`MiMeS`: Misalignment Mechanism Solver

Dimitrios Karamitros

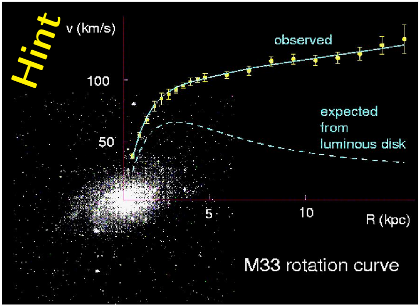
Manchester U.

23/11/2021

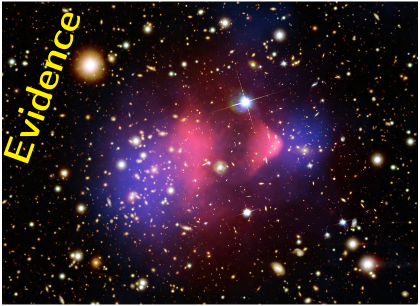
El Journal Club más Sabroso

Outline

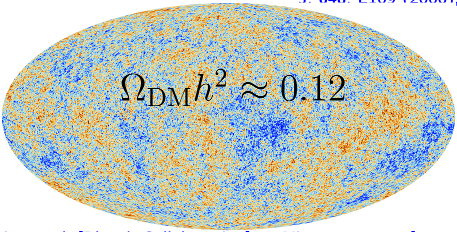
Why particle dark matter



E. Corbelli and P. Salucci, *Mon. Not. Roy. Astron. Soc.* **311** *441* (2000), arXiv:astro-ph/9909252.



M. Markevitch, ESA Spec. Publ. **604** (2006) 723, astro-ph/0511345.Clowe, Bradac, *et. al.* Astrophys. J. **648**, L109 (2006), astro-ph/0608407



N. Aghanim *et al.* [Planck Collaboration], arXiv:1807.06209 [astro-ph.CO].

"Ἓν οἶδα, ὅτι οὐδὲν οἶδα."
"I know one thing, that I know nothing."

–Socrates

The dark matter particle

"Ἓν οἶδα, ὅτι οὐδὲν οἶδα."
"I know one thing, that I know nothing."

–Socrates

- Gravitational interactions.
- Mostly electrically neutral.
- Stable or very slow decay rate.
- Non-Baryonic.
- Cold/Warm and non-relativistic today.

The axion (like) particle

Notably, the original Axion was originally introduced in order to solve the *strong-CP problem* of the SM. Axion-Like-Particles (ALPs) arise in a number of new physics models, beyond the SM.

Axions and ALPs generally:

- Have suppressed interactions with photons.
- Are (mostly) stable.
- Were non-relativistic around the epoch of structure formation.

The axion (like) particle

Notably, the original Axion was originally introduced in order to solve the *strong-CP problem* of the SM. Axion-Like-Particles (ALPs) arise in a number of new physics models, beyond the SM.

Axions and ALPs generally:

- Have suppressed interactions with photons.
- Are (mostly) stable.
- Were non-relativistic around the epoch of structure formation.
- Non-baryonic (by definition), and of-course interact gravitationally.

The axion (like) particle

Notably, the original Axion was originally introduced in order to solve the *strong-CP problem* of the SM. Axion-Like-Particles (ALPs) arise in a number of new physics models, beyond the SM.

Axions and ALPs generally:

- Have suppressed interactions with photons.
- Are (mostly) stable.
- Were non-relativistic around the epoch of structure formation.
- Non-baryonic (by definition), and of-course interact gravitationally.

Maybe DM has Axionic nature!

Calculating the Relic Abundance

The axion EOM

Axions and ALPs follow a similar equation of motion (EOM):

$$\left(\frac{d^2}{dt^2} + 3H(t)\ \frac{d}{dt}\right)\theta(t) + \tilde{m}_a{}^2(t)\ \sin\theta(t) = 0\ ,$$

where $\theta = A\,f_a$, with A the axion filed, and f_a some energy scale that characterises the potential (Peccei-Quinn breaking scale).

How hard can it be?

Hard (in general).

How hard can it be?

Hard (in general).

The classical analogue is the dumped pendulum with both frequency (length) and friction being time-dependent:

- There is no closed form solution.
- There are no constants of motion (wait a minute).
- No package/library/program available!

Hard (in general).

The classical analogue is the dumped pendulum with both frequency (length) and friction being time-dependent:

- There is no closed form solution.
- There are no constants of motion (wait a minute).
- No package/library/program available!

