ThunderBoltz C++ Manual

Release 0.1

Ryan Park, Brett Scheiner, Mark Zammit

Los Alamos National Laboratory, Los Alamos, NM, 87545

Oct 30, 2023
LA-UR-23-31893

CONTENTS

This documentation includes instruction on installation and compilation of ThunderBoltz source code and descriptions of the input and output simulation parameters involved.

CHAPTER

ONE

INSTALLATION

For now, the code must be downloaded from a repository. Use the following command to clone the code into a local repository.

```
git clone git@gitlab.com/Mczammit/thunderboltz.git
```

You may need to set up SSH keys in order to access gitlab. See the Gitlab SSH Guide to set up access to GitLab repositories.

The basic ThunderBoltz functionality is available either as an executable in `bin/thunderboltz.bin` or can be compiled from the source in `src/thunderboltz`.

CHAPTER

TWO

COMPILATION

ThunderBoltz requires a g++ of clang compiler and should be compiled from source directories as

```
g++ -std=c++17 -o thunderboltz.bin DSMC0D.cpp
```

Then run with

```
./thunderboltz.bin inputfile.in
```

to use a manually constructed indeck file. The code is maintained with the standard *-Wall* and *-Werror* compiler options.

Here is an example of how to run a simple ThunderBoltz calculation.

```
# Make a directory for testing
mkdir example_sim
cd example_sim
# Copy the source over
cp ../src/thunderboltz/* .
# Copy example input files over
cp -r ../indecks/N2/* .
# Compile
g++ -std=c++17 -o thunderboltz.bin DSMC0D.cpp
# Run
./thunderboltz.bin N2vib.in
```

CHAPTER

THREE

SIMULATION PARAMETERS

3.1 Input Parameters

The ThunderBoltz simulation settings and their default values.

`B`	(list[int]) Magnetic field vector (Tesla), default is `[0.0, 0.0, 0.0]`.
`CR`	(int) If `1`, then the remainder of N_{pairs} is carried into the next N_{pairs} evaluation of the same process, default is `0`.
`DT`	(float) Time increment interval (s).
`E`	(float) Electric field in z-direction (V/m), default is `-0`.
`ET`	(float) E-field oscillation frequency (Hz), default is `0`.
`EX`	(int) Cross section extrapolation — options are `0` (extrapolated cross sections are set to 0 m ^{2}), or `1` (linearly extrapolated from last two points), default is `0`.
`FV`	(list[int]) Output velocity dump settings [start, stride, species ID].
`L`	(float) Cell length (m).
`LV`	(list[str,int]) Optionally load particle velocities from a comma separated text file; specify the name of the file at index `0` and the particle species index it applies to at index `1`.
`MEM`	(float) Request memory (GB) for particle arrays.
`MP`	(list[float]) Mass of each particle species (amu).
`NP`	(list[int]) Number of particles for each species.
`NS`	(int) Number of time steps.
`OS`	(int) Time step stride for output parameters, default is `100`.
`QP`	(list[int]) Charge (elementary units) of each particle species.
`SE`	(int) When using a SLURM manager on HPC, auto dump particle velocity data before job allocation runs out — options are `0` (don't auto dump) | `1` (dump using SLURM setup).
`SP`	(int) Number of species.
`TP`	(list[float]) Temperature (eV) of each particle species.
`VS`	(int) Number of random samples used to find $\max_\epsilon(v\sigma(\epsilon))$ for each process, default is `1000`.
`VV`	(list[float]) Flow velocity for each particle.

3.2 Output Parameters

3.2.1 Banner Output

A listing of the main output parameters of the simulation. By default, these are printed to standard out every "OS" time steps.

