How to control NUM from UI in SPIS 5

This html page presents the full list of parameters used by the numerical kernel to control the simulation. It constitutes a key complement to the SPIS5 User Manual.

Introduction

A first way of controlling the execution of SPIS/NUM solvers is through the source code. The object oriented (OO) language Java allows an easy handling of objects like a Spacecraft, a Plasma or a VolumeDistribution. In practice, this can be done either:

- directly in the NUM Java source code as documented in Java for NUM.html (Java basics), NUM architecture.html (code architecture) and NUM integration in framework.html (practical file integration)
- through the Jython command line of SPIS/UI (still to be documented)

The second simplified way of controlling the execution of SPIS/NUM is through a more classical user interface, offering the capability to modify parameters, either global or local. This is the subject of this page. Of course it reduces somewhat the range of possibilities with respect to what is really supported by the solvers.

The Advanced users may look at the *source code of* ..\API\public\spis\Top\Simulation\SimulationFromUIParams.html, or sometimes other routines, to see how the global parameters control the simulation at top level. A practical way to find what a global parameter is used for, is to search where is extracted, which can easily be done by looking where the String variable containing its name is used. All these static Strings are defined in the Common class (and a global parameter name should always be taken from these common variables, never hard-coded anywhere in the code).

Global parameters are presented first, local parameters, i.e. fields living either in the volume or on a surface (spacecraft or external boundary), are presented next.
Note the last column of the tables below, stating whether each property is in use or not as of this software version (currently 5.0.2).

As explained in UI documentation, local parameters are based on Material/Electric/Plasma properties that can be edited via GUI, and are then to be assigned locally on your CAD model group per group thanks to UI Group Editor. Global parameters are simply edited in UI Global Parameter Editor.

Global parameters

The general behaviour of NUM solvers is ruled by global parameters. They are organised by sections:

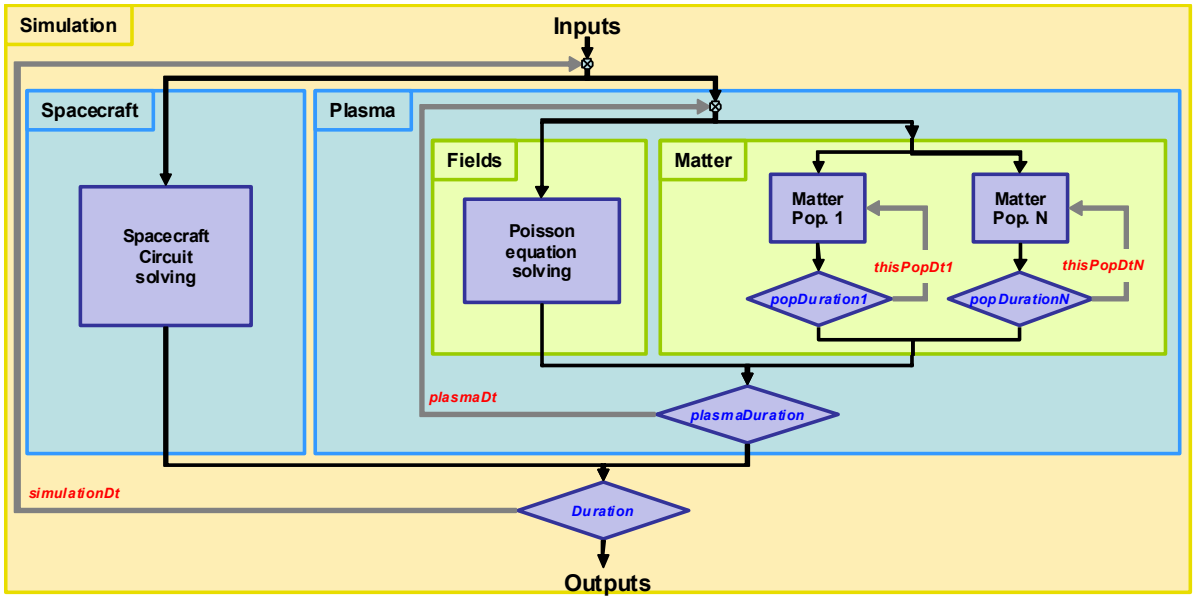
- Volume interactions
- Outputs
- Scenario
- Transitions

In each section the parameters are first reviewed, then listed in a table. All parameters are given with default values and level of expertise. Lower level of expertise can easily be changed by users while more expert levels need better understanding of numerical models. The default values given here are thought to be relevant of most situations but this is the responsibility of the user to check their relevance. The next pages should help understanding the role of each parameter.

Simulation control

The hierarchical structure of a simulation is outlined in the figure below. The nested boxes reflect the object structure of the code (what the basic user user may not care about) while the arrows represent the time evolution of a simulation (what he needs to be aware of).

The time integration process was described in SPIS 5 User Manual. We only give here some extra etails for advanced understanding of the numerical loops.



The **largest allowable time steps** at each nested level are written in red. They are controlled by user defined parameters:

- At particle population level (noted *thisPopDt*): *in the past* the maximum allowable time steps was to be defined so that particles do not cross more than a fraction of a cell at each times step (CFL-like condition). These maximum time steps are defined below in sections plasma / particle sources / interactions respectively for ambient particles, actively emitted particles and secondary particles (parameters *xxxxDt*). They can either be user-defined (best if user can do that!) or automatically determined by the code (default). The user must be warned that the code automated time step determination is rather coarse as it is based on the particle steps at injection. If particles are strongly accelerated after injection, time step can get

too coarse and result in inaccuracies in trajectory integration (define them manually in such cases). However *today*, the improved trajectory integration scheme (see PIC model) suppresses this constraint. The integration is either exact (parabolic trajectories), or with an adaptive time step (subcycling) with controlled accuracy (Runge-Kutta Cash-Karp method, since SPIS v4.0; and more efficient dichotomy-like method since SPIS v5).

- At plasma level (noted *plasmaDt*): here the maximum authorised time step *plasmaDt* should be smaller than a plasma period ($\sim 0.2 T_p = 0.2*2\pi/\omega_p$). In principle if matter populations are sped up (see numerical times below), the allowed *plasmaDt* is increased by the same factor (since the numerical integration time of these populations is reduced by that factor). If *plasmaDt* is set to 0, it is determined semi automatically: the lower level time step (smallest time step allowed among matter populations) is used, which is a degraded criterion compared to the plasma period (however for cells larger than Debye length, the CFL condition for particles is stronger than this plasma stability criterion, hence this semi-automatic setting is sufficient for stability, even though not optimal).
- At simulation level (noted *simulationDt*): the top level time step *simulationDt* is constrained by the stability of the SC-plasma coupling (for a floating SC). The smaller the SC capacitances, the faster the SC dynamics, the smaller this time step must be. Theoretical limits are *simulationDt* << *Capacitance* × *Potential* / *CollectedCurrent* (at eigenvalue level for local values). In practice SC absolute capacitance C_{sat} is the smallest capacitance and yields the largest eigenvalue, hence the stability criterion is often *simulationDt* << C_{sat} × *spacecraftPotential* / *totalCollectedCurrent*. If the exact time evolution of the SC absolute potential is not of major concern to the user (if equilibrium only is pursued) the value of C_{sat} can thus be overestimated to improve stability and/or maximum allowed *simulationDt* (C_{sat} becomes a parameter of numerical convergence to steady state). If *simulationDt* is set to 0, it is determined semi automatically: the lower level time step (*plasmaDt*) is used, which is a degraded criterion compared to the *CU*/*I* eigenvalue criterion (but usually stronger, hence insuring stability)

Since SPIS v4.0 an extra constraint applies to the maximum allowable time scale, the validity of dV. As explained in the circuit solver technical note, in the new implicit solver, current change estimations are supplied to the circuit solver with a given validity range for potential changes. On the other hand some of the above constraints can be relaxed with this new feature, in particular the one on *simulationDt* which is in a way taken into account automatically by the solver.
Combined with the automatic determination of the times step of the new circuit solver (cf. *simulationDt* parameter), much faster convergence can often by achieved.

At each level involving different time structures, if ever one of the characteristic times is faster than the others it may be sped up by considering that the fast process dynamics is quasi-static as compared to the slow one. In such a case (to be assessed by the user), the dynamics of the slower process may not need to be modelled during as long a duration as the fast one, simply because it reached its steady state in a smaller amount of time. Among the main three nested levels of the chart (Simulation, Plasma, Matter), this method, that we may call **numerical times**, can be used at the two lower levels:

- at matter level: computation of faster populations dynamics (typically electrons or sometimes fast ions) can be sped up by only integrating over a smaller time for these populations than for the others. This is controlled by two parameters per population (see below in sections plasma / particle sources / interactions respectively for ambient particles, actively emitted particles and secondary particles). It is the responsibility of the user to control the validity of the quasi-steadiness assumption. These parameters *pop#Dt*, *pop#Duration* are described in SPIS5 User Manual.

- at plasma-SC level: plasma dynamics is often faster than spacecraft charging (at least differential charging). Plasma can thus often be considered as stationary at the large time scale of surface potential dynamics (to be checked by user in each case). If so, plasma dynamics can be sped up thanks to the parameters *plasmaDt* and *plasmaDuration* also described in SPIS5 User Manual.

NB: in some modelling cases, these speed up parameters are irrelevant (Boltzmann electrons, SC at fixed potential...).

Since SPIS v4.3, it is possible for the user to define the integration duration for the plasma and the populations (as e.g. for ambient ions). For the sake of completeness, the compatibility with older SPIS versions is described in Time_steps_RC_4.3.

WARNING: We recommend to use the parameters described in SPIS 5 User Manual : i.e. : simulationDt, plasmaDuration, plasmaDt, pop#Dt and pop#Duration and to avoid automatic modes instead of the *speed up, fixedDt* parameters.

Name	Type	Default Value	Unit (cf allowed units)	Description	Expertise Level	In use
didvRelaxationTime	double	1E+30	[s]	dIdV relaxation time	Advanced	since SPIS5
dimensionality	int	3	[-]	Physical dimensionality of the assumptions done in code	Advanced	since SPIS5
duration	double	1	[s]	Duration of the simulation	Low	yes
fixedDt	int	0	[-]	flag to have fixed integration duration dt of all populations (if yes, durations will be each population dtMax) (0 is *recommended = no*, 1=yes)	Expert	yes
fixedSimulationDtFlag	int	0	None	flag to define the time step evolution mode: 0 => automatic calculation (as a function of the validity: maximum time step equal simulationDt), 1 => fixed time step equal to SimulationDt (Infinite validity for the current scalers - if activated)	Expert	no
noCurrentScalerFlag	int	0	None	flag to desactivate the dI/dV calculation: 0 => activated, 1 => disactivated (zero current variation asumed in the implicit circuit solver)	Expert	no
plasmaDt	double	1E-05	[s]	Time step for global plasma dynamics (semi-automatic if 0: determined by Lower level time step = smallest matter dt)	Medium	yes
plasmaDuration	double	1E-05	[s]	Integration duration of the plasma dynamics (automatic if 0: plasma dynamics is only integrated over a fraction 1/plasmaSpeedUp of actual physical time)	Medium	yes
plasmaSpeedUp	double	1	[-]	Numerical times speed-up factor for plasma (plasma dynamics is only integrated over a fraction 1/plasmaSpeedUp of actual physical time): *not recommended use*	Expert	yes
plasmaUnderRelaxTimeCstt	double	0	[s]	under-relaxation time constant for plasma (default=0 => no under-relaxation). If not 0, at each step of the Poisson-matter loop: - Poisson eq. is solved, giving the E_{soled} solution - The new electric field is computed as $(1-w)* E_{soled} + w*E_{old}$ where E_{old} is the old electric field, $w = e^{-dt/underRelaxationTimeConstant}$ a weight function, .and *dt* the time step of the loop.	Expert	yes

				It amounts to underrelaxing with time constant plasmaUnderRelaxTimeCstt (leading e.g. to an exponential decay with this time constant in case a step-like variation of density in Poison eq.)		
scenario	String	Scenario	None	(possible) scenario for the simulation. Name of the scenario used to run successive simulations (or simulation with externally-induced changes). The default value, scenario = "Scenario", is trivial (no changes). See the Scenario section for the general rules, the example of the PotentialSweep Scenario, and the meaning of the scenario parameters in this case.	Expert	yes
scenarioParameter1, scenarioParameter2, etc.	String		[-]	Scenario parameters, with a specific meaning depending on each Scenario. See the Scenario section for the meaning of the scenario parameters for the case of the PotentialSweep Scenario.	Expert	yes
simulationDt	double	0.05	[s]	see SPIS 5 User Manual (recommended : positive value)	Low	yes
simulationDtInit	double	0.0001	[s]	initial time step for global simulation dynamics (only used if simulationDt >0)	Medium	yes
simulationDtMaxFactor	double	5	[s]	maximum amplification factor of the global simulation dynamics time step	Advanced	yes
spisGEO	int	0	None	flag to define SPIS-GEO-MEO automatic settings (1: activated, *recommended for Geo/MEO surface charging applications*)	Medium	since SPIS5

Back to top.