`MiMeS` *simulates the evolution of the Axion/ALP, for (virtually) any cosmological scenario and Axion/ALP (thermal) mass.*

Initial conditions

Some time at the very early Universe, $\tilde{m}_a \ll H(T)$, [1] with

$$\ddot{\theta} + 3H\ \dot{\theta} \approx 0\ .$$

The solution is

$$\theta = \theta_{\rm ini} + C \int_0^t dt' \ \left(\frac{a(t'=0)}{a(t')} \right)^3 \ .$$

So, $\dot{\theta} \sim a^{-3}$. Since we are interested in θ at much later times (once the potential becomes relevant), $\dot{\theta} \approx 0$. [2] Therefore, we can start integration at some point $(t = t_{\rm ini})$ with $3H \gg \tilde{m}_a$, and set $\theta(t = t_{\rm ini}) = \theta_{\rm ini}$ and $\dot{\theta}(t = t_{\rm ini}) = 0$.

---

[1] This is an assumption that `MiMeS` has to make, for the sake of generality.

[2] Standard misalignment mechanism. For the kinetic one see R. T. Co, L. J. Hall and K. Harigaya, Phys. Rev. Lett. 124 (2020) no.25, 251802 [arXiv:1910.14152 [hep-ph]], C. F. Chang and Y. Cui, Phys. Rev. D 102 (2020) no.1, 015003 [arXiv:1911.11885 [hep-ph]], or B. Barman, N. Bernal, N. Ramberg and L. Visinelli, [arXiv:2111.03677 [hep-ph]] .

(Bad) Analytical approximations

Once we agree on the initial conditions, we move to the next important things:

[3] Defined from $s(T_0) = \gamma \ {a_{\rm osc}}^3 \ s_{\rm osc}$.

(Bad) Analytical approximations

Once we agree on the initial conditions, we move to the next important things:

- Assume $\theta \ll 1$, and linearise the EOM. *Not always that bad.*

[3] Defined from $s(T_0) = \gamma \; {a_{\rm osc}}^3 \; s_{\rm osc}$.

(Bad) Analytical approximations

Once we agree on the initial conditions, we move to the next important things:

- Assume $\theta \ll 1$, and linearise the EOM. *Not always that bad.*
- Assume that at $\tilde{m}_a(T_{\rm osc}) \approx 3H(T_{\rm osc})$, the oscillations begin with $\dot{\theta}(T_{\rm osc}) = 0$. *This is in general quite bad.*

[3] Defined from $s(T_0) = \gamma \; {a_{\rm osc}}^3 \; s_{\rm osc}$.

(Bad) Analytical approximations

Once we agree on the initial conditions, we move to the next important things:

- Assume $\theta \ll 1$, and linearise the EOM. *Not always that bad.*
- Assume that at $\tilde{m}_a(T_{\rm osc}) \approx 3H(T_{\rm osc})$, the oscillations begin with $\dot{\theta}(T_{\rm osc}) = 0$. *This is in general quite bad.*
- Assume that $\theta_{\rm osc} \approx \theta_{\rm ini}$. *Generally quite bad.*

[3] Defined from $s(T_0) = \gamma \; {a_{\rm osc}}^3 \; s_{\rm osc}$.

(Bad) Analytical approximations

Once we agree on the initial conditions, we move to the next important things:

- Assume $\theta \ll 1$, and linearise the EOM. *Not always that bad.*
- Assume that at $\tilde{m}_a(T_{\rm osc}) \approx 3H(T_{\rm osc})$, the oscillations begin with $\dot{\theta}(T_{\rm osc}) = 0$. *This is in general quite bad.*
- Assume that $\theta_{\rm osc} \approx \theta_{\rm ini}$. *Generally quite bad.*

Then, we get the "WKB"-approximate solution

$$\theta(t) \approx \theta_{\rm ini} \left(\frac{3}{4}\right)^{1/4} \sqrt{\frac{\tilde{m}_a(T_{\rm osc})}{\tilde{m}_a(T)}} \left(\frac{a}{a_{\rm osc}}\right)^{-3/2} \cos\left(\int_{t_{\rm osc}}^{t} dt' \; \tilde{m}_a(t')\right) .$$

The advantage of this approximation is that we get an easy formula for the axion/ALP energy density today:

$$\rho_{a,0} = \gamma^{-1} \frac{s_0}{s_{\rm osc}} \, \frac{1}{2} \, {f_a}^2 \, m_a \, \tilde{m}_{a,{\rm osc}} \, {\theta_{\rm ini}}^2 \, ,$$

where γ the amount of entropy injection between $T_{\rm osc}$ and today. [3]

[3] Defined from $s(T_0) = \gamma \; {a_{\rm osc}}^3 \; s_{\rm osc}$.

