MC simulation of the solar KK axions density
- Documentation -

Cyprien BEAUFORT - *beaufort@lpsc.in2p3.fr*

- Spring 2020 -

Contents

1 Installation and running

1.1 Dependencies

- C++ ($\geq$ C++11)
- CMake ($\geq$ 3.3)
- GSL (GNU Scientific Library : https://www.gnu.org/software/gsl/)
- ROOT ($\geq$ 6) (https://root.cern.ch/)

1.2 Files description

They are 3 main files in the directory:

- `AxionDensityMC.cxx`: this is the main program that will run the code. It uses multithreading (one thread for each mass considered), it then runs the MC simulations in each threads. When all threads have been joined, it reads their outputs and generates a final histogram with the density.
- `MCfunctions.h`: contains the class `AxionMC` that is used for the simulation. It also contains the function `ThreadRunner` used to properly run the threads.
- `MCfunctions.cpp`: the corresponding implementations of all methods in the class `AxionMC`.

The simulation will generate output files located in the directory `output`. It creates one file per thread (*i.e.* per mass) containing a list of trapped axions located in a 1cm^3 box, with a corresponding weight for each of them. The file `density.txt` is the concatenation of all other files, so it is the main result. The directory also contains a ROOT file with the final histogram of the KK axion density.

1.3 Set the parameters

Numerical parameters used in the simulations can all be set inside the code, calling the appropriate `Set` function. For instance, for setting the number of energy points to 10^6 one must call `axionMC->SetNenergy(1e6);` in the initialization part of `AxionDensityMC.cxx`. Default values (used to obtain the published results) are set in the constructor of the class `AxionMC`.

One must be careful when changing the parameters in order to still obtain physical results. We here describe quickly the parameters:

- `nIterMC` is the number of iterations per mass and per energy. Increasing this number improves the statistics.
- `nEnergy` is the number of energy points. It must be large enough to cover the range of trapped KK axions having an orbit that passes in the 1cm^3 box used in the code. It should be larger than 10^5 for precise simulations.
- `tMinPower` will set the minimum time to evaluate the position of the axion. It must be large compared to the period of the orbit.
- `tMaxPower` is a limit on this duration in order to accelerate the code (do not solve the EoM for too long duration). Increasing `tMaxPower` improves a bit the precision but it slows down the code.
- `boxWidthFactor` is a scaling factor for the size of the box in which we count the axions. The box must be large enough to contain statistically enough axions, but not too large in order not to bias the result. We typically use 2×10^{11}. One can play with this parameter: if it is set correctly, varying a bit `boxWidthFactor` should not modify the density obtained.
- `minVelocity` and `maxVelocity` set the range of the axion velocity. We know that for an axion to be trapped, its velocity should be lower than the escape velocity, but if it is too low the axion will not reach the Earth. We then set a range on the axion velocity to focus on a region of interest.

1.4 Running the code

First, one needs to compile the code with the command `make clean ; make`. If you encounters compiling issues please have a look to the `Makefile` to see if ROOT and GSL are correctly linked. Compiling will generate an executable file, `AxionDensityMC.bin`.

To run the simulation, execute the binary with the parameters δ, $R\,[\text{keV}^{-1}]$, and g_{10} [no unit] as arguments. For instance:
`./AxionDensityMC.bin 2 1e3 9.2e-4`

This MC simulation is very long: with the default values and 25 mass points (so 25 threads), the computing time is about 5 days.

2 Working principles

The number density of KK axions at a distance r from the Sun can be obtained from a Monte Carlo (MC) simulation. The axion density is a crucial quantity for phenomenology and we aim to cross-check the analytic calculations with the MC simulation, both approaches using a different philosophy.

The backbone of the simulation is to produce a given amount of KK axions in the Sun and to follow the trajectory of each of them by solving the Equations of Motion (EoM). In this way one knows the position of each KK axion at any time and one can compute the number density at a distance r. The number of KK axions of mass m produced by the Sun evolves in time as:

$$\frac{dN_a(t,m)}{dt} = -\Gamma(m)\, N_a(t,m) + P_a(m) \tag{1}$$

where $P_a(m)$ is the solar production rate. It implies that, at present time $t = t_\odot$,

$$N_a(m,t_\odot) = \frac{P_a(m)}{\Gamma(m)}\left(1 - e^{-t_\odot \Gamma(m)}\right) \tag{2}$$

Summing over the mode multiplicity yields to

$$N_a\big(\delta, R, g_{10}, t_\odot\big) = \frac{2\pi^{\delta/2}}{\Gamma[\delta/2]}\, R^\delta \int_0^\infty dm\, m^{(\delta-1)}\, \frac{P_a(m)}{\Gamma(m)} \left(1 - e^{-t_\odot \Gamma(m)}\right) \tag{3}$$

We already have seen that the coalescence mechanism is the main source of trapped axions. The solar axion production, per energy and time unit, through the coalescence process is given by

$$P_a(m, E) = \frac{g_{10}^2}{32\pi^2}\, m^4 \sqrt{E^2 - m^2} \int_{Sun} dr\, \frac{r^2}{e^{E/T(r)} - 1} \tag{4}$$

in which the integration is over a solar model. In this work we use the Saclay solar model [**?**]. The present total number of KK axions susceptible of being trapped is then given by

$$\begin{aligned} N_a\big(\delta, R, g_{10}, t_\odot\big) = \frac{2\pi^{\delta/2}}{\Gamma[\delta/2]}\, R^\delta \int_0^\infty dm\, \frac{m^{(\delta-1)}}{\Gamma(m)} \left(1 - e^{-t_\odot \Gamma(m)}\right) \\ \times \int_m^\infty dE\, \frac{g_{10}^2}{32\pi^2}\, m^4 \sqrt{E^2 - m^2} \int_{Sun} dr\, \frac{r^2}{e^{E/T(r)} - 1} \end{aligned} \tag{5}$$

The principle of the MC simulation is to generate n_{MC} KK axions of mass m and energy E, to follow them in their trajectory around the Sun, and to count the proportion that will be located in a $1\,\mathrm{cm}^3$ box at distance r to the Sun. Since the solar axion production is isotropic this box can be located at any position in the (ϕ, θ) plane. We then scale n_{MC} according to Eq. (5) to get the present number density of KK axions.

In order to know the position of the axion at present time $t_\odot$, the axion being produced at time $t_a \leq t_\odot$, one must solve the EoM from t_a to $t_\odot$. We want to stress out that numerically t_a does not need to be chosen in the range $]0, t_\odot]$ but instead it can be taken randomly in the range $[t_{rand}, t_\odot]$ as long as t_{rand} is large enough compared to the orbit period of the trapped axion around the Sun. The reasons are that we do not consider interactions between axions, that we assume the production rate of the Sun to be constant in time and isotropic, and that the eventual decay of the axion is already embedded in Eq. (5).

The randomness of the MC comes from the choice of the initial conditions in the EoM solver: r_0 is taken randomly in the distribution of Eq. (4) ; θ_0 and ϕ_0 are constructed in order to have an isotropic angular distribution $d\Omega = d\cos\theta_0\, d\phi_0$; and the duration of the integration t_{rand} is taken randomly in a power law distribution.