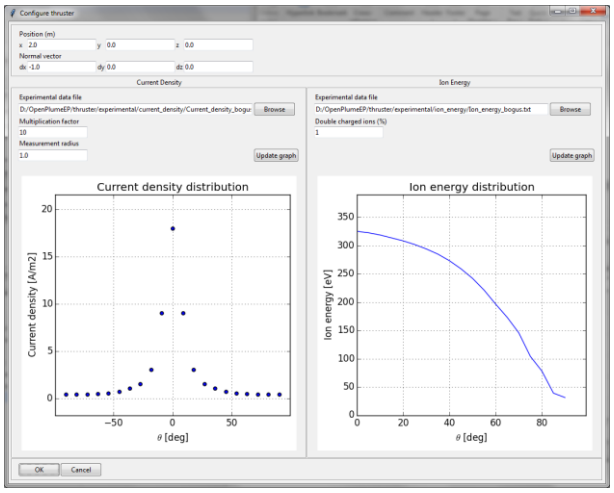

Figure 5: Thruster experimental data loaded in openPlumeEP

When calculating the current densities at the nodes of the mesh, openPlumeEP will interpolate linearly between the points of the given current density distribution file.

It is assumed that the velocities (i.e. the energies) of the ions remain constant with the distance from the thruster. openPlumeEP will also linearly interpolate the ion energy distribution to evaluate the ion energy at the mesh nodes.
The data can be easily loaded as a simple text file with the angles as first column, density or energy in the second column.

- *Particle-In-Cell*

In case of experimental data absence, an interfacing with Particle-In-Cell simulations is possible. PIC sumulations can be done at a smaller domain and the thruster can be characterized by using a hybrid PIC approach. Currently, openPlumeEP handles PicPluS inputs but can be connected to any PIC tool like SPIS. In addition, subsystem suppliers could provide their own assessment of their thruster plume.
This option is working in the same procedure as for the experimental data. Current densities and energies can be extracted from the PIC simulations and injected into the ray-tracing tool.

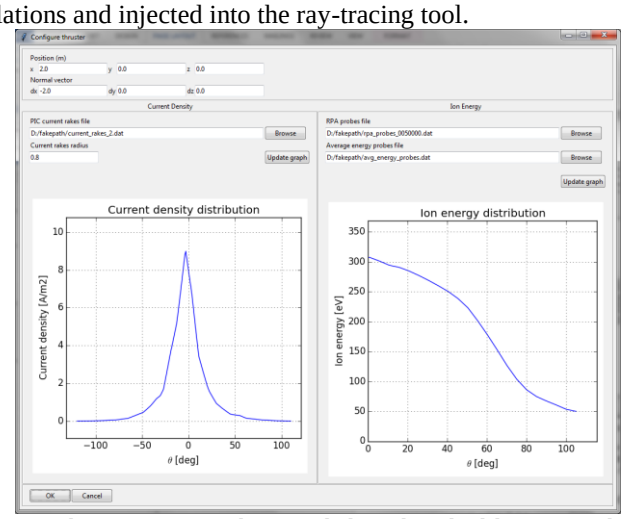

Figure 6: Thruster experimental data loaded in openPlumeEP

- *Custom*

The third method for modelling the thruster is a special tool to create current density and ion energy distributions. This can be useful when no experimental data or PIC simulations are available.
As a physical evaluation criterion for the current density distribution, openPlumeEP informs the user the amount of current that will be injected.
Additionally, this method can be used to create an adjusted version of experimental data. For example when it is known that the plume to be modelled is less wide than the plume from experimental measurements, the current density in the outer angles can be set to lower values.
The fitting of the current density is then possible based on a probability density functions (Normal, Gaussian, Student). The function is used to fits a current density distribution at a certain radius from the thruster.

$$cd(r,\theta) = p(\theta) \cdot \frac{r_{ref}^2}{r^2}$$

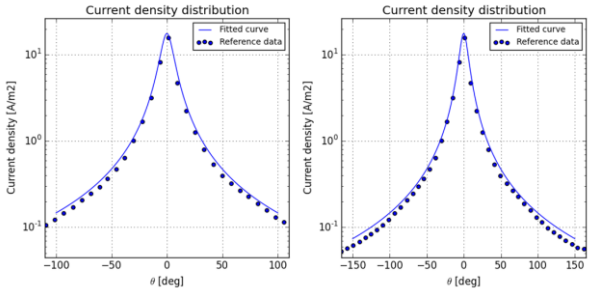

Figure 7 Fitting experimental data

D. Erosion and material modelling

The erosion modelling depends on the energy of the impinging ions and their angles. The Erosion yield Y represents the ratio between the sputtered atoms and the incident ions. Generally, experiments are carried out to characterize monolayers materials for some energy and angles ranges.

