Physics of the Dark Universe 5–6 (2014) 18–28

Contents lists available at ScienceDirect

Physics of the Dark Universe

journal homepage: www.elsevier.com/locate/dark

MadDM v.1.0: Computation of dark matter relic abundance using MadGraph 5

Mihailo Backović [a,*], Kyoungchul Kong [b], Mathew McCaskey [b]

[a] Department of Particle Physics and Astrophysics, Weizmann Institute of Science, Rehovot 76100, Israel

[b] Department of Physics and Astronomy, University of Kansas, Lawrence, KS 66045, USA

ARTICLE INFO

Keywords:
Beyond standard model
MadGraph
Dark matter
Relic density
Numerical tools

ABSTRACT

We present MadDM v.1.0, a numerical tool to compute dark matter relic abundance in a generic model. The code is based on the existing MadGraph 5 architecture and as such is easily integrable into any MadGraph collider study. A simple Python interface offers a level of user-friendliness characteristic of MadGraph 5 without sacrificing functionality. MadDM is able to calculate the dark matter relic abundance in models which include a multi-component dark sector, resonance annihilation channels and co-annihilations. We validate the code in a wide range of dark matter models by comparing the relic density results from MadDM to the existing tools and literature.

1. Introduction

A high demand for computational tools in collider physics resulted in a myriad of monte-carlo event generators. Much care and effort has also been dedicated to link mass spectrum generators, detector simulators, and parton showering models with the existing tools. On the other hand, the experimental effort of dark matter (DM) searches sparked a demand for another type of numerical tool aimed at dark matter phenomenology. The gap between dark matter tools and collider tools has not been completely bridged, with very few numerical packages offering the ability to perform both collider and dark matter calculations.

Perhaps the only set of tools able to integrate dark matter phenomenology with collider physics is CalcHEP [1] and micrOMEGAs [2], while other popular collider tools such as MadGraph [3] and Sherpa [4] lack this ability. In addition, model specific dark matter tools such as DarkSusy [5] allow for detailed calculations of galactic dark matter relic signals in the framework of super-symmetric models but without the ability to *easily* integrate into collider tools.

MadDM aims to bridge the gap between collider oriented event generators and dark matter physics tools. We chose to build MadDM on the existing MadGraph 5 infrastructure for two reasons. First, MadGraph is widely used among the experimental collaborations in searches for Beyond the Standard Model (BSM) physics and we wish to provide them with the ability to easily incorporate relic density constraints into searches for models which support a dark matter candidate. Second, the Python implementation of the MadGraph 5 code allows for a fairly straightforward development of MadGraph add-ons without sacrificing functionality.

Very much like MadGraph/MadEvent, the MadDM code is split into the process generating module written in Python and a FORTRAN numerical module for calculating relic abundance. The Python module automatically determines the dark matter candidates from a user specified model and generates relevant dark matter annihilation diagrams. The numerical FORTRAN code then solves the dark matter density evolution equation for a given parameter set and outputs the resulting relic density. We provide a simple, MadGraph 5 inspired interface for the convenience of the user, as well as the ability to perform multi-dimensional parameter scans.

MadDM is compatible with any existing MadGraph 5 model, as well as model-specific mass spectrum or decay width calculator which can produce a Les Houches formatted MadGraph parameter card. The code is able to handle the full effects of co-annihilations and resonant annihilation in a generic model. In models which contain more than one dark matter particle, MadDM can solve either the full set of coupled density evolution equations or a single equation involving effective thermally averaged cross sections. The former gives MadDM the ability to calculate relic density in models which allow for dark matter conversions [6].

We use micrOMEGAs as a benchmark for MadDM validation. Standard Model extensions with singlet scalar fields serve as

* Corresponding author.
E-mail addresses: mihailo.backovic@weizmann.ac.il (M. Backović), kckong@ku.edu (K. Kong), mccaskey@ku.edu (M. McCaskey).

http://dx.doi.org/10.1016/j.dark.2014.04.001

simple tests of the MadDM performance in models which feature resonant annihilation channels and co-annihilations. In addition, we also consider more complex models such as the Minimal Supersymmetric Standard Model (MSSM) and Minimal Universal Extra Dimensions (MUED). We find good agreement between MadDM and micrOMEGAS in a wide range of dark matter models, with the exception of regions of the MSSM parameter space where details of the running b mass can cause large effects.

We begin in Section 2 by reviewing relic abundance calculation and introduce MadDM in Section 3. We reserve Section 4 for validation and include an example for MadDM Python script in the Appendix.

2. Calculation of relic abundance

2.1. The density evolution equation with co-annihilations

We begin with a simple case in which a single dark matter particle is stable and annihilates only with itself,[1] while we postpone the discussion of a more general treatment of dark matter relic density until the following sections. The calculation of relic abundance for a dark matter candidate χ is governed by the density evolution equation

$$\frac{dn_\chi}{dt} + 3Hn_\chi = -2\langle\sigma_{\rm eff}v\rangle\left(n_\chi^2 - (n_\chi^{EQ})^2\right), \tag{1}$$

where n_X is the number density of dark matter particles, and H is the Hubble expansion constant. The indexes i, j run over all the co-annihilating particles partners in the model, while $\langle\sigma_{\rm eff}|v|\rangle$ is the effective thermally averaged annihilation cross section[2]

$$\langle\sigma_{\rm eff}v\rangle \equiv \sum_{i,j=1}^{N}\langle\sigma(\chi_i\chi_j \to SM)v\rangle\frac{n_{\chi_i}^{EQ}n_{\chi_j}^{EQ}}{(n_\chi^{EQ})^2},$$

$$\langle\sigma_{\chi\bar\chi\leftrightarrow X\bar X}|v|\rangle = \frac{2Tm_\chi^2}{(2\pi)^4 n_{eq}^2(1+\delta_{\chi\bar\chi})}\int_0^1 d\beta\,\frac{\beta}{(1-\beta^2)^2} \times\sqrt{\frac{\lambda(s,m_\chi^2,m_{\bar\chi}^2)}{s}}K_1\left(\frac{\sqrt{s}}{T}\right)W_{ij}(s), \tag{2}$$

where k runs over all the particles in the process, while n_χ^{EQ} is the thermal equilibrium density of the dark matter species with mass m_χ and g_χ internal degrees of freedom at temperature T:

$$n_\chi^{EQ} \approx g_\chi\left(\frac{m_\chi T}{2\pi}\right)^{3/2}e^{-m_\chi/T}. \tag{3}$$

