PICLas Documentation

Institute for Aerodynamics and Gas Dynamics (IAG)
Institute for Space Systems (IRS)
University of Stuttgart, Germany

January 22, 2019

Contents

1 Introduction

PICLas is a three-dimensional simulation framework for Particle-in-Cell and Direct Simulation Monte Carlo methods that can be coupled for the simulation of collisional plasma flows. It features a high-order discontinuous Galerkin (DG) simulation module for the solution of the time-dependent Maxwell equations on unstructured hexahedral elements in three space dimensions. The code was specifically designed for very high order accurate simulations on massively parallel systems. It is licensed under GPLv3, written in Fortran and parallelized with MPI. Implemented features are

- Coupled Particle-in-Cell with Direct Simulation Monte Carlo methods
- Arbitrary order nodal polynomial tensor product basis using Gauss or Gauss Lobatto collocation points for electrostatic and electromagnetic solvers
- Matching high order curved mesh generation from external mesh formats (CGNS, GMSH) or simple analytic blocks via the open source preprocessor **HOPR**
- Nonconforming interfaces based on the mortar approach (electromagnetic solver)
- Non-reflecting boundary conditions via CFS-PMLs (electromagnetic solver)
- Automatic domain decomposition for parallel simulations based on a space filling curve
- High order low-storage explicit Runge-Kutta time integration
- I/O using the HDF5 library optimized for massively parallel jobs

1.1 How this documentation is organized

This user guide is organized to both guide the first steps as well as provide a complete overview of the simulation code's features from a user and a developer point of view.

- Chapter 2 contains step by step instructions from obtaining the source code up to running a first simulation and visualizing the simulation results. In addition, it provides an overview of the whole simulation framework and the currently implemented features.
- Chapter 3 is meant as a complete user guide with a detailed description how to use and apply the features of **PICLas** from a user's point of view. This includes setting up solver settings, initial and boundary conditions, the mesh interface, parallel execution and the currently available post processing capabilities.
- Chapter 5 lists all tools contained in the **PICLas** repository, including **POSTI** post-processing tools.
- Simulation tutorials are contained in Chapter 6.

- The unit test system used to test key routines with CTest is described in Chapter 7.

2 Getting started

2.1 Installation

2.1.1 Prerequisites

PICLas has been tested for various Linux distributions. This includes Ubuntu 14.04 LTS, 16.04 LTS and 18.04 LTS, OpenSUSE 42.1 and CentOS 7. The suggested packages in this section can of course be replaced by self compiled versions.

The required packages for the Ubuntu Linux distributions are listed in table 2.1. Under Ubuntu, they can be obtained using the apt environment:

```
sudo apt-get install git
```

Table 2.1: Debian/Ubuntu packages. x: required, o: optional, -: not available

Package	Ubuntu 14.04	Ubuntu 16.04	Ubuntu 18.04
git	x	x	x
cmake	x	x	x
cmake-curses-gui	o	o	o
liblapack3	x	x	x
liblapack-dev	x	x	x
gfortran	x	x	x
g++	x	x	x
mpi-default-dev	x	x	x
zlib1g-dev	-	x	x
exuberant-ctags	o	o	o

The required packages for OpenSUSE and CentOS are listed in table 2.2.

Under OpenSUSE, packages are installed by the following command.

```
sudo zypper install git
```

The `PATH` variable must be extended by the openmpi path

```
export PATH=$PATH:/usr/lib64/mpi/gcc/openmpi/bin
```

Under CentOS, packages are installed by the following command.

```
sudo yum install git
```

Additionally, the PATH variable must be extended by the openmpi path

```
export PATH=$PATH:/usr/lib64/openmpi/bin
```

Table 2.2: OpenSUSE/CentOS packages. x: required, o: optional, -: not available

Package	OpenSUSE 42.1	CentOS 7
git	x	x
cmake	x	x
lapack-devel	x	x
openmpi	x	x
openmpi-devel	x	x
zlib-devel	x	x
gcc-fortran	x	x
gcc	x	-
gcc-c++	x	x
ctags-etags	-	o

On some systems it may be necessary to increase the size of the stack (part of the memory used to store information about active subroutines) in order to execute **PICLas** correctly. This is done using the command

```
ulimit -s unlimited
```

from the command line. For convenience, you can add this line to your .bashrc.

2.1.2 Obtaining the source

The **PICLas** repository is available at GitHub. To obtain the most recent version you have two possibilities:

- Clone the **PICLas** repository from Github

  ```
  git clone https://github.com/piclas-framework/piclas.git
  ```

- Download **PICLas** from Github:

  ```
  wget https://github.com/piclas-framework/piclas/archive/
   master.tar.gz
  tar xzf master.tar.gz
  ```

Note that cloning **PICLas** from GitHub may not be possible on some machines, as e.g. the HLRS at the University of Stuttgart restricts internet access. Please refer to section 8.1 of this user guide.

2.1.3 Compiling the code

- Open a terminal
- Change into the **PICLas** directory
- Create a new subdirectory and use CMake to configure and compile the code

```
mkdir build; cd build
cmake ../
make
```

The executables **PICLas** and **posti_visu** are contained in your **PICLas** directory in `build/bin/`.

Custom configuration of compiler options may be done using

```
ccmake ../
```

For a list of all compiler options see Section 3.2.

2.1.3.1 Directory paths

In the following, we write `$PICLASROOT` as a substitute for the path to the **PICLas** repository. Please replace `$PICLASROOT` in all following commands with the path to your **PICLas** repository *or* add an environment variable `$PICLASROOT`.

Furthermore, the path to executables is omitted in the following, so for example, we write `piclas` instead of `$PICLASROOT/build/bin/piclas`.

Here is some explanation for Linux beginners:

In order to execute a file, you have to enter the full path to it in the terminal. There are two different ways to enable typing `piclas` instead of the whole path (do not use both at the same time!)

1. You can add an alias for the path to your executable. Add a command of the form

```
alias piclas='$PICLASROOT/build/bin/piclas'
```

to the bottom of the file `~/.bashrc`. Source your `~/.bashrc` afterwards with

```
. ~/.bashrc
```

2. You can add the **PICLas** binary directory to your `$PATH` environment variable by adding

```
export PATH=$PATH:$PICLASROOT/build/bin
```

to the bottom of the file `~/.bashrc` and sourcing your `~/.bashrc` afterwards.

2.1.4 Running the code

For a first minimal **PICLas** run, do the following:

- Open a terminal
- Navigate to a directory, in this case *temp*

```
cd temp
```

- Copy the *cavity* tutorial folder

```
cp -r $PICLASROOT/tutorials/cavity/Basic_Re100 .
cd Basic_Re100
```

- Run piclas

```
piclas parameter_piclas.ini
```

- Convert the output files to the vtu format

```
posti_visu cavity_State_0000000.200000000.h5
```

- Visualize using e.g. ParaView.

2.2 Basic Usage

For a basic overview of the framework and the single components of the flow solver a flowchart is given in Figure 2.1.

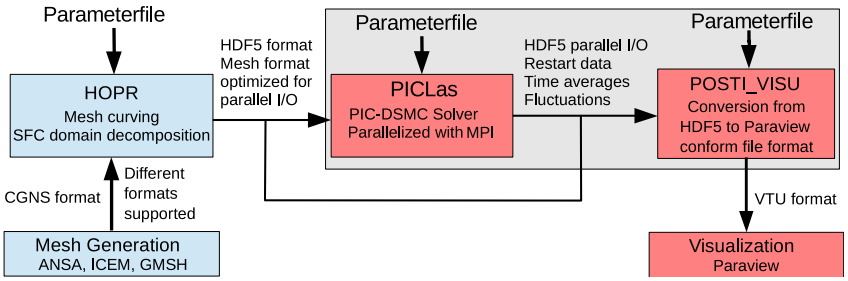

Figure 2.1: Flowchart: Basic modules and files used by **PICLas**

HOPR

A standalone high-order preprocessor **HOPR** has been developed to generate high-order meshes from input data from external linear mesh generators. Different file formats are supported. HOPR has been recently made open source under the GPLv3 license. It generates a **PICLas** conform mesh format in HDF5 for efficient parallel initialization. For a complete overview of HOPR, see https://www.hopr-project.org.

HOPR can be compiled in the same way as **PICLas** (see section 2.1.3). The basic command to run HOPR is

```
hopr parameter.ini
```

Note that the path to the **HOPR** executable is omitted in the command (see 2.1.3).

PICLas

PICLas, a high order DGSEM based CFD solver, is the core module in the tool chain. Generally **PICLas** requires two main files as input, a mesh file in HDF5 format generated by **HOPR** and a parameter file where the main settings for the CFD simulation are set. The results files generated by **PICLas** are also HDF5 files.

The basic command to run **PICLas** is

```
mpirun -np [no. processors] piclas parameter.ini
```

Note: Adding `mpirun -np [no. processors]` before the **PICLas** executable starts **PICLas** in parallel with the specified number of threads. If it is omitted, **PICLas** is run on one processor without MPI. This also applies to all other tools mentioned below.

Parameter file

The `parameter.ini` file contains the main settings for the CFD simulation. The parameter file defines e.g.

- CFL (Courant-Friedrichs-Lewy) number
- polynomial degree,
- simulation end time and dump/analyze intervals
- boundary conditions
- initial and boundary states

A complete list of all runtime options that can be set in the parameter file can be obtained with the command

```
piclas --help
```

It is also supplied in Section 9. The `--help` option also works for most other **PICLas** tools.

posti_visu tool

To visualize the results e.g. with ParaView, a converter tool is provided. The posti_visu tool takes the HDF5 files generated by **PICLas**.

The basic command to run the posti_visu tool is

```
mpirun -np [no. processors] posti_visu parameter.ini [
   piclas_outputfile.h5]
```

In this case a parameter file is specified in which options like the type and amount of the visualization nodes and mesh options are defined - see Section 3.10 for all available options. You can also omit the parameter file argument:

```
mpirun -np [no. processors] posti_visu [piclas_outputfile.h5]
```

This runs posti_visu using only standard options, i.e.

- equidistant visualization nodes
- amount of visualizaion nodes equals number of collocation points per element
- allowing for curved meshes
- visualizing the conservative variables

HDF5

HDF5 is a data model, library, and file format for storing and managing data. It supports an unlimited variety of datatypes, and is designed for flexible and efficient I/O and for high volume and complex data. For further information and to download the software, visit the HDF5 website at https://www.hdfgroup.org.

2.3 Feature list

The currently implemented features of **PICLas** include

- Direct Simulation Monte Carlo Module
  - Octree cell sub-division for particle pairing
  - ...
- Particle-in-Cell Module
  - Equation systems:
    - Maxwell's equations
  - Space discretization: DGSEM method (Kopriva and Gassner 2010; Hindenlang et al. 2012)

    - Legendre Gauss
    - Legendre Gauss Lobatto
- Time discretization: explicit Runge-Kutta methods
    - standard RK methods
    - low storage RK methods (Carpenter and Kennedy 1994)
- Two- or three-dimensional domains
- Riemann solvers:
    - Flex-Vector Splitting
- Curved Meshes
- Nonconforming Meshes via mortar interfaces (Kopriva, Woodruff, and Hussaini 2002)
- Boundary conditions
    - PEC, Silver-Mueller
- Time averaging

3 Workflow

In this chapter, the complete process of setting up a simulation in **PICLas** is detailed.

3.1 Mesh generation using HOPR

PICLas obtains its computational meshes solely from the high order preprocessor **HOPR** (available under GPLv3 at https://www.hopr-project.org) in HDF5 format. The design philosophy is that all tasks related to mesh organization, different input formats and the construction of high order geometrical mappings are separated from the *parallel* simulation code. These tasks are implemented most efficiently in a *serial* environment.

The employed mesh format is designed to make the parallel read-in process as simple and fast as possible. For details concerning the mesh format please refer to the **HOPR** HDF5 Curved Mesh Format documentation.

Using **HOPR**, simple, structured meshes can be directly created using an inbuilt mesh generator. More complex geometries can be treated by importing meshes generated by external mesh generators in CGNS or GMSH format. A number of strategies to create curved boundaries are also included in HOPR.

The test cases provided in Chapter 6 come with both a ready to use mesh file as well as a parameter file for **HOPR**, which can be used to generate or modify the meshes as needed.

Provided the mesh file has been set up, its location must be specified in the parameter file.

```
MeshFile=[path to mesh file.h5]
```

3.2 Compiler options

This section describes the main configuration options which can be set when building **PICLas** using CMake. Some options are dependent on others being enabled (or disabled), so the available ones may change.

The first set of options describe general CMake behaviour:

- `CMAKE_BUILD_TYPE`:

This statically specifies what build type (configuration) will be built in this build tree. Possible values are

  - **Release**

    "Normal" execution.

  - **Profile**

    Performance profiling using gprof.

  - **Debug**

    Debug compiler for detailed error messages during code development.

  - **SANI**

    Sanitizer compiler for even more detailed error messages during code development.

- `CMAKE_HOSTNAME`:

  This will display the host name of the machine you are compiling on.

- `CMAKE_INSTALL_PREFIX`:

  If "make install" is invoked or INSTALL is built, this directory is prepended onto all install directories. This variable defaults to /usr/local on UNIX.

For some external libraries and programs that **PICLas** uses, the following options apply:

- `CTAGS_PATH`:

  This variable specifies the Ctags install directory, an optional program used to jump between tags in the source file.

- `PICLas_BUILD_HDF5`: ON/OFF

  This will be set to ON if no prebuilt HDF5 installation was found on your machine. In this case a HDF5 version will be build and used instead.

- `HDF5_DIR`:

  If you want to use a prebuilt HDF5 library that has been build using the CMake system, this directory should contain the CMake configuration file for HDF5 (optional).

3.3 Solver settings

Before setting up a simulation, the code must be compiled with the desired parameters. The most important compiler options to be set are

- `PICLAS_EQNSYSNAME`, e.g. *Navier-Stokes*

- `PICLAS_NODETYPE`, the nodal collocation points used during the simulation. Available options are either GAUSS or GAUSS-LOBATTO.

All other options are set in the parameter file. The most important steps are

- **Set the polynomial degree `N`**

  Defines the polynomial degree of the solution. The order of convergence follows as $N+1$. Each grid cell contains $(N+1)^3$ collocation points to represent the solution.

- **Choose a de-aliasing approach.**

  For under-resolved Navier-Stokes simulations, e.g. in an LES setting, de-aliasing is important for numerical stability. Various choices are available and set using `OverintegrationType`.

  - `OverintegrationType=1`

    The first option is a filtering strategy. The complete operator is first evaluated at `N` (U_t^N) and then filtered to a lower effective degree `NUnder` (U_t^{Nunder}).

    To use this variant, specify `Nunder` to a value smaller than `N`.

  - `OverintegrationType=2`

    In this variant of the first option, the operator in reference space, e.g. JU_t, is first projected to the `NUnder` node set before converting it to physical space $U_t^{Nunder} = JU_t^{Nunder}/J^{Nunder}$. This implementation enforces conservation.

    To use this variant, specify `Nunder` to a value smaller than `N`.

  An alternative approach to guaranteeing non-linear stability is to use split form DG methods. This needs to be specified during compile time using the `PICLAS_SPLIT_DG` option, and the specific split flux formulation must be set using the parameter `SplitDG`.

- **Choose a Riemann solver**

  The Riemann solver defines how inter-element coupling is accomplished. The available variants are listed in Section 9. Use the `Riemann` and the `RiemannBC` options to specify which Riemann solver is to be used at internal interfaces and at Dirichlet boundary conditions, respectively. The default Riemann solver is "Roe with entropy fix".

- **Choose a time discretization method**

  The time discretization method is set using the option `TimeDiscMethod`. Various explicit Runge-Kutta variants are available and listed in Section 9. By default, the low-storage fourth order Runge-Kutta scheme by (Carpenter and Kennedy 1994) is employed.

3.4 Initial and boundary conditions

Initial and boundary conditions are controlled via the so-called `RefState` and `ExactFunction` constructs.

The `RefState` basically specifies a state vector in primitive form $(\rho, u, v, w, p)^T$. An arbitrary number of `RefStates` can be defined:

```
RefState=(/1,1,0,0,0.71428571/)
RefState=(/1,0.3,0,0,0.71428571/)
```

In this example, the first state would result in a parallel flow in x direction at $Ma = 1$, the second state at $Ma = 0.3$.

3.4.1 Initial conditions

The code contains a number of pre-defined analytic solution fields (ExactFunctions), which are invoked by specifying their respective number. For instance the initialization of a simple constant freestream is achieved by setting

```
IniExactFunc=1
```

The associated state vector to be used is determined by

```
IniRefState=1
```

This implies that the first of the two available `RefStates` is used for initialization.

Note: currently, the ExactFunctions contained in the code are not documented yet. They can be looked up in the source file `src/equations/navierstokes/equation.f90`.

3.4.2 Boundary conditions

The names of the boundaries are contained in the mesh file and can be used in the **PICLas** parameter file to override the boundary conditions already set in the parameter file, if necessary.

PICLas lists the boundaries and their respective boundary conditions during initialization:

```
|                  Name       Type      State      Alpha
|       BC_periodicz-          1          0          3
|       BC_periodicy-          1          0          2
|       BC_periodicx+          1          0         -1
|       BC_periodicy+          1          0         -2
|       BC_periodicx-          1          0          1
|       BC_periodicz+          1          0         -3
```

Suppose, we wish to apply a Dirichlet boundary condition with `RefState` 2 at the two lateral boundaries. Therefor, we have to add the following lines to the parameter file

```
BoundaryName=BC_periodicy-
BoundaryType=(/2,2/)
BoundaryName=BC_periodicy+
BoundaryType=(/2,2/)
```

Note that the first index within brackets specifies `BC_TYPE`, while the second one specifies `BC_STATE`, in this case the number of the `RefState` to be used. In general, `BC_STATE` identifies either a `RefState`, an `ExactFunction` or remains empty, dependent on the `BC_TYPE`. Currently implemented boundary types for *Navier-Stokes* are listed in table 3.1.

