`CollCalc`: a code for fast numerical calculation of collision integrals

Maxim Laletin

Abstract

This document provides a manual for `CollCalc`, a C++ package designed to efficiently compute multi-dimensional integrals that appear in statistical mechanics of relativistic particles, especially in its applications in cosmology. This manual covers installation, structure of the code and usage examples, as well as the theoretical aspects of the collision integrals.

Contents

1 Introduction

Collision integrals arise in numerous applications of kinetic theory and serve as a measure of the rate at which the distribution function that describes a statistical system changes with time under the impact of microscopic interaction processes between the elements of this system. Some cosmological applications of kinetic theory deal with the time evolution of various particle species in the hot and dense plasma of the early Universe and the typical microscopic interaction processes involve decays, annihilations, elastic scatterings, etc. The collision integral takes into account all physically allowed configurations (energies and momenta) of particles in a given reaction and from the mathematical point of view acts on a function in a phase space of dimension $D = 3(N-1)$ with N being the number of vertices in a reaction (one vertex corresponds to the particle whose evolution we study). For instance, 2-body decay has 3 vertices and elastic scattering has 4 vertices, hence the respective collision integrals have $D = 6$ and $D = 9$. Although, the laws of energy and momentum conservation (mathematically expressed in the form of 4-dimensional Dirac delta function) allow to perform 4 integrations analytically and reduce the number of dimensions the resulting integral rarely has an analytical solution (or can be further analytically simplified), which requires numerical computations of multi-dimensional integrals.

`CollCalc` provides the tools for fast computation of these integrals for the physics processes specified by the user. The code is written in C++ and is optimized to reduce the CPU time by the combination of efficient algorithms and mathematical approximations without the loss of generality in physics. The values of the integrals can be tabulated on the grid of physical variables like the momentum of the particle whose distribution is solved for or (inverse) temperature for numerical interpolation, which can make the solution of the corresponding kinetical equation faster. The tabulation can be performed on several CPU cores using OpenMP.

2 Collision integrals

Collision terms $C[f_\chi]$ appear in the Boltzmann equation for the phase-space density f_χ of particle χ in the expanding Universe

$$\tilde{H}(x\partial_x - \tilde{g}q\partial_q) f_\chi(x, q_\chi) = C[f_\chi]\,, \tag{1}$$

where $x = m/T$ can be regarded as a cosmological unit of time with T being the temperature of the radiation bath and m is the characteristic mass scale (can be the mass of χ or some other particle for the sake of convenience), $q_\chi = p_\chi/T$ is the temperature-reduced momentum of χ, $\tilde{H} = H/(1+\tilde{g})$ is the Hubble parameter divided by

$$1 + \tilde{g} = 1 + \frac{1}{3}\frac{d\ln g_s}{d\ln x} \tag{2}$$

with g_s being the number of entropy degrees of freedom. $C[f_\chi]$ takes into account the impact of all the particle physics processes in which χ participates on its phase-space distribution. For more details on the Boltzmann equation and its applications we address the reader to Refs. [1, 2]

Below we discuss different types of processes and the correspondent collision integrals that are implemented in `CollCalc`. The current version of the package contains the

processes in which only one type of particles χ has an unknown distribution function f_χ while all the other particles are in thermal equilibrium with the radiation bath and their distribution function is determined by their spin statistics or is given by the Maxwell-Boltzmann distribution if their quantum properties can be neglected.

$$f_i^{\rm eq}(E) = \begin{cases} [\exp(E/T)-1]^{-1}\,, & \text{bosons,} \\ [\exp(E/T)+1]^{-1}\,, & \text{fermions,} \\ \exp(-E/T), & \text{classical,} \end{cases} \tag{3}$$

where E is the energy of i particles. From here onward all the particles with known distribution functions will be denoted with latin indices like i, j, k, etc. Also, the current version of the code only contains $2 \to 2$ kinematical reactions.

2.1 Co-annihilation

Co-annihilation is a type of processes in which one χ particle interacts with particle k to produce particles i and j. Although we use different indices here to preserve generality any two particles from the $\{i, j, k\}$ set or even all of them can be the particles of the same type. The collision term for the general co-annihilation process is given by

$$C[f_\chi] = \frac{1}{2g_\chi E_\chi} \int d\Pi_k d\Pi_i d\Pi_j \, (2\pi)^4 \delta^{(4)}(\mathcal{P}_i + \mathcal{P}_j - \mathcal{P}_\chi - \mathcal{P}_k) \, |\mathcal{M}|^2_{ij\to\chi k} \times \\ \times \Big[f_i f_j (1 \pm f_\chi)(1 \pm f_k) - f_\chi f_k (1 \pm f_j)(1 \pm f_i) \Big] \,, \tag{4}$$