K_1 is the modified Bessel function of the second kind, $\beta \equiv \sqrt{1-4m_\chi^2/s}$ is the relativistic velocity of the initial state particles, while the numerically convenient W_{ij} proxy is defined as:

$$W_{ij}(s) = \frac{\sqrt{\lambda(s,m_X^2,m_{\bar X}^2)}}{(1+\delta_{X\bar X})8\pi s}\int\sum_{\rm spins}|\mathcal{M}|^2\left(\frac{d\Omega_{CM}}{4\pi}\right), \tag{4}$$

where $\lambda(a,b,c)$ is the usual two body kinematic function:

$$\lambda(a,b,c) \equiv a^2+b^2+c^2-2(ab+bc+ac). \tag{5}$$

Notice that the definition of effective thermally averaged cross section reduces to $\langle\sigma(\chi\chi \to SM)|v|\rangle$ in case of only one dark matter particle and no-coannihilating partners.

To calculate the dark matter relic abundance one must solve Eq. (1). Due to the presence of both n and n^2 terms, the numeric solution to Eq. (1) in the present form is not practical. Tracking the number density of dark matter particles per unit entropy ($Y \equiv n_\chi/s$) instead of n_χ,[3] as well as substituting temperature for $x \equiv m_\chi/T$ allows for Eq. (1) to be cast into a simpler form which MadDM uses to solve for relic density:

$$\frac{dY}{dx} = -\sqrt{\frac{\pi}{45\,g_*}}\frac{g_{*S}\,m_{pl}\,m_\chi}{x^2}\langle\sigma_{\rm eff}|v|\rangle[Y^2 - Y_{EQ}^2], \tag{6}$$

$$Y_{EQ}(x) \approx \frac{45}{4\pi^4}\sqrt{\frac{\pi}{2}}\frac{g}{g_{*S}}x^{3/2}e^{-x} \quad (x \gg 1). \tag{7}$$

It is possible to find approximate numerical solutions to Eq. (6) by integrating the equation from the freeze-out temperature x_f to infinity, whereas a more precise solution can be obtained by fully integrating Eq. (6) (as explained in Section 2.2). The result of the calculation is Y_∞, the number of dark matter particles per unit entropy in the present universe. Y_∞ is related to the relic density parameter Ωh^2 as

$$\Omega h^2 \equiv \frac{\rho_\chi}{\rho_c} = \frac{m_\chi\,Y_\infty\,s_\infty}{1.05\times10^{-5}\,{\rm GeV}^2\,{\rm cm}^{-3}}, \tag{8}$$

where ρ_c is the critical energy density of the universe, ρ_χ is the energy density of dark matter and $s_\infty \approx 2889.2\,{\rm cm}^{-3}$ is the entropy density in the present universe.

2.2. Thermal freeze-out and the numerical integration of the density rate equation

The density evolution equation should, in principle, be integrated from the beginning of the Universe (appropriately at $t = 0$) to today ($t = \infty$). In a numerical sense this is not possible, much less practical. As we will show momentarily, the differential equation only needs to be integrated over a much smaller range.

The thermal evolution of the dark matter in the early universe, $x \sim O(1)$, is characterized by a rapid dark matter annihilation rate

$$\Gamma_\chi \equiv \langle\sigma_A|v|\rangle\,n_\chi^{EQ}. \tag{9}$$

The expansion rate of the universe (characterized by the Hubble parameter H) in the low x regime is much lower than Γ_χ due to the large suppression of $1/m_{pl}$. In addition, the dark matter annihilation rate at high temperature is compensated by the inverse process (creation rate), thus guaranteeing that dark matter remains in thermal equilibrium; *i.e.* $Y \approx Y_{EQ}$.

As the universe cools down and x increases, the annihilation rate Γ_χ obtains an exponential suppression. At some point in thermal evolution, $H > \Gamma_\chi$, meaning that the annihilation rate effectively stops and the number density of dark matter becomes constant. The so-called "thermal freeze-out" occurs roughly at

$$H \approx \Gamma_\chi, \tag{10}$$

with the resulting temperature called the "freeze-out temperature".[4]

[1] The case where dark matter has a unique anti-particle is a straightforward extension.

[2] The factor of 2 represents the fact that there are either two dark matter particles being created or destroyed in the annihilation/creation process. This factor is usually canceled with an additional factor of 1/2 which comes from double counting in the initial state phase space. Our definition of the thermally averaged annihilation cross section will explicitly have this factor for clarity.

[3] The change of variables assumes that the universe undergoes an adiabatic expansion

[4] Note that the in cases in which $\langle\sigma_A|v|\rangle$ has a non-trivial dependence on x, the temperature at which dark matter density Y decouples from the equilibrium distribution and the temperature at which the relic density freezes out can be different. Resonant dark matter annihilation is a notable example, where large resonance widths can prolong the efficient dark matter annihilation rates [7–9].