Table 3.1: Boundary conditions.

Boundary Condition	BC_TYPE	BC_STATE	Comment
Periodic BC	1	-	Can only be defined in HOPR
Weak Dirichlet	2	`RefState`	
Weak Dirichlet	12	-	Like 2, but using an external state set by `BCStateFile`
Weak Dirichlet	22	`ExactFunction`	Like 2, but using an `ExactFunction`
Wall adiabatic	3	-	
Wall isothermal	4	`RefState`	Isothermal wall, temperature is specified via p and ρ contained in the `RefState`
Wall slip	9	-	Slip, symmetry or Euler wall [1]
Outflow Mach no.	23	`RefState`	
Outflow Pressure	24	`RefState`	
Outflow Subsonic	25	`RefState`	
Inflow Total pressure / Temp.	27	`RefState`	**Special Refstate:** *total* quantities $(T_t, \alpha, \beta, 0, p_t)$

3.5 Material properties

At present, the only available equation of state in the *Navier-Stokes* solver of **PICLas** is the ideal gas. The gas constant, adiabatic exponent, Prandtl number and viscosity are specified in the parameter file using `R`, `kappa`, `Pr` and `mu0`.

[1] see (Carlson 2011) for details on the listed inflow/outflow boundary conditions.

3.6 Output time interval

Set the end time of the simulation using `TEnd` and the interval in which the solution is dumped to the hard drive with `Analyze_dt`.

Note that evaluation of body forces and other runtime analysis routines are also invoked once in every analyze interval determined by `Analyze_dt`. Set e.g. `nWriteData=10` to a value greater one to restrict the solution output to every 10th `Analyze_dt`.

Restart the simulation

The simulation may be restarted from an existing state file

```
piclas parameter.ini [restart_file.h5]
```

Note: when restarting from an earlier time (or zero), all later state files possibly contained in your directory are deleted!

3.7 Evaluation during runtime

At every `Analyze_dt`, the following evaluations are possible:

- `CalcErrorNorms=T`: Calculate the L_2 and L_∞ error norms based on the specified `ExactFunc` as reference. This evaluation is used for e.g. convergence tests.
- `CalcBodyForces=T`: Calculate the pressure and viscous forces acting on every wall boundary condition (BC_TYPE=3,4 or 9) separately. The forces are written to .dat files.
- `CalcBulkState=T`: Calculate the bulk quantities (e.g. bulk velocity in channel flow).
- `CalcWallVelocity=T`: Due to the discontinuous solution space and the weakly enforced boundaries, the no-slip condition is not exactly fulfilled. The deviation depends mainly on the resolution in the near-wall region. Thus, this evaluation can be used as a resolution measure at the wall.

3.8 Parallel execution

The simulation code is specifically designed for (massively) parallel execution using the MPI library. For parallel runs, the code must be compiled with `PICLAS_MPI=ON`.

Parallel execution is then controlled using `mpirun`

```
mpirun -np [no. processors] piclas parameter.ini
```

3.8.1 Domain decomposition

The grid elements are organized along a space-filling curved, which gives a unique one-dimensional element list. In a parallel run, the mesh is simply divided into parts along the space filling curve. Thus, domain decomposition is done *fully automatic* and is not limited by e.g. an integer factor between the number of cores and elements. The only limitation is that the number of cores may not exceed the number of elements.

3.8.2 Choosing the number of cores

Parallel performance heavily depends on the number of processing cores. The performance index is defined as

(1)
$$PID = \frac{WallTime}{nCores \cdot nDOF \cdot nTimeSteps}$$

and measures the CPU time per degree of freedom and time step. During runtime, the average PID is displayed in the output

```
CALCULATION TIME PER STAGE/DOF: [ 5.59330E-07 sec ]
```

When compared to the single performance, it can be used as a parallel efficiency measure. The PID is mainly dependent on the load per core

(2)
$$Load = nDOF/nCores$$

and the polynomial degree N. Load values for optimal performance lie in the range $Load = 2000 - 5000$. A detailed parallel performance analysis at the example of a Cray XC-40 system is given in (Atak et al. 2016).

3.9 Test case environment

The test case environment can be used as to add test case-specific code for e.g. custom source terms or diagnostics to be invoked during runtime.

The compiler option `PICLAS_TESTCASE` sets the current test case. The test cases are contained in the *src/testcase/* folder.

Standardized interfaces are defined for initialization, source terms and analysis routines

- **InitTestcase**

  Read in testcase related parameters from the `parameter.ini`, initialize the corresponding data structures.

- **FinalizeTestcase**

  Deallocate test case specific data structures.

- **ExactFuncTestcase**

  Define test case specific analytic expressions for initial or boundary conditions.

- **CalcForcing**

  Impose test case specific source terms, e.g. the pressure gradient in test case `channel`.

- **AnalyzeTestCase**

  Perform test case specific diagnostics.

Currently supplied test cases are

- **default**
- **channel**: turbulent channel flow with steady pressure gradient source term
- **phill**: periodic hill flow with controlled pressure gradient source term
- **riemann2d**: a two dimensional Riemann problem
- **taylorgreenvortex**: automatic diagnostics for the Taylor-Green vortex flow

Note that the test case environment is currently only applicable to the *Navier-Stokes* equation system.

3.10 Post processing / posti_visu tool

PICLas comes with a posti_visu tool for visualization with Paraview. The posti_visu tool takes the HDF5 files generated by **PICLas** and converts them to vtu-files readable by Paraview.

```
posti_visu [posti-prm-file [piclas-prm-file]] statefile [
   statefiles]
```

The posti_visu tool runs in parallel with activated `PICLAS_MPI` flag

```
mpirun -np [no. processors] posti_visu [posti-prm-file [piclas-prm
   -file]] statefile [statefiles]
```

Multiple HDF5 files can be passed to the posti_visu tool at once. The runtime parameters to be set in `parameter_postiVisu.ini` are

Table 3.2: Runtime parameters for the posti_visu tool.

NodeTypeVisu	VISU	Node type of the visualization basis: VISU,GAUSS,GAUSS-LOBATTO,CHEBYSHEV-GAUSS-LOBATTO
NVisu		Polynomial degree at which solution is sampled for visualization.

Table 3.2: Runtime parameters for the posti_visu tool.

VarName	Names of variables, which should be visualized.

Some of these options are duplicates from options for **PICLas**. You can use the same parameter file for both executables.

The default value for NodeTypeVisu uses equidistant nodes which include the boundary points of elements. The default value for NVisu is to use the same number of sampling points as collocation points on each element. For high quality visualization, it is usually adviseable to choose NVisu higher than the polynomial degree of the computation in order to keep interpolation errors small. The parameter VarName specifies the variables being visualized. It can be used multiple times for each variable. Visualization of derived quantities such as Velocities and Pressure is possible. If no VarName is specified, the five conservative variables are visualized.

The following lines can be used as an example for the parameter_postiVisu.ini file.

```
NVisu   = 10
varName = MomentumX
varName = VelocityX
varName = Density
varName = Pressure
varName = Temperature
```

4 Features & Models

4.1 Particle Tracking

4.1.1 Linear

4.1.2 Curved

4.2 Boundary Conditions

4.2.1 Field

4.2.2 Particle

4.2.2.1 Porous Wall

The porous boundary condition uses a removal probability to determine whether a particle is deleted or reflected of the boundary.

4.3 Particle Emission

4.3.1 Surface Flux

4.3.1.1 Adaptive Boundaries

Multiple adaptive particle emission/boundary conditions can be defined. Sources...

4.4 Particle in Cell

4.5 Direct Simulation Monte Carlo

4.5.1 Relaxation

4.5.2 Chemistry

4.5.3 Surface Chemistry

5 Tools Overview

This section gives an overview over the tools contained in the **PICLas** repository. It also provides references to the tutorials where their usage is explained. There are two different kinds of tools:

- **POSTI**-tools can be compiled together with **PICLas** given the according `cmake` options.
- In the `tools` folder, a collection of mostly shell or python scripts can be found. They are mostly used to manage **PICLas** runs and **PICLas** output files.

5.1 POSTI

POSTI tools are mostly documented in section 3.10 and in the tutorials (chapter 6). Here, an overview is given together with references to the respective tutorials. A list and description of input parameters can be shown with the command

```
[posti_toolname] --help
```

for all **POSTI** tools using a parameter file.

5.1.1 Visualization

posti_visu	
Brief description	Converts **PICLas** state files from HDF5 format to ParaView readable `.vtu` format. Spatial resolution of output as well as calculated variables and further options can be given in the parameter file.
Basic usage	`posti_visu [parameter.ini] [statefile1.h5 ...]`
Further info / usage example	3.10, 6.1.3, **??**, **??**, **??**, **??**

Paraview plugin	
Brief description	A ParaView reader based on `posti_visu` to load **PICLas** state files in ParaView. Provides the interface to adjust `posti_visu` parameters in the ParaView GUI.
Basic usage	`libVisuReader.so` is loaded as a Plugin in ParaView

Paraview plugin	
Further info / usage example	No tutorials so far

5.1.2 Swap meshes

posti_swapmesh	
Brief description	Interpolates state file data from one mesh to another. Uses high-order interpolation and a Newton coordinate search algorithm. Meshes do not have to be conforming. A reference state can be given for areas in the target mesh not covered by the source mesh.
Basic usage	`posti_swapmesh [parameter.ini] [statefile.h5]`
Further info / usage example	No tutorials so far

5.1.3 Record points

posti_preparerecordpoints	
Brief description	Enables **PICLas** to record values at a set of physical points over time with a higher temporal sampling rate than the state file output interval. The record point coordinates and the **PICLas** mesh are defined in the parameter file. Creates an additional `.h5` file, whose path is passed to **PICLas** as a parameter.
Basic usage	`posti_preparerecordpoints [parameter_prepareRP.ini]`
Further info / usage example	**??**

posti_visualizerecordpoints	
Brief description	Performs post-processing of the `*_RP_*` files written by **PICLas**: merges several time steps and writes output such as value over time or spectra.
Basic usage	`posti_visualizerecordpoints [parameter_visuRP.ini] [projectname_RP_*.h5]`
Further info / usage example	**??**

posti_evaluaterecordpoints	
Brief description	Evaluate the values at recorpoints a posteri from existing statefiles. Can be used if the recordpoints have not been set during the simulation, but will only give coarse temporal resolution.
Basic usage	`posti_evaluaterecordpoints [parameter.ini] [statefile.h5]`
Further info / usage example	No tutorials so far

5.1.4 Time averaging

posti_mergetimeaverages	
Brief description	Averages several **PICLas** `State` or `TimeAverage` files. If `TimeAverage` files are the input, each files is weighted with its time averaging period. `State` files are all weighted equally. All HDF5 data sets are averaged. No parameter file is needed.
Basic usage	`posti_mergetimeaverages [statefile1.h5 statefile2.h5 ...]`
Further info / usage example	No tutorials so far

posti_calcfluctuations	
Brief description	Calculates Flucuations from `Mean` and `MeanSquare` as given in the (merged) `TimeAverage` files. Fluctuations are written into an additional data set in the same HDF5 file. All applicable fluctuations are calculated. No parameter file is needed.
Basic usage	`posti_calcfluctuations [statefile1.h5 statefile2.h5 ...]`
Further info / usage example	No tutorials so far

5.1.5 Test cases

posti_channel_fft	
Brief description	Calculates the mean velocity and Reynolds stress profiles of the turbulent channel flow test case by averaging both in the direction parallel to the wall and by averaging the upper and lower half of the channel. Furthermore, kinetic energy spectra dependent on the distance to the wall are computed.
Basic usage	`posti_channel_fft [parameter_channelfft.ini] [ statefile1.h5 statefile2.h5 ...]`
Further info / usage example	**??**

5.2 tools folder

The scripts provided in the tools folder are generally not part of the tutorials. They are briefly described below. The path to the python files (of the form $PICLASROOT/tools/SUBDIR/) is omitted in the following.

For most python tools, possible arguments and syntax can be shown with the -h argument:

```
python2 [toolname.py] -h
```

5.2.1 animate

The python script **animate_paraview.py** creates movies from a series of state files using PvBatch, a GUI-less interface to ParaView. You need ParaView installed on your system (details can be found in the ParaView section) and the directory containing the PvBatch executable needs to be a part of your $PATH. Before running this script, you have to visualize your **PICLas** state file with ParaView and save the current view via Save State..., e.g. under the name pvstate.pvsm. You also need the MEncoder tool installed. The basic command to run the script is

```
python2 animate_paraview.py -l [pvstate.pvsm] -r [
   path_to_posti_paraview_plugin] [statefile1.h5 statefile2.h5 ...
   ]
```

Apart from the movie file, the script also outputs a .png-file for each HDF5 file given as input. In order to visualize a set of .vtu-files, e.g. from posti_visu output, omit the -r argument and pass .vtu-files instead of .h5-files. Further options can be shown with the -h argument.

There are further tools for image handling in this folder:

The tool **concatenatepics.py** stitches several pairs of images (e.g. creates a time series of stitched images from two time series of images). A possible command could look like this (*Further options can be shown with the -h argument*):

```
python concatenatepics.py -d e -p left*.png  -a right*.png
```

The tool **crop.py** crops several images to the same size. Simply pass all images as arguments:

```
python crop.py [image*.png]
```

The script **pics2movie.py** creates a movie from several images using the mencoder tool (which is also done as part of the animate_paraview.py script. Basic usage is again

```
python pics2movie.py [image*.png]
```

and further options can again be shown with the -h argument.

5.2.2 convergence_test

The python scripts convergence and convergence_grid provide automated convergence tests for p- and h-convergence, respectively. The basic command is

```
python2 convergence piclas [parameter.ini]
```

where convergence can be replaced by convergence_grid for h-convergence. Further options can again be shown with the -h option.

Note that for h convergence, the mesh names are hard-coded to the form CART_HEX_PERIODIC_MORTAR_XXX_2D_mesh.h5, where XXX is the number of elements in each direction, and MORTAR and 2D are optional. The polynomial degree in the parameter file is *always* overwritten by that passed to the script as an optional argument, with a default value of 3, if no such argument is passed.

5.2.3 userblock

The userblock contains complete information about a **PICLas** run (git branch of the repository, differences to that branch, cmake configuration and parameter file) and is prepended to every .h5 state file. The parameter file is prepended in ASCII format, the rest is binary and is generated automatically during the build process with the generateuserblock.sh script. It can be extracted and printed using the extract_userblock.py script. Its basic usage is

```
python2 extract_userblock.py -XXX [statefile.h5]
```

where -XXX can be replaced by

- -s to show all available parts of the userblock (such as CMAKE or GIT BRANCH)
- -a to print the complete userblock
- -p [part] to print one of the parts listed with the -s command.

The second python tool in this folder is rebuild.py. It extracts the userblock from a state file and builds a **PICLas** repository and binary identical to the one that the state file was created with. In order to do so, it clones a **PICLas** git repository, checks out the given branch, applies the stored changes to the git HEAD and builds **PICLas** with the stored cmake options. If run with the parameter file given in the INIFILE part of the userblock, this binary should reproduce the same results/behaviour (possible remaining sources of different output are for example differences in restart files, compilers, linked libraries or machines). The basic usage is

```
python2 rebuild.py [dir] [statefile.h5]
```

where dir is an empty directory that the repository is cloned into and where the piclas executable is built. statefile.h5 is the state file whose userblock is used to rebuild the piclas executable. Help can be shown via -h for both userblock scripts.

5.2.4 sortfiles.sh

The sortfiles.sh script sorts all .h5-files in subfolders State, BaseFlow, TimeAvg and RP, while keeping the last time instance at the upper level. It also copies Log.*.sdb, .log and .out files into a logs subdirectory. The project name is hard-coded in the script and has to be adapted there, the directory that is to be sorted is passed as an argument.

5.2.5 getload.py

This script is specific to runs on HPC systems. It calculates a suitable number of nodes and cores to achieve

- a number of degrees of freedom per core which is close to a target
- an average number of elements per core which is just below a close integer, such that parallel efficiency is not impaired by a few cores with higher load that the others have to wait for.

No arguments are passed to this script, all input values are hard-coded and have to be adjusted in the script.

5.2.6 testcases

The python script **plotChannelFFT.py** creates plots of the mean velocity and the Reynolds stress profiles as well as the turbulent energy spectra based on the posti_channel_fft HDF5 output files. Basic usage is:

```
python plotChannelFFT.py -p projectname -t time
```

Further options can be shown with the -h argument.

6 Tutorials

This chapter will give a detailed overview of flow simulations with **PICLas**. It assumes that you are familiar with how to set the compiler options and how to compile the code. The path to all executables is omitted. It is assumed that you either added aliases for **piclas**, **hopr** and all posti tools, or that you added the binary directories to your `$PATH` variable as described in 2.1.3.

Each tutorial is equipped with .ini files *parameter_hopr.ini*, *parameter_piclas.ini*, *parameter_postiVisu.ini* as well as mesh file **_mesh.h5* in HDF5 format (created with **HOPR**).

```
parameter_hopr.ini
parameter_piclas.ini
parameter_postiVisu.ini
mesh.h5
```

We suggest to copy each folder to a new directory, where you can run and modify the "INI-files".