where g_χ is the number of χ's degrees of freedom, $d\Pi_i = d^3p_i/((2\pi)^3\, 2E_i)$, $\mathcal{P}$ denotes the four-momenta of particles, $|\mathcal{M}|^2$ is the squared amplitude for the corresponding process **summed over all** initial and final spin configurations, the integral is taken over the phase space of all the particles except χ and the signs of the quantum corrections depend on the spin statistics of the corresponding particles and in general should not have the same value across this expression in all the cases. Note that the collision term also includes the inverse process, which is called co-production. In full thermal equilibrium the rates of forward and inverse processes should be the same, so the second term in the expression above should be equal to the first one. Hence, we can write the second term as

$$f_\chi f_k^{\rm eq}(1 \pm f_i^{\rm eq})(1 \pm f_j^{\rm eq}) = \frac{f_\chi}{f_\chi^{\rm eq}} f_i^{\rm eq} f_j^{\rm eq} (1 + f_\chi^{\rm eq})(1 \pm f_k^{\rm eq}) \,. \tag{5}$$

Plugging this into the collision term we get the following structure

$$C[f_\chi] = \frac{1}{2g_\chi E_\chi} \left(1 - \frac{f_\chi}{f_\chi^{\rm eq}}\right) \gamma_{ij\to\chi k} \,, \tag{6}$$

where $\gamma_{ij\to\chi k}$ can be regarded as a differential rate of χ production per unit of its phase-space volume and is given by

$$\gamma_{ij\to\chi k} = \int d\Pi_k d\Pi_i d\Pi_j \, (2\pi)^4 \delta^{(4)}(\mathcal{P}_i + \mathcal{P}_j - \mathcal{P}_\chi - \mathcal{P}_k) \, |\mathcal{M}|^2_{ij\to\chi k} \, f_i^{\rm eq} f_j^{\rm eq} (1 \pm f_k^{\rm eq}). \tag{7}$$

One possible way to analytically reduce the dimensionality of the integrand is to compute the integral over $d\Pi_i$ and $d\Pi_j$ in the center-of-momentum (CM) frame[1] transforming the distribution functions via the following substitution

$$E_i = \frac{E_i^* + p_i^* v \cdot \cos\theta_{\rm CM}}{\sqrt{1-v^2}}\,, \quad E_j = \frac{E_j^* - p_i^* v \cdot \cos\theta_{\rm CM}}{\sqrt{1-v^2}}\,, \tag{8}$$

where the asterisk denotes the quantities in the CM frame, $v = \sqrt{1 - s/(E_k + E_a)^2}$ is the velocity of the CM frame and $\theta_{\rm CM}$ is the angle between the momentum in CM frame and the velocity of the CM frame. After the integration over $d\Pi_j^*$ and E_i^* are performed with the use of the delta-function the residual angular integrals can be expressed via Mandelstam variable t and an angle ϕ between the projections of $\vec{p}_a$ and $\vec{p_i^*}$ on the plane that is orthogonal to $\vec{p_a^*}$. The integration over $d\Pi_k$ has to be performed in the plasma frame reducing it to the integral over the energy E_k in the plasma frame and s variable. The cosine of $\theta_{\rm CM}$ can be expressed via the four integration variables that we are left with. Before we arrive at the final expression for $\gamma_{ij\to\chi k}$ it is convenient to introduce some dimensionless variables by dividing each unit of energy by the temperature of the radiation bath. Thus, for example, we use $\epsilon_i = E_i/T$ and $q_i = p_i/T$ instead of energy and momentum, and $\tilde{s} = s/T^2$ and $\tilde{t} = t/T^2$ instead of s and t Mandelstam variables

$$\gamma_{ij\to\chi k} = \frac{T^2}{4\,(2\pi)^4}\int_{\epsilon_k^{\rm min}}^{\infty} d\epsilon_k\; q_k\,(1 \pm f_k(\epsilon_k)) \int_{-1}^{\cos\theta_s^{\rm max}} d\cos\theta_s\; \frac{q_i^*}{\sqrt{\tilde{s}}} \times \\ \times \int_{-1}^{1} d\cos\theta_t\;\; |\mathcal{M}|^2_{ij\to ak}\,(s,t) \int_{-1}^{1} d\cos\phi\; \frac{f_i^*(\epsilon_i) f_j^*(\epsilon_j)}{\sqrt{1-\cos\phi^2}}\,, \tag{9}$$

where $\epsilon_k^{\rm min}$ is derived from the condition that $s_{\rm max} \geq (m_i + m_j)^2$