Therefore, sputtering of materials have been characterized by choosing an empirical approach. In fact, in a spacecraft configuration, ions impinging on a surface occurs at much larger energies and angles scales than those available in one experiment. Consequently, a fitting function is needed and several models co-exist that link the erosion yield to the impinging ions energy and their angle.

The dependencies between the two parameters, the energy of impinging and the angle of impingement, is then used to calculation erosion yield.

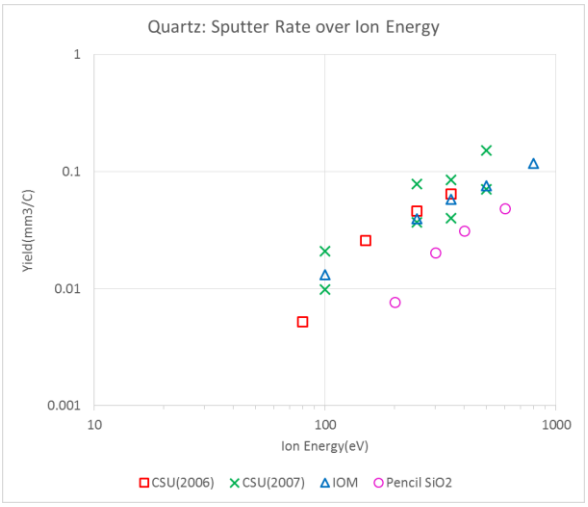

Figure 8 Energy dependent normal sputtering rate

This is the sputtering rate if the ions are impinging normal to the surface. This should be corrected when the ions are hitting a surface at an angle, where the angle is defined as:

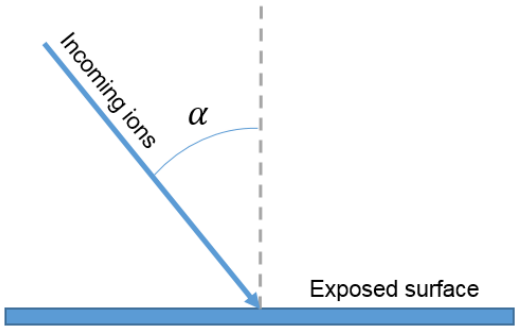

Figure 9 Definition of impingement angle

From this angle, the Angular sputtering coefficient $S(\alpha)/S(0)$ is calculated.

The atomic yield rate Y_{atom} [atoms/s] is then calculated by multiplying the ion flux with the angular sputtering rate:

$$Y_{atom} = \Phi_{\text{ions}} \cdot S(\alpha)$$

The ion flux Φ_{ions} in [ions/s] is derived from the current density [A/m2] and the cell area [m2] using the following relation:

$$\Phi_{ions} = \frac{cd \cdot A_{cell}}{q_e}$$

q_e is the charge of an electron.

The atomic yield is then converted to the depth of yield in [Angstrom/s], using:

$$Y = \frac{Y_{atom} \cdot m_{atom}}{\rho \cdot A_{cell}} \cdot f\left(\frac{\text{Å}}{m}\right)$$

Where:

- m_{atom} is the mass of an atom of the sputtered material [kg/atom]
- ρ is the density of the sputtered material in [kg/m3]
- A_{cell} is the surface area of the cell in [kg]
- $f\left(\frac{\text{Å}}{m}\right)$ is a converting factor (m to Å)

Combining the equations for yield rates and ion flux gives:

$$Y = \frac{cd \cdot S(\alpha) \cdot m_{atom}}{q_e \cdot \rho} \cdot f\left(\frac{\text{Å}}{m}\right)$$

Assigning materials in the model is quite straight forward. First, a database is needed to be built by having a once again easy set up file: first column represent the angles and the second column the energies. In addition, the angular sputtering distribution has the same format.
In the spacecraft, it is possible to assign the materials present in the database to each physical surface, previously defined in gmsh. The button "Identify surfaces" is a nice function that automatically recover the surfaces existing in the geometry file.