6.1 Freestream

The setup considers a freestream scenario with constant pressure $p = 101325.0$ Pa, density $\rho = 1.225$ kg/m^3 and velocity vector $\mathbf{U} = (1, 1, 1)^T$ m/s.

Copy the `freestream` tutorial folder

```
    cp -r $PICLAS$_TUTORIALS/freestream .
    cd freestream
```

6.1.1 Mesh Generation with HOPR

The mesh files used by **PICLas** are created by supplying an input file *parameter_hopr.ini* with the appropriate information.

```
hopr parameter_hopr.ini
```

This creates the mesh file *cartbox_mesh.h5* in HDF5 format. Alternatively, if you do not want to run **hopr**, you can also use the provided mesh.

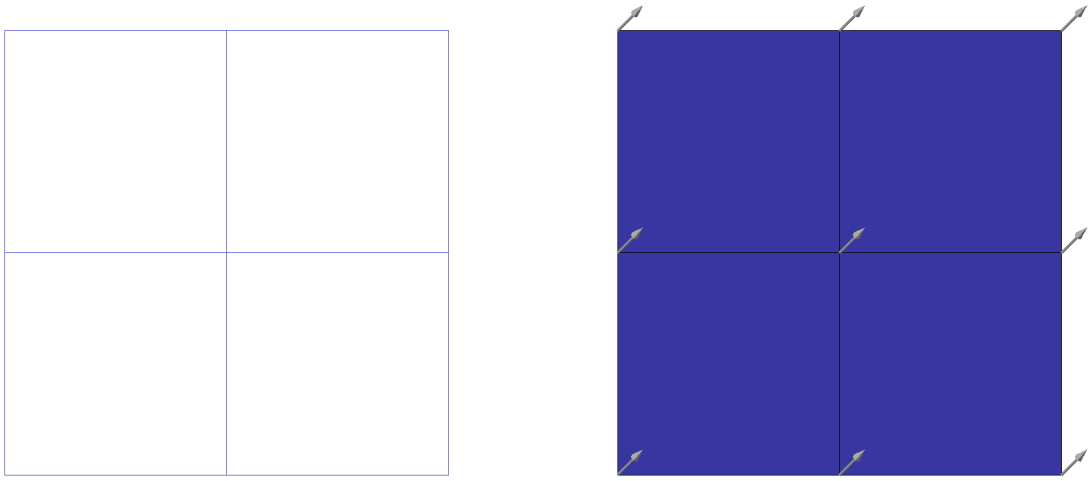

Figure: Mesh and flow field solution with velocity vector plot of the velocity field. View in x-y-plane.

6.1.2 Flow Simulation with PICLas

The simulation setup is defined in *parameter_piclas.ini*. The initial condition is selected via the variable vector **RefState=(/1.225,1.,1.,1.,101325./)** which represents the solution vector $(\rho, u, v, w, p)^T$. **PICLas** allows for multiple **RefState** vectors and numerates them respectively for them to be used by different functions. In this example a single **RefState** is supplied and therefore is given the number **1**.

IniRefState = 1 : the initial condition uses **RefState 1** for the initial flow field solution.

IniExactFunc = 1 : the used exact function routine uses **RefState 1**, e.g., for the calculation of the L_2 error norm.

Constant flow properties like the gas constant are given in table 6.1 and define the gas behavior in combination with the ideal gas law $p = \rho RT$.

Table 6.1: Numerical settings

Property	Variable	Value
dynamic viscosity μ	mu0	0.000018547
ideal gas constant R	R	276
isentropic coefficient κ	kappa	1.4

6.1.3 Numerical settings

The DG solution on the mesh is represented by piecewise polynomials and the polynomial degree in this tutorial is chosen as $N = 3$.

The default settings for these properties are displayed in table 6.2.

Table 6.2: Numerical settings

Variable	Description	Value
N	polynomial degree	3
MeshFile		cartbox_mesh.h5
tend		1e-6
Analyze_dt		1e-6
nWriteData		1
CFLscale		0.99
DFLscale		0.4

The command

```
piclas parameter_piclas.ini > std.out
```

runs the code and dumps all output into the file *std.out*. If the run has completed successfully, which should take only a brief moment, the contents of the working folder should look like in figure 6.1

```
total 104K
lrwxrwxrwx 1    20 Jul 22 17:05 convert -> ../build/bin/convert
-rw-r--r-- 1 2.3K Jul 22 17:06 parameter_hopr.ini
-rw-r--r-- 1 1.3K Jul 22 17:06 parameter_flexi.ini
-rw-r--r-- 1  10K Jul 22 17:06 cartbox_mesh.h5
lrwxrwxrwx 1    18 Jul 22 17:07 flexi -> ../build/bin/flexi
-rw-r--r-- 1  31K Jul 22 17:17 cartbox_State_0000000.000000000.h5
-rw-r--r-- 1  31K Jul 22 17:17 cartbox_State_0000000.000001000.h5
-rw-r--r-- 1  17K Jul 22 17:17 std.out
```

Figure 6.1: The folder contents after a successful run

Two additional files have been created, which are are named **Projectname_State_Timestamp.h5**. They contain the solution vector of the conserved variables at each interpolation node at the given time, which corresponds to multiplies of **Analyze_dt**. If these files are not present, something went wrong during the execution of **PICLas**. In that case, check the *std.out* file for an error message.

After a successful completion, the last lines in this files should look like in figure 6.2

To visualize the solution, the *State*-files must be converted into a format suitable for **ParaView**. Issue the command

```
================================================================================================================================
 INITIALIZATION DONE! [    0.03 sec ]
================================================================================================================================
--------------------------------------------------------------------------------------------------------------------------------
#GridCells :    8.0000000E+00
#DOFs      :    5.1200000E+02
#Procs     :    1.0000000E+00
#DOFs/Proc :    5.1200000E+02
 WRITING INITIAL SOLUTION:
 WRITE STATE TO HDF5 FILE...DONE  [.004s]
--------------------------------------------------------------------------------------------------------------------------------
 Errors of initial solution:
 Sim time   :    0.0000000E+00
 L_2        :    1.878429114E-16   1.878429114E-16   1.878429114E-16   1.878429114E-16   7.101037057E-11
 L_Inf      :    8.881784197E-16   8.881784197E-16   8.881784197E-16   8.881784197E-16   2.619344741E-10
--------------------------------------------------------------------------------------------------------------------------------
Initial Timestep  :    1.0636682E-04
 CALCULATION RUNNING...
--------------------------------------------------------------------------------------------------------------------------------
 Sys date   :    22.07.2016 17:17:58
 CALCULATION TIME PER STAGE/DOF: [ 1.62339E-06 sec ]
 Timestep   :    1.0636682E-04
#Timesteps  :    1.0000000E+00
 WRITE STATE TO HDF5 FILE...DONE  [.006s]
 Sim time   :    1.0000000E-06
 L_2        :    1.878429114E-16   4.948834301E-16   5.307730445E-16   5.692408050E-16   7.101037057E-11
 L_Inf      :    8.881784197E-16   3.996802889E-15   4.662936703E-15   5.107025913E-15   2.619344741E-10
................................................................................................................................
 FLEXI RUNNING cartbox... [    0.05 sec ]
--------------------------------------------------------------------------------------------------------------------------------

================================================================================================================================
 FLEXI FINISHED! [    0.05 sec ]
================================================================================================================================
```

Figure 6.2: The *std.out* file after a successful run

```
posti_visu parameter_postiVisu.ini parameter_piclas.ini
   cartbox_State_0000000.00000*
```

to generate the corresponding *vtu*-files, which can then be loaded into **ParaView**.

7 Unit tests

Unit tests are used to test individual key units of the source code. Currently these key routines include:

- Calculation of node positions and integration weights.
- Calculation of Vandermonde matrices.
- Calculation of derivative matrices.
- Algorithms to interpolate data from one set of nodes to another.
- Algorithm to prolong volume data to sides.
- Algorithm to perform a surface integral.
- Functionality of Read-In tools.

7.1 Integration of unit test with CTest

These unit tests are integrated into the **PICLas** build process using the CTest tool. Usually CTest will be run every time you build **PICLas** and give you an overview on the exit status of each test that looks something like this:

```
Run unit tests
Test project /home/PICLAS/build
    Start 1: NodesAndWeights
1/6 Test #1: NodesAndWeights ..............   Passed    0.01 sec
    Start 2: Vandermonde
2/6 Test #2: Vandermonde ..................   Passed    0.01 sec
    Start 3: DerivativeMatrix
3/6 Test #3: DerivativeMatrix .............   Passed    0.00 sec
    Start 4: ChangeBasis
4/6 Test #4: ChangeBasis ..................   Passed    0.00 sec
    Start 5: SurfInt
5/6 Test #5: SurfInt ......................   Passed    0.00 sec
    Start 6: ProlongToFace
6/6 Test #6: ProlongToFace ................   Passed    0.00 sec

100% tests passed, 0 tests failed out of 6

Total Test time (real) =   0.05 sec
```

To manually run the tests after a build use the CTest command

```
ctest
```

in your build directory. The manual page of CTest can give you an overview of all available options.

If you don't want to run the test after each build there is a CMake option called PICLAS_UNITTESTS that can be used to turn the tests on and off. This is an advanced option that CCMake will only show if you enter the advanced mode by pressing the t key.

7.2 Implementation of unit tests

All unit tests are implemented in FORTRAN and can be found in the subdirectory unitTests in your **PICLas** directory alongside a seperate CMakeLists.txt and some binary input and reference files.

7.2.1 CMakeLists.txt

The CMakeLists.txt defines a custom function called add_unit_test which can be used in the CMakeLists.txt to add a single test to the CTest tool. The syntax is

```
add_unit_test(NAME SOURCEFILE.F90)
```

All tests are defined using this function. At the end of the CMakeLists.txt a custom target all_tests is defined which includes all unit tests and will run the ctest command after it has been build.

The whole CMakeLists.txt content is included in the main CMakeLists.txt if the option PICLAS_UNITTESTS is set to ON (default) by CMake.

7.2.2 General unit test structure

The general structure of the unit tests is the same in all cases. They are implemented as FORTRAN programs. The unit test will call a function or subroutine from the **PICLas** framework with input either set in the program itself or read from a binary file. The output of this call will then be compared to some precomputed reference results (also stored as binary files) with a certain tolerance to account for differences in e.g. compiler versions and system architecture. If the results are within the given tolerance, the test will be passed, otherwise it will fail by returning a value other than 0.

The programs usually also contain a command line option that can be uses to generate the reference solution from a code version that is known to work correctly.

Have a look at the source code of one of the already implemented unit tests if you want to have a more detailed idea about how to implement your own tests.

7.2.3 Generation of reference mesh data

Some of the unit tests require parts of the mesh data structure to be able to call the funtions to be tested. For this purpose, a curved single element is created and all the mesh data stored as a binary file called `UnittestElementData.bin`. This binary file can then be read during runtime by the unit test programs.

To generate the curved single element mesh, run **HOPR** with the parameter file provided in the `unitTest` subdirectory of **PICLas**. To generate the binary file, run **PICLas** with the following command line argument and the parameter file provided in the `unitTest` subdirectory:

```
piclas --generateUnittestReferenceData parameter.ini
```

8 Installation guidelines

This chapter contains guidelines to install the code from Github on specific systems.

8.1 Cloning and compiling at the HLRS

Unfortunately, the GitHub server is not available on machines at the HLRS, such as the Hazelhen, due to restricted internet access. The workaround is to use ssh tunnels to access the GitHub repositories. Note that the reomte repositories hosted at teh GitLab at the Institute of Aerodynamics and Gasdynamics (IAG), no ssh tunnel is required and cloning works straight forwardly.

The following instructions to access the GitHub repositories on HLRS machines is taken from the HLRS wickie page, see https://wickie.hlrs.de/platforms/index.php/Secure_Shell_ssh#Git.

8.1.1 HTTPS

Unfortunately, just using a SSH tunnel as with the SSH and git protocols is not sufficient in this case. Instead, one has to connect via an additional SOCKS proxy on a machine that has unlimited access to the internet, e.g. your local machine.

In order to do so, establish a proxy by using a special feature of OpenSSH:

```
ssh -N -D 1080 localhost
```

This will establish some kind of a "loopback" SSH connection from your local machine to itself which will not execute any command (-N) but act as an SOCKS proxy on port 1080 (-D 1080).

On a second shell, now login to the desired HWW-system and forward a port on the remote machine (e.g. 7777) to the port on your local machine where the newly established SOCKS proxy is listening on (1080):

```
ssh -R 7777:localhost:1080 <system-name>.hww.de
```

By doing so, you have a SOCKS proxy listening on port 7777 of the HWW-system. Hence you can use this proxy for accessing remote git repositories. Unfortunately, the default versions of

git installed on the HWW-systems are not capable of doing this. You hence have to load an appropriate version first:

```
module load tools/git
```

In order to use the proxy, you can now add "-c https.proxy=‘socks5://localhost:7777’ " to your git commands, e.g.:

```
git -c https.proxy='socks5://localhost:7777' clone https://
github.com/piclas-framework/piclas.git
```

In order to avoid typing this in every git call, you can also set the respective port to be used whenever git talks to a remote repository via HTTPS by

```
git config --global https.proxy 'socks5://localhost:7777'
```

Unfortunately, to connect with GitHub for pulling or pushing, the connection to Hazelhen has to be done via the ssh tunnel.

9 Parameter file options

A `parameter.ini` file is needed to control the code. An overview of all options in the parameter file can be generated by following command in the terminal:

```
piclas --help
```

Generally following types are used:

```
INTEGER = 1
REAL    = 1.23456
LOGICAL = T          ! True
LOGICAL =            ! False
STRING  = PICLAS
VECTOR  = (/1.0,2.0,3.0/)
```

The concept of the parameter file is described as followed:

- each single line is saved and examined for specific variable names
- the examination is case-insensitive
- comments can be set with symbol "!" in front of the text

```
! commented text
```

- numbers can also be set by using "pi"

```
vector = (/1,2Pi,3Pi/)
```

- the order of defined variables is with one exception generally indifferent, but it is preferable to group similar variables
- the order becomes important only by modifying boundary conditions, if you want to modify a specific boundary by addressing its name, the related boundary type has to be defined

```
BoundaryName=inflow          ! BC_Name defined in mesh file
BoundaryType=(/2,0,0,0/)
BoundaryName=outflow         ! BC_Name defined in mesh file
BoundaryType=(/2,0,0,0/)
```

The following tables describe the main configuration options which can used in the parameter file.

MPI		
Variable	**Default**	**Description**
GroupSize	0	Define size of MPI subgroups, used to e.g. perform grouped IO, where group master collects and outputs data.

IO_HDF5		
Variable	**Default**	**Description**
gatheredWrite	F	Set true to activate gathered HDF5 IO for parallel computations. Only local group masters will write data after gathering from local slaves.

LoadBalance		
Variable	**Default**	**Description**
DoLoadBalance	F	Set flag for doing dynamic LoadBalance.
LoadBalanceSample	1	Define number of iterations (before analyze_dt) that are used for calculation of elemtime information
PartWeightLoadBalance	F	Set flag for doing LoadBalance with partMPIWeight instead of elemtimes. Elemtime array in state file is filled with nParts*PartMPIWeight for each Elem. If Flag [TRUE] LoadBalanceSample is set to 0 and vice versa.
Load-DeviationThreshold	0.10	Define threshold for dynamic load-balancing. Restart performed if (Maxweight-Targetweight)/Targetweight $>$ defined value.

LoadBalance		
Particles-MPIWeight	0.02	Define weight of particles for elem loads. (only used if ElemTime does not exist or DoLoadBalance=F).
WeightDistributionMethod		Method for distributing the elem to procs. DEFAULT: 1 if Elemtime exits else -1 -1: elements are equally distributed 0: distribute to procs using elemloads 1: distribute to procs using elemloads, last proc recieves least 2: NOT WORKING 3: TODO DEFINE 4: TODO DEFINE 5/6: iterative smoothing of loads towards last proc

Interpolation		
Variable	**Default**	**Description**
N		Polynomial degree of computation to represent to solution

Restart		
Variable	**Default**	**Description**
DoInitialAutoRestart	F	Set Flag for doing automatic initial restart with loadbalancing routines after first 'InitialAutoRestartSample'-number of iterations. Restart is done if Imbalance > 'Load-DeviationThreshold'.

Restart		
InitialAutoRestartSample		Define number of iterations at simulation start used for elemtime sampling before performing automatic initial restart. IF 0 than one iteration is sampled and statefile written has zero timeflag. DEFAULT: LoadBalanceSample.
InitialAutoRestart-PartWeightLoadBalance	F	Set flag for doing initial auto restart with partMPIWeight instead of elemtimes. Elemtime array in state file is filled with nParts*PartMPIWeight for each Elem. If Flag [TRUE] InitialAutoRestartSample is set to 0 and vice versa.
RestartNullifySolution	F	Set the DG solution to zero (ignore the DG solution in the state file)

Output		
Variable	**Default**	**Description**
ProjectName		Name of the current simulation (mandatory).
Logging	F	Write log files containing debug output.
WriteErrorFiles	T	Write error files containing error output.
OutputFormat	None	File format for visualization: None, Tecplot, TecplotASCII, ParaView. Note: Tecplot output is currently unavailable due to licensing issues.
ASCIIOutputFormat	CSV	File format for ASCII files, e.g. body forces: CSV, Tecplot.
doPrintStatusLine	F	Print: percentage of time, ...

Output		
WriteStateFiles	T	Write HDF5 state files. Disable this only for debugging issues. NO SOLUTION WILL BE WRITTEN!