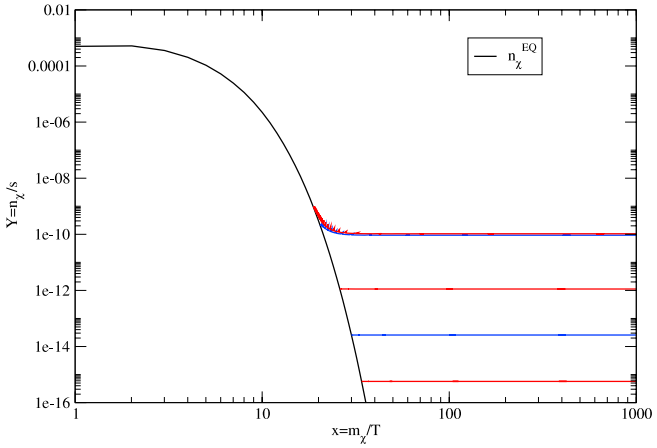

Fig. 1. Practical integration of the density evolution equation without knowing the freeze-out temperature before hand. The alternating red and blue lines show the solution starting at smaller initial values for $x = m_\chi/T$. Once two consecutive solutions are within a specified percent error the solution is considered the correct one. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

The above argument implies that it is not necessary to integrate the rate equation from $x = 0$, but only from the freeze-out time x_f to some large time, which we take to be $x = 1000$. Since we do not know *a priori* at what temperature the DM candidate will freeze out we take the following approach that both finds the freeze-out temperature and gives the correct solution to the density evolution equation today. We start the integration at a temperature x_i that is generally far below the freeze-out temperature at (i.e. $x = 30$) and calculate Y_∞. We then step back the starting temperature and again calculate Y_∞. This process is repeated until there are two consecutive values for Y_∞ which are within a certain precision from each other defined by the parameter:

$$\frac{Y_\infty(i) - Y_\infty(i-1)}{Y_\infty(i)} \leq \texttt{eps\_ode},$$

where i marks a particular starting temperature x_i. Fig. 1 shows an example calculation of the freeze-out temperature. Starting with a large x_i the code continues to step back until two consecutive values of Y_∞ are within `eps_ode` from each other. This method of calculating relic density is numerically more stable and faster than starting at $x = 1$, since precision issues of small dY/dx are avoided.

2.3. General density evolution equation

A generic DM model can in principle contain any number of particles and types of interactions. To calculate the relic abundance in such a model, one needs to track the densities of all DM candidates as well as all possible co-annihilation partners $(\chi_1, \ldots, \chi_n)$,[5] while the rest of the particles are assumed to be in thermal equilibrium or have already decayed to $\chi_1 \ldots \chi_n$.[6]

In the most general case, one needs to categorize the interactions that will affect the dark matter particle densities, as only the interactions which change the number of dark matter particles are relevant. We can classify them into three different groups:

- DM pair annihilations into SM particles: $\chi_i\chi_j \to SM$.
- DM particle scattering off of a thermal background particle: $\chi_i X \to \chi_j Y$.
- Decays of one DM species to another.

Next, with all the interactions categorized we can write down the set of density evolution equations:

$$\begin{aligned}
\frac{dn_{\chi_i}}{dt} &= -3Hn_{\chi_i} - \sum_j (1+\delta_{ij})\langle\sigma(\chi_i\chi_j \leftrightarrow SM)|v|\rangle \\
&\quad \times \left[n_{\chi_i} n_{\chi_j} - n^{EQ}_{\chi_i} n^{EQ}_{\chi_j}\right] \\
&\quad - \sum_{j\neq i}\langle\sigma(\chi_i X \leftrightarrow \chi_j Y)|v|\rangle \left[n_{\chi_i} - \frac{n^{EQ}_{\chi_i}}{n^{EQ}_{\chi_j}} n_{\chi_j}\right] n^{EQ}_{\chi_i} \\
&\quad - \sum_{j,k,l}(1+\delta_{ij}-\delta_{ik}-\delta_{il})\langle\sigma(\chi_i\chi_j \leftrightarrow \chi_k\chi_l)|v|\rangle \\
&\quad \times \left[n_{\chi_i} n_{\chi_j} - \frac{n^{EQ}_{\chi_i} n^{EQ}_{\chi_j}}{n^{EQ}_{\chi_k} n^{EQ}_{\chi_l}} n_{\chi_k} n_{\chi_l}\right] \\
&\quad - \sum_{j\neq i}\langle\Gamma(\chi_i \to \chi_j X)\rangle n_{\chi_i} + \langle\sigma(\chi_j X \to \chi_i)|v|\rangle n_{\chi_j} n^{EQ}_{\chi_j} \\
&\quad + \sum_{j\neq i}\langle\Gamma(\chi_j \to \chi_i X)\rangle n_{\chi_j} - \langle\sigma(\chi_i X \to \chi_j)|v|\rangle n_{\chi_i} n^{EQ}_{\chi_i}, \quad (11)
\end{aligned}$$

where n_{χ_i} is the number density of the ith dark matter species, $n^{EQ}_{\chi_i}$ is the number density of the ith dark matter species in thermal equilibrium defined in Eq. (3). The indices j, k, and l in Eq. (11) run over all of the DM species. The coefficient for each term in Eq. (11) is either a thermally averaged cross section or thermally averaged decay rate. Both terms are similarly constructed, with thermally averaged annihilation section defined in Eq. (2) and decay rate being

$$\begin{aligned}
\langle\Gamma(\chi_i \to SM)\rangle &= \frac{1}{n^{EQ}_{\chi_i}} \int \prod_k \frac{d^3p_k}{(2\pi)^3}\frac{1}{2E_k} \sum_{\text{all spins}} |\mathcal{M}(\chi_i \to \text{SM})|^2 \\
&\quad \times \delta^4(p_1 - p_2 - p_3) e^{-E_1/T}, \quad (12)
\end{aligned}$$

where the product is over all the external momenta of the process.