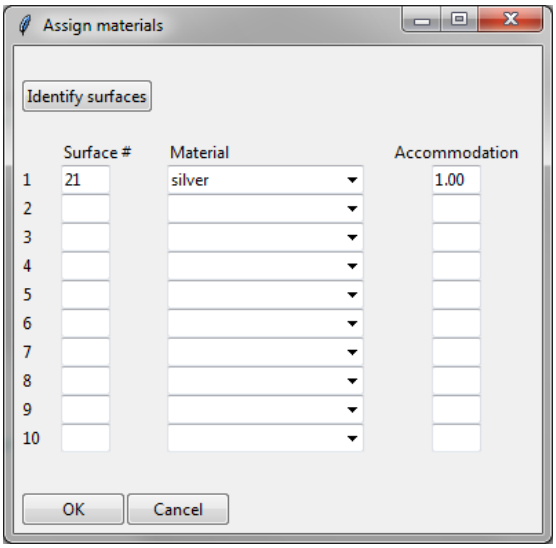

Figure 10 Materials assignment in openPlumeEP GUI

The final spacecraft model can then be displayed in paraview (**Figure *11***) to check the materials assignment. As explained later, the choice of vtk format has been done, which quite handy and easy to use in a python environement.

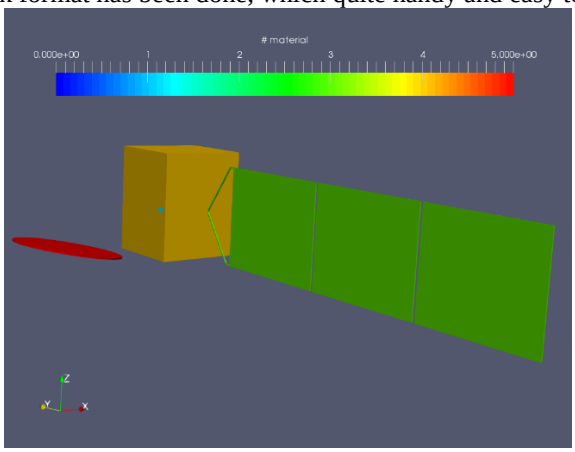

Figure 11 Materials assignment displayed in paraview (5 is CFRP, 2 is silver etc …)

D. Running and post-processing

When delivered, the openPlumeEP folder will have the following structure:

- **code**: all python scripts are existing in this folder. The user could use then the scripts version and running each scripts stand alone.
- **docs**: this contains a html environment documenting the classes and function in the Python code.
- **materials**: inside this folder is the spreadsheet with the material properties. The filename should be mymaterials.xlsx. An option of having an open format text file is under development.
- **mesh**: This is the default directory for mesh files
- **projects**: default directory for .oPEP project files.
- **results**: default directory where output files are saved
- **thruster**: default directory from where thruster characterizing data files are loaded. For example, experimental current density measurements.
- **openPlumeEP**: Executable file that will launch openPlumeEP

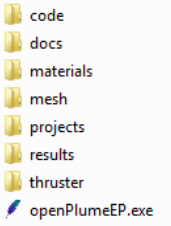

Figure 12 Executable folder of openPlumeEP

The mesh, material, thruster model are stored into a project called "oPEP" (**o**pen**P**lume**EP**). This makes the set-up and the execution quite straight forward as seen in **Figure *13***.