Need for accuracy, speed, and automation

Serious disadvantages of the approximate results:

- The approximations can be tested against numerical results in a case-by-case basis; there is no way to tell if they will work in new models and cosmological scenarios.
- There is no available tool that can help us reproduce published results obtained by numerical integration; people use their own ***private*** code.
- If someone wants to simply see if an ALP model is compatible with a cosmological scenario, they have to develop their own private code; the overall effort of the community increases.

`MiMeS`:

- *Easy* to use; anyone can run it and see if their model can work.
- Reasonably fast; less than $0.05\ s$ for the scenarios tested.
- Tools that can help determine if the algorithm is accurate enough.
- The user provides too much input. This helps the user determine is the convergence of the algorithm is consistent.

MiMeS

What *is* `MiMeS`?

From the software point of view:

What *is* `MiMeS`?

From the software point of view:

- `MiMeS` is a `C++` header-only library that contains various templated classes; there is no “installation" and no special procedures, just include the header files.

What *is* `MiMeS`?

From the software point of view:

- `MiMeS` is a `C++` header-only library that contains various templated classes; there is no "installation" and no special procedures, just include the header files.
- `MiMeS` comes with a `python` interface so that everybody can use it.

What *is* MiMeS?

From the software point of view:

- MiMeS is a C++ header-only library that contains various templated classes; there is no "installation" and no special procedures, just include the header files.
- MiMeS comes with a python interface so that everybody can use it.
- MiMeS is distributed under the MIT license; you can do whatever you want with it, and I am *not* responsible.

What *is* MiMeS?

From the software point of view:

- MiMeS is a C++ header-only library that contains various templated classes; there is no "installation" and no special procedures, just include the header files.
- MiMeS comes with a python interface so that everybody can use it.
- MiMeS is distributed under the MIT license; you can do whatever you want with it, and I am *not* responsible.

MiMeS: under the hood

MiMeS relies on NaBBODES [4] for the numerical integration, and SimpleSplines [5] for the various interpolations.

Advantages:

- You only need to have the standard C++ library.
- The two libraries are developed by myself, so their integration with MiMeS is seamless.
- There is always going to be a compatible version of these libraries that works with MiMeS.

Disadvantages:

- These are not well tested libraries.
- No community of contributors; if it doesn't work, I have to fix it.
- Slow development.

[4] https://github.com/dkaramit/NaBBODES.
[5] https://github.com/dkaramit/SimpleSplines.

MiMeS: Notation

`MiMeS` uses a notation suitable (any) underlying cosmology, since it is up to the user to define the cosmological evolution. First, we define

$$u \equiv \log\left(a/a_{\rm ini}\right) \;,$$

with $a_{\rm ini}$ some initial value of the scale factor. [6] in order to express the time derivatives as

$$\frac{d}{dt} \to H \frac{d}{du} \;, \quad \frac{d^2}{dt^2} \to H^2 \; \left(\frac{d^2}{du^2} + \frac{1}{2} \frac{d \log H^2}{du} \frac{d}{du} \right) \;.$$

Then, we express the EOM as a system of first order ordinary differential equations

$$\frac{d\zeta}{du} + \left[\frac{1}{2} \frac{d \log H^2}{du} + 3 \right] \zeta + \; \left(\frac{\tilde{m}_a}{H} \right)^2 \; \sin\theta = 0 \;.$$
$$\frac{d\theta}{du} - \zeta = 0 \;.$$

Observe that, by definition, $\zeta = d\theta/du$. The initial conditions are $\zeta(0) = 0$ and $\theta(0) = \theta_{\rm ini}$.

[6] Only the ratios $a/a_{\rm ini}$ appear in the calculations. So, the choice does not matter as long as it is consistent.

MiMeS: When do you start and stop integrating?