The first term on the right-hand side of Eq. (11) corresponds to the DM pair annihilation processes into SM particles. The second and third lines describe the DM thermal scattering processes. The final two lines correspond to the decay and creation processes of the unstable DM particles. The factors containing the Kronecker deltas are designed to properly count the number of interactions.[7] Notice that the term responsible for semi-annihilations is absent from Eq. (11), the treatment of which we plan to include in the future versions of our code. The current version of `MadDM` also omits any dark matter decay terms.

2.4. Multi-component dark matter with no co-annihilations

Many scenarios exist in which it is not possible to cast a system of coupled rate equations into a convenient form of Eq. (1). Such scenarios could arise in models with multiple DM particles that do not contain any canonical co-annihilation diagrams (i.e. $\chi_i\chi_j \leftrightarrow SM$ with $i \neq j$), as well as the thermal background scattering diagrams which are responsible for establishing the thermal

[5] We will refer to the particles in this list as a whole as DM particles.

[6] For most practical examples the "rest of the particles" simply refer to to SM particles.

[7] For self-annihilating particles there would be a factor of 2 to properly count the interaction rate. Several papers note that this is counteracted by a factor of 1/2 to avoid double counting the initial state phase space in the thermally averaged annihilation cross section for identical particles. Both factors are then canceled and subsequently ignored [5,7,10].

equilibrium between the DM species. In this case, the assumptions necessary to cast a system of coupled density evolution equations into Eq. (1) are invalid and one is forced to track the density of all the DM species individually. If we assume that there are no DM decays, we can write down the coupled set of density evolution equations as

$$\begin{aligned}\frac{dn_{\chi_i}}{dt} &= -3Hn_{\chi_i} - \sum_j (1+\delta_{ij})\langle\sigma(\chi_i\chi_j \leftrightarrow SM)|v|\rangle \\ &\times \left[n_{\chi_i}n_{\chi_j} - n^{EQ}_{\chi_i}n^{EQ}_{\chi_j}\right] \\ &- \sum_{j,k,l}(1+\delta_{ij}-\delta_{ik}-\delta_{il})\langle\sigma(\chi_i\chi_j \leftrightarrow \chi_k\chi_l)|v|\rangle \\ &\times \left[n_{\chi_i}n_{\chi_j} - \frac{n^{EQ}_{\chi_i}n^{EQ}_{\chi_j}}{n^{EQ}_{\chi_k}n^{EQ}_{\chi_l}}n_{\chi_k}n_{\chi_l}\right]. \end{aligned} \quad (13)$$

Eq. (13) tell us that the conversion rate of dark matter particles can have large effects on thermal evolution of dark matter, if $\langle\sigma(\chi_i\chi_j \leftrightarrow \chi_k\chi_l)|v|\rangle \sim \langle\sigma(\chi_i\chi_j \leftrightarrow SM)|v|\rangle$. This condition can be easily arranged in models with multiple particles in the dark sector (see Ref. [6] for a recent study of such as scenario).

For the purpose of illustration, consider the following dark sector:

$$\begin{aligned}\mathcal{L}_{DM} \supset\ & \frac{m^2_{X_1}}{2}X_1^2 + \frac{m^2_{X_2}}{2}X_2^2 + \frac{d_1}{4}X_1^4 + \frac{d_2}{4}X_1^2X_2^2 + \frac{d_3}{4}X_2^4 + \frac{\delta_1}{2}H^\dagger H X_1^2 \\ &+ \frac{\delta_2}{2}H^\dagger H X_2^2. \end{aligned} \quad (14)$$

The two scalars, $X_{1,2}$, interact with the rest of the SM particles only through the Higgs via the δ couplings and with each other through the d couplings. Notice the absence of terms proportional to proportional to $H^\dagger H X_1 X_2$, which can easily be arranged by introducing discrete symmetries on the dark matter fields. Such terms would induce mixing between the X_1 and X_2 states after the Higgs obtains its vacuum expectation value, as well as generate the canonical co-annihilation diagrams between the two DM eigenstates X_1' and X_2'. Both $X_{1,2}$ in the toy model are stable so there are no decay diagrams to consider. The $DM \rightarrow DM$ scattering diagrams are mediated by the d_i couplings where the interesting interaction comes from the term proportional to d_2.

The model in Eq. (14) is an example of a scenario in which the annihilation rate into light states can be of the same order or smaller than the conversion rate of $X_1 \leftrightarrow X_2$. In the parameter space where the conversion is of the same order as the annihilation rate, the relic density calculation can not proceed according to the rule of Eq. (1). Thus, it is necessary to solve to full set of coupled differential equations of Eq. (13).

To show this numerically we choose the following parameter space point.

$$\begin{aligned} &m_{X_1} = 200\ \text{GeV}, \qquad m_{X_2} = 220\ \text{GeV}, \\ &\delta_1 = 1, \qquad \delta_2 = 0.01. \end{aligned} \quad (15)$$

Taken by themselves, X_1 interacts strongly with the Higgs and thus will be underproduced in the early universe while X_2 is very weakly coupled to the Higgs which would cause over-production. Fig. 2 shows the calculation of the relic density in this scenario as a function of d_2, the coupling which sets the strength of the conversion rate $X_1 \leftrightarrow X_2$. In the limit of $d_2 \rightarrow 0$, the two dark matter particles are decoupled and the relic density is simply a sum of the relic densities of X_1 and X_2 which is mainly dominated by the density of X_2 (solid black line). In the limit of $d_2 \rightarrow \infty$ the high conversion rate keeps X_1 and X_2 in thermal equilibrium and the density evolution is well approximated by Eq. (1) (solid red line). However, there is a large region of parameter space in which relic density can take any value in-between the coupled and canonical co-annihilation regime (green line), illustrating that the relic density in this parameter region must be computed using Eq. (13).