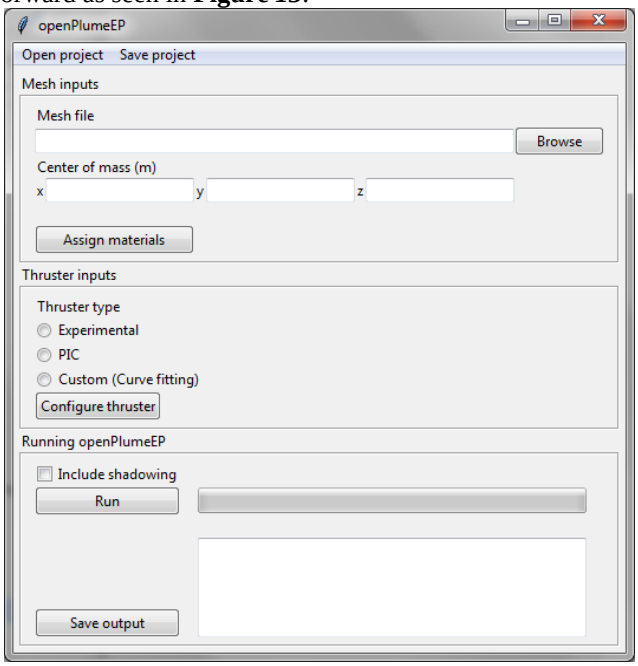

Figure 13 Project set-up

openPlumeEP produces two types of outputs:

- Distributions of calculated quantities over the mesh cells (.vtk format)
- Forces and torques acting on objects (plain text file)

VTK format for the results have been chosen thanks to its compatibility with Paraview (freeware) where the cells of the meshs are coloured according to their calculated values. In addition, a pyramid shaped object is added to represent the thruster as seen in next figure.

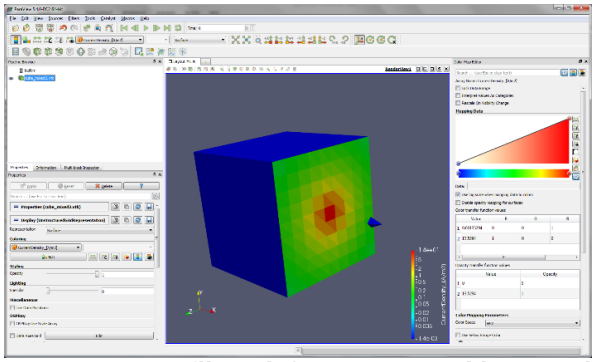

Figure 14 Output file .vtk format opened in Paraview

The forces and torques output are stored in a text file as seen in this extract:

```
Center of mass used:
x, y, z (m)
x x x
Total force:
Fx, Fy, Fz (mN)
x x x
Total torque:
Tx, Ty, Tz (mNm)
x x x
```

II. Application cases

A. Thrusters models and propellant choice

Propellant choice can be easily uploaded within openPlumeEP by choosing the right atomic properties for Krypton instead of Xenon. The plume profile can be provided either from test data, PIC simulation or from literature.
Krypton has an atomic mass of 83.8 whereas Xenon has 131. As shown earlier, the velocity of the ions are extrapolated in the far field by using the mass of the ion and their energy. In the case of the Krypton, some experiments have been carried out for different thrusters[3,4].

These showed the Krypton plume is wider than the Xenon plume. This could mean that for a system integrator, even if the use of Krypton will reduce the costs, this would probably lead to more effort on the accommodation part. A hypothetical HET thruster have been implemented within openPlumeEP to see the impact of a Krypton plume on a spacecraft accommodation. The energy distribution in this case have been calibrated with a ratio between the Krypton and the Xenon masses.

For a typical spacecraft configuration, the following results were obtained for Krypton and Xenon. It has to be noticed that the same behavior is observed for the Krypton in the experiment and in a simulated configuration of a spacecraft. The Kr plume is wider than the Xenon.