Plasma

This section defines the environment through two distributions of electrons and two of ions. The total should be neutral (not enforced).

The major point to be noted is that some of the parameters are names of classes. It means that Java generates a class from its name, which is possible thanks to the powerful introspection capabilities of Java. Reasonable defaults are provided for these classes, to which shy users can stick.
The general rule for the *environmentType* parameter, which defines the environment, is:

- this class must derive from the class Environment
- have a specific constructor including the UI-defined parameters as described in "Writing UI-supported classes" page and in ..\API\public\spis\Top\Plasma\Environment.html
- in practice in SPIS 4 following these specifications only BiMaxwellianEnvironment was implemented, which may involve two Maxwellians or only one by setting the second one(s) to zero density (as in the defaults)

The general rules for the ionDistrib* electronDistrib* parameters, which define the 4 particle populations (2 of ions, 2 electrons), is:

- these classes must derive from the class VolDistribWithIO
- have a specific constructor including the UI-defined parameters as described in "Writing UI-supported classes" page and in VolDistrib\VolDistribWithIO.html
- in practice since SPIS v4.0 the following distributions were available:

- GlobalMaxwellBoltzmannVolDistrib: describes a particle population as a global thermal equilibrium distribution (Maxwell-Boltzmann) and is usually valid when no attractive potential or potential barrier exists (density increase is limited to a linear variation for positive potential)
- UnlimitedGlobalMaxwellBoltzmannVolDistrib: similar Maxwell-Boltzmann distribution but density increase is not limited for positive potential (remains exponential)
- PICVolDistrib: really simulates this population dynamics but is much more costly in computation time and memory
- BackTrackingVolDistrib: computes currents onto spacecraft surface through backtracking (but does not compute densities!)
- BacktrackingBoltzmannCompositeVolDistrib: computes currents onto spacecraft surface through backtracking and densities through Boltzmann distribution
- BacktrackingPICCompositeVolDistrib: computes currents onto spacecraft surface through backtracking and densities through Boltzmann distribution
- HybridMZVolDistrib: hybrid multi-zone volume distribution: two different volume distributions are used in two different zones: Boltzmann in large density zone (quasi neutral) and PIC in lower density region, cf. Multi-physical modelling algorithms technical note
- NoSinkHMZVD: similar hybrid multi-zone volume distribution but this population is not in contact with a sink or unlimited source (as e.g. the ambient environment), hence a balance for these particles is to be computed (still experimented, limited stability, cf. Multi-physical modelling algorithms technical note)
- LocalMaxwellVolDistrib: simple constant distribution, mostly used for debugging.

The supported types of particle are currently electron, H+, O+, H2O+, Xe, Xe+, Xe++, Ar, Ar+, Ar++, Cs, Cs+, In, In+, C+ and Si+ but can easily be increased (see *the source of* ..\API\public\spis\Top\Default\SpisDefaultPartTypes.html).

Name	Type	Default Value	Unit	Description	Expertise Level	In use
avPartNbPerCell	double	5	None	average number of super-particle per cell. NB: the average particle number per node is more relevant because computation is mostly on the nodes. It is 6 times bigger, this is why avPartNbPerCell can be rather small ~ 5	Advanced	yes
BFieldIterativePusher	int	1	None	flag for particle pusher method used in case of magnetic field (0: RKCK algorithm from spis 4.3; 1: Dichotomy method from spis 5)	Advanced	since SPIS5
btPartNbPerSurf	int	20	None	number of super-particle generated per surface element for back tracking	Advanced	yes
chargeDepositDuringIntegrationFlag	int	1	[-]	flag for setting charge deposit in volume of PIC distribution during instead of after the trajectory integration; 0: after ; 1: during	Medium	yes
electronDensity	double	1E+06	[m-3]	Electron density (1st population)	Low	yes
electronDensity2	double	1E+06	[#/m3]	Electron density (2nd population)	Low	yes

electronDensityCutoff	double	0	[m-3]	truncation of elec density in case of fluid model	Advanced	yes
electronDistrib	String	GlobalMaxwellBoltzmannVolDistrib	None	Name of the VolDistrib class to be used for electrons	Medium	yes
electronDistrib2	String	PICVolDistrib	None	Name of the VolDistrib class to be used for the 2nd electron population	Medium	yes
electronDt	double	1E-06	[s]	Maximum integration time step for electron 1st population (see SPIS 5 User Manual)	Medium	yes
electronDt2	double	1E-07	[s]	Maximum integration time step for electron 2nd population (see SPIS 5 User Manual)	Medium Medium	yes
electronDuration	double	1E-06	[s]	Maximum integration duration for electron 1st population (see SPIS 5 User Manual)	Medium	yes
electronDuration2	double	1E-07	[s]	Maximum integration duration for electron 2nd population (see SPIS 5 User Manual)	Medium	yes
electronSpeedUp	double	1	[-]	Numerical times speed-up factor for electron 1st population	Expert	yes
electronSpeedUp2	double	1	[-]	Numerical times speed-up factor for electron 2nd population	Expert	yes
electronTemperature	double	1	[eV]	Electron temperature(1st population)	Low	yes
electronTemperature2	double	100	[eV]	Electron temperature(2nd population)	Low	yes
electronTrajFlag1	int	0	[-]	Plot ambient electron (1st population) trajectory? 0=no, 1=yes. NB: extra trajectory parameters must be defined	Advanced	yes
electronTrajFlag2	int	0	[-]	Plot ambient electron (2nd population) trajectory? 0=no, 1=yes	Advanced	yes
electronVx	double	0	[m/s]	electron drift velocity along x axis (1st population)	Medium	yes
electronVx2	double	0	[m/s]	electron drift velocity along x axis (2nd population)	Medium	yes
electronVy	double	0	[m/s]	electron drift velocity along y axis (1st population)	Medium	yes
electronVy2	double	0	[m/s]	electron drift velocity along y axis (2nd population)	Medium	yes
electronVz	double	0	[m/s]	electron drift velocity along z axis (1st population)	Medium	yes
electronVz2	double	0	[m/s]	electron drift velocity along z axis (2nd population)	Medium	yes
environmentType	String	BiMaxwellianEnvironment	None	Name of the Environment class to be used (see SPIS 5 user manual annex for extended environment)	Advanced	yes
ionDensity	double	1E+06	[m-3]	Ion density (1st population)	Low	yes
ionDensity2	double	1E+06	[#/m3]	Ion density (2nd population)	Low	yes
ionDistrib	String	BackTrackingVolDistrib	None	Name of the VolDistrib class to be used for ions	Advanced	yes
ionDistrib2	String	PICVolDistrib	None	Name of the VolDistrib class to be used for ions 2nd population	Advanced	yes
ionDt	double	0.0001	[s]	Maximum integration time step for ion 1st population (see SPIS 5 User Manual)	Medium	yes
ionDt2	double	1E-05	[s]	Maximum integration time step for ion 2nd population (see SPIS 5 User Manual)	Medium	yes

ionDuration	double	0.0001	[s]	Maximum integration duration for ion 1st population (see SPIS 5 User Manual)	Medium	yes
ionDuration2	double	1E-05	[s]	Maximum integration duration for ion 2nd population (see SPIS 5 User Manual)	Medium	yes
ionSpeedUp	double	1	[-]	Numerical times speed-up factor for ion 1st population	Expert	yes
ionSpeedUp2	double	1	[-]	Numerical times speed-up factor for ion 2nd population	Expert	yes
ionTemperature	double	1	[eV]	Ion temperature (1st population)	Low	yes
ionTemperature2	double	100	[eV]	Ion temperature (2nd population)	Low	yes
ionTrajFlag1	int	0	[-]	Plot ambient ion (1st population) trajectory? 0=no, 1=yes	Advanced	yes
ionTrajFlag2	int	0	[-]	Plot ambient ion (2nd population) trajectory? 0=no, 1=yes	Advanced	yes
ionType	String	H+	None	First ion population	Low	yes
ionType2	String	H+	None	Second ion population	Low	yes
ionVx	double	0	[m/s]	Ion drift velocity along x axis (1st population)	Medium	yes
ionVx2	double	0	[m/s]	Ion drift velocity along x axis (2nd population)	Medium	yes
ionVy	double	0	[m/s]	Ion drift velocity along y axis (1st population)	Medium	yes
ionVy2	double	0	[m/s]	Ion drift velocity along y axis (2nd population)	Medium	yes
ionVz	double	0	[m/s]	Ion drift velocity along z axis (1st population)	Medium	yes
ionVz2	double	0	[m/s]	Ion drift velocity along z axis (2nd population)	Medium	yes
iterativePusherAbsTolPos	double	1E-05	[m]	precision of iterative particle pusher (RKCK method): absolute tolerance position	Expert	yes
iterativePusherAbsTolVelo	double	1E+12	[m/s]	precision of iterative particle pusher (RKCK method): absolute tolerance velocity	Expert	yes
iterativePusherRelTolPos	double	0.001	None	precision of iterative particle pusher (either RKCK or dichotomy method): relative tolerance position	Expert	yes
iterativePusherRelTolVelo	double	0.001	None	precision of iterative particle pusher (RKCK method): relative tolerance velocity	Expert	yes
lmvdSubType	String	uniform	None	sub-type of the LocalMaxellVolDistrib if an ion/elec distrib is declared LocalMaxellVolDistrib. It can be 'uniform', 'linear', 'stepwise', 'constant-linear' or 'bubble' (see in LocalMaxwellVolDistrib source for details)	Expert	yes
maxwellEnergySamplerFactor	double	1.3	None	spacing geometric factor of the maxwellian energy sampler	Advanced	yes
maxwellEnergySamplerPointNb	int	100	None	number of points of the maxwellian energy sampler	Advanced	yes
maxwellEnergySamplerSpacing	double	0.01	None	first spacing of the maxwellian energy sampler [eV]	Advanced	yes

In addition, **since SPIS 5**, generic distributions functions are available (kappa, ASCII tabulated files) : see ***SPIS 5 user manual annex "Advanced uses for scientific applications"***). They are set with new parameters referred as pop# in next table where #is the index of the generic distributrions, which extend the bi-maxwellian environment of SPIS 4.