`E`	(float) The electric field component (V/m) in the z direction, which can change in AC scenarios.
`MEe`	(float) The mean energy (eV) of the species at index 0 (usually electrons), computed as $\langle \epsilon \rangle = \frac{m_0}{2N_0} \sum_{i=1}^{N_0} v_{0i}^2$ where m_0 and N_0 are the mass and particle count of the 0^{th} species, and v_{0i} is the velocity vector of the i^{th} particle of species 0.
`step`	(int) The number of time steps elapsed in the simulation, with `t` $= 0$ corresponding to `step` $= 0$, and with `t = DT` corresponding to `step` $= 1$.
`t`	(float) The time (s) elapsed in the simulation.

3.2.2 Particle Parameters

A listing of additional output written for each species. A text file is generated named "Particle_Type_i" for each species "i" with the following data in each column.

`Ki`	(float) The total kinetic energy (eV).
`Mi`	(float) The mean kinetic energy (eV).
`Ni`	(float) The number density (m $^{-3}$).
`Rxi`	(float) The mean x component of all particle displacements (m).
`Ryi`	(float) The mean y component of all particle displacements (m).
`Rzi`	(float) The mean z component of all particle displacements (m).
`Txi`	(float) The mean x component temperature (eV).
`Tyi`	(float) The mean y component temperature (eV).
`Tzi`	(float) The mean z component temperature (eV).
`Vxi`	(float) The mean x component velocity (m/s).
`Vyi`	(float) The mean y component velocity (m/s).
`Vzi`	(float) The mean z component velocity (m/s).
`step`	(int) The number of time steps elapsed in the simulation, with `t` $= 0$ corresponding to `step` $= 0$.
`t`	(float) The time (s) elapsed in the simulation.

CHAPTER

FOUR

ELECTRON GROWTH AND MEMORY MANAGEMENT

Depending on the ionization model and field strength, ThunderBoltz may generate a large number of electrons. In these cases, the appropriate amount of memory must be allocated. The correct amount will be allocated automatically in scenarios where no ionization process is used, or when the `IonizationNoEgen` model is used. This amount will be allocated based on the sum of all `NP` elements times 4.

However, in scenarios where there is significant electron generation, i.e. at high E fields with the `Ionization` model on, the default memory settings are not sufficient and the simulation will exit with the error "Too many particles!". To prevent this specify the `MEM` flag in the indeck. MEM will accept any float representing the number of gigabytes to be made available to the particle arrays.

Warning: If the value of `MEM` is more than the actual number of available GB, then the simulation will still run, but will exit with a segmentation fault once too many particles are created.

Warning: When using multiple cores on the same machine / node, ensure that each process has enough memory requested and that the sum of memory requests does not exceed the available pool of RAM.

CHAPTER

FIVE

INPUT CROSS SECTION FORMAT

5.1 Indeck Format

ThunderBoltz input is contained within an input deck that specifies the simulation behavior. Each line of the input deck, including comments, starts with a one or two character long specifier called an index. The deck order is unimportant with the exception that the number of species needs to be specified before other particle definitions such as number, charge, temperature, flow velocity, or mass.

Comment lines begin with the characters CC and have the following comment in quotes. The only restriction to this usage is that another index should not appear in isolation. For example, a comment `CC "The electric field index is E "` will produce an error because the parser will recognize the index E. An appropriate modification is to prevent the E from appearing in isolation such as `CC "The electric field index is {E} "`.