Piclas Initialization		
Variable	**Default**	**Description**
UseDSMC	F	Flag for using DSMC in Calculation
UseLD	F	Flag for using LD in Calculation

TimeDisc		
Variable	**Default**	**Description**
TEnd		End time of the simulation (mandatory).
CFLScale		Scaling factor for the theoretical CFL number, typical range 0.1..1.0 (mandatory)
maxIter	-1	Stop simulation when specified number of timesteps has been performed.
NCalcTimeStepMax	1	Compute dt at least after every Nth timestep.
IterDisplayStep	1	Step size of iteration that are displayed.

Mesh		
Variable	**Default**	**Description**
DoSwapMesh	F	TODO-DEFINE-PARAMETER Flag to swap mesh for calculation.
SwapMeshExePath		(relative) path to swap-meshfile (mandatory).

Mesh		
SwapMeshLevel	0	TODO-DEFINE-PARAMETER 0: initial grid 1: first swap mesh 2: second swap mesh
MeshFile		(relative) path to meshfile (mandatory) (HALOWIKI:) usually located in directory of project.ini
useCurveds	T	Controls usage of high-order information in mesh. Turn off to discard high-order data and treat curved meshes as linear meshes.
DoWriteStateToHDF5	T	Write state of calculation to hdf5-file. TODO-DEFINE-PARAMETER
interpolateFromTree	T	For non-conforming meshes, built by refinement from a tree structure, the metrics can be built from the tree geometry if it is contained in the mesh. Can improve free-stream preservation.
meshScale	1.0	Scale the mesh by this factor (shrink/enlarge).
meshdeform	F	Apply simple sine-shaped deformation on cartesion mesh (for testing).
CalcPoyntingVecIntegral	F	TODO-DEFINE-PARAMETER Calculate pointing vector integral | only perpendicular to z axis
crossProductMetrics	F	Compute mesh metrics using cross product form. Caution: in this case free-stream preservation is only guaranteed for N=3*NGeo.

Mesh		
BoundaryName		Names of boundary conditions to be set (must be present in the mesh!).For each BoundaryName a BoundaryType needs to be specified.
BoundaryType		Type of boundary conditions to be set. Format: (BC_TYPE,BC_STATE)
writePartitionInfo	F	Write information about MPI partitions into a file.

Equation		
Variable	**Default**	**Description**
c0	1.0	TODO-DEFINE-PARAMETER Velocity of light (in vacuum)
eps	1.0	TODO-DEFINE-PARAMETER Electric constant (vacuum permittivity)
mu	1.0	TODO-DEFINE-PARAMETER Magnetic constant (vacuum permeability = 4piE-7H/m)
IniExactFunc		TODO-DEFINE-PARAMETER Define exact function necessary for linear scalar advection
IniWavenumber	(/ 1.0, 1.0, 1.0 /)	TODO-DEFINE-PARAMETER
IniCenter	(/ 0.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER
IniAmplitude	0.1	TODO-DEFINE-PARAMETER
IniHalfwidth	0.1	TODO-DEFINE-PARAMETER
ACfrequency	0.0	TODO-DEFINE-PARAMETER
ACamplitude	0.0	TODO-DEFINE-PARAMETER

Equation		
chitensWhichField	-1	TODO-DEFINE-PARAMETER
chitensValue	-1.0	TODO-DEFINE-PARAMETER
chitensRadius	-1.0	TODO-DEFINE-PARAMETER
AlphaShape	2	TODO-DEFINE-PARAMETER
r_cutoff	1.0	TODO-DEFINE-PARAMETER Modified for curved and shape-function influence (c*dt*SafetyFactor+r_cutoff)

HDG		
Variable	**Default**	**Description**
NonLinSolver	1	TODO-DEFINE-PARAMETER
NewtonExactSourceDeriv	F	TODO-DEFINE-PARAMETER
AdaptIterNewton	0	TODO-DEFINE-PARAMETER
NewtonAdaptStartValue	F	TODO-DEFINE-PARAMETER
AdaptIterNewtonToLinear	100	TODO-DEFINE-PARAMETER
RelaxFacNonlinear	0.5	TODO-DEFINE-PARAMETER
AdaptIterFixPoint	10	TODO-DEFINE-PARAMETER
MaxIterFixPoint	10000	TODO-DEFINE-PARAMETER
NormNonlinearDevLimit	99999.0	TODO-DEFINE-PARAMETER
EpsNonLinear	0.10E-05	TODO-DEFINE-PARAMETER
PrecondType	2	TODO-DEFINE-PARAMETER
epsCG	0.10E-05	TODO-DEFINE-PARAMETER

HDG		
useRelativeAbortCrit	F	TODO-DEFINE-PARAMETER
maxIterCG	500	TODO-DEFINE-PARAMETER
OnlyPostProc	F	TODO-DEFINE-PARAMETER
ExactLambda	F	TODO-DEFINE-PARAMETER
HDG_N		TODO-DEFINE-PARAMETER Default: 2*N
HDG_MassOverintegration	F	TODO-DEFINE-PARAMETER

Dielectric Region		
Variable	**Default**	**Description**
DoDielectric	F	Use dielectric regions with EpsR and MuR
DielectricFluxNonConserving	F	Use non-conservative fluxes at dielectric interfaces between adielectric region and vacuum
DielectricEpsR	1.0	Relative permittivity
DielectricMuR	1.0	Relative permeability
DielectricTestCase	default	Test cases, e.g., “FishEyeLens” or “FH_lens”
DielectricRmax	1.0	Radius parameter for functions
DielectricCheckRadius	F	Use additional parameter “DielectricRadiusValue” for checking if a DOF is within a dielectric region
DielectricRadiusValue	-1.0	Additional parameter radius for checking if a DOF is within a dielectric region
xyzPhysicalMinMaxDielectric	(/ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 /)	[xmin, xmax, ymin, ymax, zmin, zmax] vector for defining a dielectric region by giving the bounding box coordinates of the PHYSICAL region

Dielectric Region		
xyzDielectricMinMax	(/ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 /)	[xmin, xmax, ymin, ymax, zmin, zmax] vector for defining a dielectric region by giving the bounding box coordinates of the DIELECTRIC region
Dielectric_E_0	1.0	Electric field strength parameter for functions

Filter		
Variable	**Default**	**Description**
FilterType	0	TODO-DEFINE-PARAMETER
HestFilterParam	(/ 36.0, 12.0, 1.0 /)	TODO-DEFINE-PARAMETER

Analyze		
Variable	**Default**	**Description**
DoCalcErrorNorms	F	Set true to compute L2 and LInf error norms at analyze step.
Analyze_dt	0.0	Specifies time intervall at which analysis routines are called.
NAnalyze		Polynomial degree at which analysis is performed (e.g. for L2 errors). Default: 2*N.
OutputTimeFixed	-1.0	fixed time for writing state to .h5
nSkipAnalyze		(Skip Analyze-Dt)
CalcTimeAverage		Flag if time averaging should be performed
VarNameAvg		Count of time average variables
VarNameFluc		Count of fluctuation variables
nSkipAvg		Iter every which CalcTimeAverage is performed

Analyze		
Field-AnalyzeStep	1	Analyze is performed each Nth time step
CalcPotentialEnergy	F	Calculate Potential Energy. Output file is Database.csv
CalcPointsPerWavelength	F	Flag to compute the points per wavelength in each cell

Analyzefield		
Variable	**Default**	**Description**
PoyntingVecInt-Planes	0	Total number of Poynting vector integral planes for measuring the directed power flow (energy flux density: Density and direction of an electromagnetic field.
Plane-Tolerance	0.1E-04	Absolute tolerance for checking the Poynting vector integral plane coordinates and normal vectors of the corresponding sides for selecting relevant sides
Plane-[$]-x-coord	0.0	TODO-DEFINE-PARAMETER
Plane-[$]-y-coord	0.0	TODO-DEFINE-PARAMETER
Plane-[$]-z-coord	0.0	TODO-DEFINE-PARAMETER
Plane-[$]-factor	1.0	TODO-DEFINE-PARAMETER
PoyntingMainDir		Direction in which the Poynting vector integral is to be measured. 1: x 2: y 3: z (default)

RecordPoints		
Variable	**Default**	**Description**
RP_inUse	F	Set true to compute solution history at points defined in recordpoints file.

RecordPoints		
RP_DefFile		File containing element-local parametric recordpoint coordinates and structure.
RP_MaxMemory	100.0	Maximum memory in MiB to be used for storing recordpoint state history. If memory is exceeded before regular IO level states are written to file.

Particle		
Variable	**Default**	**Description**
Particles-ManualTimeStep	0.0	Manual timestep [sec]
Part-AdaptiveWeightingFactor	0.001	Weighting factor theta for weighting of average instantaneous values with those of previous iterations.
Particles-SurfaceModel	0	Define Model used for particle surface interaction. If >0 then look in section SurfaceModel. 0: Maxwell scattering 1: Kisliuk / Polanyi Wigner (currently not working) 2: Recombination model 3: adsorption/desorption + chemical interaction (SMCR with UBI-QEP, TST and TCE) 4: TODO 5: SEE-E and SEE-I (secondary e-emission due to e- or i+ bombardment) by Levko2015 for copper electrondes 6: SEE-E (secondary e- emission due to e- bombardment) by Pagonakis2016 for molybdenum (originally from Harrower1956)
Part-nSpecies	1	Number of species used in calculation

Particle		
Part-nMacroRestartFiles	0	Number of Restart files used for calculation
Part-MacroRestartFile[$]	none	relative path to Restart file [$] used for calculation
Part-DoInitialIonization	F	When restarting from a state, ionize the species to a specific degree
InitialIonizationSpecies		Supply the number of species that are considered for automatic ionization
InitialIonizationSpeciesID		Supply a vector with the species IDs that are used for the initial ionization.
InitialIonizationChargeAverage		Average charge for each atom/molecule in the cell (corresponds to the ionization degree)
Part-MaxParticleNumber	1	Maximum number of Particles per proc (used for array init)
Particles-dt_part_ratio	3.8	TODO-DEFINE-PARAMETER Factors for td200/201 overrelaxation/subcycling
Particles-overrelax_factor	1.0	TODO-DEFINE-PARAMETER Factors for td200/201 overrelaxation/subcycling
Part-NumberOfRandomSeeds	0	Number of Seeds for Random Number GeneratorChoose nRandomSeeds =-1 Random = 0 Debugging-friendly with hard-coded deterministic numbers > 0 Debugging-friendly with numbers from ini.
Particles-RandomSeed[$]	1	Seed [$] for Random Number Generator
Particles-DoPoissonRounding	F	TODO-DEFINE-PARAMETER Flag to perform Poisson sampling instead of random rounding

Particle		
Particles-DoTimeDepInflow	F	TODO-DEFINE-PARAMETER Insertion and SurfaceFlux with simple random rounding. Linearly ramping of inflow-number-of-particles is only possible with PoissonRounding or DoTimeDepInflow
Part-nPeriodicVectors	0	TODO-DEFINE-PARAMETER Number of the periodic vectors j=1,. . . ,n. Value has to be the same as defined in preprog.ini
Part-PeriodicVector[$]	(/ 1.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER Vector for periodic boundaries.Has to be the same as defined in preproc.ini in their respective order.
Part-DelayTime	0.0	TODO-DEFINE-PARAMETER During delay time the particles, won't be moved so the EM field can be evolved
Particles-OutputVpiWarnings	F	TODO-DEFINE-PARAMETER Flag for warnings for rejected v if VPI+PartDensity
Part-SafetyFactor	1.0	TODO-DEFINE-PARAMETER Factor to scale the halo region with MPI
Particles-HaloEpsVelo	0.0	TODO-DEFINE-PARAMETER Halo region radius
NbrOfRegions	0	TODO-DEFINE-PARAMETER Number of regions to be mapped to Elements

Particle		
RegionBounds[$]	(/ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER RegionBounds ((xmin,xmax,ymin,...)|1:NbrOfRegions)
Part-RegionElectronRef[$]	(/ 0.0, 0.0, 1.0 /)	rho_ref, phi_ref, and Te[eV] for Region#
Part-RegionElectronRef[$]-PhiMax		max. expected phi for Region# (linear approx. above! def.: phi_ref)
Part-LorentzType	3	TODO-DEFINE-PARAMETER Used Lorentz boost
PrintrandomSeeds	F	Flag defining if random seeds are written.
Particles-NumberOfRandomVectors	100000	Option defining how many random vectors are calculated
Part-DoFieldIonization	F	Do Field Ionization. Implemented models are: * Ammosov-Delone-Krainov (ADK) model

IMD		
Variable	**Default**	**Description**
IMDTimeScale	0.1018E-13	Time unit of input file. The default value is ~10.18 fs which comes from the unit system in IMD
IMDLengthScale	0.10E-09	Length unit scale used by IMD which is 1 angstrom
IMDAtomFile	no file found	IMD data file containing the atomic states for PartState(1:6)
IMDCutOff	no_cutoff	Atom cut-off parameter for reducing the number of improrted IMD particles 1.) no_cutoff 2.) Epot 3.) coordinates 4.) velocity
IMDCutOffxValue	-999.9	Cut-off coordinate for IMDCutOff='coordiantes'
IMDnSpecies	1	Count of IMD species

IMD		
IMDInputFile	no file found	Laser data file name containing PartState(1:6)

VMPF		
Variable	**Default**	**Description**
Part-vMPF	F	TODO-DEFINE-PARAMETER Flag to use variable Macro Particle Factor.
Part-vMPFPartMerge	F	TODO-DEFINE-PARAMETER Enable Particle Merge routines.
Part-vMPFMergePolOrder	2	TODO-DEFINE-PARAMETER Polynomial degree for vMPF particle merge.
Part-vMPFCellSplitOrder	15	TODO-DEFINE-PARAMETER Order for cell splitting of variable MPF
Part-vMPFMergeParticleTarget	0	TODO-DEFINE-PARAMETER Count of particles wanted after merge.
Part-vMPFSplitParticleTarget	0	TODO-DEFINE-PARAMETER Number of particles wanted after split.
Part-vMPFMergeParticleIter	100	TODO-DEFINE-PARAMETER Number of iterations between particle merges.
Part-vMPFvelocityDistribution	OVDR	TODO-DEFINE-PARAMETER Velocity distribution for variable MPF.
Part-vMPFrelativistic	F	TODO-DEFINE-PARAMETER

Particle Sampling		
Variable	**Default**	**Description**

Particle Sampling		
Part-WriteMacroValues	F	Set [T] to activate ITERATION DEPENDANT h5 output of macroscopic values sampled every [Part-IterationForMacroVal] iterations from particles. Sampling starts from simulation start. Can not be enabled together with Part-TimeFracForSampling. (HALOWIKI:)Write macro values (e.g. rotational Temperature).
Part-WriteMacroVolumeValues	F	Similar to Part-WriteMacroValues. Set [T] to activate iteration dependant sampling and h5 output for each element. Is automatically set true if Part-WriteMacroValues is true. Can not be enabled if Part-TimeFracForSampling is set.
Part-WriteMacroSurfaceValues	F	Similar to Part-WriteMacroValues. Set [T] to activate iteration dependant sampling and h5 output on surfaces. Is automatically set true if Part-WriteMacroValues is true. Can not be enbaled if Part-TimeFracForSampling is set.
Part-IterationForMacroVal	1	Set number of iterations used for sampling if Part-WriteMacroValues is set true.