Name	Type	Default Value	Unit	Description	Expertise Level	In use
environmentType	String	BiMaxwellianEnvironment	None	Name of the Environment class to be used (see SPIS 5 user manual annex for extended environment)	Advanced	yes
ExtendedPopNbr	int	0	None	if *environmentType = ExtendedEnvironment*, number of extended populations	Advanced	yes
pop1Density	double	0	[m-3]	Population density	Advanced	yes
pop1DF_FileName	String	None	[-]	name of the file describing the population distribution function in the environment	Advanced	yes
pop1DFBasis_Vect1_X	double	1	[-]	x coordinate of Vect1 defining the basis of the population distribution function	Advanced	yes
pop1DFBasis_Vect1_Y	double	0	[-]	y coordinate of Vect1 defining the basis of the population distribution function	Advanced	yes
pop1DFBasis_Vect1_Z	double	0	[-]	z coordinate of Vect1 defining the basis of the population distribution function	Advanced	yes
pop1DFBasis_Vect2_X	double	0	[-]	x coordinate of Vect2 defining the basis of the population distribution function	Advanced	yes
pop1DFBasis_Vect2_Y	double	1	[-]	y coordinate of Vect2 defining the basis of the population distribution function	Advanced	yes
pop1DFBasis_Vect2_Z	double	0	[-]	z coordinate of Vect2 defining the basis of the population distribution function	Advanced	yes
pop1Distrib	String	PICVolDistrib	[-]	distribution type of 1st extended population	Advanced	yes
pop1Dt	double	-1	[s]	Maximum integration time step for 1st extended population	Advanced	yes
pop1Duration	double	0	[s]	Maximum integration duration for 1st extended population	Advanced	yes
pop1EnvironmentDF	String	IsotropicMaxwellianDF	[-]	Population distribution function in the environment	Advanced	yes
pop1Kappa	double	9	[-]	Population kappa parameter (if kappa distribution)	Advanced	yes
pop1Optimization	double	0	[-]	optimize population statistics by injecting new particles. Example: if 0.5 => add 50 % of optimized particles to the original list	Advanced	yes
pop1OptimizationMode	double	1	[-]	if pop1Optimization is positive, mode of	Advanced	yes

				statistics collection ? 1 : after particle integration; 2 : during particle integration		
pop1SEEFlag	int	0	[-]	secondary Emission Flag Under Electron or Proton Impact: bits go by groups of 3 (bit0=on/off, bit1=simulate_secondary_elec_dynamics/don't, bit2=allow_secondaries_of-secondaries/don't)	Advanced	yes
pop1SpeedUp	double	1	[-]	Numerical times speed-up factor for population	Advanced	yes
pop1Temperature	double	1	[eV]	Population temperature (if isotropic)	Advanced	yes
pop1TrajFlag	int	0	[-]	plot population trajectory ? 0=no, 1=yes	Advanced	yes
pop1Tx	double	0	[eV]	Population temperature along x axis	Advanced	yes
pop1Ty	double	0	[eV]	Population temperature along y axis	Advanced	yes
pop1Type	String	H+	None	Population type	Advanced	yes
pop1Tz	double	0	[eV]	Population temperature along z axis	Advanced	yes
pop1Vx	double	0	[m/s]	Population drift velocity along x axis	Advanced	yes
pop1Vy	double	0	[m/s]	Population drift velocity along y axis	Advanced	yes
pop1Vz	double	0	[m/s]	Population drift velocity along z axis	Advanced	yes
pusherThreadNb	int	4	None	Number of parallel particle pushers	Low	since SPIS5

Back to top.

MultiZone

This section describes the fine tuning parameters for the MultiZone modelling (HybridMZVolDistrib) of a population.
They should only be modified by Advanced users.
Beyond the short description below, the Advanced user can find a detailed effect of these parameters in the source code of HybridMZVolDistrib.

Name	Type	Default Value	Unit	Description	Expertise Level	In use
hmzvdPoissonVlasovLoopNb	int	3	None	number of Poisson-Vlasov loops within each jCL factor adjusting iteration in a HybridMZVolDistrib	Expert	yes
jclfAdjustLoopNb	int	3	None	number of loops for adjusting jCL factor within a HybridMZVolDistrib	Expert	yes
jclfCVSpeed	double	1	None	convergence speed for jCL factor	Expert	yes
jclfExtractingFieldWeight	double	1	None	weighing factor for the presence of an extracting electric field at boundary between zones, which leads to a jCLfact increase	Expert	yes
jclfLowerBound	double	0.01	None	Lower bound for jCL factor	Expert	yes

jclfPosChargeWeight	double	1	None	weighing factor for the presence of positive space charge in a positive sheath, which leads to jCLfact reduction to avoid instabilities	Expert	yes
jclfReattractingFieldWeight	double	1	None	weighing factor for the presence of a re-attracting electric field at boundary between zones, which leads to a jCLfact reduction	Expert	yes
jclfSmoothing	double	1	None	smoothing strengh for jCL factor at each iteration	Expert	yes
jclfUpperBound	double	100	None	upper bound for jCL factor	Expert	yes
neLowerBoundCoeff	double	1	None	coefficient ruling the Lower boundary on Ne estimate for jTh computation: small => less constraint, big => ne close to ni	Expert	yes
zoneBdElecDensification	double	1	None	densification coefficient (increases superparticle number, decreasing their weight) for PIC electrons emitted at zone boundary	Expert	yes

Back to top.

Poisson equation

Poisson boundary conditions are:
- always Dirichlet on the spacecraft (fixed potential), the initial potential being controlled by the global parameter initPotFlag (spacecraft section) and possibly defined locally (see the local parameters)
- Fourier on the external boundary (mixed Dirichlet-Neumann) with parameters defined so as to give an asymptotic behaviour in r^{-n}. Dirichlet is also possible on the external boundary (this eliminates the observed detrimental interactions at an edge between Fourier BCs on two nearby external surfaces when one of the Fourier is used to mimic a quasi Dirichlet BC).

They are controlled by the *poissonBCType* parameter.

The non-linear Poisson equation includes one (or two) Maxwellian distributions of electrons:

$$-\Delta\phi = e(n_i - n_{e1}\, e^{e\phi/kTe1}) / \varepsilon_0 \qquad \text{or} \qquad -\Delta\phi = e(n_i - n_{e1}\, e^{e\phi/kTe1} - n_{e2}\, e^{e\phi/kTe2}) / \varepsilon_0$$

where $e\, n_i$ is the total charge density of other particles (usually PIC-modelled ions, but possibly also other PIC-modelled electrons), and n_{ex} is the electrons density of the x-th electron distribution (a scalar, contrarily to n_i which is a field) and T_{ex} its temperature.
If the non linear solver is selected (linearPoisson = 0), the Boltzmann electron distribution(s) of the environment (Plasma section above) are automatically inserted in the non-linear Poisson solver (but not electron distributions that you defined as PIC, which are handled like ions in the above non-linear equation).
The non-linear Poisson solver follows an implicit scheme (Newton type), which has the major advantage to be stable even for cells larger than Debye length.

The next parameters, controlling the maximum iteration number or tolerance of the conjugate gradient Poisson equation solver, are rather for specialists.

Name	Type	Default Value	Unit	Description	Expertise Level	In use
iterGradient	int	1000	None	Maximum iteration number for conjugate gradient Poisson Solver	Expert	yes
iterGradientNl	int	1000	None	Maximum iteration number for conjugate gradient non-linear Poisson Solver	Expert	yes
iterLinearSys	int	10000	None	Maximum iteration number for linear system solver (used for capacitance matrix inversion)	Expert	yes
iterNewton	int	100	None	Maximum iteration number for Newton algorithm in non-linear Poisson solving	Expert	yes
linearPoisson	int	0	None	if 1 linear Poisson solver, if 0 non-linear. 0- no: use non-linear Poisson equation solver (implicit Newton scheme): $-\Delta\Phi = [q_i\, n_i - e\, n_{e1} \exp(e\Phi/kT_{e1}) - e\, n_{e2} \exp(e\Phi/kT_{e2})] / \varepsilon_0$ where n_{e1} = first electronDensity if a Boltzmann electron distribution is selected (same for 2nd distribution). It can use a truncated electron exponential instead, depending on electron GlobalMaxwellBoltzmannVolDistrib subtype 1- yes: use linear Poisson solver: $-\Delta\Phi = (q_i\, n_i - e\, n_e) / \varepsilon_0$ the local electron density n_e being possibly computed through a Boltzmann law (if selected), but computed with the old potential (i.e. not taking into account its evolution in the equation solving) NB: this distinction is meaningless if no Boltzmann distribution is selected => SPIS-NUM shifts to linear and emits a warning NB: in case of small Debye length (smaller than cell size), only non-linear solver is stable.	Advanced	yes
neutrality	int	0	None	neutrality switch, 0=off => regular Poisson computation, 1=on => imposes neutrality instead of solving Poisson. If on (neutrality = 1), imposes neutrality instead of solving Poisson: $q_i\, n_i - e\, n_{e1} \exp(e\Phi/kT_{e1})$ Only the first ambient electron density is considered, and it must be defined as an UnlimitedGlobalMaxwellBoltzmannVolDistrib (if a PICVolDistrib its PIC density would be taken into account) If off (neutrality = 0), regular Poisson computation	Expert	yes
poissonBCParameter1	double	0	[varies]	Parameter that can be used by some BC types (e.g. 1/rn exponent)	Advanced	yes
poissonBCParameter2	double	0	[varies]	2nd parameter that can be used by some BC types	Advanced	yes
poissonBCType	int	2	None	Poisson boundary conditions type. Defines Fourier (*alpha pot + d(pot)/dn = value*) or Dirichlet boundary condition (*pot = value*) on computation box external boundary: 0- Use the BC defined as fields through plasma group editor (BdFourAlpha and BdFourValue Fourier BC; or BdDiriPot Dirichlet BC) 1- alpha parameter mimicking a $1/r$ decay (~vacuum) 2- alpha parameter mimicking a $1/r^2$ decay (~pre-sheath) 3- alpha parameter mimicking a $1/r^n$ decay, n being next parameter (*poissonBCParameter1*) NB: in any case, BC are Dirichlet on SC, defined by local plasma parameter SCDiriPot	Advanced	yes
tolGradient	double	0.0001	[-]	Tolerance for conjugate gradient Poisson Solver	Expert	yes

tolGradientNl	double	0.0001	[-]	Tolerance for conjugate gradient Poisson Solver when non-linear solving	Expert	yes
tolLinearSys	double	1E-08	[-]	Tolerance for linear system solver (used for capacitance matrix inversion). May have to be further reduced when strongly multiscale SC mesh is used (resulting in very variable areas and capacitances of elements). If not, local surface potential is not solved on the smallest surface elements.	Expert	yes
tolNewton	double	0.02	[-]	Tolerance for Newton algorithm loop in non-linear Poisson solving	Expert	yes
vacuum	int	0	None	flag for vacuum computation (0=off, 1=on), if on and linearPoisson is on solves Laplace equation, if on and linearPoisson is off only sets ion space charge to zero in Poisson eq.	Expert	yes
variableTe	int	0	None	flag to use a variable Te in Boltzmann equation (physically meaningless), 0=off, 1=on. If on (variableTe = 1) and neutrality is on, the temperature used in Boltzmann distribution in neutrality equation is derived from $k_B\ T_e\ n_e^{-\gamma+1} = constant$ NB: plugging this adiabatic law directly in Boltzmann distribution like this is physically wrong (one must go back to Boltzmann distribution derivation), but it was implemented for comparison to other codes, where this is sometimes done.	Expert	yes
variableTeConstant	double	1	[eV.m$^{3(\gamma-1)}$] i.e. computed from k_BT_e in [eV] and n_e in [m^{-3}], but not checked by the code (user can fill in any unit)	constant in the variable Te law	Expert	yes
variableTeGamma	double	1.1	[-]	gamma adiabatic exponent in the variable Te law	Expert	yes

Back to top.

B field

A uniform B field can be defined this way.