Here is a full example indeck:

```
CC "Comment lines start with CC and have comment in quotes"
CC "Length of Box"
L 1e-6
CC "Particle Species Information"
CC "------------------------------"
CC "number of species {SP}"
SP 4
CC "number of particles for each species {NP}"
CC "NP 101250 135000 "
NP 6000 10000 0 0
CC "temperature of each particle species"
TP  1.0 0.0 0.0 0.0
CC "flow velocity"
VV 50.0 0.0 0.0 0.0
CC "Charge of each particle species"
QP -1.0 0.0 0.0 0.0
CC "Mass of each particle species"
MP 5.4857e-4 28.0 28.0 28.0
CC "------------------------------"
NS 33001
VS 1000
DT 1.0e-11
CC "note -10000V is 100Td, -100V is 1Td"
E -5000.0
ET 10000000
CC "0.001 = 10Hx"
B 0.0 0.00 0.0
CC " Collision Model Specification "
```

```
CC "----------------------"
CC "List cross sections first start with {CS}"
CS Data/N2/1.dat 0 1 ElasticFixedParticle2 0.0 0 1
CS Data/N2/v01.dat 0 1 InelasticChangeParticle2 0.275 0 2
CS Data/N2/v02.dat 0 1 InelasticChangeParticle2 0.59 0 3
CS Data/N2/v10.dat 0 2 InelasticChangeParticle2 0.0  0 1
CS Data/N2/v20.dat 0 3 InelasticChangeParticle2 0.0  0 1
CS Data/N2/v12.dat 0 2 InelasticChangeParticle2 0.315  0 3
CS Data/N2/v21.dat 0 3 InelasticChangeParticle2 0.0  0 2
CS Data/N2/1.dat 0 2 ElasticFixedParticle2 0.0 0 2
CS Data/N2/1.dat 0 3 ElasticFixedParticle2 0.0 0 3
CS Data/N2/4.dat 0 1 InelasticFixedParticle2 0.88 0 1
CS Data/N2/5.dat 0 1 InelasticFixedParticle2 1.17 0 1
CS Data/N2/6.dat 0 1 InelasticFixedParticle2 1.47 0 1
CS Data/N2/7.dat 0 1 InelasticFixedParticle2 1.76 0 1
CS Data/N2/8.dat 0 1 InelasticFixedParticle2 2.06 0 1
CS Data/N2/9.dat 0 1 InelasticFixedParticle2 2.35 0 1
CS Data/N2/10.dat 0 1 InelasticFixedParticle2 6.17 0 1
CS Data/N2/11.dat 0 1 InelasticFixedParticle2 7.00 0 1
CS Data/N2/12.dat 0 1 InelasticFixedParticle2 7.35 0 1
CS Data/N2/13.dat 0 1 InelasticFixedParticle2 7.36 0 1
CS Data/N2/14.dat 0 1 InelasticFixedParticle2 7.80 0 1
CS Data/N2/15.dat 0 1 InelasticFixedParticle2 8.16 0 1
CS Data/N2/16.dat 0 1 InelasticFixedParticle2 8.40 0 1
CS Data/N2/17.dat 0 1 InelasticFixedParticle2 8.55 0 1
CS Data/N2/18.dat 0 1 InelasticFixedParticle2 8.89 0 1
CS Data/N2/19.dat 0 1 InelasticFixedParticle2 11.03 0 1
CS Data/N2/20.dat 0 1 InelasticFixedParticle2 11.88 0 1
CS Data/N2/21.dat 0 1 InelasticFixedParticle2 12.25 0 1
CS Data/N2/22.dat 0 1 InelasticFixedParticle2 13.0 0 1
CS Data/N2/23.dat 0 1 InelasticFixedParticle2 15.6 0 1
CC "-----Output Control-----"
CC "Dump particles of type i: {FV} {start time in steps} {Stride in steps} {int type}"
FV 1000 150000 0
```

Note that the orderings of all parameters with multiple arguments is consistent with respect to each species. Typically the first species is assumed to be electrons.

5.2 Cross Section and Collision Model Format

The index `CS` specifies the particle interaction pairs, cross section data file, collision model, reaction products, and threshold energy (or exothermic energy release) for a reaction. The general form of input for this input is

```
CS [data] [reactant_1] [reactant_2] [model] [energy] [product_1] [product_2].
```

The required input options are the following:

5.2.1 [data] : The cross section data file

This is a path to a cross section data file for the energy dependent cross section $\sigma(E)$ and should be indicated by a string. The file is assumed to be in two column format with the first column being the energy in eV and the second column the cross section in m^2. For example, the first few lines of the cross section file in Data/N2/1.dat are the following:

```
0.000000e+0     1.100000e-20
1.000000e-3     1.360000e-20
2.000000e-3     1.490000e-20
3.000000e-3     1.620000e-20
2.000000e-3     1.810000e-20
7.000000e-3     2.000000e-20
8.500000e-3     2.100000e-20
1.000000e-2     2.190000e-20
1.500000e-2     2.550000e-20
2.000000e-2     2.850000e-20
3.000000e-2     3.400000e-20
4.000000e-2     3.850000e-20
5.000000e-2     4.330000e-20
7.000000e-2     5.100000e-20
```

The code linearly interpolates the data for the DSMC collision evaluation so the input should be sufficiently well resolved for the intended purpose.