Particle Sampling		
Part-TimeFracForSampling	0.0	Set value greater 0.0 to enable TIME DEPENDANT sampling. The given simulation time fraction will be sampled. Sampling starts after TEnd*(1-Part-TimefracForSampling). Can not be enabled together with Part-WriteMacroValues.
Particles-NumberForDSMCOutputs	0	Give the number of outputs for time fraction sampling. Default value is 1 if Part-TimeFracForSampling is enabled.
Particles-DSMC-CalcSurfaceVal	F	Set [T] to activate sampling, analyze and h5 output for surfaces. Therefore either time fraction or iteration sampling have to be enabled as well.
DSMC-HOSampling-Type	cell_mean	TODO-DEFINE-PARAMETER
Particles-DSMC-OutputOrder	1	TODO-DEFINE-PARAMETER
DSMC-HOSampling-NodeType	visu	TODO-DEFINE-PARAMETER
DSMCSampVolWe-BGMdeltas	(/ 0.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER
DSMCSampVolWe-FactorBGM	(/ 1.0, 1.0, 1.0 /)	TODO-DEFINE-PARAMETER
DSMCSampVolWe-VolIntOrd	50	TODO-DEFINE-PARAMETER
DSMC-nSurfSample	1	Define polynomial degree of particle BC sampling. Default: NGeo

Particle SurfCollis		
Variable	**Default**	**Description**

Particle SurfCollis		
Particles-CalcSurfCollis_OnlySwaps	F	TODO-DEFINE-PARAMETER Count only wall collisions being SpeciesSwaps
Particles-CalcSurfCollis_Only0Swaps	F	TODO-DEFINE-PARAMETER Count only wall collisions being delete-SpeciesSwaps
Particles-CalcSurfCollis_Output	F	TODO-DEFINE-PARAMETER Print sums of all counted wall collisions
Particles-AnalyzeSurfCollis	F	TODO-DEFINE-PARAMETER Output of collided/swaped particles during Sampling period?
Particles-DSMC-maxSurfCollisNumber	0	TODO-DEFINE-PARAMETER Max. number of collided/swaped particles during Sampling
Particles-DSMC-NumberOfBCs	1	TODO-DEFINE-PARAMETER Count of BC to be analyzed
Particles-DSMC-SurfCollisBC		BCs to be analyzed (def.: 0 = all)
Particles-CalcSurfCollis_NbrOfSpecies	0	TODO-DEFINE-PARAMETER Count of Species for wall collisions (0: all)
Particles-CalcSurfCollis_Species		TODO-DEFINE-PARAMETER Help array for reading surface stuff
Part-WriteFieldsToVTK	F	TODO-DEFINE-PARAMETER Not in Code anymore, but read-in has to be deleted in particle_init.f90
Part-ConstPressAddParts	T	TODO-DEFINE-PARAMETER
Part-ConstPressRemParts	F	TODO-DEFINE-PARAMETER

Particle Species

Variable	Default	Description
Part-Species[$]-nInits	0	Number of different initial particle placements for Species [$]
Part-Species[$]-Reset	F	Flag for resetting species distribution with init during restart
Part-Species[$]-ChargeIC	0.0	[TODO-DEFINE-PARAMETER] Particle Charge (without MPF) of species[$] dim
Part-Species[$]-MassIC	0.0	Particle Mass (without MPF) of species [$] [kg]
Part-Species[$]-MacroParticleFactor	1.0	Number of Microparticle per Macroparticle for species [$]
Part-Species[$]-IsImplicit	F	TODO-DEFINE-PARAMETER Flag if specific particle is implicit
Part-Species[$]-UseForInit	T	Flag to use species[$] for initialization.
Part-Species[$]-UseForEmission	T	Use species[$] for volume emission. (set EmissionType)
Part-Species[$]-SpaceIC	cuboid	Specifying Keyword for particle space condition of species [$] in case of one init. - point - line_with_equidistant_distribution - line - disc - gyrotron_circle - circle_equidistant - cuboid - cylinder - cuboid_vpi - cylinder_vpi - LD_insert - cell_local - cuboid_equal - cuboid_with_equidistant_distribution - sin_deviation - IMD

Particle Species		
Part-Species[$]-velocityDistribution	constant	Used velocity distribution. constant: all particles have the same defined velocity.(VeloIC, VeloVec) maxwell: sampled from maxwell distribution.(for MWTemperatureIC) maxwell_lpn: maxwell with low particle number (better maxwell dist. approx. for lpn).
Part-Species[$]-rotation	1	TODO-DEFINE-PARAMETER Direction of rotation, similar to TE-mode
Part-Species[$]-velocityspread	0.0	TODO-DEFINE-PARAMETER Velocity spread in percent
Part-Species[$]-velocityspreadmethod	0	TODO-DEFINE-PARAMETER Method to compute the velocity spread
Part-Species[$]-InflowRiseTime	0.0	TODO-DEFINE-PARAMETER Time to ramp the number of inflow particles,linearly from zero to unity
Part-Species[$]-initialParticleNumber	0	TODO-DEFINE-PARAMETER Initial particle number
Part-Species[$]-RadiusIC	1.0	TODO-DEFINE-PARAMETER Radius for IC circle
Part-Species[$]-Radius2IC	0.0	TODO-DEFINE-PARAMETER Radius for IC cylinder (ring)
Part-Species[$]-RadiusICGyro	1.0	TODO-DEFINE-PARAMETER Gyrotron radius
Part-Species[$]-NormalIC	(/ 0.0, 0.0, 1.0 /)	TODO-DEFINE-PARAMETER Normal orientation of circle.

Particle Species		
Part-Species[$]-BasePointIC	(/ 0.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER Base point for IC cuboid and IC sphere
Part-Species[$]-BaseVector1IC	(/ 1.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER First base vector for IC cuboid
Part-Species[$]-BaseVector2IC	(/ 0.0, 1.0, 0.0 /)	TODO-DEFINE-PARAMETER Second base vector for IC cuboid
Part-Species[$]-CuboidHeightIC	1.0	TODO-DEFINE-PARAMETER Height of cuboid if SpaceIC=cuboid
Part-Species[$]-CylinderHeightIC	1.0	TODO-DEFINE-PARAMETER Height of cylinder if SpaceIC=cylinder
Part-Species[$]-CalcHeightFromDt	F	TODO-DEFINE-PARAMETER Calculated cuboid/cylinder height from v and dt?
Part-Species[$]-VeloIC	0.0	Absolute value of initial velocity. (ensemble velocity)
Part-Species[$]-VeloVecIC	(/ 0.0, 0.0, 0.0 /)	Normalized velocity vector for given VeloIC
Part-Species[$]-Amplitude	0.01	TODO-DEFINE-PARAMETER Amplitude for sin-deviation
Part-Species[$]-WaveNumber	2.0	TODO-DEFINE-PARAMETER Wave number for sin-deviation
Part-Species[$]-maxParticleNumber-x	0	TODO-DEFINE-PARAMETER MaximumNumber of all particles in x-direction
Part-Species[$]-maxParticleNumber-y	0	TODO-DEFINE-PARAMETER MaximumNumber of all particles in y-direction
Part-Species[$]-maxParticleNumber-z	0	TODO-DEFINE-PARAMETER MaximumNumber of all particles in z-direction

Particle Species		
Part-Species[$]-Alpha	0.0	TODO-DEFINE-PARAMETER Factor for normal speed in gyrotron simulations.
Part-Species[$]-MWTemperatureIC	0.0	Initial translational temperature for Maxwell distribution initialization.
Part-Species[$]-ConstantPressure	0.0	TODO-DEFINE-PARAMETER Pressure for an area with constant pressure
Part-Species[$]-ConstPressureRelaxFac	1.0	TODO-DEFINE-PARAMETER Relaxation Factor for constant pressure sampling.
Part-Species[$]-PartDensity	0.0	Define particle density for species [$]. PartDensity (real particles per m^3). Used for DSMC with (vpi_)cuboid/cylinder and cell_local initial inserting. Also for LD_insert or (vpi_)cub./cyl. / cell_local as alternative to Part.Emis. in Type1
Part-Species[$]-ParticleEmissionType	2	Define Emission Type for particles (volume emission) 1 = emission rate in part/s, 2 = emission rate part/iteration 3 = user def. emission rate 4 = const. cell pressure 5 = cell pres. w. complete part removal 6 = outflow BC (characteristics method)
Part-Species[$]-ParticleEmission	0.0	Emission rate in part/s or part/iteration.
Part-Species[$]-NSigma	10.0	TODO-DEFINE-PARAMETER Sigma multiple of maxwell for virtual insert length.

Particle Species		
Part-Species[$]-NumberOfExcludeRegions	0	TODO-DEFINE-PARAMETER Number of different regions to be excluded
Part-Species[$]-MJxRatio	0.0	TODO-DEFINE-PARAMETER x direction portion of velocity for Maxwell-Juettner
Part-Species[$]-MJyRatio	0.0	TODO-DEFINE-PARAMETER y direction portion of velocity for Maxwell-Juettner
Part-Species[$]-MJzRatio	0.0	TODO-DEFINE-PARAMETER z direction portion of velocity for Maxwell-Juettner
Part-Species[$]-WeibelVeloPar	0.0	TODO-DEFINE-PARAMETER Parallel velocity component for Weibel
Part-Species[$]-WeibelVeloPer	0.0	TODO-DEFINE-PARAMETER Perpendicular velocity component for Weibel
Part-Species[$]-OneDTwoStreamVelo	0.0	TODO-DEFINE-PARAMETER Stream Velocity for the Two Stream Instability
Part-Species[$]-OneDTwoStreamTransRatio	0.0	TODO-DEFINE-PARAMETER Ratio between perpendicular and parallel velocity
Part-Species[$]-vpiDomainType	perpendicular_extrusion	TODO-DEFINE-PARAMETER Specifying Keyword for virtual Pre-Inserting region implemented: - perpendicular_extrusion (default) - freestream - orifice - . . . more following. . .

Particle Species		
Part-Species[$]-vpiBV1BufferNeg	T	TODO-DEFINE-PARAMETER incl. buffer region in -BV1 direction?
Part-Species[$]-vpiBV1BufferPos	T	TODO-DEFINE-PARAMETER incl. buffer region in +BV1 direction?
Part-Species[$]-vpiBV2BufferNeg	T	TODO-DEFINE-PARAMETER incl. buffer region in -BV2 direction?
Part-Species[$]-vpiBV2BufferPos	T	TODO-DEFINE-PARAMETER incl. buffer region in +BV2 direction?
Part-Species[$]-IsIMDSpecies	F	TODO-DEFINE-PARAMETER
Part-Species[$]-MacroRestartFileID	0	Define File ID of file used for Elem specific cell_local init of all macroscopic values
Part-Species[$]-ElemTemperatureFileID		Define File ID of file used for Elem specific cell_local init of translational temperature. (x,y,z are used from State) DEFAULT: MacroRestartFileID
Part-Species[$]-ElemPartDensityFileID		Define File ID of file used for Elem specific cell_local init of number density. DEFAULT: MacroRestartFileID
Part-Species[$]-ElemVelocityICFileID		Define File ID of file used for Elem specific cell_local init of drift velocity. (x,y,z are used from State) DEFAULT: MacroRestartFileID
Part-Species[$]-ElemTVibFileID		Define File ID of file used for Elem specific cell_local init of vibrational temperature. DEFAULT: MacroRestartFileID only used if DSMC + collismode>1

Particle Species	
Part-Species[$]-ElemTRotFileID	Define File ID of file used for Elem specific cell_local init of rotational temperature. DEFAULT: MacroRestartFileID only used if DSMC + collismode>1
Part-Species[$]-ElemTElecFileID	Define File ID of file used for Elem specific cell_local init of electronic temperature. DEFAULT: MacroRestartFileID only used if DSMC + collismode>1 + electronicmodel

Particle Species Ninits		
Variable	**Default**	**Description**
Part-Species[$]-Init[$]-UseForInit	T	TODO-DEFINE-PARAMETER Flag to use Init/Emission for init
Part-Species[$]-Init[$]-UseForEmission	F	TODO-DEFINE-PARAMETER Flag to use Init/Emission for emission
Part-Species[$]-Init[$]-SpaceIC	cuboid	Specifying Keyword for particle space condition of species [$] in case of multiple inits
Part-Species[$]-Init[$]-velocityDistribution	constant	TODO-DEFINE-PARAMETER Specifying keyword for velocity distribution
Part-Species[$]-Init[$]-rotation	1	TODO-DEFINE-PARAMETER Direction of rotation, similar to TE-mode
Part-Species[$]-Init[$]-velocityspread	0.0	TODO-DEFINE-PARAMETER Velocity spread in percent
Part-Species[$]-Init[$]-velocityspreadmethod	0	TODO-DEFINE-PARAMETER Method to compute the velocity spread

Particle Species Ninits		
Part-Species[$]-Init[$]-InflowRiseTime	0.0	TODO-DEFINE-PARAMETER Time to ramp the number of inflow particles linearly from zero to unity
Part-Species[$]-Init[$]-initialParticleNumber	0	TODO-DEFINE-PARAMETER Number of Particles at time 0.0
Part-Species[$]-Init[$]-RadiusIC	1.0	TODO-DEFINE-PARAMETER Radius for IC circle
Part-Species[$]-Init[$]-Radius2IC	0.0	TODO-DEFINE-PARAMETER Radius2 for IC cylinder (ring)
Part-Species[$]-Init[$]-RadiusICGyro	1.0	TODO-DEFINE-PARAMETER Radius for Gyrotron gyro radius
Part-Species[$]-Init[$]-NormalIC	(/ 0.0, 0.0, 1.0 /)	TODO-DEFINE-PARAMETER Normal / Orientation of circle
Part-Species[$]-Init[$]-BasePointIC	(/ 0.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER Base point for IC cuboid and IC sphere
Part-Species[$]-Init[$]-BaseVector1IC	(/ 1.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER First base vector for IC cuboid
Part-Species[$]-Init[$]-BaseVector2IC	(/ 0.0, 1.0, 0.0 /)	TODO-DEFINE-PARAMETER Second base vector for IC cuboid
Part-Species[$]-Init[$]-CuboidHeightIC	1.0	TODO-DEFINE-PARAMETER Height of cuboid if SpaceIC = cuboid. (set 0 for flat rectangle),negative value = opposite direction
Part-Species[$]-Init[$]-CylinderHeightIC	1.0	TODO-DEFINE-PARAMETER Third measure of cylinder (set 0 for flat rectangle), negative value = opposite direction

Particle Species Ninits		
Part-Species[$]-Init[$]-CalcHeightFromDt	F	TODO-DEFINE-PARAMETER Calculate cuboid/cylinder height from v and dt?
Part-Species[$]-Init[$]-VeloIC	0.0	TODO-DEFINE-PARAMETER Velocity for inital Data
Part-Species[$]-Init[$]-VeloVecIC	(/ 0.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER Normalized velocity vector
Part-Species[$]-Init[$]-Amplitude	0.01	TODO-DEFINE-PARAMETER Amplitude for sin-deviation initiation.
Part-Species[$]-Init[$]-WaveNumber	2.0	TODO-DEFINE-PARAMETER WaveNumber for sin-deviation initiation
Part-Species[$]-Init[$]-maxParticleNumber-x	0	TODO-DEFINE-PARAMETER Maximum Number of all Particles in x direction
Part-Species[$]-Init[$]-maxParticleNumber-y	0	TODO-DEFINE-PARAMETER Maximum Number of all Particles in y direction
Part-Species[$]-Init[$]-maxParticleNumber-z	0	TODO-DEFINE-PARAMETER Maximum Number of all Particles in z direction
Part-Species[$]-Init[$]-Alpha	0.0	TODO-DEFINE-PARAMETER WaveNumber for sin-deviation initiation.
Part-Species[$]-Init[$]-MWTemperatureIC	0.0	TODO-DEFINE-PARAMETER Temperature for Maxwell Distribution
Part-Species[$]-Init[$]-ConstantPressure	0.0	TODO-DEFINE-PARAMETER Pressure for an Area with a Constant Pressure
Part-Species[$]-Init[$]-ConstPressureRelaxFac	1.0	TODO-DEFINE-PARAMETER Relaxation Factor for constant pressure sampling.

Particle Species Ninits		
Part-Species[$]-Init[$]-PartDensity	0.0	TODO-DEFINE-PARAMETER PartDensity (real particles per m^3) for LD_insert or (vpi_)cub./cyl. as alternative to Part.Emis. in Type1
Part-Species[$]-Init[$]-ParticleEmissionType	2	TODO-DEFINE-PARAMETER Emission Type 1 = emission rate in 1/s, 2 = emission rate 1/iteration 3 = user def. emission rate 4 = const. cell pressure 5 = cell pres. w. complete part removal 6 = outflow BC (characteristics method)
Part-Species[$]-Init[$]-ParticleEmission	0.0	TODO-DEFINE-PARAMETER Emission in [1/s] or [1/Iteration]
Part-Species[$]-Init[$]-NSigma	10.0	TODO-DEFINE-PARAMETER Sigma multiple of maxwell for virtual insert length
Part-Species[$]-Init[$]-NumberOfExcludeRegions	0	TODO-DEFINE-PARAMETER Number of different regions to be excluded
Part-Species[$]-Init[$]-MJxRatio	0.0	TODO-DEFINE-PARAMETER x direction portion of velocity for Maxwell-Juettner
Part-Species[$]-Init[$]-MJyRatio	0.0	TODO-DEFINE-PARAMETER y direction portion of velocity for Maxwell-Juettner
Part-Species[$]-Init[$]-MJzRatio	0.0	TODO-DEFINE-PARAMETER z direction portion of velocity for Maxwell-Juettner
Part-Species[$]-Init[$]-WeibelVeloPar	0.0	TODO-DEFINE-PARAMETER Parallel velocity component for Weibel

Particle Species Ninits		
Part-Species[$]-Init[$]-WeibelVeloPer	0.0	TODO-DEFINE-PARAMETER Perpendicular velocity component for Weibel
Part-Species[$]-Init[$]-OneDTwoStreamVelo	0.0	TODO-DEFINE-PARAMETER Stream Velocity for the Two Stream Instability
Part-Species[$]-Init[$]-OneDTwoStreamTransRatio	0.0	TODO-DEFINE-PARAMETER Ratio between perpendicular and parallel velocity
Part-Species[$]-Init[$]-vpiDomainType	perpendicular_extrusion	TODO-DEFINE-PARAMETER Specifying Keyword for virtual Pre-Inserting region implemented: - perpendicular_extrusion (default) - freestream - orifice - . . . more following. . .
Part-Species[$]-Init[$]-vpiBV1BufferNeg	T	TODO-DEFINE-PARAMETER incl. buffer region in -BV1 direction?
Part-Species[$]-Init[$]-vpiBV1BufferPos	T	TODO-DEFINE-PARAMETER incl. buffer region in +BV1 direction?
Part-Species[$]-Init[$]-vpiBV2BufferNeg	T	TODO-DEFINE-PARAMETER incl. buffer region in -BV2 direction?
Part-Species[$]-Init[$]-vpiBV2BufferPos	T	TODO-DEFINE-PARAMETER incl. buffer region in +BV2 direction?
Part-Species[$]-Init[$]-MacroRestartFileID	0	Define File ID of file used for Elem specific cell_local init of all macroscopic values
Part-Species[$]-Init[$]-ElemTemperatureFileID		Define File ID of file used for Elem specific cell_local init of translational temperature. (x,y,z are used from State) DEFAULT: MacroRestartFileID