Name	Type	Default Value	Unit	Description	Expertise Level	In use
Bx	double	0	[T]	x-component of the magnetic field (uniform over the computation box)	Low	yes
By	double	0	[T]	y-component of the magnetic field	Low	yes
Bz	double	0	[T]	z-component of the magnetic field	Low	yes
magnetizedPlasmaFlag	int	1	[T]	flag for taking account the effect of the magnetic field and of the magnetically induced electric field (due to spacecraft motion) on particle trajectories. 1: yes; 0: no (un-magnetized plasma)	Medium	since SPIS5

NB: like in many domains, the solver can indeed handle more general situations, here a local B field, the restriction to a uniform B field coming from the UI only. A dipolar B field can thus e.g. easily be handled by coding it in the software (the only work is to generate such a local map; solvers are then apt to use it directly).

Back to top.

Spacecraft

If *electricCircuitIntegrate* = 0, spacecraft potentials are constant, if *electricCircuitIntegrate* = 1, the spacecraft floats, the relative capacitances being derived from material properties, whereas the spacecraft absolute capacitance is defined by the parameter *CSat*.
See Spacecraft circuit description for details on circuit model.

Since SPIS 5, it is possible to apply an orbital velocity directly to the spacecraft instead of applying the opposite velocity to the plasma. In general, the two approaches are strictly identical, except when there is a magnetic field since it induces a VcrossB electric field on spacecraft surface. In this case, the VcrossBfield is calculated by using the scveloX,Y,Z, which define the velocity components of the spacecraft in the (possibly drifting) plasma referential frame. As e.g., for a spacecraft flowing at -100 km/s wrt to a referential at rest, and a plasma drifting at 400 km/s wrt to the same referential at rest : scVeloX = -500 km/s in the referential of the drifting plasma. When using this parameter, it is unecessary to use the "popVx" parameters (except to add some population extra velocity vs drift velocity).

Name	Type	Default Value	Unit	Description	Expertise Level	In use
circuitSolverMode	int	0	None	flag to define the circuit solver mode: 0 => implicit solver, 1 => explicit solver	Expert	no
CSat	double	1E-06	[F]	Spacecraft absolute capacitance. It represents the capacitive coupling between spacecraft and infinity (the capacitance electrodes are the spacecraft and its sheath). This capacitance is spread over all electric nodes proportionally to their areas, i.e. split into several capacitors between infinity and the electric nodes grounds (as is the real capacitive coupling in space). An alternative is to define a negative *Csat* = *-x*. The absolute capacitance used is then *x* (positive), and it is plugged between infinity and spacecraft ground only (electric node 0) in that case. It can sometimes be useful. See Spacecraft circuit description.	Low	yes
electricCircuitFilename	String	circuit.txt	None	File name of extra electric devices (RLCV). Name of the file describing extra electric devices between electric (super-)nodes. See below for syntax of circuit file. The file must be in the "SpisUI/defaultValues" directory (if no project loaded) or your project	Low	yes

				directory (subfolder NumKernel/Input)		
electricCircuitIntegrate	int	1	None	SC electric circuit integration: 0=no change, 1=floating	Medium	yes
exactCSat	double	0	[-]	flag to ask for an exact computation of spacecraft capacitance (if > 0). More precisely Gauss theorem (integral Poisson equation) is used at each time step to determine the SC potential so as to insure exact charge conservation (a variable Csat is derived from that)	Advanced	yes
implicitCircuitSolver	int	0	None	type of linear system solver used in case of implicit circuit solver; 1: Gauss method; 0: Conjugate Gradient Squared (recommended)	Advanced	since SPIS5
initPot	double	0	[V]	initial potential	Medium	yes
initPotFlag	int	1	None	flag to define initial pot: 0 => set to 0, 1 => set to global initPot, 2 => set to local potential defined as SC Dirichlet condition	Medium	yes
scVeloCrossBFlag	int	1	[-]	flag to take account the effect induced by the spacecraft drift on the spacecraft surface potential (in the reference frame of the plasma)	Advanced	since SPIS5
scVeloX	double	0	[m/s]	x-component of the spacecraft velocity in the reference frame of the plasma. E.g., for a spacecraft flowing at -100 km/s wrt to a referential at rest, and a plasma drifting at 400 km/s wrt to the same referential at rest : scVeloX = -500 km/s in the referential of the drifting plasma. When using this parameter, it is unecessary to use the "popVx" parameters (except to add some population extra velocity vs drift velocity).	Advanced	since SPIS5
scVeloY	double	0	[m/s]	y-component of the spacecraft velocity in the reference frame of the plasma. See above.	Advanced	since SPIS5
scVeloZ	double	0	[m/s]	z-component of the spacecraft velocity in the reference frame of the plasma. See above.	Advanced	since SPIS5
smoothingI	double	0	None	strength of spacecraft surface intensity smoothing at each step (1.0 => 1 step on nearest elements, can be smaller or larger than 1.0)	Advanced	since SPIS5
smoothingPot	double	2	[-]	strength of spacecraft surface potential smoothing at each step	Advanced	yes
validityRenormalisation	double	0.5	[-]	Scaling parameter to globally renormalise validity of scalable currents	Low	yes

Circuit file syntax

The file describing the electric circuit is composed of an arbitrary number of lines, each with the syntax:

```
componentDescriptor node1Id node2Id value
```

with:

- `componentDescriptor` (a string) one of
  - `C` : it is a capacitor of capacitance `value`
  - `R` : it is a resistor of resistance `value`
  - `V` : it is a voltage generator of potential difference `value` ($V_{node2} = V_{node1} + value$)
- `node1Id` and `node2Id` (integers): the Ids of the (super) electric nodes between which to plug the component (same Id as in ElecNodeId)
- `value` (a float): the value of the component (resistance…)

Example file :

```
V 0 1 -10
R 0 2 1.e6
```

```
C 0 3 1.e-10
C 2 3 1.e-10
```

- line 1: Electric super node 1 is biased of -10 V with respect to node 0, which is SC ground (it may be a Langmuir probe).
- line 2: Electric super node 2 is related by a 1 MΩ resistor to SC ground (it may be a solar array).
- line 3: Electric super node 3 is not related by any "real" component to SC ground, so it was chosen to model its capacitive coupling to the ground (this is not necessary, a fraction of SC absolute capacitance CSat is attributed to each electric node, proportionally to its area, so that it does not have zero capacitance, resulting in infinite potential as soon as it collects some charge).
- line 4: the capacitive coupling between nodes 2 and 3 has been added (seldom useful).

Back to top.

Particle sources on spacecraft

These parameters allow the embedding of a plasma sources on the spacecraft (e.g. a thruster). Several sources are allowed, currently 4, but their number can easily be increased by modifying DefaultGlobalParam.py in SpisUI/DefaultValues folder.
Each source is controlled by the following global parameters, which allow turning the source on, defining the source class and particle type. The general rules for the sourceType parameter, which defines the source class, is:

- this class must derive from the class NonPicSurfDistrib
- have a specific constructor including the UI-defined parameters as described in "Writing UI-supported classes" page and in ..\API\public\spis\Surf\SurfDistrib\NonPICSurfDistrib.html
- in practice as of today LocalMaxwellSurfDistrib, AxisymTabulatedVelocitySurfDistrib, TwoAxesTabulatedVelocitySurfDistrib, FowlerNordheimSurfDistrib and MaxwellianThruster are supported.

Extra local parameters allow to switch locally between the sources (sourceId), and to define their parameters (current, temperature, Mach number). It is up to the source model to use or not these local parameters. They will usually use the local source current density, but local temperature and Mach number were really designed for Maxwellian sources and may not be used by others (AxisymTabulatedVelocitySurfDistrib and TwoAxesTabulatedVelocitySurfDistrib use angular distributions defined in files, while FowlerNordheimSurfDistrib is self contained).

Multi-species/multi-sources on the same location are also supported. The rules to perform that are:
- declare sourceTypeX of source number X as a MultipleSurfDistrib
- define the number sourceNbX of its so-called "sub sources" (sourceX.Y is the Y^{th} sub source of sourceX)
- extra global parameters (sourceCurrentFactorX.Y, sourceTempFactorX.Y and sourceMachFactorX.Y) are used to define the multiplication factor to apply to local parameters (current, temperature, Mach number) which are initially defined for sourceX.

Name	Type	Default Value	Unit	Description	Expertise Level	In use

fieldFactor	double	1	[-]	field factor enhancement for Fowler-Nordheim sources (usually named beta)	Advanced	
sourceCurrentFactor1.1	double	1	None	Multiplication factor defining the current of sub source No 1 of source No 1 with respect to source No 1	Advanced	
sourceCurrentFactor1.2	double	1	None	Multiplication factor defining the current of sub source No 2 of source No 1 with respect to source No 1	Advanced	
sourceCurrentFactor1.3	double	1	None	Multiplication factor defining the current of sub source No 1 of source No 1 with respect to source No 1	Advanced	
sourceCurrentFactor1.4	double	1	None	Multiplication factor defining the current of sub source No 2 of source No 1 with respect to source No 1	Advanced	yes
sourceDt1	double	-1	[s]	Maximum integration time step for particles from 1st source (see SPIS5 User Manual)	Advanced	yes
sourceDt1.1	double	-1	[s]	Maximum integration time step for particles from sub source No 1 of source No 1 (automatic if negative)	Advanced	yes
sourceDt1.2	double	-1	[s]	Maximum integration time step for particles from sub source No 2 of source No 1 (see SPIS5 User Manual)	Advanced	yes
sourceDt1.3	double	-1	[s]	Maximum integration time step for particles from sub source No 1 of source No 1 (see SPIS5 User Manual)	Advanced	yes
sourceDt1.4	double	-1	[s]	Maximum integration time step for particles from sub source No 2 of source No 1 (see SPIS5 User Manual)	Advanced	yes
sourceDt2	double	-1	[s]	Maximum integration time step for particles from 2nd source (see SPIS5 User Manual)	Advanced	
sourceDt3	double	-1	[s]	Maximum integration time step for particles from 3rd source (see SPIS5 User Manual)	Advanced	yes
sourceDt4	double	-1	[s]	Maximum integration time step for particles from 4th source (see SPIS5 User Manual)	Advanced	yes
sourceDuration1	double	0	[s]	Maximum integration duration for particles from 1st source (automatic if 0)	Advanced	yes
sourceDuration1.1	double	0	[s]	Maximum integration duration for particles from sub source No 1 of source No 1 (automatic if 0)	Advanced	yes
sourceDuration1.2	double	0	[s]	Maximum integration duration for particles from sub source No 2 of source No 1 (automatic if 0)	Advanced	yes
sourceDuration1.3	double	0	[s]	Maximum integration duration for particles from sub source No 1 of source No 1 (automatic if 0)	Advanced	yes
sourceDuration1.4	double	0	[s]	Maximum integration duration for particles from sub source No 2 of source No 1 (automatic if 0)	Advanced	yes
sourceDuration2	double	0	[s]	Maximum integration duration for particles from 2nd source (automatic if 0)	Advanced	yes
sourceDuration3	double	0	[s]	Maximum integration duration for particles from 3rd source (automatic if 0)	Advanced	yes
sourceDuration4	double	0	[s]	Maximum integration duration for particles from 4th source (automatic if	Advanced	