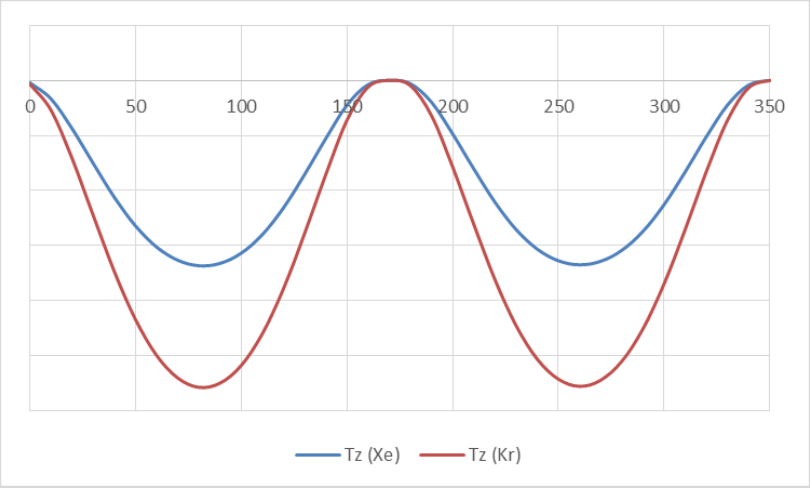

Figure 15 Xenon vs Krypton implementation with openPlumeEP: Forces and torques impact

B. In-orbit validation

A significant work[5] have been carried out to fit the thrusters' model to on-ground and in-orbit data. These papers mention the connection of openPlumeEP to tools like particle in Cell that model the thruster. In this case, openPlumeEP relies on the good fitting of the plume distribution. Indirect validation however is possible by observing OHB GeoSat induced torques. In this case, both approaches can be compared to real in-orbit data: thruster model (from the PIC part) and the extrapolation approach of openPlumeEP.

Figure 16 shows discrepancies with in-orbit data since the $1/r^2$ is underestimating the measured torques due to a HET thruster.

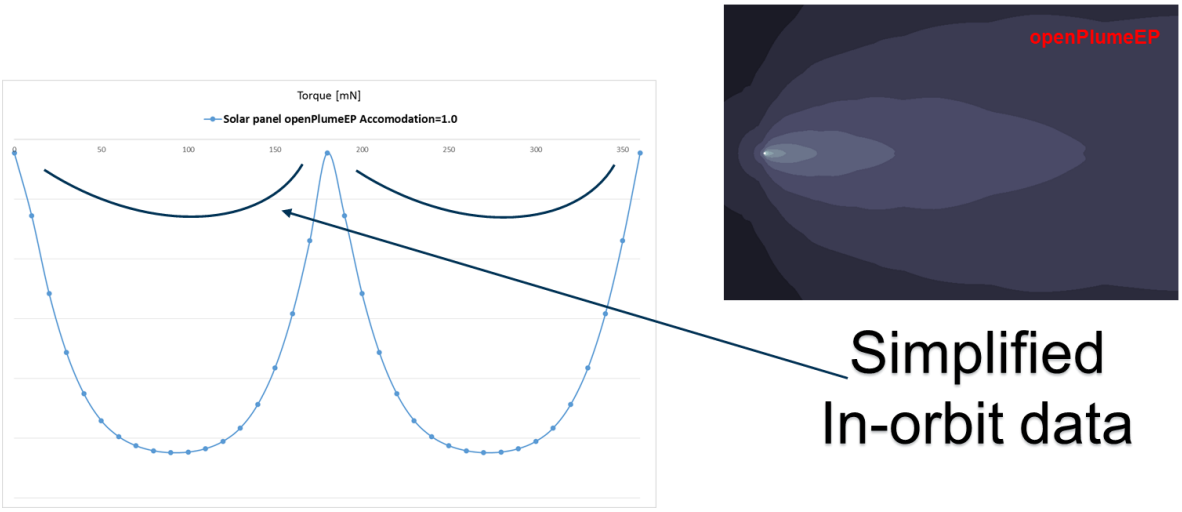

Figure 16 Forces and torques impact with $1/r^2$ rule

In order to better the modelling, different extrapolation formulas have been investigated. The EP community investigated these effects in-orbit since a while and there is no clear agreement for using the adiabatic expansion for HET thrusters. A significant work have been performed by TSNiiMASH[6] in this field and several mathematical models have been developed to model different type of thrusters. Soreq[7] did also model a model that is applicable for the far field. The Plume model depends in fact on the distance from the thruster exit on the impinged surface. We can distinguish then:

- The near field
- The transition zone
- The Far field

The near field where the electron temperature still play a role can be modelled as adiabatic, this questions however the delimitation of the zone and where the different regimes starts. For HET thrusters, the plume is supposed to be adiabatic next to the thruster exit plan and rather isothermal after a certain distance (to be defined). By exploring the Self Similar Model, one can think of the following distribution of the different models.