Particle Species Ninits		
Part-Species[$]-Init[$]-ElemPartDensityFileID		Define File ID of file used for Elem specific cell_local init of number density. DEFAULT: MacroRestartFileID
Part-Species[$]-Init[$]-ElemVelocityICFileID		Define File ID of file used for Elem specific cell_local init of drift velocity. (x,y,z are used from State) DEFAULT: MacroRestartFileID
Part-Species[$]-Init[$]-ElemTVibFileID		Define File ID of file used for Elem specific cell_local init of vibrational temperature. DEFAULT: MacroRestartFileID only used if DSMC + collismode>1
Part-Species[$]-Init[$]-ElemTRotFileID		Define File ID of file used for Elem specific cell_local init of rotational temperature. DEFAULT: MacroRestartFileID only used if DSMC + collismode>1
Part-Species[$]-Init[$]-ElemTElecFileID		Define File ID of file used for Elem specific cell_local init of electronic temperature. DEFAULT: MacroRestartFileID only used if DSMC + collismode>1 + electronicmodel

Particle Species Init RegionExculdes		
Variable	**Default**	**Description**
Part-Species[$]-Init[$]-ExcludeRegion[$]-SpaceIC	cuboid	TODO-DEFINE-PARAMETER Specified keyword for excluded particle space condition of species[$] in case of multiple inits
Part-Species[$]-Init[$]-ExcludeRegion[$]-RadiusIC	1.0	TODO-DEFINE-PARAMETER Radius for excluded IC circle

Particle Species Init RegionExculdes		
Part-Species[$]-Init[$]-ExcludeRegion[$]-Radius2IC	0.0	TODO-DEFINE-PARAMETER Radius2 for excluded IC cylinder (ring)
Part-Species[$]-Init[$]-ExcludeRegion[$]-NormalIC	(/ 0.0, 0.0, 1.0 /)	TODO-DEFINE-PARAMETER Normal orientation of excluded circle
Part-Species[$]-Init[$]-ExcludeRegion[$]-BasePointIC	(/ 0.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER Base point for excluded IC cuboid and IC sphere
Part-Species[$]-Init[$]-ExcludeRegion[$]-BaseVector1IC	(/ 1.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER First base vector for excluded IC cuboid
Part-Species[$]-Init[$]-ExcludeRegion[$]-BaseVector2IC	(/ 0.0, 1.0, 0.0 /)	TODO-DEFINE-PARAMETER Second base vector for excluded IC cuboid
Part-Species[$]-Init[$]-ExcludeRegion[$]-CuboidHeightIC	1.0	TODO-DEFINE-PARAMETER Height of excluded cuboid, if Part-Species[$]-Init[$]-ExcludeRegion[$]-SpaceIC=cuboid (set 0 for flat rectangle), negative value = opposite direction
Part-Species[$]-Init[$]-ExcludeRegion[$]-CylinderHeightIC	1.0	TODO-DEFINE-PARAMETER Height of excluded cylinder, if Part-Species[$]-Init[$]-ExcludeRegion[$]-SpaceIC=cylinder (set 0 for flat circle),negative value = opposite direction

Particle Boundaries		
Variable	**Default**	**Description**
Part-nBounds	1	TODO-DEFINE-PARAMETER Number of particle boundaries.

Particle Boundaries		
Part-Boundary[$]-NbrOfSpeciesSwaps	0	TODO-DEFINE-PARAMETER Number of Species to be changed at wall.
Part-Boundary[$]-Condition	open	TODO-DEFINE-PARAMETER Used boundary condition for boundary[$]. - open - reflective - periodic - simple_anode - simple_cathode. If condition=open, the following parameters are used: (Part-Boundary[$]-=PB) PB-Ambient ,PB-AmbientTemp,PB-AmbientMeanPartMass,PB-AmbientVelo,PB-AmbientDens,PB-AmbientDynamicVisc,PB-AmbientThermalCond,PB-Voltage If condition=reflective: PB-MomentumACC,PB-WallTemp,PB-TransACC,PB-VibACC,PB-RotACC,PB-WallVelo,Voltage,SpeciesSwaps.If condition=periodic:Part-nPeriodicVectors,Part-PeriodicVector[$]
Part-Boundary[$]-AmbientCondition	F	TODO-DEFINE-PARAMETERUse ambient condition (condition "behind" boundary).
Part-Boundary[$]-AmbientConditionFix	T	TODO-DEFINE-PARAMETER
Part-Boundary[$]-AmbientTemp	0.0	TODO-DEFINE-PARAMETERAmbient temperature
Part-Boundary[$]-AmbientMeanPartMass	0.0	TODO-DEFINE-PARAMETERAmbient mean particle mass

Particle Boundaries		
Part-Boundary[$]-AmbientVelo	(/ 0.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETERAmbient velocity
Part-Boundary[$]-AmbientDens	0.0	TODO-DEFINE-PARAMETERAmbient density
Part-Boundary[$]-AmbientDynamicVisc	0.172326582572253E-04	TODO-DEFINE-PARAMETERAmbient dynamic viscosity
Part-Boundary[$]-AmbientThermalCond	0.242948500556027E-01	TODO-DEFINE-PARAMETERAmbient thermal conductivity
Part-Boundary[$]-Adaptive	F	Define if particle boundary [$] is adaptive [.TRUE.] or not [.FALSE.]
Part-Boundary[$]-AdaptiveType	2	Define type of adaptive boundary [$] [1] (STREAM INLET) with define temperature and pressure and pressurefraction [2] (STREAM OUTLET) with defined pressure and pressurefraction
Part-Boundary[$]-AdaptiveMacroRestartFileID	0	Define FileID of adaptive boundary [$] macro restart if macro restart is used
Part-Boundary[$]-AdaptiveTemp	0.0	Define temperature for adaptive particle boundary [$] (in [K])
Part-Boundary[$]-AdaptivePressure	0.0	Define pressure for adaptive particle boundary [$] (in [Pa])
Part-Boundary[$]-Species[$]-Pressurefraction	0.0	If particle boundary [$] adaptive, define pressurefractions for each species, so sum of all species for this adaptiveis 1.0. Results in abort if not set right.
Part-Boundary[$]-Voltage	0.0	TODO-DEFINE-PARAMETERVoltage on boundary [$]

Particle Boundaries		
Part-Boundary[$]-WallTemp	0.0	Wall temperature (in [K]) of reflective particle boundary [$].
Part-Boundary[$]-MomentumACC	0.0	Momentum accommodation coefficient of reflective particle boundary [$].
Part-Boundary[$]-TransACC	0.0	Translation accommodation coefficient of reflective particle boundary [$].
Part-Boundary[$]-VibACC	0.0	Vibrational accommodation coefficient of reflective particle boundary [$].
Part-Boundary[$]-RotACC	0.0	Rotational accommodation coefficient of reflective particle boundary [$].
Part-Boundary[$]-ElecACC	0.0	Electronic accommodation coefficient of reflective particle boundary [$].
Part-Boundary[$]-Resample	F	TODO-DEFINE-PARAMETERResample Equilibrum Distribution with reflection
Part-Boundary[$]-WallVelo	(/ 0.0, 0.0, 0.0 /)	Velocity (global x,y,z in [m/s]) of reflective particle boundary [$].
Part-Boundary[$]-SolidState	T	Flag defining if reflective BC is solid [TRUE] or liquid [FALSE].
Part-Boundary[$]-SolidReactive	F	Flag for defining solid surface to be treated catalytically (for surfacemodel>0).
Part-Boundary[$]-SolidSpec	0	Set Species of Solid Boundary. (currently not used)
Part-Boundary[$]-SolidPartDens	0.10E+20	If particle boundary defined as solid set surface atom density (in [part/m^2]).
Part-Boundary[$]-SolidMassIC	0.32395E-24	Set mass of solid surface particles (in [kg]).
Part-Boundary[$]-SolidAreaIncrease	1.0	TODO-DEFINE-PARAMETER
Part-Boundary[$]-SolidCrystalIndx	4	Set number of interaction for hollow sites.

Particle Boundaries		
Part-Boundary[$]-LiquidSpec	0	Set used species of Liquid Boundary
Part-Boundary[$]-ParamAntoine	(/ 0.0, 0.0, 0.0 /)	Parameters for Antoine Eq (vapor pressure)
Part-Boundary[$]-ProbOfSpeciesSwaps	1.0	TODO-DEFINE-PARAMETERProbability of SpeciesSwaps at wall
Part-Boundary[$]-SpeciesSwaps[$]	(/ 0, 0 /)	TODO-DEFINE-PARAMETERSpecies to be changed at wall (out=: delete)
Part-Boundary[$]-SourceName		TODO-DEFINE-PARAMETERNo Default. Source Name of Boundary[i]. Has to be selected for allnBounds. Has to be same name as defined in preproc tool
Part-Boundary[$]-UseForQCrit	T	TODO-DEFINE-PARAMETERFlag to use Boundary for Q-Criterion
Part-nAuxBCs	0	TODO-DEFINE-PARAMETERNumber of auxillary BCs that are checked during tracing
Part-AuxBC[$]-NbrOfSpeciesSwaps	0	TODO-DEFINE-PARAMETERNumber of Species to be changed at wall.
Part-AuxBC[$]-Condition	open	TODO-DEFINE-PARAMETERUsed auxillary boundary condition for boundary[$].- open- reflective- periodic)-> more details see also Part-Boundary[$]-Condition
Part-AuxBC[$]-MomentumACC	0.0	TODO-DEFINE-PARAMETERMomentum accommodation
Part-AuxBC[$]-WallTemp	0.0	TODO-DEFINE-PARAMETERWall temperature of boundary[$]

Particle Boundaries		
Part-AuxBC[$]-TransACC	0.0	TODO-DEFINE-PARAMETERTranslation accommodation on boundary [$]
Part-AuxBC[$]-VibACC	0.0	TODO-DEFINE-PARAMETERVibrational accommodation on boundary [$]
Part-AuxBC[$]-RotACC	0.0	TODO-DEFINE-PARAMETERRotational accommodation on boundary [$]
Part-AuxBC[$]-ElecACC	0.0	TODO-DEFINE-PARAMETERElectronic accommodation on boundary [$]
Part-AuxBC[$]-Resample	F	TODO-DEFINE-PARAMETERResample Equilibirum Distribution with reflection
Part-AuxBC[$]-WallVelo	(/ 0.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETEREmitted velocity on boundary [$]
Part-AuxBC[$]-ProbOfSpeciesSwaps	1.0	TODO-DEFINE-PARAMETERProbability of SpeciesSwaps at wall
Part-AuxBC[$]-SpeciesSwaps[$]	(/ 0, 0 /)	TODO-DEFINE-PARAMETERSpecies to be changed at wall (out=: delete)
Part-AuxBC[$]-Type	plane	TODO-DEFINE-PARAMETERType of BC (plane, . . .)
Part-AuxBC[$]-r_vec	(/ 0.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER
Part-AuxBC[$]-radius		TODO-DEFINE-PARAMETER
Part-AuxBC[$]-n_vec	(/ 1.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER
Part-AuxBC[$]-axis	(/ 1.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER

Particle Boundaries		
Part-AuxBC[$]-lmin		TODO-DEFINE-PARAMETER
Part-AuxBC[$]-lmax		TODO-DEFINE-PARAMETER
Part-AuxBC[$]-inwards	T	TODO-DEFINE-PARAMETER
Part-AuxBC[$]-rmax	0.0	TODO-DEFINE-PARAMETER
Part-AuxBC[$]-halfangle	45.0	TODO-DEFINE-PARAMETER
Part-AuxBC[$]-zfac	1.0	TODO-DEFINE-PARAMETER

Tracking		
Variable	**Default**	**Description**
DoRefMapping	T	Refmapping [T] or Tracing [F] algorithms are used for tracking of particles.
TriaTracking	F	Using Triangle-aproximation [T] or (bi-)linear and bezier (curved) description [F] of sides for tracing algorithms. Currently flag is only used in DSMC timediscs. Requires DoRefMapping=F.
Write-Tria-DebugMesh	F	Writes per proc triangulated Surfacemesh used for Triatracking. Requires TriaTracking=T.
TriaSurfaceFlux		Using Triangle-aproximation [T] or (bi-)linear and bezier (curved) description [F] of sides for surfaceflux. Default is set to TriaTracking
Write-TriaSurfaceFlux-DebugMesh	F	Writes per proc triangulated Surfacemesh used for TriaSurfaceFlux. Requires TriaSurfaceFlux=T.

Tracking		
CountNbOfLostParts	F	Count number of lost particles during tracking that can not be found with fallbacks.
PartOut	0	If compiled with CODE_ANALYZE flag: For This particle number every tracking information is written as STDOUT.
MPIRankOut	0	If compiled with CODE_ANALYZE flag: This MPI-Proc writes the tracking information for the defined PartOut.
MeasureTrackTime	F	If .TRUE. then the time how long the tracking routines are called are sampled and written for each MPI-Proc.
CartesianPeriodic	F	Simplified treatment for periodic box with Refmapping. Not computation of intersection points at periodic BCs.
FastPeriodic	F	Further simplification by directly moving particle into grid. Instead of moving the particle several times the periodic displacements, the particle is mapped directly back into the domain.
RefMappingGuess		Initial guess of the Newton for mapping the particle into reference coordinates. 1 -linear pseudo-Cartesian coordinates 2 - Xi of closest Gauss point 3 - Xi of closest XCL_ngeo point 4 -trival guess (0,0,0)^t
RefMappingEps	0.1E-03	Tolerance for mapping particle into reference element measured as L2-norm of deltaXi

Tracking		
BezierEpsilonBilinear	0.1E-05	Bi-linear tolerance for the bi-linear - planar decision.
BezierElevation	0	Use BezierElevation>0 to tighten the bounding box. Typical values>10
BezierSampleN	0	TODO-DEFINE-PARAMETER Default value: NGeo equidistant sampling of bezier surface for emission
Part-FIBGMdeltas	(/ 1.0, 1.0, 1.0 /)	Define the deltas for the cartesian Fast-Init-Background-Mesh. They should be of the similar size as the smallest cells of the used mesh for simulation.
Part-FactorFIBGM	(/ 1.0, 1.0, 1.0 /)	Factor with which the background mesh will be scaled.
printMPINeighborWarnings	F	Print warning if the MPI-Halo-region between to procs are not overlapping. Only one proc find the other in halo
printBezierControlPointsWarnings	F	Print warning if MIN-VAL(BezierControlPoints3d(iDir,:,:,newSideID)) and global boundaries are too close
BezierNewtonAngle	1.570796326	BoundingBox intersection angle for switching between Bezierclipping and BezierNewton.
BezierClipTolerance	0.1E-07	Tolerance for BezierClipping
BezierNewtonTolerance	0.1E-03	Tolerance for BezierNewton
BezierNewtonGuess	1	Initial guess for BezierNewton 1 - linear projected face 2 - closest projected BeziercontrolPoint 4 - (0,0)^t
BezierNewtonMaxIter	100	TODO-DEFINE-PARAMETER

Tracking		
BezierSplitLimit	0.6	Limit for splitting in BezierClipping. Value allows to detect multiple intersections and speed up computation. Parameter is multiplied by 2
BezierClipMaxIter	100	Max iteration of BezierClipping
BezierClipLineVectorMethod	2	TODO-DEFINE-PARAMETER
epsilontol	0.0	TODO-DEFINE-PARAMETER
BezierClipHit	0.0	Tolerance in [-1,1] of BezierFace
BezierNewtonHit	0.0	Tolerance in [-1,1] of BezierNewton
BezierClipMaxIntersec		Max. number of multiple intersections. Default: 2*(NGeo+1)

Particle Analyze		
Variable	**Default**	**Description**
Part-AnalyzeStep	1	Analyze is performed each Nth time step
CalcTotalEnergy	F	Calculate Total Energy. Output file is Database.csv
PIC-VerifyCharge	F	Validate the charge after each depositionand write an output in std.out
CalcIonizationDegree	F	Compute the ionization degree in each cell
CalcPointsPerShapeFunction	F	Compute the points per shape function in each cell
CalcPlasmaParameter	F	Compute the plasma parameter N_D in each cell
CalcPointsPerDebyeLength	F	Compute the points per Debye length in each cell
CalcDebyeLength	F	Compute the Debye length in each cell
CalcPICTimeStep	F	Compute the HDG time step in each cell

Particle Analyze		
CalcElectronTemperature	F	Compute the electron temperature in each cell
CalcElectronIonDensity	F	Compute the electron density in each cell
CalcPlasmaFrequency	F	Compute the electron frequency in each cell
CalcCharge	F	TODO-DEFINE-PARAMETER Compute the whole deposited charge, absolute and relative charge error
CalcKineticEnergy	F	TODO-DEFINE-PARAMETER Calculate Kinetic Energy.
CalcInternalEnergy	F	TODO-DEFINE-PARAMETER Calculate Internal Energy.
CalcTemp	F	TODO-DEFINE-PARAMETER Calculate Translational temperature.
CalcPartBalance	F	TODO-DEFINE-PARAMETER Calculate the Particle Power Balance- input and outflow energy of all particles
CalcVelos	F	TODO-DEFINE-PARAMETER Calculate thermal and flow velocities.if CalcVelos = T VelocityDirections = (/[int],[int],[int],[int]/) Switching dimensions for CalcVelos on (1) or off (0) (/v_x,v_y,v_z,|v|/)
CalcLaserInteraction	F	Compute laser-plasma interaction properties such as maximum particle energy per species.