				0)		
sourceFlag1	double	0	[-]	Flag for defining artificial source No 1 on the spacecraft: 0 => none, 1 => yes, x => number of super-particles densified by x	Advanced	yes
sourceFlag1.1	double	0	[-]	Flag for defining artificial sub source No 1 of source No 1 on the spacecraft (source1.1): 0 => none, 1 => yes, x => number of super-particles densified by x	Advanced	
sourceFlag1.2	double	0	[-]	Flag for defining artificial sub source No 2 of source No 1 on the spacecraft (source1.2): 0 => none, 1 => yes, x => number of super-particles densified by x	Advanced	yes
sourceFlag1.3	double	0	[-]	Flag for defining artificial sub source No 3 of source No 1 on the spacecraft (source1.1): 0 => none, 1 => yes, x => number of super-particles densified by x	Advanced	yes
sourceFlag1.4	double	0	[-]	Flag for defining artificial sub source No 4 of source No 1 on the spacecraft (source1.2): 0 => none, 1 => yes, x => number of super-particles densified by x	Advanced	yes
sourceFlag2	double	0	[-]	Flag for defining artificial source No 2 on the spacecraft: 0 => none, 1 => yes, x => number of super-particles densified by x	Advanced	yes
sourceFlag3	double	0	[-]	Flag for defining artificial source No 3 on the spacecraft: 0 => none, 1 => yes, x => number of super-particles densified by x	Advanced	yes
sourceFlag4	double	0	[-]	Flag for defining artificial source No 4 on the spacecraft: 0 => none, 1 => yes, x => number of super-particles densified by x	Advanced	yes
sourceMachFactor1.1	double	1	None	Multiplication factor defining the mach number of sub source No 1 of source No 1 with respect to source No 1	Advanced	yes
sourceMachFactor1.2	double	1	None	Multiplication factor defining the mach number of sub source No 2 of source No 1 with respect to source No 1	Advanced	yes
sourceMachFactor1.3	double	1	None	Multiplication factor defining the mach number of sub source No 1 of source No 1 with respect to source No 1	Advanced	
sourceMachFactor1.4	double	1	None	Multiplication factor defining the mach number of sub source No 2 of source No 1 with respect to source No 1	Advanced	yes
sourceNb	int	4	None	Number of particle sources: not to be modified in UI.	Advanced	yes
sourceNb1	int	0	None	Number of particles sources of the multi-source 1 (if source 1 is a MultipleSurfDistrib). Nb: create extra parameters for these sub sources	Advanced	yes
sourceNb2	int	0	None	Number of particles sources of the multi-source 2 (if source 2 is a MultipleSurfDistrib). Nb: create extra parameters for these sub sources	Advanced	yes
sourceNb3	int	0	None	Number of particles sources of the multi-source 3 (if source 3 is a MultipleSurfDistrib). Nb: create extra parameters for these sub sources	Advanced	yes
sourceNb4	int	0	None	Number of particles sources of the multi-source 4 (if source 4 is a MultipleSurfDistrib). Nb: create extra parameters for these sub sources	Advanced	
sourceParticleType1	String	Xe+	None	Type of particles emitted by source 1 (a string that must be found in the particle types)	Advanced	yes
sourceParticleType1.1	String	electron	None	Type of particles emitted by sub source No 1 of source No 1	Advanced	

sourceParticleType1.2	String	electron	None	Type of particles emitted by sub source No 2 of source No 1	Advanced	yes
sourceParticleType1.3	String	electron	None	Type of particles emitted by sub source No 3 of source No 1	Advanced	yes
sourceParticleType1.4	String	electron	None	Type of particles emitted by sub source No 4 of source No 1	Advanced	yes
sourceParticleType2	String	electron	None	Type of particles emitted by source 2	Advanced	yes
sourceParticleType3	String	Cs+	None	Type of particles emitted by source 3	Advanced	
sourceParticleType4	String	In+	None	Type of particles emitted by source 4	Advanced	yes
sourceSpeedUp1	double	1	[-]	Numerical times speed-up factor for 1st source population	Advanced	yes
sourceSpeedUp1.1	double	1	[-]	Numerical times speed-up factor for sub source No 1 of source No 1	Advanced	yes
sourceSpeedUp1.2	double	1	[-]	Numerical times speed-up factor for sub source No 1 of source No 1	Advanced	yes
sourceSpeedUp1.3	double	1	[-]	Numerical times speed-up factor for sub source No 1 of source No 1	Advanced	yes
sourceSpeedUp1.4	double	1	[-]	Numerical times speed-up factor for sub source No 1 of source No 1	Advanced	yes
sourceSpeedUp2	double	1	[-]	Numerical times speed-up factor for 2nd source population	Advanced	
sourceSpeedUp3	double	1	[-]	Numerical times speed-up factor for 3rd source population	Advanced	yes
sourceSpeedUp4	double	1	[-]	Numerical times speed-up factor for 4th source population	Advanced	yes
sourceTempFactor1.1	double	1	None	Multiplication factor defining the temperature of sub source No 1 of source No 1 with respect to source No 1	Advanced	yes
sourceTempFactor1.2	double	1	None	Multiplication factor defining the temperature of sub source No 2 of source No 1 with respect to source No 1	Advanced	yes
sourceTempFactor1.3	double	1	None	Multiplication factor defining the temperature of sub source No 1 of source No 1 with respect to source No 1	Advanced	yes
sourceTempFactor1.4	double	1	None	Multiplication factor defining the temperature of sub source No 2 of source No 1 with respect to source No 1	Advanced	yes
sourceTrajFlag1	int	0	None	plot source 1 trajectory (and sub sources if any)? 0=no, 1=yes. NB: in the case source 1 is a multiple source, plot trajectories of each PIC sub source.	Advanced	yes
sourceTrajFlag2	int	0	None	plot source 2 trajectory (and sub sources if any)? 0=no, 1=yes	Advanced	yes
sourceTrajFlag3	int	0	None	plot source 3 trajectory (and sub sources if any)? 0=no, 1=yes	Advanced	yes
sourceTrajFlag4	int	0	None	plot source 4 trajectory (and sub sources if any)? 0=no, 1=yes	Advanced	yes
sourceType1	String	LocalMaxwellSurfDistrib	None	Name of the SurfDistrib class to be used on the spacecraft as source No 1. (ex: LocalMaxwellSurfDistrib, which will use the source flux, source temperature and source Mach user-defined local fields, whereas a specific EP model could only use the source flux and define internally its velocity distribution, see above)	Advanced	yes
sourceType1.1	String	AxisymTabulatedVelocitySurfDistrib	None	Name of the SurfDistrib class to be used on the spacecraft as sub source No 1 of source No 1	Advanced	yes
sourceType1.2	String	MaxwellianThruster	None	Name of the SurfDistrib class to be used on the spacecraft as sub source No 2 of source No 1	Advanced	yes
sourceType1.3	String	FowlerNordheimSurfDistrib	None	Name of the SurfDistrib class to be used on the spacecraft as sub source No 3 of source No 1	Advanced	yes
sourceType1.4	String	TwoAxesTabulatedVelocitySurfDistrib	None	Name of the SurfDistrib class to be used on the spacecraft as sub source	Advanced	yes

				No 4 of source No 1		
sourceType2	String	MaxwellianThruster	None	Name of the SurfDistrib class to be used on the spacecraft as source No 2	Advanced	yes
sourceType3	String	LocalMaxwellSurfDistrib	None	Name of the SurfDistrib class to be used on the spacecraft as source No 3	Advanced	
sourceType4	String	LocalMaxwellSurfDistrib	None	Name of the SurfDistrib class to be used on the spacecraft as source No 4	Advanced	yes

Back to top.

Surface interactions

Surface interactions are related to a population of particle, the one at the origin of the interaction (possibly with a specific handling as for photoemission).
They may or may not also be a source of particles (secondary emission does, but Radiation Induced Conductivity does not).

From the user viewpoint there are two types of interactions:
- a list of predefined interactions (photo-emission, SEE under electron impact, SEE under proton impact, erosion…)
- interactions handled on a generic basis, with the possibility to easily define new ones (since SPIS V4, only CathodeSpot is implemented)

They are described in the two subsections below.
A specific paragraph on the presence of a potential barrier on top of a sunlit surface (important for GEO charging) is to be found at the end of the first subsection.

Predefined surface interactions

These parameters are mostly flags to turn interactions on or off, see their definition in the table below.
For the definition of the interactions, see the classes implementing the interaction:
- Photo-emission: BasicPhotoEmInteractor
- Secondary emission from electrons: BasicSEEEInteractor, SEEEYieldFunction1, ElecBackscatterFunction
- Secondary emission from protons: BasicSEEPInteractor, SEEPYieldFunction1
- Induced conductivity: BasicInducedConductInteractor
- Erosion: ErosionInteractor, GRBOErosionYield, TonduErodedProductSampler

When invoked from UI, these Interactors use a GenericParamSet (since v4.3) witch is composed by a:
- Nascap Parameters for materials, described in NascapParamSet
- Erosion Parameters for materials, described in ErosionParamSet

with the database of built-in materials in SpisDefaultMaterials or using extended NASCAP based materials (since v4.3, see Material Properties).

When photoemission is turned on, the photoelectron current density on illuminated surfaces is calculated as a function of the distance to the Sun. SunX, SunY and SunZ define both the direction of the Sun and the amplification factor wrt the reference flux at 1 AU. (e.g.: SunX=2, SunY = 0, SunZ = 0, will consider a Sun in X direction with a flux multiplied by 2 wrt to conditions at 1 AU).

Name	Type	Default Value	Unit	Description	Expertise Level	In use
electronSecondaryDensification	double	1	[-]	densification coefficient for secondary electron superparticles (from electron impact)	Medium	yes
electronSecondaryEmission	int	3	None	Bits go by groups of 3 : - bit 0: turn on secondary emission under electron impact (if 1), - bit 1: simulate secondary electron dynamics by PIC model (if 1), - bit 2 = model secondary emission form secondary electrons ("hoping") Six groups of 3 bits are used, successively for: - ambient electron population 1 - ambient electron population 2 - source 1 - source 2 - source 3 - source 4 Examples: - binary 011011 = decimal 30 => model secondary emission from both ambient populations, with secondary electron dynamics but no secondaries from secondaries - binary 111000000 = decimal 448 => model secondary emission from source 1 electrons with secondary electron dynamics and secondaries from secondaries ("hoping") NB: in the code, when turned on, the hoping is simulated by a second interactor, which is differentiated from the first interactor for primary electrons in the emittedCurrents.txt file. NB: when secondary emission is turned on for a multipleSurfDistrib source, it is turned on for each electron sub source.	Medium	yes
electronSecondaryEmissionTrajFlag	int	0	None	plot secondary electron trajectory? 0=no, 1=yes	Advanced	yes
electronSecondaryTemperature	double	2	[eV]	secondary electron temperature (from electron impact)	Medium	yes
erosion	int	0	None	bits go by groups of 3 (bit0=on/off, bit1=eroded_products_dynamics/don t, bit2=unused), while groups of 3 bits are for ambient population 1, ambient population 2, source 1, source 2, source 3 and source 4 resp. Similarly to electronSecondaryEmission, bits go by groups of 3 : - bit 0: turn on erosion (if 1), - bit 1: simulate eroded products dynamics by PIC model (if 1), - bit 2 = unused Six groups of 3 bits are used, successively for: - ambient ion population 1	Medium	yes