	Near field r ∈ [0, 0.5m]	**Transition r ∈ [0.5, 1.5m]**	**Far field r ∈ [1.5m, ∞]**
Models	Adiabatic	Conical	Isothermal
openPlume	PIC or exper.	Conical	
SSM	Adiabatic	conical	Isothermal SSM
Soreq	-		Isothermal Soreq

The mathematical model of Soreq is based on a self-similar flow, the used of cylindrical coordinates where the flow is characterized at a plane. In this case, near field effects are neglected and the plume is supposed to be isothermal.
The density depends on the stream-line function f(z):

$$\rho = \frac{\rho_{00}}{f(z)^2}[1 + \frac{r^2}{d^2 f(z)^2}]^{-d^2/2R^2}$$

The radial component of the velocity is solved thanks to the following equation:

$$v_r = v_{r0}[1 + \alpha^2 d^2 ln(f(z))]^{1/2}$$

Unfortunately, this model shows less impact that the openPlumeEP simple extrapolation. After investigation, it was found out that the plume is impinging the solar panels at closer distances than the Soreq model is able to represent. Therefore, in this case, the model is not adapted but it will not be neglected for future application.

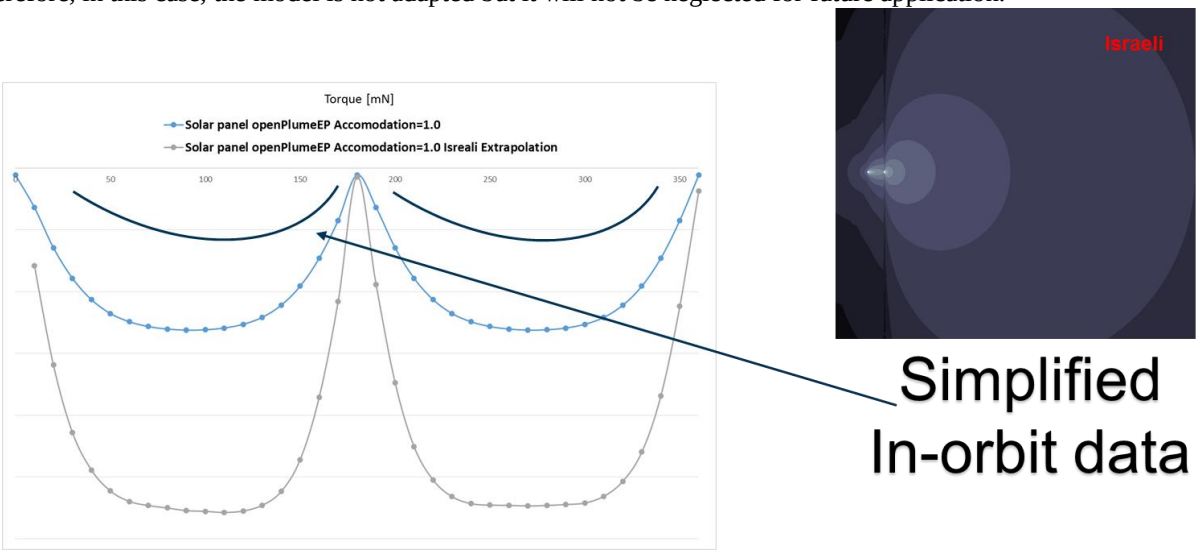

Figure 17 Forces and torques impact with Soreq model

The Self Similar Model (SSM) is a complete set of mathematical equation that can model dense low temperature jets such as arcject or rarified HET thruster with a simplified isothermal model. Details of equations are discarded in

this current paper. The general idea is to solve the density and the velocity step by step with some quantities being specific to the studied thruster.
The results of the generated torques are more promising in this case since it got closer to the final in-orbit data:

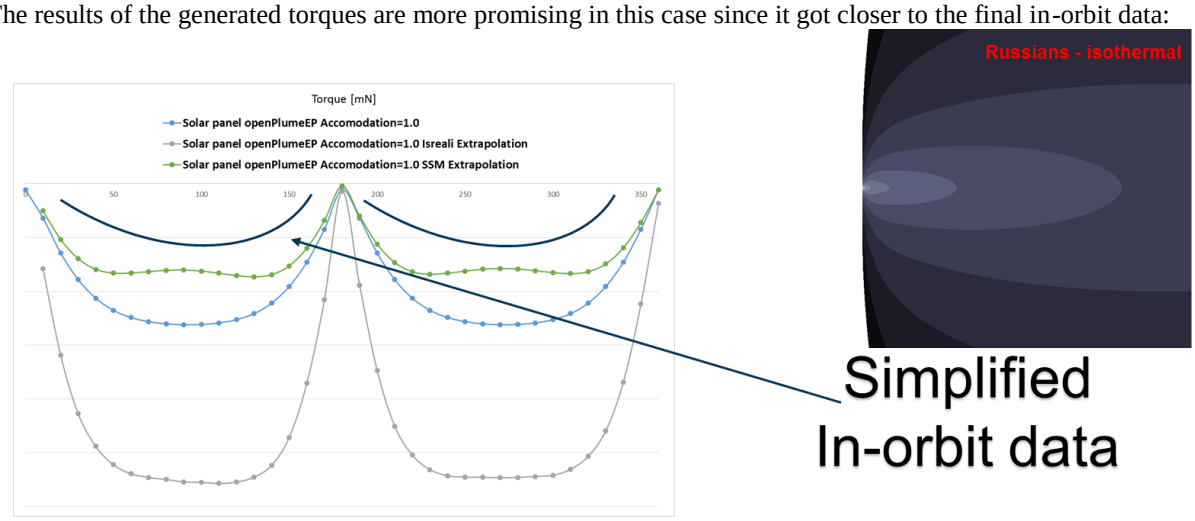

Figure 17 Forces and torques impact with SSM model

It has to be noticed that these implementations were quite easy to perform within openPlumeEP thanks to its modular approach and plugins where the main extrapolation in the far field can be overwritten.
This validation gave OHB the confidence in the method especially that a lot of efforts have been done in the thrusters' models from the PIC codes. openPlumeEP shows its powerful approach and any user could code his own extrapolation in the far field

III. Conclusion

openPlumeEP offers to the analysists in OHB the tools to assess the impact of the plume by investigating several methods. The thrusters' providers could also profit from such a tool to convince prime companies about the good plume behavior of their developed products. By being able to load any kind of thrusters' data (experiment, in-orbit data), openPlumeEP can be used beyond its application within OHB.

Extrapolation in the far field is possible by implementing mathematical models within the tools. Next steps in this case would consist in switching from diverse mathematical models depending on the distance. This work in progress within OHB in parallel to the PIC models.

It has to be noticed that openPlumeEP is distributed under GNU-GNP v3 license for free. Special offers for training are available by contacting the main author.

IV. Acknowledgments

The main author would like to thank all his OHB colleagues for their constant support.

References

[1] J. Laube and et al., "EP plasma plume: Modelling, validation status and way forward," in Space Propulsion Conference, Seville, 2018.

[2] B. Zitouni and et al., "Electric Propulsion Plume Modelling: OHB approach," in *the 35th International Electric Propulsion Conference, Atlanta, Georgia, USA, IEPC-2017-135*, 2017.

[3] M. R. Nakles, W.A. Hargus et al. "A performance Comparison of Xenon and Krypton Propellant on an SPT-100 Hall Thruster", *in the 32th International Electric Propulsion Conference, Weisbaden, Germany, IEPC-2011-003*, 2011

[4] J. Szabo et al. "Measurements of a Krypton Fed 1.5 kW Hall Effect Thruster with a centrally Located Cathode", in the 35th International Electric Propulsion Conference, Atlanta, Georgia, USA, IEPC-2017-26, 2017

[5] J. Laube and et. al., "EP plasma plume in orbit: Diagnostics and analysis correlation" in *the 36th International Electric Propulsion Conference, Vienna, Austria, IEPC-2019-294*, 2019.

[6] A.G. Korsun, E.M. Tverdokhlebova et. al., "Mathematical model of hypersonic plasma flows expanding in vacuum", October 2004

[7] J. Ashkenazy et. al "Plasma Plume Far Field Analysis", October 2001