Particle Analyze		
LaserInteractionEkinMaxRadius		maximum radius (x- and y-dir) of particle to be considered for Ekin maximum calculation (default is HUGE) OR if LaserInteractionEkinMaxZPosMin is true
LaserInteractionEkinMaxZPosMin		minimum z-position of particle to be considered for Ekin maximum calculation (default is -1.*HUGE) OR if LaserInteractionEkinMaxRadius is true
VelocityDirections	(/ 1, 1, 1, 1 /)	TODO-DEFINE-PARAMETER x,y,z,abs -> 0/1 = T/F. (please note: CalcVelos)
Part-TrackPosition	F	TODO-DEFINE-PARAMETER Track particle position
printDiff	F	TODO-DEFINE-PARAMETER
printDiffTime	12.0	TODO-DEFINE-PARAMETER
printDiffVec	(/ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER
CalcNumSpec	F	TODO-DEFINE-PARAMETER Calculate species count.
CalcCollRates	F	TODO-DEFINE-PARAMETER Calculate the collision rates per collision pair
CalcReacRates	F	TODO-DEFINE-PARAMETER Calculate the reaction rate per reaction
CalcShapeEfficiency	F	TODO-DEFINE-PARAMETER Use efficiency methods for shape functions.

Particle Analyze		
CalcShapeEfficiencyMethod	AllParts	TODO-DEFINE-PARAMETER Choose between "AllParts" and "SomeParts", to either use all particles or a certain percentage (ShapeEfficiencyNumber) of the currently used particles
ShapeEfficiencyNumber	100	TODO-DEFINE-PARAMETER Percentage of currently used particles is used.
IsRestart	F	TODO-DEFINE-PARAMETER Flag, if the current calculation is a restart.

TTM		
Variable	**Default**	**Description**
DoImportTTMFile	F	Read IMD Two-Temperature Model (TTM) data (FD grid data with electrontemperature and other field data)
TTMLogFile	no file specified	TTM Log file path
TTMFile	no file found	TTM Data file path
TTMGridFDdim	(/ 0, 0, 0 /)	Number of FD grid cells in each direction (/x,y,z/)
TTMElemBaryTolerance	0.1E-05	TTM FD bary center tolerance to DG bary center. The tolerance is used for finding the corresponding DG element to which the FD data is saved.

PIC		
Variable	**Default**	**Description**

PIC		
PIC-Interpolation-Type	particle_position	TODO-DEFINE-PARAMETER Type of Interpolation-Method to calculate the EM field's value for the particle
PIC-InterpolationElemLoop	T	TODO-DEFINE-PARAMETER Interpolate with outer iElem-loop (notfor many Elems per proc!)
PIC-externalField	(/ 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER External field is added to themaxwell-solver-field
PIC-scaleexternalField	1.0	TODO-DEFINE-PARAMETER
PIC-DoInterpolation	T	TODO-DEFINE-PARAMETER Compute the self field's influence on the Particle
PIC-BG-Field	T	TODO-DEFINE-PARAMETER BGField data (1:x,0:NBG,0:NBG,0:NBG,1:PP_nElems) If PIC-BG-Field=T Define: PIC-BGFilename PIC-BGFieldScaling PIC-NBG
PIC-BGFileName	none	TODO-DEFINE-PARAMETER File name for background field ([character].h5)
PIC-NBG	1	TODO-DEFINE-PARAMETER Polynomial degree that shall be used for background field during simulation
PIC-BGFieldScaling	1.0	TODO-DEFINE-PARAMETER Space scaling of background field
PIC-curvedexternalField	none	TODO-DEFINE-PARAMETER File to curved external field data.

PIC		
PIC-variableexternalField	none	TODO-DEFINE-PARAMETER File containing the external field CSV table
PIC-nCollectChargesBCs	0	TODO-DEFINE-PARAMETER
PIC-CollectCharges[$]-BC	0	TODO-DEFINE-PARAMETER
PIC-CollectCharges[$]-NumOfRealCharges	0.0	TODO-DEFINE-PARAMETER
PIC-CollectCharges[$]-ChargeDist	0.0	TODO-DEFINE-PARAMETER
PIC-NormVecOfWall	(/ 1.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER Normal vector for pushTimeStep
PIC-DeltaType	1	TODO-DEFINE-PARAMETER Flag
PIC-DeltaType-N	1	TODO-DEFINE-PARAMETER Polynomial degree of delta distribution
PIC-BGMdeltas	(/ 0.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER Dimensions of PIC background mesh
PIC-FactorBGM	(/ 1.0, 1.0, 1.0 /)	TODO-DEFINE-PARAMETER Denominator of PIC-BGMdeltas
PIC-OutputSource	F	TODO-DEFINE-PARAMETER Writes the source to hdf5

PIC Deposition		
Variable	**Default**	**Description**
PIC-DoDeposition	T	TODO-DEFINE-PARAMETER Switch deposition on/off

PIC Deposition		
PIC-Deposition-Type	nearest-blurrycenter	TODO-DEFINE-PARAMETER (HALOWIKI:) If Deposition-Type=shape_function Define: PIC-shapefunction-radius PIC-shapefunction-alpha. If Deposition-Type =(cartmesh_volumeweighting/ cartmesh_splines) Define: PIC-BGMdeltas PIC-FactorBGM
PIC-TimeAverageFile	none	TODO-DEFINE-PARAMETER
PIC-epanechnikov-radius	1.0	TODO-DEFINE-PARAMETER
PIC-shapefunction-radius	1.0	TODO-DEFINE-PARAMETER Radius of shape function
PIC-shapefunction-alpha	2	TODO-DEFINE-PARAMETER Exponent of shape function
PIC-shapefunction-equi	F	TODO-DEFINE-PARAMETER Use equidistant points for shapefunction
PIC-shapefunction1d-direction	1	TODO-DEFINE-PARAMETER Direction of 1D shape function
PIC-shapefunction-radius0	1.0	TODO-DEFINE-PARAMETER Minimal shape function radius
PIC-shapefunction-scale	0.0	TODO-DEFINE-PARAMETER Scaling factor of shape function radius
PIC-NbrOfSFdepoFixes	0	TODO-DEFINE-PARAMETER Number of fixes for shape func depo at planar BCs
PrintSFDepoWarnings	F	TODO-DEFINE-PARAMETER Print the shapefunction warnings

PIC Deposition		
PIC-SFdepoFixesEps	0.0	TODO-DEFINE-PARAMETER Epsilon for defined planes
PIC-SFdepoFixes[$]-Basepoint	(/ 0.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER
PIC-SFdepoFixes[$]-Normal	(/ 1.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER
PIC-SFdepoFixes[$]-ChargeMult	1.0	TODO-DEFINE-PARAMETER Multiplier for mirrored charges (wall: -1.0, sym: 1.0)
PIC-SFdepoFixes[$]-xmin		TODO-DEFINE-PARAMETER -> SFdepoFixesBounds(:,:,:) 1:nFixes;1:2(min,max);1:3(x,y,z)?
PIC-SFdepoFixes[$]-ymin		TODO-DEFINE-PARAMETER
PIC-SFdepoFixes[$]-zmin		TODO-DEFINE-PARAMETER
PIC-SFdepoFixes[$]-xmax		TODO-DEFINE-PARAMETER
PIC-SFdepoFixes[$]-ymax		TODO-DEFINE-PARAMETER
PIC-SFdepoFixes[$]-zmax		TODO-DEFINE-PARAMETER
PIC-NbrOfSFdepoFixLinks	0	TODO-DEFINE-PARAMETER Number of linked SFdepoFixes
PIC-SFdepoFixLink[$]	(/ 1, 2 /)	TODO-DEFINE-PARAMETER (1:nLinks) 1:3 (2 fixes are linked with each other!) :,3 is fraction of 180 deg
PIC-NbrOfSFdepoLayers	0	TODO-DEFINE-PARAMETER Number of const. source layer for sf-depo at planar BCs
PIC-ConstantSFdepoLayers	F	Do deposition of SFdepoLayers just once
PIC-SFdepoLayers[$]-Basepoint	(/ 0.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER

PIC Deposition		
PIC-SFdepoLayers[$]-Normal	(/ 1.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER
PIC-SFdepoLayers[$]-xmin		TODO-DEFINE-PARAMETER -> SFdepoLayersBounds(:,:,:) 1:nFixes;1:2(min,max);1:3(x,y,z)?
PIC-SFdepoLayers[$]-ymin		TODO-DEFINE-PARAMETER
PIC-SFdepoLayers[$]-zmin		TODO-DEFINE-PARAMETER
PIC-SFdepoLayers[$]-xmax		TODO-DEFINE-PARAMETER
PIC-SFdepoLayers[$]-ymax		TODO-DEFINE-PARAMETER
PIC-SFdepoLayers[$]-zmax		TODO-DEFINE-PARAMETER
PIC-SFdepoLayers[$]-UseFixBounds	T	TODO-DEFINE-PARAMETER Use alls planes of SFdepoFixes as additional bounds
PIC-SFdepoLayers[$]-Space	cuboid	TODO-DEFINE-PARAMETER Name of space (cuboid or cylinder)
PIC-SFdepoLayers[$]-BaseVector1	(/ 0.0, 1.0, 0.0 /)	TODO-DEFINE-PARAMETER Base Vector 1
PIC-SFdepoLayers[$]-BaseVector2	(/ 0.0, 0.0, 1.0 /)	TODO-DEFINE-PARAMETER Base Vector 2
PIC-SFdepoLayers[$]-SFdepoLayersRadius	1.0	TODO-DEFINE-PARAMETER Radius for cylinder-space
PIC-SFdepoLayers[$]-Chargedens	1.0	TODO-DEFINE-PARAMETER
PIC-SFdepoLayers[$]-Spec	1	TODO-DEFINE-PARAMETER Particle species for respective layer
PIC-SFdepoLayers[$]-MPF		MPF for respective layer (def.: MPF of resp. species)
PIC-SFResampleAnalyzeSurfCollis	F	TODO-DEFINE-PARAMETER

PIC Deposition		
PIC-SFResampleSurfCollisBC		TODO-DEFINE-PARAMETER BCs to be analyzed (def.: 0 = all)
PIC-SFResampleReducePartNumber	F	TODO-DEFINE-PARAMETER Reduce PartNumberSamp to PartNumberReduced
PIC-PartNumThreshold	0	TODO-DEFINE-PARAMETER Threshold for checking inserted parts per deposition (otherwise abort)
PIC-SFResampleNumberOfBCs	1	TODO-DEFINE-PARAMETER Number of BC to be analyzed
PIC-SFResamplePartNumberReduced	0	TODO-DEFINE-PARAMETER Max. allowed number of parts to be saved
PIC-SFResampleNbrOfSpeciesForDtCalc	1	TODO-DEFINE-PARAMETER Number of species used for SFResample-dt
PIC-SFResampleSpeciesForDtCalc		TODO-DEFINE-PARAMETER Species used for SFResample-dt (def.: 0 = all)
PIC-SFResampleRestart	F	TODO-DEFINE-PARAMETER Read-in old DSMCSurfCollis-file for restart
PIC-SFResampleRestartFile	dummy	TODO-DEFINE-PARAMETER Name of the new DSMCSurfCollis-file to read-in by restart
PIC-SFResample-xmin		TODO-DEFINE-PARAMETER
PIC-SFResample-ymin		TODO-DEFINE-PARAMETER
PIC-SFResample-zmin		TODO-DEFINE-PARAMETER
PIC-SFResample-xmax		TODO-DEFINE-PARAMETER

PIC Deposition		
PIC-SFResample-ymax		TODO-DEFINE-PARAMETER
PIC-SFResample-zmax		TODO-DEFINE-PARAMETER
PIC-SFResample-UseFixBounds	T	TODO-DEFINE-PARAMETER Use all planes of SFdepoFixes as additional bounds?

Particle Emission		
Variable	**Default**	**Description**
Part-Species[$]-nSurfacefluxBCs	0	TODO-DEFINE-PARAMETER Number of SF emissions
Part-Species[$]-Surfaceflux[$]-BC	0	TODO-DEFINE-PARAMETER PartBound to be emitted from
Part-Species[$]-Surfaceflux[$]-velocityDistribution	constant	TODO-DEFINE-PARAMETER Specifying keyword for velocity distribution
Part-Species[$]-Surfaceflux[$]-VeloIC	0.0	TODO-DEFINE-PARAMETER Velocity for inital Data
Part-Species[$]-Surfaceflux[$]-VeloIsNormal	F	TODO-DEFINE-PARAMETER VeloIC is in Surf-Normal instead of VeloVecIC
Part-Species[$]-Surfaceflux[$]-VeloVecIC	(/ 0.0, 0.0, 0.0 /)	TODO-DEFINE-PARAMETER Normalized velocity vector
Part-Species[$]-Surfaceflux[$]-SimpleRadialVeloFit	F	TODO-DEFINE-PARAMETER Fit of veloR/veloTot=-r(*A*exp(B*r)+C)
Part-Species[$]-Surfaceflux[$]-preFac	0.0	TODO-DEFINE-PARAMETER A , see SimpleRadialVeloFit
Part-Species[$]-Surfaceflux[$]-powerFac	0.0	TODO-DEFINE-PARAMETER B , see SimpleRadialVeloFit

Particle Emission		
Part-Species[$]-Surfaceflux[$]-shiftFac	0.0	TODO-DEFINE-PARAMETER C , see SimpleRadialVeloFit
Part-Species[$]-Surfaceflux[$]-axialDir	1	TODO-DEFINE-PARAMETER Axial direction of coordinates in polar system
Part-Species[$]-Surfaceflux[$]-origin	(/ 0.0, 0.0 /)	TODO-DEFINE-PARAMETER Origin in orth(ogonal?) coordinates of polar system
Part-Species[$]-Surfaceflux[$]-rmax	0.1E+22	TODO-DEFINE-PARAMETER Max radius of to-be inserted particles
Part-Species[$]-Surfaceflux[$]-rmin	0.0	TODO-DEFINE-PARAMETER Min radius of to-be inserted particles
Part-Species[$]-Surfaceflux[$]-MWTemperatureIC	0.0	TODO-DEFINE-PARAMETER Temperature for Maxwell Distribution
Part-Species[$]-Surfaceflux[$]-PartDensity	0.0	TODO-DEFINE-PARAMETER PartDensity (real particles per m^3) for LD_insert or (vpi_)cub./cyl. as alternative to Part.Emis. in Type1
Part-Species[$]-Surfaceflux[$]-ReduceNoise	F	TODO-DEFINE-PARAMETER Reduce stat. noise by global calc. of PartIns
Part-Species[$]-Surfaceflux[$]-AcceptReject	T	TODO-DEFINE-PARAMETER Perform ARM for skewness of RefMap-positioning
Part-Species[$]-Surfaceflux[$]-ARM_DmaxSampleN	1	TODO-DEFINE-PARAMETER Number of sample intervals in xi/eta for Dmax-calc.
DoForceFreeSurfaceFlux	F	TODO-DEFINE-PARAMETER Flag if the stage reconstruction uses a force

Particle Emission		
OutputSurfaceFluxLinked	F	Flag to print the SurfaceFlux-linked Info

DSMC		
Variable	**Default**	**Description**
Particles-DSMC-OutputMeshInit	F	not working currently | Writeoutput mesh for constant pressure BC at initialization.
Particles-DSMC-OutputMeshSamp	F	not working currently | Write output mesh for constant pressure BC with samplingvalues at t_analyze.
Particles-DSMC-CollisMode	1	Define mode of collision handling in DSMC. 0: No Collisions (=free molecular flow with DSMC-Sampling-Routines). 1: Elastic Collision 2: Relaxation + Elastic Collision 3: Mode 2 + Chemical Reactions.
Particles-DSMC-SelectionProcedure	1	Mode of Selection Procedure 1: Laux 2: Gimelsheim.
Particles-DSMC-RotRelaxProb	0.2	Define the rotational relaxation probability upon collision of molecules (HALOWIKI:)Choice of the vibrational relaxation probability calculation 0-1: constant 2: variable, Boyd)
Particles-DSMC-VibRelaxProb	0.02	Define the vibrational relaxation probability upon collision of molecules
Particles-DSMC-ElecRelaxProb	0.01	Define the elextronic relaxation probability upon collision of molecules
Particles-DSMC-GammaQuant	0.5	Set the GammaQuant for zero point energy in Evib (perhaps also Erot) should be 0.5 or 0.

DSMC		
Particles-DSMC-BackwardReacRate	F	Set [TRUE] to enable the automatic calculation of the backward reaction rate coefficientusing the equilibrium constant calculated by partition functions [FALSE] if they are defined as separate reactions.
Particles-DSMC-PartitionMaxTemp	20000.0	Define temperature limit for pre-stored partition function that are used for calculation of backwards rates
Particles-DSMC-PartitionInterval	10.0	Define temperature interval for pre-stored partition functions that are used for calculation of backwards rates
Particles-DSMC-veloMinColl-Spec[$]	0.0	min velo magn. for spec allowed to perform collision
Particles-DSMC-CalcQualityFactors	F	Enables [TRUE] / disables [FALSE] the calculation and output of flow-field variable. Maximal collision probability Time-averaged mean collision probability Mean collision separation distance over mean free path
Particles-DSMCReservoirSim	F	Only TD=Reservoir (42). Set [TRUE] to disable particle movement. Use for reservoir simulations.
Particles-DSMCReservoirSimRate	F	Only TD=Reservoir (42). Set [TRUE] to disable particle reactions.Only probabilities (rates) are calculated.
Particles-DSMCReservoirStatistic	F	Only TD=Reservoir (42). Probabilities (rates) are calculated [TRUE] counting reacting particles. [FALSE] summing reaction probabilities.