				- ambient ion population 2 - source 1 - source 2 - source 3 - source 4 Ex: all on = 011011011011011011 =112347 (or 111111111111111111 = 2^{18}-1 = 262143) NB: when erosion is turned on for a multipleSurfDistrib source, it is turned on for each ion sub source.		
erosionProductDensification	double	1	[-]	densification coefficient for erosion product superparticles	Medium	yes
erosionProductDt	double	-1	[s]	Maximum integration time step for erosion products (see SPIS 5 User Manual)	Medium	yes
erosionProductDuration	double	0	[s]	Maximum integration duration for erosion products (see SPIS 5 User Manual)	Medium	yes
erosionProductSpeedUp	double	1	[-]	Numerical times speed-up factor erosion products	Expert	yes
erosionProductsTrajFlag	int	0	None	plot erosion products trajectory? 0=no, 1=yes	Medium	yes
inducedConductivity	int	1	None	if 0 no induced conductivity, if 1 induced conductivity turned on	Medium	yes
photoElectronDensification	double	1	[-]	densification coefficient for photo electron superparticles	Medium	yes
photoElectronTemperature	double	2	[eV]	photo-electron temperature	Medium	yes
photoElectronTrajFlag	int	0	None	plot photo electron trajectory? 0=no, 1=yes	Advanced	yes
photoEmission	int	3	None	- if 0, no photo-emission - if 1, photo-emission is turned on with the sun direction defined below (from sun vector (sunX...), no shading for now) - if 3, photo-emission is turned on with the sun direction defined below (from sun vector (sunX...)) and photo-electron dynamics is modelled (PIC) - if 5, photo-emission is turned on with a sun flux defined locally (local parameter sunFlux) - if 7, photo-emission is turned on with a sun flux defined locally (local parameter sunFlux)and photo-electron dynamics is modelled (PIC) NB: these values stem for a bit per bit definition: bit0 => on/off, bit1 => dynamics of photo-electrons is modelled / not, bit2 => sun flux locally defined / from sun direction (e.g. all on => binary 111 = decimal 7)	Low	yes
protonSecondaryDensification	double	1	[-]	densification coefficient for secondary electron superparticles (from proton impact)	Medium	yes
protonSecondaryEmission	int	3	None	if 0, no secondary emission, if 1, secondary emission turned on	Medium	yes
protonSecondaryTemperature	double	2	[eV]	secondary electron temperature (from proton impact)	Medium	yes
secondaryDt	double	1E-06	[s]	Maximum integration time step for all types of secondary electrons (see SPIS 5 User Manual)	Expert	yes
secondaryDuration	double	1E-06	[s]	Maximum integration duration for all types of secondary electrons (see SPIS 5 User Manual)	Expert	yes
secondarySpeedUp	double	1	[-]	Numerical times speed-up factor for all types of secondary electrons	Expert	yes
sunX	double	0	[-]	x-component of sun direction, see photoemission documentation	Low	yes
sunY	double	0	[-]	y-component of sun direction, see photoemission documentation	Low	yes

sunZ	double	1	[-]	z-component of sun direction, see photoemission documentation	Low	yes
surfaceConductivity	int	1	None	if 0 no surface conductivity, if 1 surface conductivity turned on	Medium	yes
volumeConductivity	int	1	None	if 0 no volume conductivity, if 1 volume conductivity turned on	Medium	yes

The following parameters are used to turn on (barrierCSFlag) or tune (the next ones, for experts only) the CurrentScaler used by the implicit circuit solver when a potential barrier shows up on top of a photo-emissive surface.
The regular user should simply turn on this CurrentScaler through barrierCSFlag when such a situation is expected (typically charging in GEO in sunlight), while the advance user may enter into the source code of the derived classes of CurrentScaler for a more detailed understanding of the consequences of the tuning parameters.

Name	Type	Default Value	Unit	Description	Expertise Level	In use
barrierCSFlag	int	0	[-]	flag for the current scaler specific to GEO potential barrier phenomenon for photo/secondary electron recollection (0 = off, 1 = on). To be turned on when potential barrier typical of GEO is expected.	Advanced	yes
bcsGlobalFactor	double	10	[-]	global temperature factor for the current scaler specific to GEO potential barrier phenomenon (BarrierCurrentScaler) for photo/secondary electron recollection (for Expert users only)	Expert	yes
bcsLocalFactor	double	1	[-]	local temperature factor for the current scaler specific to GEO potential barrier phenomenon (BarrierCurrentScaler) for photo/secondary electron recollection (for Expert users only)	Expert	yes
bcsRelValid	double	200	[-]	relative validity (relative to temp*bcsLocalFactor) for the current scaler specific to GEO potential barrier phenomenon (BarrierCurrentScaler) for photo/secondary electron recollection (for Expert users only)	Expert	yes
bcsSmoothdIdV	int	30	[-]	number of smoothing steps (each step => nearest elements) for dI/dV of recollected electrons when the barrier current scaler is on (for Expert users only)	Expert	yes
bcsSmoothI	int	0	[-]	number of smoothing steps (each step => nearest elements) for the collected intensity when the barrier current scaler is on (for Expert users only)	Expert	yes
bcsSmoothPot	int	10	[-]	number of smoothing steps (each step => nearest elements) for the potential when the barrier current scaler is on (for Expert users only)	Expert	yes

Generic surface interactions

Extra surface interactions can also be defined generically as a "plug in" class similarly ar for volume distributions, surface sources, etc.
Although only one of these classes was currently implemented (CathodeSpot), extra ones can easily be added as explained in .

The general rules for the interactorType* parameters, which define the model to be used (a class name), are:
- this class must derive from the class Interactor
- have a specific constructor including the UI-defined parameters as described in Surf\SurfInteract\Interactor.html
- in practice in SPIS v4.0 only CathodeSpot is available.

Name	Type	Default Value	Unit	Description	Expertise Level	In use
interactorDt1	double	-1	[s]	Maximum integration time step for particles from first interactor on SC (automatic if negative)	Expert	yes
interactorDuration1	double	0	[s]	Maximum integration duration for particles from first interactor on SC (automatic if 0)	Expert	yes
interactorFlag1	double	0	[-]	flag for defining a first generic interactor on the spacecraft: 0 => none, 1 => yes, x => number of super-particles densified by x if relevant	Expert	yes
interactorNb	int	0	None	number of interactors	Expert	yes
interactorParticleType1	String	O+	None	Type of particles emitted by the first interactor if it is an emitter	Expert	yes
interactorPopSource1	String	electrons1	None	volume population to be used as source of the interaction of this first interactor (must be one of the predefined volume population names ions1, elec1, source1, photoElec...)	Expert	yes
interactorSpeedUp1	double	1	[-]	Numerical times speed-up factor for particles from first artificial interactor on SC	Expert	yes
interactorType1	String	CathodeSpot	None	Name of the first Interactor class to be used for an interactor on the spacecraft	Expert	yes

User defined interactions

Since SPIS 5, it is possible to define distribution functions of secondary particles, see ***SPIS5 User Manual annex on "Advanced uses for scientific applications"***.
It leads to define new global parameters:

Back to top.

Volume interactions

These parameters allow to turn interactions on or off, define the type of interaction, the incoming and resulting populations and particle types.

The general rules for the volInteractType parameter, which defines the volume interaction type (class), are:

- this class must derive from the class VolInteractor
- have a specific constructor including the UI-defined parameters as described in "Writing UI-supported classes" page and in ..\API\public\spis\Vol\VolInteract\VolInteractor.html
- in practice as of today only CEXInteractor is implemented.

Name	Type	Default Value	Unit	Description	Expertise Level	In use
crossSectionVolInteract1	String	1.0e-18	[m2] or None	Cross section for 1st volume interaction, either a float (its value [m2]) or the name of the file describing sigma(E). Can be either: - a float: the value of the cross section σ [m^2] - a String: the name of the file (to be found in your project NumKernel/Input folder later) where the cross section versus energy is defined (two ASCII columns E[eV], σ [m2]) (see below for its format) The rule is the following: an attempt to traduce this String into a float, of which success depends the switching to first or second option	Advanced	yes
crossSectionVolInteract2	String	1.0e-18	[m2] or None	Cross section for 2nd volume interaction, either a float (its value [m2]) or the name of the file describing sigma(E)	Advanced	yes
inPart1VolInteract1	String	Xe+	None	Type of particles for first interacting population (a string that must be found in the particle types) for the 1st volume interaction. NB: may be redundant with the definition of the interacting population, but has to be defined anyway.	Advanced	yes
inPart1VolInteract2	String	Xe+	None	Type of particles for first interacting population of 2nd volume reaction	Advanced	yes
inPart2VolInteract1	String	Xe	None	Type of particles for second interacting population (a string that must be found in the particle types) in 1st volume interaction	Advanced	yes
inPart2VolInteract2	String	Xe	None	Type of particles for second interacting population of 2nd volume reaction	Advanced	yes
inPop1VolInteract1	String	source1	None	Defines first interacting population (ions for CEX) of first volume interaction. Must be one of: - source1, source2, source3, source4 (indeed the PICVolDistrib alimented by source_x), or sourceX.Y - ions1, ions2, elec1, elec2 - photoElec, secondElec, secondElecUnderProton to select respectively a population issued from an artificial source, the ambient plasma, or a surface interaction.	Advanced	yes
inPop1VolInteract2	String	source1	None	Defines first interacting population of 2nd volume reaction (e.g. ions for CEX), must be one of ions1, ions2, elec1, elec2, source1... source4, photoElec, secondElec	Advanced	yes

inPop2VolInteract1	String	fractionOfFirstPopSource	None	Defines second interacting population (neutrals for CEX) for 1st volume interaction. Must be one of: - source1, source2, source3, source4 (or sourceX.Y, etc.) - ions1, ions2, elec1, elec2 - photoElec, secondElec, secondElecUnderProton fractionOfFirstPopSource, which is special feature for CEXInteractor applied to EP thrusters plume: the first population must be defined from an artificial source (inPop1VolInteract = source1, source2, sourceX.Y...) and this second population (of neutrals) will be emitted on the same SC surfaces, with a flux reduced with respect to the first one by a factor = parameter1VolInteract, and a uniform temperature equal to parameter2VolInteract [eV]. So, e.g. for a thruster of ionisation efficiency of 97% and neutrals emitted at 1160K, simply define parameter1VolInteract = 0.03 and parameter1VolInteract = 0.1 [eV].	Advanced	yes
inPop2VolInteract2	String	fractionOfFirstPopSource	None	Defines second interacting population of 2nd volume reaction(e.g. neutrals for CEX), must be one of ions1, ions2, elec1, elec2, source1... source4, photoElec, secondElec	Advanced	yes
outPart1VolInteract1	String	Xe+	None	Type of particles for first population produced in 1st volume interaction	Advanced	yes
outPart1VolInteract2	String	Xe+	None	Type of particles for first population produced in 2nd volume interaction	Advanced	yes
outPart2VolInteract1	String	Xe	None	Type of particles for second population produced in 1st volume interaction. NB: not used in the current version of CEXInteractor, but might be later ("fast" neutrals)	Advanced	yes
outPart2VolInteract2	String	Xe	None	Type of particles for second population produced in 2nd volume interaction	Advanced	yes
outPop1DtVolInteract1	double	-1	[s]	Maximum integration time step for first population produced in 1st volume interaction (automatic if negative)	Advanced	yes
outPop1DtVolInteract2	double	-1	[s]	Maximum integration time step for first population produced in 2nd volume interaction (automatic if negative)	Advanced	yes
outPop1DurationVolInteract1	double	0	[s]	Maximum integration duration for first population produced in 1st volume interaction (automatic if 0)	Advanced	yes
outPop1DurationVolInteract2	double	0	[s]	Maximum integration duration for first population produced in 2nd volume interaction (automatic if 0)	Advanced	yes
outPop1SpeedUpVolInteract1	double	1	[-]	Numerical times speed-up factor for first population produced in 1st volume interaction	Advanced	yes
outPop1SpeedUpVolInteract2	double	1	[-]	Numerical times speed-up factor for first population produced in 2nd volume interaction	Advanced	yes
outPop1VolInteractTrajFlag	int	0	None	plot 1st produced population trajectory? 0=no, 1=yes	Advanced	yes