5.2.2 [reactant 1] and [reactant 2] : Colliding Particle Pairs/Reactants

This option specifies the colliding particle species taking part in the interaction. Each species is identified as an integer starting at 0 and counting to $N_{\text{species}} - 1$. The species integer identifier is ordered in the same order as `NP`, `MP`, `QP`, and `TP`.

5.2.3 [product 1] and [product 2] : Reaction Products

This option specifies the reactant species from a successful collision. This data is not utilized for all collisions, but must be specified for each instance of `CS`. See the details for each collision model for more information.

5.2.4 [energy]: Threshold/Exothermic Energy

This option is the threshold energy for a collision to occur and is specified in electron volts (eV). If the value is negative, it indicates that the reaction releases energy into the products during the reaction. Here it is assumed that the cross section file is zero below the threshold energy. It this is not the case some collision model evaluation will on occasion produce collision pairs where the collision energy is below the threshold for the inelastic reaction to proceed. In this case, the code will produce a warning and will set the reactant energy to zero. The user should ensure that the cross section file meets the required specification.

5.2.5 [model]: Collision Model

All collision models utilize the energy dependent cross section file specified by the file path [data] to evaluate the probability of interaction. This subsection lists the particle interaction models that specify how energy/momentum is transferred between particles. The current collision model options are listed in the next section.

CHAPTER

SIX

DESCRIPTION OF AVAILABLE COLLISION MODELS

6.1 Isotropic Elastic Collisions

In this collision model the energy and momentum exchange between particles is calculated assuming a uniformly random isotropic post-collision scattering angle. This model is intended to be used with elastic momentum cross section data, where in many cross section databases momentum transfer cross sections are included instead of elastic cross sections. These can be used in conjunction with the isotropic and elastic collision model to produce transport data that captures the momentum transfer statistics of the anisotropic scattering when models for the anisotropic scattering angle distribution are unavailable.

6.2 Anisotropic Elastic Collisions

ThunderBoltz includes the framework needed for the implementation of anisotropic scattering models. Generally, independent of the particular anisotropic model, the post collision scattering angle is determined in the center of mass frame. The relative velocity vector is rotated in the center of mass frame throughout the collision and needs to be transformed back to the original laboratory (simulation) frame to determine the post collision velocities. The mechanism for performing this transformation is available in the code for the implementation of anisotropic models. See the discussion in Ref. [2] for details of the procedure.

At present, ThunderBoltz includes the anisotropic elastic scattering model from Ref. [4]. The model provides an invertible angular distribution function that allows the post collision scattering angle of an electron to be determined by its relative collision energy and a uniform random number between 0 and 1 such that the scattering angle distribution of electrons approximates that given by the elastic differential scattering cross section when collisions are evaluated with an appropriate elastic integrated scattering cross section. The model currently has known fit parameters for the inverted angular distribution function and the elastic integrated scattering cross section of H, He, H_2, and their isotopes.

6.3 Non-Reactive Collisions

This collision option can be combined with inelastic collisions and neglects the products produced in reactions, i.e the reactants are the same as the products. The appropriate energy transfer or threshold energy cost of the reaction is calculated and subtracted from the reactants post collision velocities. This collision type is useful for emulating the behavior of Boltzmann solvers by maintaining an unchanging background, or when the tracking of excited states of some subset of species in reactive kinetics simulations is unimportant. As an example, if the final density of $N_2(A^3\Sigma_u^+)$ is unimportant, the reaction e + $N_2 \rightarrow$ e + $N_2(A^3\Sigma_u^+)$ is modeled as e + $N_2 \rightarrow$ e + N_2 with the excitation energy of the $A^3\Sigma_u^+$ state subtracted from the relative collision energy, but no particles are removed and no new particle products are generated.