DSMC		
Particles-DSMCReservoirSurfaceRate	F	Only TD=Reservoir (42). Set [TRUE] to disable particle adsorption and desorption and keep surface coverage constant. Only probabilities (rates) are calculated.
Particles-ModelForVibrationEnergy	0	Define model used for vibrational degrees of freedom. 0: SHO 1:TSHO.
Particles-DSMC-TEVR-Relaxation	F	Flag for T-V-E-R [TRUE] or more simple T-V-R T-E-R [FALSE] relaxation.
Particles-DSMC-ElectronicModel	F	Set [TRUE] to model electronic states of atoms and molecules.
Particles-DSMCElectronicDatabase	none	If electronic model is used give (relative) path to (h5) Name of Electronic State Database
EpsMergeElectronicState	0.1E-03	Percentage parameter of electronic energy level merging.
Particles-DSMC-UseQCrit	F	Set [TRUE] to enable steady state detection and sampling start using Q-criterion (Burt/Boyd).
Particles-DSMC-UseSSD	F	Set [TRUE] to enable steady state detection and sampling start using 3SD routines.
Particles-DSMCBackgroundGas	0	Define Species number that is used as background gas species
Particles-DSMCBackgroundGasDensity	0.0	Define Species number density for background gas

DSMC		
Particles-DSMC-PolyRelaxSingleMode	F	Set [TRUE] for separate relaxation of each vibrational mode of a polyatomic in a loop over all vibrational modes. Every mode has its own corrected relaxation probability, comparison with the same random number while the previous probability is added to the next
Particles-DSMC-CompareLandauTeller	F	Only TD=Reservoir (42).
Particles-DSMC-UseOctree	F	Use octree method for dynamic grid resolution
Particles-OctreePartNumNode	80	Resolve grid until the maximum number of particles in a subcell equals OctreePartNumNode.
Particles-OctreePartNumNodeMin	50	Allow grid division until the minimum number of particles in a subcell is above OctreePartNumNodeMin.

DSMC Species		
Variable	**Default**	**Description**
Part-Species[$]-SpeciesName	none	Species name of Species[$]
Part-Species[$]-InteractionID	0	ID for identification of particles 1: Atom 2: Molecule 4: Electron 10: Atomic Ion 20: Molecular Ion 40: Excited Atom 100: Excited Atomic Ion 200: Excited Molecule 400: Excited Molecular Ion)
Part-Species[$]-VHSReferenceTemp	0.0	Reference temperature for variable hard sphere model.
Part-Species[$]-VHSReferenceDiam	1.0	Reference diameter for variable hard sphere model.
Part-Species[$]-omegaVHS	0.0	Reference value for exponent omega for variable hard sphere model.

DSMC Species		
Part-Species[$]-CharaTempVib	0.0	Characteristic vibrational temperature.
Part-Species[$]-CharaTempRot	0.0	Characteristic rotational temperature
Part-Species[$]-Ediss_eV	0.0	Energy of Dissoziation in [eV].
Part-Species[$]-VFDPhi3	0.0	Factor of Phi3 in VFD Method: Phi3 = 0 => VFD
Part-Species[$]-CollNumRotInf	0.0	Factor of Phi3 in VFD Method: Phi3 = 0 => VFD -> TCE, ini_2
Part-Species[$]-TempRefRot	0.0	Referece temperature for rotational relaxation according to Parker orZhang, ini_2 -> model dependent!
Part-Species[$]-CollNumVib	0.0	Vibrational collision number according to Boyd, ini_2
Part-Species[$]-TempVib	0.0	Vibrational temperature.
Part-Species[$]-TempRot	0.0	Rotational temperature.
Part-Species[$]-TempElec	0.0	Electronic temperature.
Part-Species[$]-Init[$]-TempVib	0.0	Vibrational temperature.
Part-Species[$]-Init[$]-TempRot	0.0	Rotational temperature.
Part-Species[$]-Init[$]-TempElec	0.0	Electronic temperature.
Part-Species[$]-Surfaceflux[$]-TempVib	0.0	Vibrational temperature.
Part-Species[$]-Surfaceflux[$]-TempRot	0.0	Rotational temperature.
Part-Species[$]-Surfaceflux[$]-TempElec	0.0	Electronic temperature.
Part-Species[$]-HeatOfFormation_K		Heat of formation of the respective species [Kelvin]
Part-Species[$]-PreviousState	0	Species number of the previous state (e.g. N for NIon)
Part-Species[$]-NextIonizationSpecies	0	SpeciesID of the next higher ionization level (required for field ionization)
Part-Species[$]-NumElectronicLevels	0	Max elec quantum number + 1
Part-Species[$]-ElectronicDegeneracy-Level[$]	0	Electronic degeneracy level of respective species

DSMC Species		
Part-Species[$]-ElectronicEnergyLevel-Level[$]	0.0	Electronic energy level of respective species
Part-Species[$]-SymmetryFactor	0	TODO-DEFINE-PARAMETER
Part-Species[$]-IonizationEn_eV	0.0	Energy of Ionization in [eV].
Part-Species[$]-RelPolarizability	0.0	Relative Polarizability
Part-Species[$]-NumEquivElecOutShell	0	Number of equivalent electrons in outer shells
Part-Species[$]-NumOfProtons	0	Number of protons for respective species.

DSMC Species Polyatomic		
Variable	**Default**	**Description**
Part-Species[$]-PolyatomicMol	F	Allow usage of polyatomic moleculs?
Part-Species[$]-LinearMolec	F	Flag if it is a linear molecule
Part-Species[$]-NumOfAtoms	0	Number of Atoms in Molecule
Part-Species[$]-CharaTempVib[$]	0.0	Characteristic vibrational temperature.
Part-Species[$]-CharaTempRot[$]	0.0	Characteristic rotational temperature

DSMC Chemistry		
Variable	**Default**	**Description**
DSMC-NumOfReactions	0	Number of reactions.
DSMC-Reaction[$]-NumberOfNonReactives	0	TODO-DEFINE-PARAMETER
DSMC-Reaction[$]-NonReactiveSpecies		Array with the non-reactive collision partners for dissociation
DSMC-Reaction[$]-ReactionType	none	Used reaction type I: electron impact ionization R: molecular recombination D: molecular dissociation E: molecular exchange reaction X: simple charge exchange reaction)

DSMC Chemistry		
DSMC-Reaction[$]-QKProcedure	F	Flag to use quantum-kinetic model
DSMC-Reaction[$]-QK-Method	0	Recombination Method for Q-K model 1: by Bird 2: by Gallis) If using bird, define the variables: DSMC-Reaction[$]-QK-Coeff1 DSMC-Reaction[$]-QK-Coeff2
DSMC-Reaction[$]-QK-Coeff1	0.0	First Q-K coefficient for Birds method.
DSMC-Reaction[$]-QK-Coeff2	0.0	Second Q-K coefficient for Birds method.
DSMC-Reaction[$]-Reactants	(/ 0, 0, 0 /)	Reactants of Reaction[$] (SpecNumOfReactant1, SpecNumOfReactant2, SpecNumOfReactant3)
DSMC-Reaction[$]-Products	(/ 0, 0, 0 /)	Products of Reaction[j] (Product1, Product2, Product3)
DSMC-Reaction[$]-Arrhenius-Prefactor	0.0	TODO-DEFINE-PARAMETER
DSMC-Reaction[$]-Arrhenius-Powerfactor	0.0	TODO-DEFINE-PARAMETER
DSMC-Reaction[$]-Activation-Energy_K	0.0	Activation energy (relativ to k_Boltzmann) for Reaction[$].
DSMC-Reaction[$]-CEXa	-27.2	CEX log-factor (g-dep. cross section in Angstrom, def.: value for Xe+)
DSMC-Reaction[$]-CEXb	175.269	CEX const. factor (g-dep. cross section in Angstrom, def.: value for Xe+)
DSMC-Reaction[$]-DoScat	F	Perform scattering-based charge-exchange instead of isotropic (model of Samuel Araki by lookup table)
DSMC-Reaction[$]-ELa	-26.8	with DoScat=T: EL log-factor (g&cut-off-angle-dep. cs in Angstrom, def.: value for Xe+)

DSMC Chemistry		
DSMC-Reaction[$]-ELb	148.975	with DoScat=T: EL const. factor (g&cut-off-angle-dep. cs in Angstrom, def.: value for Xe+)
DSMC-Reaction[$]-MEXa	-27.2	with DoScat=F: MEX log-factor (g-dep. cross section in Angstrom, def.: value for Xe+)
DSMC-Reaction[$]-MEXb	175.269	with DoScat=F: MEX const. factor (g-dep. cross section in Angstrom, def.: value for Xe+)
DSMC-Reaction[$]-TLU_FileName	0	with DoScat=F: No TLU-File needed (def.:)

LD		
Variable	**Default**	**Description**
LD-ModelForMultiTemp	0	TODO-DEFINE-PARAMETER Modell choice for MultiTemperature (see Paper) 0 = no MultiTemperature Modeling 1 = LD1 2 = LD2 3 = LD3
LD-InitialGuess	10.0	TODO-DEFINE-PARAMETER 2nd guess, plus user defined value[m/s], (default 10 m/s)
LD-MaxIterNumForLagVelo	100	TODO-DEFINE-PARAMETER Max. number of iterations for LAGRANGIAN vell calculation
LD-AccuracyForLagVelo	0.001	TODO-DEFINE-PARAMETER Accuracy for LAGRANGIAN velocity calculation
LD-RepositionsFaktor	0.0	TODO-DEFINE-PARAMETER
LD-RelaxationsFaktor	0.0	TODO-DEFINE-PARAMETER

LD		
LD-DSMC-RelaxationsFaktorForBufferA	0.0	TODO-DEFINE-PARAMETER
LD_CalcResidual	F	TODO-DEFINE-PARAMETER

SurfaceModel		
Variable	**Default**	**Description**
Part-Species[$]-MaximumCoverage	0.0	Maximum coverage of surfaces with species [$] (used for surfacemodel=1)
Part-Species[$]-InitialStick	0.0	Initial sticking coefficient (S_0) of species [$] for surfaces (used for Kisliuk model, surfacemodel=1)
Part-Species[$]-PrefactorStick	0.0	Prefactor of sticking coefficient of species [$] for surfaces (used for Kisliuk model, surfacemodel=1)
Part-Species[$]-Adsorbexp	1	Adsorption exponent of species [$] for surfaces (used for Kisliuk model, surfacemodel=1)
Part-Species[$]-Nu-a	0.0	TODO-DEFINE-PARAMETER Nu exponent a for surface n
Part-Species[$]-Nu-b	0.0	TODO-DEFINE-PARAMETER Nu exponent b for surface n
Part-Species[$]-Desorption-Energy-K	1.0	TODO-DEFINE-PARAMETER Desorption energy (K) for surface n
Part-Species[$]-Intensification-K	0.0	TODO-DEFINE-PARAMETER Intensification energy (K) for surface n
Part-Species[$]-Recomb-PartnerSpec	-1	TODO-DEFINE-PARAMETER Partner recombination species (nSpecies)

SurfaceModel		
Part-Species[$]-Recomb-ResultSpec	-1	TODO-DEFINE-PARAMETER Resulting recombination species (nSpecies)
Part-Species[$]-PartBound[$]-RecombinationCoeff	0.0	TODO-DEFINE-PARAMETER
Part-Species[$]-PartBound[$]-RecombinationEnergy	0.0	TODO-DEFINE-PARAMETER Energy transformed by reaction (nPartBound,nSpecies)
Part-Species[$]-PartBound[$]-RecombinationAccomodation	1.0	Define energy accomodation coefficient. Describes the percentage of reaction enthalpy of surface reaction transmitted to surface.
Part-Species[$]-PartBound[$]-Coordination	0	Coordination at which particle of species [$] is bound on surface of boundary [$]. 1=hollow 2=bridge 3=on-top[surfacemodel=3]
Part-Species[$]-PartBound[$]-DiCoordination	0	If particles of species [$] are di-, polyatomic and bind with additional coordination at Boundary [$]. 0: no DiCoordination 1: bound via bridge bonding 2: chelating binding [surfacemodel=3
Part-Species[$]-PartBound[$]-HeatOfAdsorption-K	0.0	Define heat of adsorption [K] on clear surface for binding atom of species [$] on boundary [$]. [Assumption of on-top side bind, surfacemodel=3]

SurfaceModel		
Particles-SurfCoverageFile	none	Give relative path to Surface-state-file ([Project-name]_DSMCSurfState***.h5) used for coverage init. File must be of same format as calculation (same mesh, same amount of species, cells and surfacemodel). If no file specified: Define Part-Species[$]-PartBound[$]-InitialCoverage for initial coverage different than 0.
Part-Species[$]-PartBound[$]-InitialCoverage	0.0	Initial coverage used for species [$] and surfaces of boundary [$] in case of no surface-state-file init
Particles-Surface-MacroParticleFactor		Weighting factor used for particles adsorbed on surface in case of reconstruction [surfacemodel=3]. If one surface contains less then 10 surface atoms program abort is called. Default: Species(1)%MPF: Uses macro particle factor of species1.
Surface-MaxDissNum	0	TODO-DEFINE-PARAMETER
Surface-Nbr-DissocReactions	0	TODO-DEFINE-PARAMETER Resulting species for given dissoc (2,MaxDissNum,nSpecies)
Surface-Nbr-ExchangeReactions	0	TODO-DEFINE-PARAMETER
Part-Species[$]-Adsorption-Powerfactor	0.0	TODO-DEFINE-PARAMETER
Part-Species[$]-Adsorption-Prefactor	0.0	TODO-DEFINE-PARAMETER

SurfaceModel		
Part-Species[$]-Adsorption-EDissBond	0.0	TODO-DEFINE-PARAMETER Bond dissociation energy (K) for diss into resultingspecies (ReactNum,nspecies)?
Part-Species[$]-Adsorption-EDissBondPoly1	0.0	TODO-DEFINE-PARAMETER
Part-Species[$]-Adsorption-EDissBondPoly2	0.0	TODO-DEFINE-PARAMETER
Part-Species[$]-SurfDiss[$]-Products	(/ 0, 0 /)	TODO-DEFINE-PARAMETER
Part-Species[$]-SurfDiss[$]-Powerfactor	0.0	TODO-DEFINE-PARAMETER
Part-Species[$]-SurfDiss[$]-Prefactor	0.0	TODO-DEFINE-PARAMETER
Part-Species[$]-SurfDiss[$]-EDissBond	0.0	TODO-DEFINE-PARAMETER Bond dissociation energy (K) for diss into resultingspecies (ReactNum,nspecies)?
Part-Species[$]-Surf-ER[$]-Powerfactor	0.0	TODO-DEFINE-PARAMETER
Part-Species[$]-Surf-ER[$]-Prefactor	0.0	TODO-DEFINE-PARAMETER
Surface-ExchReact[$]-Reactants	(/ 0, 0 /)	TODO-DEFINE-PARAMETER
Surface-ExchReact[$]-Products	(/ 0, 0 /)	TODO-DEFINE-PARAMETER
Surface-ExchReact[$]-DissBond_K-Reactants	(/ 0.0, 0.0 /)	TODO-DEFINE-PARAMETER
Surface-ExchReact[$]-DissBond_K-Products	(/ 0.0, 0.0 /)	TODO-DEFINE-PARAMETER
Surface-Adsorption-CalcTST	0	TODO-DEFINE-PARAMETER
Surface-AdsorptionTST-PartitionMaxTemp	10000.0	TODO-DEFINE-PARAMETER Temperature limit for pre-stored partition function (DEF: 20 000K)

SurfaceModel		
Surface-AdsorptionTST-PartitionInterval	20.0	TODO-DEFINE-PARAMETER Temperature interval for pre-stored partition function (DEF: 10K)

Surface Analyze		
Variable	**Default**	**Description**
Surface-AnalyzeStep	1	Analyze is performed each Nth time step for surfaces
Surf-CalcNumSpec	F	TODO-DEFINE-PARAMETER Calculate the number of simulatedparticles per species on surfaces
Surf-CalcCoverage	F	TODO-DEFINE-PARAMETER Calculate the surface coverages foreach species
Surf-CalcAccomodation	F	TODO-DEFINE-PARAMETER Calculate the surface accomodation coefficient
Surf-CalcEvaporation	F	TODO-DEFINE-PARAMETER Calculate rate of evaporation [kg/s]
Surf-CalcAdsorbRates	F	TODO-DEFINE-PARAMETER Calcualte the adsorption probabilities of species
Surf-CalcSurfRates	F	TODO-DEFINE-PARAMETER Calculate the surface reaction rate per reaction (k_r)

References

Atak, Muhammed, Andrea Beck, Thomas Bolemann, David Flad, Hannes Frank, and Claus-Dieter Munz. 2016. "High Fidelity Scale-Resolving Computational Fluid Dynamics Using the High Order Discontinuous Galerkin Spectral Element Method." In *High Performance Computing in Science and Engineering' 15*, 511–30. Springer.

Carlson, Jan-Reneé. 2011. "Inflow/Outflow Boundary Conditions with Application to Fun3d."

Carpenter, M., and C. Kennedy. 1994. "Fourth-Order 2N-Storage Runge-Kutta Schemes." NASA TM 109111.

Hindenlang, Florian, Gregor J Gassner, Christoph Altmann, Andrea Beck, Marc Staudenmaier, and Claus-Dieter Munz. 2012. "Explicit Discontinuous Galerkin Methods for Unsteady Problems." *Computers & Fluids* 61: 86–93.

Kopriva, David A., and Gregor Gassner. 2010. "On the Quadrature and Weak Form Choices in Collocation Type Discontinuous Galerkin Spectral Element Methods." *Journal of Scientific Computing* 44 (2): 136–55.

Kopriva, David A, Stephen L Woodruff, and M Yousuff Hussaini. 2002. "Computation of Electromagnetic Scattering with a Non-Conforming Discontinuous Spectral Element Method." *International Journal for Numerical Methods in Engineering* 53 (1): 105–22.