outPop2DtVolInteract1	double	-1	[s]	Maximum integration time step for first population produced in 1st volume interaction (automatic if negative)	Advanced	yes
outPop2DtVolInteract2	double	-1	[s]	Maximum integration time step for first population produced in 2nd volume interaction (automatic if negative)	Advanced	yes
outPop2DurationVolInteract1	double	0	[s]	Maximum integration duration for 2nd population produced in 1st volume interaction (automatic if 0)	Advanced	yes
outPop2DurationVolInteract2	double	0	[s]	Maximum integration duration for 2nd population produced in 2nd volume interaction (automatic if 0)	Advanced	yes
outPop2SpeedUpVolInteract1	double	1	[-]	Numerical times speed-up factor for first population produced in 1st volume interaction	Advanced	yes
outPop2SpeedUpVolInteract2	double	1	[-]	Numerical times speed-up factor for first population produced in 2nd volume interaction	Advanced	yes
outPop2VolInteractTrajFlag	int	0	None	plot 2nd produced population trajectory? 0=no, 1=yes	Advanced	yes
parameter1VolInteract1	double	0.05	[variable]	1st parameter of 1st volume interactor: - for CEX : if parameter3VolInteract=0, it is the ratio between neutral and ion fluxes at source surfaces.	Advanced	yes
parameter1VolInteract2	double	0.05	[variable]	1st parameter of 2nd volume interactor	Advanced	yes
parameter2VolInteract1	double	0.1	[variable]	2nd parameter of 1st volume interactor: - for CEX : temperature of neutrals	Advanced	yes
parameter2VolInteract2	double	0.1	[variable]	2nd parameter of 2nd volume interactor	Advanced	yes
parameter3VolInteract1	double	0	[variable]	3rd parameter of 1st volume interactor: - for CEX: flag to turn on the lambertian distribution (0) or constant neutral density (1)	Advanced	yes
parameter3VolInteract2	double	0	[variable]	3rd parameter of 2nd volume interactor	Advanced	yes
parameter4VolInteract1	double	0	[variable]	4th parameter of 1st volume interactor: - for CEX: if parameter3VolInteract=1, it is the pressure in default unit (kg/m/s2)	Advanced	yes
parameter4VolInteract2	double	0	[variable]	4th parameter of 2nd volume interactor	Advanced	yes
volInteract1	double	0	None	Flag to turn on 1st volume interaction : 0 => off, 1 => on, x>0 => on, superparticles densified by x	Advanced	yes
volInteract2	double	0	None	Flag to turn on 2nd volume interaction: 0 => off, 1 => on, x>0 => on, superparticles densified by x	Advanced	yes
volInteractNb	int	2	None	Number of volume interactors : not to be modified in UI, but only in defaultGlobalParam.py if the number of sources is modified in defaultGlobalParam.py	Advanced	yes
volInteractType1	String	CEXInteractor	None	Type of 1st volume interaction, UI-buildable class name derived from VolInteract. For now only CEXInteractor is supported (cf also CEX model documention). That choice has the following consequences: - inPop1VolInteract are the ions and must be a PICVolDistrib - inPop2VolInteract are the neutrals and can only be generated	Advanced	yes

				from an artificial surface source defined by a LocalMaxwellSurfDistrib		
volInteractType2	String	CEXInteractor	None	Type of 2nd volume interaction, UI-buildable class name derived from VolInteract	Advanced	yes

Example of cross section definition file:

```
E (eV)   cross section (m2)
 0.0        2.0e-18
 100.0      1.2e-18
 300.0      1.0e-18
 1000.0      .9e-18
```

The syntax is:
- first line = header (unread)
- next lines: energy in eV and cross section in square meters (separator = space)

Remark: it is still possible to use the single volume interaction version by setting the parameters volInteract to parameter4VolInteract (without numbering as in volInteract1). In that case, it is not possible to define multiple volume reactions. It may be the case when using projects built with a version older than SPIS4.3.

Back to top.

Outputs

These parameters mostly the periodicity for storing data for postprocessing (these data are then returned to UI and can be plotted).
Other parameters tune the detail level for screen printing (or verbosity level).

Name	Type	Default Value	Unit	Description	Expertise Level	In use
cumulateBetweenSteps	int	1	None	cumulate currents and densities between monitoring steps for improved statistics (0=no, 1=yes(improved only), 2=both)?	Advanced	yes
currentLogPlotCutoff	double	1E-12	[A/m2]	cutoff for current log plots	Advanced	yes
currentLogPlotFlag	int	2	None	plot log10 of currents? 0=no, 1=yes(log only), 2=both	Advanced	yes
currentMapMonitorStep	double	-10	[s]	time step for current density vectors monitoring (0 => none, -n => n times)	Low	yes
densitiesMapsMonitorStep	double	-10	[s]	time step for densities monitoring (0 => none, -n => n times)	Low	yes
densityChargeState	int	4	None	control of output density type, either amu/m3 or #/m3, 1=amu, 2=#, 4=automatic	Advanced	yes

				(from known particle type)		
densityLogPlotCutoff	double	0.001	[ecu/m3]	cutoff for density log plots	Advanced	yes
densityLogPlotFlag	int	2	None	plot log10 of densities? 0=no, 1=yes(log only), 2=both	Advanced	yes
energyMapMonitorStep	double	-10	[s]	time step for kinetic energy monitoring (0 => none, -n => n times)	Low	yes
exportAllDataFields	String	None	None	Select the export mode for all data fields (None=no export, ASCII=ASCII multi-files)	Advanced	yes
exportDensity	String	None	None	Select the export mode for density data fields (None=no export, ASCII=ASCII multi-files)	Advanced	yes
exportPotential	String	None	None	Select the export mode for potential data fields (None=no export, ASCII=ASCII multi-files)	Advanced	yes
finalCumulation	int	2	None	cumulate currents and densities at the end of simulation ? 0=no, 1or2=yes	Low	yes
finalCumulationStartTime	double	0.8	[s] or [-]	if finalCumulation=1 starting time for final dens-current cumulation, if finalCumulation=2 fraction of the simulation at which cumulation starts	Low	yes
fluxChargeState	int	4	None	control of output collected fluxes type, either C/m2/s = A/m2 (i.e. a current) or #/m2/s (i.e. a flux), 1=currents, 2=fluxes, 4=automatic (from known particle type)	Advanced	yes
materialPropertyPlots	int	1	None	flag for plotting material properties: 0=no, 1=yes	Low	yes
numericsMapsMonitorStep	double	-10	[s]	time step for numerical behaviour monitoring through 3D maps of superparticle numbers, one in #/element and one in #/node (0.0 => none, -n => n times)	Low	yes
numericsMonitorStep	double	-100	[s]	time step for numerical behaviour monitoring (0.0 => none, -n => n times)	Low	yes
particleTrajectoriesNb	int	0	None	number of particle trajectories per PIC population	Advanced	yes
particleTrajectoriesPeriod	int	1000	None	Probability to folLow a particle trajectory = one over particleTrajectoriesPeriod	Advanced	yes
plasmaElecFieldMapMonitorStep	double	-10	[s]	time step for plasma electric field monitoring (0 => none, -n => n times)	Low	yes
plasmaPotMapMonitorStep	double	-10	[s]	time step for plasma potential monitoring (0 => none, -n => n times)	Low	yes
poissonVerbose	int	3	None	Same as verbose, but specific to Poisson solver	Advanced	yes
scCurrentMapMonitorStep	double	-10	[s]	time step for spacecraft local currents monitoring (0 => none, -n => n times)	Low	yes
scElecFieldMapMonitorStep	double	-10	[s]	time step for spacecraft electric field monitoring (0 => none, -n => n times)	Low	yes
scPotMapMonitorStep	double	-10	[s]	time step for spacecraft local potential monitoring (0 => none, -n => n times)	Low	yes
scPotMonitorStep	double	-100	[s]	time step for spacecraft ground potential monitoring (0 => none, -n => n times)	Low	yes
taskDurationVerbose	int	3	None	Same as verbose, but specific to CPU monitoring	Advanced	since SPIS5
verbose	int	3	None	Verbosity level (level of screen messages about code execution): 0 = no print at all 1 = prints errors and warnings only 2 = 1 + minimal information 3 = 1 + more information (remains yet readable) 4 = even more information … (next levels for debugging)	Advanced	yes

Back to top.

Scenario

The default Scenario is Scenario, which is transparent (no real scenario, everything in and out is simply transferred from/to the top level Scenario to/from the regular Simulation).

Generic "plug-in" scenarios can be implemented and easily integrated. The general rules for the scenario parameter, which defines the Scenario type (class), are:

- this class must derive from the class Scenario
- have a specific constructor including the UI-defined parameters as described in "Writing UI-supported classes" page and in Scenario

In practice as of today the only non trivial scenario implemented is PotentialSweep. It consists in chaining successive simulations for different surface potentials. Results are extracted in the form of current-voltage (IV) characteristics for the populations, nodes, types of current selected.
When PotentialSweep is in use, the scenario parameters are understood the following way:

Name	Type	Default Value	Unit	Description	Expertise Level	In use
scenarioParameter1	int	0	[-]	If PotentialSweep: Number of steps of the potential sweep	Advanced	yes
scenarioParameter10	int	0	[-]	If PotentialSweep: Maximal Id node number	Advanced	yes
scenarioParameter11	int	0	[-]	If PotentialSweep: Flag for type of current monitored (0=all, 1=collected, 2=emitted)	Advanced	yes
scenarioParameter12	int	0	[-]	If PotentialSweep: Number of nodes whose potential changes. For each node, the potential sweep is linear between the initial and final voltages.	Advanced	yes
scenarioParameter13	int	0	[-]	If PotentialSweep: Id of 1st node with pot change	Advanced	yes
scenarioParameter14	double	0	[V]	If PotentialSweep: Initial potential of 1st node	Advanced	yes
scenarioParameter15	double	1	[V]	If PotentialSweep: Final potential of 1st node	Advanced	yes
scenarioParameter16	int	0	[-]	If PotentialSweep: Id of 2nd node with pot change	Advanced	yes
scenarioParameter17	double	0	[V]	If PotentialSweep: Initial potential of 2nd node	Advanced	yes
scenarioParameter18	double	1	[V]	If PotentialSweep: Final potential of 2nd node	Advanced	yes
scenarioParameter19	int	0	[-]	If PotentialSweep: Id of 3rd node with pot change	Advanced	yes
scenarioParameter2	double	0	[V]	Initial voltage (only used for monitoring) NB: this voltage and the Final voltage of scenarioParameter3 define the table of potentials used in the results files (and not the potential of each node)	Advanced	yes
scenarioParameter20	double	0	[V]	If PotentialSweep: Initial potential of 3rd node	Advanced	yes

scenarioParameter21	double	1	[V]	If PotentialSweep: Final potential of 3rd node	Advanced	yes
scenarioParameter22	int	0	[-]	If PotentialSweep: Id of 4th node with pot change	Advanced	yes
scenarioParameter23	double	0	[V]	If PotentialSweep: Initial potential of 4th node	Advanced	yes
scenarioParameter24	double	1	[V]	If PotentialSweep: Final potential of 4th node	Advanced	yes
scenarioParameter25	int	0	[-]	If PotentialSweep: Id of 5th node with pot change	Advanced	yes
scenarioParameter26	double	0	[V]	If PotentialSweep: Initial potential of 5th node	Advanced	yes
scenarioParameter27	double	1	[V]	If PotentialSweep: Final potential of 5th node	Advanced	yes
scenarioParameter28	int	0	[-]	If PotentialSweep: Id of 6th node with pot change	Advanced	yes
scenarioParameter29	double	0	[V]	If PotentialSweep: Initial potential of 6th node	Advanced	yes
scenarioParameter3	double	1	[V]	If PotentialSweep: Final voltage (used for monitoring, not to define node potentials)	Advanced	yes
scenarioParameter30	double	1	[V]	If PotentialSweep: Final potential of 6th node	Advanced	yes
scenarioParameter4	double	2E-06	[s]	Duration of first I-V step NB: if PotentialSweep is in use, the parameter duration is used for monitoring concerns only.	Advanced	yes
scenarioParameter5	double	1E-06	[s]	If PotentialSweep: Duration of other steps	Advanced	yes
scenarioParameter6	double	0.5	[-]	If PotentialSweep: Fraction of step duration used for IV sweeps results	Advanced	yes
scenarioParameter7	int	-1	[-]	Flag for populations monitored. Populations taken into account for I-V sweeps: bits for populations to be taken into account. 9 bits are used for successive: - all populations - elec1 - elec2 - ions1 - ions2 - source1 - source2 - source3 - source4 ex: binary 000100011 = decimal 35 enables I- V sweep for all populations, elec1 and source1. ex: decimal -1 makes IV sweeps for each population	Advanced	yes
scenarioParameter8	int	-1	[-]	If PotentialSweep: Flag for nodes monitored (-1: all nodes; else: minimum and maximum nodes Id to be defined)	Advanced	yes
scenarioParameter9	int	0	[-]	If PotentialSweep: Minimal Id node number	Advanced	yes

Back to top.