$$\begin{cases} \epsilon_k^{\rm min} = \varepsilon_k(q_\chi) & \text{if } \varepsilon_k < \left[(m_i+m_j)^2 - m_x^2 - m_k^2\right]/2\epsilon_\chi, \\ \epsilon_k^{\rm min} = m_k & \text{if } \varepsilon_k \geq \left[(m_i+m_j)^2 - m_x^2 - m_k^2\right]/2\epsilon_\chi, \end{cases} \tag{10}$$

where $\varepsilon_k(q_\chi)$ is calculated as follows

$$\varepsilon_k = m_\chi^{-2}\Big(\epsilon_\chi\left[(m_i+m_j)^2 - m_k^2 - m_x^2\right] - \\ - q_\chi\sqrt{\left[(m_i+m_j+m_\chi)^2 - m_k^2\right]\left[(m_\chi - m_i - m_j)^2 - m_k^2\right]}\Big)\,. \tag{11}$$

The maximal value of the $\cos\theta_s$ is derived from the condition that $s > (m_i + m_j)^2$

$$\cos\theta_s^{\rm max} = \min\left\{1, \frac{m_\chi^2 + m_k^2 + 2\epsilon_k\epsilon_\chi - (m_i+m_j)^2}{2q_\chi p_k}\right\}\,. \tag{12}$$

The Mandelstam variables we use here can be expressed as follows

$$s = (\mathcal{P}_a + \mathcal{P}_k)^2 \;=\; m_a^2 + m_k^2 + 2E_kE_a - 2p_ap_k\cos\theta_s\,; \tag{13}$$

$$t = (\mathcal{P}_a - \mathcal{P}_i)^2 \;=\; m_i^2 + m_k^2 - 2E_i^*E_k^* + 2p_i^*p_k^*\cos\theta_t\,. \tag{14}$$

[1]Note that $\gamma_{ij\to\chi k}$ is not Lorentz invariant due to the presence of distribution functions and has to be computed in the plasma frame in which the Boltzmann equation is formulated. However, parts of this integral can be transformed into a more convenient frame for the calculation and then transformed back into the plasma frame.

The energies in the distribution functions f_i^* and f_j^* are derived via the variables of integration in the following manner

$$E_{i,j} = \frac{E^*_{i,j}}{\sqrt{1-v^2}} \pm \frac{p^*_{i,j}\left[p^*_k \cos\theta_t - p_a\left(\cos\theta_t \cos\theta_{\rm aCM} - \sin\theta_t \sin\theta_{\rm aCM}\cos\phi\right)\right]}{E_k + E_a}\,, \tag{15}$$

where $\theta_{\rm aCM}$ is the angle between $\vec{v}$ and $\vec{p_a}$ and is given by

$$\cos\theta_{\rm aCM} = \frac{p_a^2 + \gamma^L(\vec{p_a}\cdot\vec{v})(\gamma^L(\vec{p_a}\cdot\vec{v})/(\gamma^L+1) - E_a)}{p_a p_k^*}\,, \tag{16}$$

with $\gamma^L = 1/\sqrt{1-v^2}$ being the Lorentz factor and the scalar product $(\vec{p_a}\cdot\vec{v}) = (p_a^2 + p_a p_k \cos\theta_s)/(E_x + E_k)$.

`CollCalc` computes $\gamma_{ij\to\chi k}$ as a function of x and q_χ for a given model (masses, particle spin types, $|\mathcal{M}|^2$ and potentially other particle physics parameters). An executable is compiled for the model specified in the corresponding file in the `processes` folder and stored in the `bin` folder. The executable can be called directly with 3 arguments: x, q_χ and the desired relative accuracy r. If the executable is called with just one argument r a file will be created in `results` folder that contains the tabulated values of $\gamma_{ij\to\chi k}$ for different grid values of x and q_χ specified in the corresponding tables in `parameters` folder. See the examples of usage below for more information.

2.2 Annihilation

Annihilation is the type of processes in which two χ particles interact to produce particles i and j. As we mentioned in the previous section, i and j particles can be of the same type, but we use different indices for generality. The collision term for the general annihilation process is given by

...

3 Installation

To install the `CollCalc` simply clone the github repository in the terminal or command prompt

```
cd /path/to/directory
git clone https://github.com/Maxim-Laletin/
   CollCalc.git
cd CollCalc
```

where `/path/to/directory` is the path to the directory where you want to install `CollCalc`. Alternatively, you can manually download a ZIP file from `CollCalc` repository and unzip it in the desired folder.

4 Structure of the package

Let us browse through various folders of `CollCalc` package to get acquainted with the structure of the code, functionality and prerequisites.