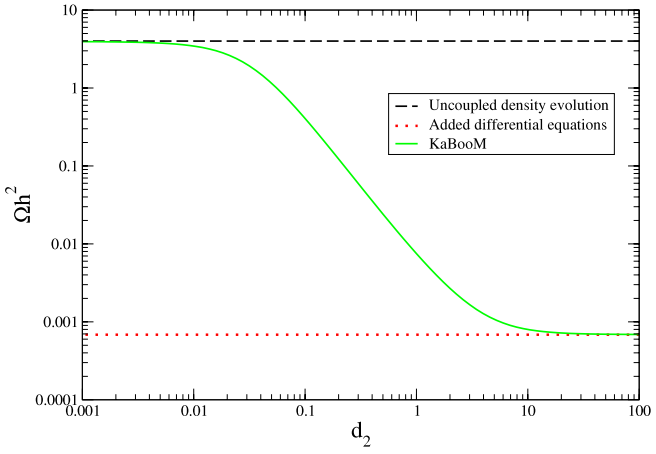

Fig. 2. Relic density calculation scanned over the strength of the d_2 coupling. With a very small d_2 the relic density matches that obtained if the two density evolution equations are solved separately (black line). Larger d_2 couplings give a relic density that is closer to the canonical co-annihilation result that assumes strong thermal scattering interactions (red line). The green line is the result of solving the coupled system of Eq. (13). For the purpose of the illustration we took $m_{X_1} = 200$ GeV, $m_{X_2} = 220$ GeV, $\delta_1 = 1$, and $\delta_2 = 0.01$.

3. MadDM code

We proceed to discuss the installation of the MadDM code as well as the general structure of the code and relevant functions, while we postpone an example MadDM script until the Appendix.

3.1. Installation

The current version of MadDM is designed to run on most Linux and Mac OS X distributions. Before attempting to run the code, make sure that the following are already installed:

- Python 2.6 or 2.7. This is also a requirement for current distributions of MadGraph 5.
- gfortran.
- If the computer is running Mac OS X 10.8, make sure that make or gmake is installed as they are not included in Xcode by default.
- MadGraph 5 v.1.5 or later.

Installing MadDM is fairly simple and does not require any pre-compilation. To install, follow the instructions from

```
https://launchpad.net/maddm,
```

to download the MadDM folder inside the main MadGraph folder (the path to which we will refer to as MG from here on). We suggest creating a separate MadGraph installation dedicated to MadDM.

3.2. Running MadDM "out of the box"

For the convenience of the user, we have created a user-interface Python script which executes the MadDM code. The script can also be used as a skeleton for more elaborate calculations. To run, using the terminal, navigate to the folder in which MadDM is installed and run

```
./maddm.py
```

We designed the MadDM executable script in a way that guides the user through the necessary steps to complete the calculation of dark matter relic density. MadDM will prompt the user to select a dark matter model and automatically find a dark matter candidate. The user can then choose whether they would like to include possible co-annihilation channels into the calculation, as well as change the model and MadDM parameters.

If the user chooses to change the parameters, MadDM will open a MadGraph model parameter card in the vi editor. Any modifications made will not affect the parameter card stored in the MadGraph models directory. To modify the numerical values of the parameters, you must first press **i** (for "insert mode"). Do not modify the names of the parameters. To go back to the navigation mode, press **Esc**. To save the changes, first make sure you are in navigation mode. Then press **:** followed by **w**. To exit, press **:** followed by **q**. The two commands can be combined by typing **:wq**. **ctrl + f** and **ctrl + b** can be used navigate the page down and up respectively through the file in the navigation mode.

After all of the parameters are set, MadDM will compile the numerical code and display the resulting relic density.

Alternatively, instead of relying on the "out of the box" script, one can write customized Python scripts (see the Appendix for an example) using the MadDM classes we discuss in the next Section.

3.3. Structure of the code

The MadDM code is divided into two main units: the MadGraph interface (in Python) and the numerical analysis code (in FORTRAN). The backbone of the Python code is in the darkmatter.py and init.py files, which are stored in the base directory of MadDM. Class definitions, functions which set up the relic density calculation, functions which find a dark matter candidate, generate relevant Feynman diagrams, and handle all the file output are all defined in darkmatter.py. These will be discussed in more detail in the following sections.

The FORTRAN side of the code performs the numerical calculation of the relic density using the FORTRAN matrix elements obtained from MadGraph. These are defined in relic_coupled.f (for solving the full set of coupled density evolution equations) and relic_canon.f (for the canonical treatment of co-annihilations using the $\langle\sigma_{\rm eff}v\rangle$ proxy).

The MadDM code is structured as follows:

- Projects: Contains the MadDM output for a user specified project.
- Templates: Contains skeleton files which are used by the FORTRAN part of the MadDM code. The files are divided into several sub-sub-folders:
  - include: Contains various data used by the FORTRAN numerical module during the calculation of relic density. Files named boson.txt, fermion.txt, g_star.txt, and g_star_S.txt contain information about the number of dark matter and light degrees of freedom used in the integration of the density evolution equations. The MadDM_card.inc file contains parameters which set the precision of the numerical integrators for the FORTRAN code as well as the flags which turn the status message print-outs on and off. See Section 3.6 for more details.
  - matrix_elements: Contains the templates for the FORTRAN version of the MadGraph matrix element output.
  - output: Empty folder. It is used to store the output of test routines. See Section 3.7 for more details.
  - src: Contains all the MadDM FORTRAN code. Different integration methods, including Romberg, Runge–Kutta and Vegas are defined in integrate.f. First and third order interpolation routines as well as the polynomial expansions of the Bessel functions are defined in interpolate.f. The density evolution equation is defined and solved by routines in relic_canon.f and relic_coupled.f. Finally, maddm.f is the file containing the main function calls which perform the calculation of relic abundance. It also contains calls (commented out by default) to test routines for de-bugging of matrix-elements, cross sections and thermally averaged cross sections. All of the test routines write their output to the output folder.