Transitions

The Scenario class was made somewhat obsolete by SPIS 5. We recommend to use *Transitions* instead defined because they are more flexible and defined by much less parameters and by ASCII tables, see **SPIS 5 User Manual annex on "Advanced uses for scientific applications"**.

Name	Type	Default Value	Unit	Description	Expertise Level	In use
transitionNb	int	0	None	number of transitions	Medium	yes
transitionFlag1	double	0	None	flag for activating transition 1 (sun flux change) on the simulation configuration: 0 => none, 1.0 => yes	Low	yes
transitionFlag2	double	0	None	flag for activating transition 2 (conductivity change) on the simulation configuration: 0 => none, 1.0 => yes	Low	yes
transitionType1	String	BasicEclipseExit	None	Name of the Transition class to be used for transition 1 on the simulation	Advanced	yes
transitionType2	String	ConductivityEvolution	None	Name of the Transition class to be used for transition 2 on the simulation	Advanced	yes
transitionDt1	double	0.01	[s]	maximal time step when the transition 1 evolves	Medium	yes
transitionDt2	double	0.01	[s]	maximal time step when the transition 2 evolves	Medium	yes

Local parameters

These local parameters are scalar fields living either in the volume or on a surface (spacecraft or external boundary). Not all of them are used in the present version of the code. Some come in addition to global parameters that they override when some flag declares that a property is to be considered as local (e.g. turning on an interaction only locally).

They can be defined through the group editor. It allows to define them group by group (a uniform value on each group).
See the SPIS5 User Manual for practical usage of the group editor.

The local fields are described now. They are grouped somewhat arbitrarily as:

- Electrical node model
- External Boundary Electric field Boundary Condition
- External Boundary Plasma population Boundary Condition
- Spacecraft Conductivity model
- Spacecraft electric field Boundary Condition
- Spacecraft Macroscopic characteristics
- Spacecraft Material model
- Spacecraft Plasma Population Boundary Condition
- Spacecraft sources and interactors

- Spacecraft thin elements
- Volume plasma model

NB: Only the properties that may be modified by the users are in **bold** here. You will find a few others which should not be modified indeed.

Name	Description	Live on (spacecraft, external boundary or volume)	Sub-group of properties	Centring, or localisation (0=node, 1=edge, 2=surface, 3=volume)	Unit	Default value	Comment (in use or not)
ElecNodeId	The electric (super) node this element is related to (SC ground, array ground…)	SC	Electrical node model	2	[-]	0	Yes
EdgeElecNodeId	The (macro) electric node edges are related to (SC ground, array ground...)	SC	Electrical node model	1	[-]	0	Yes
BdDiriFlag	If 1, Dirichlet condition for Poisson equation on external boundary (fixed potential)	Boundary	Ext. Bound. Electric field BC	0	[-]	0	Yes,Since SPIS V4.0 (used to be Fourier only)
BdDiriPot	The potential to be used for Dirichlet condition	Boundary	Ext. Bound. Electric field BC	0	[V]	0	Yes,Since SPIS V4.0
BdFourFlag	If 1, Fourier condition for Poisson equation on external boundary	Boundary	Ext. Bound. Electric field BC	2	[-]	1	Yes,Since SPIS V4.0
BdFourAlpha	Alpha parameter in Fourier condition: alpha pot + d(pot)/dn = value (used if poissonBCType = 0)	Boundary	Ext. Bound. Electric field BC	2	[m-1]	0	Yes
BdFourValue	Right hand side parameter in Fourier condition: alpha pot + d(pot)/dn = value (used if poissonBCType = 0)	Boundary	Ext. Bound. Electric field BC	0	[V/m]	0	Yes
IncomPart	If 0, no particle are injected. If 1, particles are injected (following the defined environment)	Boundary	Ext. Bound. Plasma population BC	2	[-]	1	Yes (since SPIS V4.2)
OutgoPart	If 0, outgoing particles are lost (sink) If 1, they bounce specularly (extra options possible)	Boundary	Ext. Bound. Plasma population BC	2	[-]	0	Yes (since SPIS V4.2)
VolConduct	If 1, volume conductivity through the bulk material is turned on locally	SC	S/C Conductivity model	2	[-]	0	No (global flag only is under use)

IndConduct	If 1, induced volume conductivity is turned on locally and simulated (if 0, the raw volume conductivity above is used)	SC	S/C Conductivity model	2	[-]	0	No (global flag only is under use)
SurfConduct	If 1, surface conductivity is turned on locally and simulated (from the top of a cell to the next ones)	SC	S/C Conductivity model	2	[-]	0	No (global flag only is under use)
SCDiriFlag	If 1, Dirichlet condition for Poisson equation on SC (fixed potential)	SC	S/C electric field BC	0	[-]	1	No: set to 1, always Dirichlet on SC
SCDiriPot	The potential to be used for Dirichlet condition	SC	S/C electric field BC	0	[V]	0	Yes
SCDiriPotSurf	The potential to be used for Dirichlet condition, the one used for physical SC surfaces	SC	S/C electric field BC	2	[V]	0	Yes
SCDiriPotEdge	The potential to be used for Dirichlet condition, the one used for physical thin wires	SC	S/C electric field BC	1	[V]	0	Yes
SCFourFlag	If 1, Fourier condition for Poisson equation on SC: alpha pot + d(pot)/dn = value	SC	S/C electric field BC	2	[-]	0	No: set to 0, always Dirichlet on SC
SCFourApha	Alpha parameter in Fourier condition: alpha pot + d(pot)/dn = value	SC	S/C electric field BC	2	[m-1]	0	No: always Dirichlet on SC
SCFourValue	Right hand side parameter in Fourier condition : alpha pot + d(pot)/dn = value NB: note the centring different from other Fourier parameters	SC	S/C electric field BC	0	[V/m]	0	No: always Dirichlet on SC
MatThickness	Material thickness (overrided by the material thickness defined in the generic material property set, if negative)	SC	S/C Macroscopic characteristics	2	[m]	-1	Yes **Warning : risk of confusion with DTM material property (not used)**
Temperature	Surface temperature	SC	S/C Macroscopic characteristics	2	[K]	300	No (needed by no interaction for now)
MatModelId	Id of the material model used for this material	SC	S/C Material model	2	[-]	0	Yes
MatTypeId	Id of this material in its material model	SC	S/C Material model	2	[-]	0	Yes
PhotoEmis	If 1, photo emission is turned on locally and simulated	SC	S/C Plasma Population BC	2	[-]	0	No (global flag only is under use)

ElecSecEmis	If 1, secondary electron emission under electron impact is turned on locally and simulated	SC	S/C Plasma Population BC	2	[-]	0	No (global flag only is under use)
ProtSecEmis	If 1, secondary electron emission under proton impact is turned on locally and simulated	SC	S/C Plasma Population BC	2	[-]	0	No (global flag only is under use)
SunFlux	Sun flux on spacecraft, normalised to sun flux at 1 AU (only used when global parameter photoEmission = 5 or 7)	SC	S/C Plasma Population BC	2	[-]	0	Yes
SourceId	Id of the local artificial plasma source defined on the spacecraft: between 1 and sourceNb (=4) to have source 1 to 4 emitting at this place. Value 0 or negative => no source. NB: Possible to have multiple sources at single place, using this id as the mother source id.	SC	S/C sources and interactors	2	[-]	0	Yes
SourceCurrent	Emitted current, or flux, of an artificial source defined on the spacecraft. NB: As each local parameter, SourceCurrent has only one unit during a single simulation. The first defined spacecraft source imposes the unit of currents to other ones. It is preferable to set the same units for all sources.	SC	S/C sources and interactors	2	[depends]	0	Yes (flux units, [#/m2/s], or[kg/m2/s], only since SPIS V4.3)
SourceTemp	Temperature of the emitted Maxwellian distribution, if sourceType of the corresponding sourceId (hence sourceType1 where sourceId = 1, or sourceType2 where sourceId = 2, etc.) is a LocalMaxwellSurfDistrib (if not, the interpretation of this value can be different, see each class description)	SC	S/C sources and interactors	2	[eV]	0	Yes
SourceMach	Source Mach number (0 => Lambertian). The exact definition of this parameter can be different depending on the surface distribution using it (see e.g. MaxwellianThruster, while LocalMaxwellSurfDistrib does not use it and only generates local Lambertian distributions)	SC	S/C sources and interactors	2	[-]	0	Yes
SurfThicknessS	Thickness of 2D surfaces (used when surfaces are tagged as thin, their thickness not meshed)	SC	S/C thin elements	2	[m]	0	Yes
EdgeRadiusS	Radius of 1D elements (used when edges are tagged as physical thin wires, their thickness not meshed)	SC	S/C thin elements	1	[m]	0	Yes
MatModelIdOnWire	Id of the material model used for this material on a wire element (0: default model = NASCAP-properties-based)	SC	S/C thin elements	1	[-]	0	Yes (since SPIS 5)

MatTypeIdOnWire	Id of this material in its material model on a wire element (-1 : no coating: bare metal, no interaction (except collection))	SC	S/C thin elements	1	[-]	-1	Yes (since SPIS 5)
PhotoEmisOnWire	If 1, photo emission is turned on and simulated on a wire element	SC	S/C thin elements	1	[-]	0	No (global flag only is under use)
ElecSecEmisOnWire	If 1, secondary electron emission under electron impact is turned on and simulated on a wire element	SC	S/C thin elements	1	[-]	0	No (global flag only is under use)
ProtSecEmisOnWire	If 1, secondary electron emission under proton impact is turned on and simulated on a wire element	SC	S/C thin elements	1	[-]	0	No (global flag only is under use)
SunFluxOnWire	Sun flux on spacecraft on a wire element [sun at 1 AU] (compute it from sun direction, possibly including shades)	SC	S/C thin elements		[-]	-1	Yes (since SPIS 5)
SourceIdOnWire	Id/type of an artificial plasma source defined on the spacecraft on a wire element (e.g. thruster or ion gun) (-1 : no source)	SC	S/C thin elements	1	[-]	-1	Yes (since SPIS 5)
SourceCurrentOnWire	Current of an artificial source defined on the spacecraft on a wire element (NB: for some sources the unit can be different, e.g. the particle number, or the total current)	SC	S/C thin elements	1	[A/m2]	0	Yes (since SPIS 5)
SourceTempOnWire	Temperature of the emitted Maxwellian distribution on a wire element	SC	S/C thin elements	1	[eV]	1	Yes (since SPIS 5)
SourceMachOnWire	Source Mach number on a wire element (0 => Lambertian) [-]	SC	S/C thin elements	1	[-]	0	Yes (since SPIS 5)
InstrumentSupport	Localization index for instruments localized on the spacecraft	SC	SC Instruments	2	[-]	0	Yes
VolInteracFlag	If 1, volume interaction is computed locally in that region (typically charge exchange)	Volume	Volume plasma model	3	[-]	0	No (global flag only is under use)
BackgroundDens	Fixed background density used to compute volume interaction (typically: neutral density)	Volume	Volume plasma model	3	[m-3]		No

Back to top.