4.1 bin

Stores the executables for different processes and collision integrals. The executable's name typically has the following structure: `model_CItype`, where `model` is the name of the process file in `processes` folder and `CItype` is the type of collision integral function (see below for more details). The executable can be run directly from the terminal or command line and its functionality depends on the structure of the arguments. Below we describe the two possible options:

```
./model_CItype arg1 ... argN rel_acc
```

Here `arg1 ... argN` are the values of the physical variables on which the collision integral function depends. For example, the arguments for co-annihilation type of CI should be x and q_χ, while for annihilation CI it should be x, q_χ and q_k (see Sec. 2 for more details). `rel_acc` is the desired relative accuracy for the computation of the CI function. The following structure of the arguments returns the value of the CI function for the provided variables.

```
./model_CItype rel_acc
```

If the executable is called with only one argument `rel_acc` a table of the values of the CI function (with the same name) is created in the `results` folder. The table is based on the grid of values of the physical variables stored in the `parameters` folder.

4.2 include

Contains the header files for all the relevant source files.

4.3 main

Contains the main source file `CollCalc.cpp` with the most general functions and the initialization of the nested integration routine for the given type of CI function. The file also contains instructions for the output and writing the result files and for the parallel computing of the CI functions for the case when the executable is called with just one argument.

In the current version the `CollCalc.cpp` file also declares various parameters, such as the memory allocated for the calculation of each nested integral, its relative accuracy (`rel_ecc` only sets the relative accuracy of the base integration) and integer keys that define the numbers of points used for the Gauss-Kronrod quadrature method. The user is recommended to test the alternative values of these parameters if the integration fails or in order to reduce the integration time.

4.4 parameters

Contains the files with the values of numerical parameters and arrays of the physical variables. The current version only requires the latter and the folder contains two example files `x.csv` and `q.csv` by default. These files contain the values of x and q_χ variables that can be used for the co-annihilation type of CI function. When the corresponding executable is run with only one argument these values are used to produce a grid of the CI function outputs, which is stored in the `results` folder.

4.5 processes

Contains process files with the physical parameters of a given particle-physics process and its amplitude squared function $|\mathcal{M}|^2$. The user should create the process file for the process they want to study according to the following template

```
#include <string>
#include "Model.h"

double Mi = 2.0;
double Mj = 1.0;
double Mk = 2.0;
double Mx = 0.1;

char s_i = 'f';
char s_j = 'b';
char s_k = 'f';

// Process name used to create the file with the
   results
std::string pr_name = "model";

// Amplitude squared (without the constant part
   in front)
double M2(double s, double t)
{
        return t*t/(s - Mi*Mi)/(s + t - Mi*Mi);
}
```

The process file should necessarily provide the values of the 4 masses {`Mi,Mj,Mk,Mx`}, the spin types of i, j and k particles ("f" for fermions, "b" for bosons), the name of the process (that is used to create the result table) and the amplitude squared as a function of s and t variables, as well as the values of all the parameters that might appear in the `M2` function. The values of the masses (and their corresponding units) and the expression for the amplitude squared in the template above are arbitrary. The default `process` folder contains the examples of two process files: `Muon_annihilation.cpp` for muon annihilation into a photon and axion (being the particle χ with the unknown distribution) and `Primakoff.cpp` for the Primakoff scattering of photons on muons.

4.6 results

Stores the tables with the results of the integration for various values of the corresponding physical parameters. These tables can be used for interpolation in the applications which require the CI function.

4.7 source

Contains `.cpp` source files for various types of CI functions. In the current version of the code this folder contains only `Annihilation.cpp`, which includes the instructions for nested integrations used for both annihilation and co-annihilation types of CI functions.

5 Building projects

To create an executable for a given particle-physics process (defined by the amplitude squared $|\mathcal{M}|^2$ as a function of s and t together with other particle physics parameters) and for a given type of collision integral (CI), which primarily depends on how many χ particles participate in the process, run the following `make` instruction

```
make CItype process=model
```

Here `model.cpp` (the extension should be omitted in the `make` instruction) is the name of the particle-physics process file (see below) stored in the `processes` folder and `CItype` can have one of the two values: `annihilation` and `co-annihilation`. The executable with the name `model_processtype` is created in the `bin` folder.

6 Usage Examples

This section demonstrates how to use the `CollCalc` in various contexts.

6.1 Basic Example

Here is a simple example showing how to use the `CollCalc` package:

```
# Build an executable
make co-annihilation Process=Primakoff
# Navigate to the bin folder
cd bin/
# Run the executable with x=0.1, q=1.0 and
   rel_acc=0.1
./Primakoff_coann 0.1 1.0 0.1
# The command should display the result of the
   calculation
```

References

[1] T. Binder, T. Bringmann, M. Gustafsson and A. Hryczuk, *Early kinetic decoupling of dark matter: when the standard way of calculating the thermal relic density fails*, *Phys. Rev. D* **96** (2017) 115010, [1706.07433].

[2] M. Badziak and M. Laletin, *Precise predictions for the QCD axion contribution to dark radiation with full phase-space evolution*, 2410.18186.