Compiling maddm.f, by typing make in the base folder of a user's project produces an executable maddm.x. This is the file which MadDM uses to calculate relic density. Please note that at this stage maddm.x is a standalone MadDM executable which does not require any additional MadGraph component to run.

3.4. The darkmatter Python class

We designed the MadDM code in a modular way, so that it can be incorporated into any Python script. Here we give a detailed description of the base darkmatter class as well as the user interface functions from init.py which can be used to construct user-specific Python interfaces for the MadDM code. In the following list we label the functions as

<class name>: <return values>: <function name>(<arguments>).

We use the convention of a '_' prefix to indicate that a variable name is a darkmatter class member. The darkmatter class inherits the native MadGraph classes base_objects and Particle, allowing for easier implementation of the particle properties. The relevant member functions are:

- init: [new_proj, model_name, project_name]: initialize_MadDM_session(print_banner = False): This function starts the user interface of the MadDM program. It asks the user to input the dark matter model name and checks it against the available dark matter models. It follows with the input of the folder name (subfolder to Projects) in which to write all the MadDM output. If the project already exists, the function prompts the user to overwrite it. The print_banner argument is used to determine whether to print the MadDM logo at the start of the program. The default value is False. The return values of the function are a string name of the model and a string name of the project folder with respect to Projects, as well as the logical flag which determines whether the project already exists. If the user wishes to hard-code the project name and model, this function can be omitted.
- darkmatter: darkmatter_object: darkmatter(): Creates an instance of a darkmatter class object. All of the functions below act on an instance of a darkmatter class. A darkmatter object contains all relevant information about the dark matter candidate, co-annihilation candidates, and the technical details about the MadDM project. Here we give an overview of a subset of relevant data members of a darkmatter class (all of which are lists):
  - self._dm_particles: Contains a list of Particle objects of potential dark matter candidates. Information about each particle can be accessed with a simple print command (i.e. print dm._dm_particles[0]).
  - self._coann_particles: Contains a list of Particle objects of potential dark matter co-annihilation partners.
  - self._dm_names_list: Contains the list of names of potential dark matter candidates in the MadGraph 5 user input format. For instance, n1 is the lightest neutralino of the MSSM etc.
  - self._dm_antinames_list: Same as self._dm_names_list but for anti-particles.

- `self._bsm_particles`: Contains a list of all particles in a model with a Particle Data Group (PDG) code higher than 25.
- `self._bsm_masses`: Contains a list of numerical values for the masses of all the BSM particles contained in `self._bsm_particles`.
- `self._bsm_final_states`: Contains a list of possible BSM final state particles (in `Particle` object form) that the dark matter candidate can annihilate into.
- `self._projectname`: Stores the name of the current project.
- `self._projectpath`: Stores the location of the project folder.
- `self._paramcard`: Contains the location of the `param_card.dat` file used in the project.
- `self._modelname`: Contains the name of the MadGraph 5 model used in the project.

- `darkmatter: None: init_from_model (model_name, project_name, new_proj = True)`: This function initializes the member variables of the `darkmatter` object using the model `model_name`. The name of the model should correspond to a subfolder in the `models` directory of `MadGraph`. `project_name` is the name of the folder within `/Projects` where the output of `MadDM` will be stored. `init_from_model` will create the `Projects/project_name` folder and copy the contents of the `/Template` folder to it. If `new_proj = True` the code will overwrite any existing project with the name `project_name`. The function returns no value.
- `darkmatter: None: FindDMCandidate(dm_candidate='', prompts = True)`: Based on the criteria that the particle is non-Standard Model and stable, this function finds the dark matter candidates in a model defined by the `_modelname` member. It returns no value, but it sets the member variables of a `darkmatter` object. Information about the dark matter candidates is stored in the `_dm_particles` member array. It also adds all the BSM particles to the `_bsm_particles` member array. With the default options, the function will prompt the user to change the model parameter card before determining the dark matter candidates. It will also prompt the user to enter the dark matter candidate by hand. If no user input is provided, the function will automatically determine the dark matter candidate from the default parameter card in the `MadGraph` model directory and print out the properties of the resulting dark matter candidate. The prompts can be turned off by setting the `prompts` flag to `False`. The user can also hard-code the dark matter candidate by specifying the `dm_candidate` input parameter.
- `darkmatter: None: Findco-annihilation Candidates (prompts = True, coann_eps = 0.1)`: Determines the co-annihilation candidates based on the mass splitting criteria

$$\texttt{coann\_eps} \equiv \frac{|m_{\chi_0} - m_{\chi_1}|}{m_{\chi_0}}, \tag{16}$$