6.4 Fixed Heavy Particle Collisions

To calculate transport of a charged species in a fixed background gas it is necessary to maintain the statistical properties of the background (e.g. temperature, energy, flow and other moments of the VDF). If energy exchange in elastic or inelastic collisions is allowed the background will gain energy from the field accelerated particles. For this reason, it is desirable to have a collision model that fixes the background particles. In this case, the electron or ion energy loss is calculated as in an elastic collision, but the neutral heavy particle energy and velocity is not updated, maintaining the statistical properties of the background gas distribution.

6.5 Isotropic Inelastic Collisions

The energy and momentum exchange between particles is calculated using a uniformly random isotropic post collision scattering angle and the threshold energy for the inelastic collision process is subtracted in the center of mass frame. This model can be combined with the Fixed Heavy Particle and Non-Reactive Collision options as described in Secs. 6.3 and 6.4, or the species type of the reactants is switched to that of the products given in the collision specification line. Other than ionization, currently all inelastic collisions use this scattering model.

6.6 Electron Impact Ionization

For ionizing collisions, a model is needed to determine how to partition energy in excess of the ionization energy between the products. The residual ion maintains the velocity and direction of the target neutral/ion, while the scattering angle of the product electrons is assumed to be isotropic. The excess energy available for sharing between the electrons is calculated by assuming the post-collision residual ion maintains the same velocity as the pre-collision target neutral/ion. The energy balance in the rest frame of the target is $\epsilon_{\rm s} = \epsilon - \epsilon_{\rm ion} - \epsilon_{\rm ej}$, where ϵ, $\epsilon_{\rm s}$, and $\epsilon_{\rm ej}$ are the incident, scattered, and ejected electron energies, and $\epsilon_{\rm ion}$ is the ionization energy of the bound electron. Generally, the ejected electron can take on a distribution of values which can be represented by an electron energy sharing distribution. Three models for this distribution are implemented. These are the *one takes all* model: where one electron is ejected with 0 eV and the other with $\epsilon - \epsilon_{\rm ion}$ [3], the *equal energy sharing* model[3]: where each electron is ejected with energy $(\epsilon - \epsilon_{\rm ion})/2$, and the *uniform energy sharing* model[1]: where the energy of one electron has a uniform distribution in the range $[0, (\epsilon - \epsilon_{\rm ion})/2]$ and the other is determined from conservation of energy.

BIBLIOGRAPHY

[1] H.-K. Chung, M.H. Chen, W.L. Morgan, Y. Ralchenko, and R.W. Lee. Flychk: Generalized population kinetics and spectral model for rapid spectroscopic analysis for all elements. *High Energy Density Physics*, 1(1):3–12, 2005.

[2] Zoltán Donkó, Aranka Derzsi, Máté Vass, Benedek Horváth, Sebastian Wilczek, Botond Hartmann, and Peter Hartmann. edupic: an introductory particle based code for radio-frequency plasma simulation. *Plasma Sources Science and Technology*, 30(9):095017, 2021.

[3] G J M Hagelaar and L C Pitchford. Solving the boltzmann equation to obtain electron transport coefficients and rate coefficients for fluid models. *Plasma Sources Science and Technology*, 14(4):722–733, oct 2005.

[4] Ryan M Park, Willem Kupets, Mark C Zammit, James Colgan, Christopher J Fontes, Brett S Scheiner, Eddy Timmermans, Xian-Zhu Tang, Liam H Scarlett, Dmitry V Fursa, Igor Bray, and Nathan A Garland. Anisotropic angular scattering models of elastic electron-neutral collisions for monte carlo plasma simulations. *Plasma Sources Science and Technology*, 31(6):065013, jul 2022.