The choice of a good starting point is important, as need to start at a temperature where $\zeta = 0$ is a good approximation. So you can start at some $T_{\rm ini}$ with a given ratio $3H(T_{\rm ini})/\tilde{m}_a(T_{\rm ini}) \gg 1$. This needs to be chosen carefully, as low values of $3H(T_{\rm ini})/\tilde{m}_a(T_{\rm ini})$ result in inaccurate result, while high values may result in a slow calculation. *Advice*: Use various values of that ratio, and find where the relic abundance becomes $T_{\rm ini}$-independent.

The stopping condition is more difficult. You should stop at some point where the axion/ALP evolves "adiabatically". Find a quantity that becomes constant as the system relaxes, and use this to determine when adiabaticity is reached. Once θ starts to evolve adiabatically, the amplitude of its oscillation is known at later times!

If a system exhibits closed orbits, the quantity

$$J \equiv C \oint p \, d\theta \; ,$$

is the adiabatic invariant. In this case, it becomes

$$J = a^3 \; \tilde{m}_a \; {\theta_{\rm peak}}^2 \, f(\theta_{\rm peak}) \; ,$$

with

$$f(\theta_{\rm peak}) = \frac{2\sqrt{2}}{\pi {\theta_{\rm peak}}^2} \int_{-\theta_{\rm peak}}^{\theta_{\rm peak}} d\theta \sqrt{\cos\theta - \cos\theta_{\rm peak}} \; ,$$

the so-called anharmonic factor.

If a system exhibits closed orbits, the quantity

$$J \equiv C \oint p \, d\theta \,,$$

is the adiabatic invariant. In this case, it becomes

$$J = a^3 \; \tilde{m}_a \; {\theta_{\rm peak}}^2 \, f(\theta_{\rm peak}) \,,$$

with

$$f(\theta_{\rm peak}) = \frac{2\sqrt{2}}{\pi {\theta_{\rm peak}}^2} \int_{-\theta_{\rm peak}}^{\theta_{\rm peak}} d\theta \sqrt{\cos\theta - \cos\theta_{\rm peak}} \,,$$

the so-called anharmonic factor.

Important: $\theta_{\rm peak}$ is the peak of the oscillation. So, J can be used to determine how $\theta_{\rm peak}$ changes with time. By definition, at $\theta = \theta_{\rm peak}$, $p \sim \dot{\theta} = 0$. This means that we can find $\rho_{a,0}$ on the peak of today's θ, as

$$\rho_{a,0} = \gamma^{-1} \; \frac{s_0}{s_*} \; m_a \; \tilde{m}_{a,*} \; \frac{1}{2} \; {f_a}^2 \; \theta_{{\rm peak},*}^2 \;\; f(\theta_{{\rm peak},*}) \,,$$

where T_* the temperature at which adiabaticity was reached, and γ the entropy injection between T_* and today (*i.e.* $s_0 = \gamma a_*^3 s_*$).

Using MiMeS

How to get MiMeS

There are several ways you can get a stable version of `MiMeS`:

1. `git clone -b stable https://github.com/dkaramit/MiMeS.git`. This is the preferred way, as it is guaranteed to be the latest stable version.
2. Go to mimes.hepforge.org/downloads, and download it.
3. Go to github.com/dkaramit/MiMeS/releases, and download a released version.

You can get the most up-to-date code – not always the most stable one – including the latest version of `NaBBODES` and `SimpleSplines`, by running

```
git clone https://github.com/dkaramit/MiMeS.git
cd MiMeS
git submodule init
git submodule update --remote
```

Configure (and make)

There is no need to install anything if you are going to use `MiMeS` in a `C++` program. The only thing you *must* do is run

```
bash configure.sh
```

However, you can run Commend from D: List all `make` ... that make useful things.

Know your constructors: `mimes::AxionMass<LD>`

Commend from D: Show the constructor for `mimes::AxionMass`, and discuss the input.

Know your constructors: `mimes::Axion<LD,Solver,Method>`

Commend from D: Show the constructor for `mimes::Axion`, and discuss the input.

What `MiMeS` expects from you

Commend from D: Discuss template arguments

MiMeS from python

Commend from D: Discuss `Definitions.mk` and show that the constructors are the same as in `C++`

Examples

Commend from D: can you show example code?

Commend from D: can you do a live presentation of `MiMeS` in jupyter?

Outlook

Summing up

Commend from D: blah blah, what you did

Future

Commend from D: Freeze-out/in, kinematic misalignment mechanism, automatically compare with searches.

Thank you!

Backup

(equations, derivations, tables)

Adiabatic invariant

Input – mimes::AxionMass<LD>

Input – `mimes::Axion<LD,Solver,Method>`