where χ_0 is the dark matter candidate with mass m_{χ_0} and χ_1 is the co-annihilating partner with mass m_{χ_1}. The default is `coann_eps = 0.1`, and can be passed to the function as a parameter. If the mass splitting is less than `coann_eps` and a particle is a BSM particle (i.e. PDG $>$ 25), it is added to the `_dm_particles` member array. If `prompts = True` the function will prompt the user to enter the co-annihilation candidates by hand. If no input is provided, the co-annihilation candidates are determined automatically, based on the masses in the default parameter card. If the user inputs the co-annihilation partners by hand, the mass splitting criteria is omitted. The default value for `prompts` will also result in a print-out of the co-annihilation candidates. The prompts and messages can be turned off by setting the `prompts` flag to False.
The manual selection of the co-annihilation partners is particularly useful in a parameter scan where the range of particle masses is large. See Section 3.5 for more details.
- `darkmatter: None: GenerateDiagrams()`: This function generates the $2 \to 2$ dark matter annihilation diagrams. The initial states are formed from all valid combinations of particles stored in the `_dm_particles` member array, whereas the final states are all Standard Model particles in addition to particles stored in the `_bsm_final_states` member array.
- `darkmatter: None: CreateNumericalSession (prompts = True)`: Outputs the FORTRAN files containing the matrix elements and necessary numerical code to perform the relic density calculation. It also compiles the code and creates the executable `maddm.x` within the `Projects/<proj_name>` folder. If `prompts` is set to True, it will print out status messages and a list of annihilation processes included in the calculation. The function will also prompt the user to edit the `maddm_card.inc` file (see Section 3.6). If `prompts = False` the default `maddm_card.inc` from the `Templates/include` directory will be used.
If the user wishes to turn off the prompts, changes to the `maddm_card.inc` can still be made in the `include` subfolder of the `Projects/<proj_name>` directory. Note however that the numerical code must be recompiled for changes to take effect.
- `darkmatter: double: CalculateRelicAbundance()`: This function executes `maddm.x` in the project folder `Projects/<proj_name>`. It returns a numerical value for Ωh^2. This function can not be executed before calling `GenerateDiagrams` and `CreateNumericalSession`.
- `darkmatter: double: GetMass(pdg_id)`: Returns the numerical value for the mass of a particle with a PDG code `pdg_id`.
- `darkmatter: double: GetWidth(pdg_id)`: Returns the numerical value for the width of a particle with a PDG code `pdg_id`.
- `darkmatter: None: ChangeParameter(parameter_name, parameter_value)`: This function changes the numerical value of an entry in the `param_card.dat` file (specified by the `self._param_card` member variable). `parameter_name` is a string and must correspond to the parameter label in `param_card.dat`. For instance, `ChangeParameter('Mt', 175.0)` will change the numerical value of the top mass in the parameter card to 175.0 GeV.
Note that default settings in the `include/maddm_card.inc` can cause issues with output formatting in a parameter scan. if you wish to eliminate the status messages from the FORTRAN side of the code during the parameter scan, you must set the `print-out` flag in the `include/maddm_card.inc` file to `.false.`, and then recompile `maddm.x` by typing `make` in the base project directory.
- `darkmatter: string : GetProjectName()`: Returns the name of the folder in the `Projects` directory where the `MadDM` output is stored and checks whether or not the project folder already exists.
- `darkmatter: string: Convertname(name)`: Converts particle names (e.g. e+ $\to$ ep) so that they can be used as names for files and folders.
- `darkmatter: None: init_param_scan(name_of_script)`: Creates the Python script which executes a parameter scan. The function takes in one parameter, `name_of_script` which is the name of the file the parameter scan script should be saved to in the `Projects/<proj_name>` folder. The function will edit a default template script in the `vi` editor. Make sure to save the script upon editing it by typing `:wq` in the navigation mode of `vi`. The script will be automatically executed upon exiting `vi`.

3.5. A remark about parameter scans

When performing parameter scans with MadDM, it is important to be aware of the algorithm the code uses to calculate relic density, so as to avoid possible bugs in the computation.

One issue that might arise is in models with co-annihilations. If the initial mass splitting between the dark matter and co-annihilating partners is too large when the diagram generation step takes place, the co-annihilation diagrams will not be included in the calculation. The parameter scan takes place only upon the compilation of the numerical code, and it is not possible for MadDM to add diagrams midway through the scan without recompiling the entire numerical module. In order to bypass this issue select co-annihilation candidates by manually selecting which particles you would like to consider at the diagram generation step. Alternatively, one can also initialize the masses in the `param_card.dat` to values which are within the `coann_eps` of Eq. (16) and let MadDM automatically find the co-annihilation partners.

3.6. The MadDM include file

The `MadDM_card.inc` file contains relevant parameters for the numerical calculation of relic density. These parameters should be set in the pre-compilation step of the calculation and once the numerical code is compiled, they can be modified by editing the `MadDM_card.inc` and recompiling the code. The relevant variables in the file are

- `print_out`: A logical flag which determines whether the numerical code should print out status messages. It is set to `.false.` by default.
- `relic_canonical`: A logical flag which switches between the canonical treatment of relic density in Eq. (1) and the full, coupled equation treatment of Eq. (13). It is set to `.true.` (canonical treatment) by default.
- `calc_taacs_ann_array`: A logical flag which when set to `.true.` will calculate the thermally averaged annihilation cross sections and store them in an array for interpolation during the integration of the density evolution equation.
- `calc_taacs_dm2dm_array`: In the case of coupled density evolution equations (see Section 2.4) this is a logical flag that calculates and stores the thermally averaged annihilation cross section for all of the dark matter conversion processes.
- `calc_taacs_scattering_array`: Similar to `calc_taacs_dm2dm_array` but the thermally averaged annihilation cross sections are for DM/SM elastic scattering processes This function is useful only for de-bugging purposes. In principle, if SM elastic scattering processes are present then one should use the canonical co-annihilation formalism as opposed to the coupled density evolution equations.
- `eps_ode`: A parameter which sets the precision required from the ODE integrating routines in determination of the freeze-out temperature. The integration of the density evolution equations stops stepping back in x (see Section 2.2) when

$$\frac{Y_\infty(i) - Y_\infty(i-1)}{Y_\infty(i)} \leq \texttt{eps\_ode},$$

  where $Y \equiv n/s$ is the dark matter number density per unit entropy.
- `eps_wij, eps_taacs`: Parameters which set the precision requirements of the Romberg method integrator for the W_{ij} values (see Eq. (4)), and the thermally averaged cross section. The integration stops when

$$\frac{I(i) - I(i-1)}{I(i)} \leq \texttt{eps},$$

  where I is the value of the integral in question, i is the iteration of the Romberg algorithm and `eps` is the desired precision (i.e. `eps_taacs` or `eps_wij`).
- `iter_wij, iter_taacs`: The minimum number of Romberg iterations, independent of the required precision. These parameters are particularly useful when considering narrow s-channel resonances. The default values are set to 10 iterations.
- `beta_step_min, beta_step_max`:
  The minimum and maximum step size for the calculation of W_{ij} values (see Eq. (4)). The routines that calculate the W_{ij} arrays with respect to beta will automatically adapt the step sizes between the minimum and maximum values specified here.

3.7. Using the test routines

For debugging purposes, we have added a number of test routines to the MadDM code:

- `bessel_check (min_x, max_x, nsteps, logscale)`: prints out the numerical values of the Bessel function approximations used in the relic density evolution equation. `x_min` and `x_max` define the range of $x \equiv m/T$ values, while `nsteps` defines the number of output points. `logscale` is a flag which can take values of 0 (linear scale output) and 1 (log scale output).
- `dof_check(min_temp, max_temp, nsteps, logscale)` : same as `bessel_checl` except for the interpolation of the functions $g_*(T)$, the number of relativistic degrees of freedom, and $g_S(T)$, the number of degrees of freedom which contributes to the entropy density.
- `Wij_check()`: prints out the values of W_{ij} (see Eq. (4)) as a function of the center of mass energy into a text file.
- `taacas_check(min_x, max_x, nsteps, logscale)`: prints out the values of the velocity averaged cross section $\langle\sigma v\rangle$ as a function of $x \equiv m/T$ into a text file. `min_x` and `max_x` define the range of x values and `nsteps` defines the number of data points to output. The flag `logscale` can be used to output the $\langle\sigma v\rangle$ values on the log scale (1 for log scale, 0 for linear).
- `odeint_check(new_or_old)`: prints out the values of $Y \equiv n/s$ as a function of x. The flag `new_or_old` defines whether the calculation should be performed in the canonical formulation (see Eq. (1)) (0) or whether the full set of coupled equations (see Eq. (13)) should be considered (1).
- `cross_check_scan_all(xinit_min, xinit_max, nsteps, p_or_E, logscale)`: prints out the cross section σ summed over all the relevant dark matter annihilation processes as a function of initial state energy or momentum in the center of mass frame. `nsteps` determines how many data points to output. `p_or_E` is a flag which determines whether the cross section should be output as a function of initial state momentum (0) or energy (1). As before, `logscale` determines whether the output should be log scale (1) or linear (0). The parameters `xinit_min` and `xinit_max` define the range of momentum/energy (depending on the value of the `p_or_E` flag) for the cross section calculation.
- `cross_check_scan(dm_i, dm_j, process_k, xinit_min, xinit_max, nsteps, p_or_E, logscale)`: prints out the cross section σ for an individual process as a function of initial state energy or momentum. The parameters `dm_i` and `dm_j` select which particles to consider in the initial state. For instance, `dm_i = 1` and `dm_j = 2` will select the processes in which dark matter co-annihilates with the next lightest BSM particle. `process_k` selects a particular annihilation channel to be calculated. `xinit_min`, `xinit_max` define the range of values of momentum or energy (depending on the `p_or_E` flag for the cross section calculation).

- `cross_check_process(dm_i, dm_j, process_k, x1init, x2init, p_or_E)`: same as the previous function, but for a single momentum/energy defined by `x1init` and `x2init`.
- `matrix_element_check_all(x1init, x2init, p_or_E)`: same as the `cross_check_scan_all` function except the numerical value of the matrix element is printed out instead of the cross section.
- `matrix_element_check_process(dm_i, dm_j, process_k, x1init, x2init, p_or_E)`: same as the `cross_check_process` function except the numerical value of the matrix element is printed out.

The test routines are automatically added to every project in the `src/.f` file. In addition, calls to the test routines are included as commented-out parts of the `maddm.f` file. To use the test routines, simply navigate to the `src` folder within your project directory and edit the `maddm.f` file. Uncomment the test routines you wish to use and save the changes. Navigate back to the project folder and recompile the MadDM code by executing `make`.

The output of the test routines is stored in the text files in the `output` folder within your project directory.

3.8. Interfacing MadDM with mass spectrum generators

Many models of dark matter involve more than one particles in the BSM sector and complex mass spectra. Dedicated pieces of software are often available to translate the model parameters into the masses of physical states and decay rates to light particles. The most notable examples come from supersymmetry where a myriad of tools such as SUSPECT [11], SUSYHIT [12], etc. have been developed.

MadDM is easily interfaced with any model specific spectrum or decay width code which can produce a Les Houches formatted MadGraph parameter card. After producing a `maddm.x` executable, simply copy the parameter file from the spectrum generator as the `param_card.dat` in the `/Cards` subfolder of your project. Upon re-running `maddm.x`, the code will read the new parameters and output the corresponding value for relic density.

4. Validation

We performed detailed validation of the MadDM code in a wide range of dark matter models against results obtained with `micrOMEGAs` 2.4.5 [13]. We chose the models specifically to test the non-standard scenarios such as resonant annihilations and co-annihilations. For models which combine these features we compare the MadDM calculation of relic density in the simplest forms of the extra dimensional and supersymmetric theories to knows results.

4.1. Real Scalar Extension of the Standard Model (RXSM)

The first and simplest model we consider is the Real Scalar Extension of the Standard Model (RXSM) [14]. The model consists of an additional scalar singlet field, which couples to the Standard Model only through the Higgs:

$$V = \frac{m^2}{2}H^\dagger H + \frac{\lambda}{4}(H^\dagger H)^2 + \frac{\delta_1}{2}H^\dagger H S + \frac{\delta_2}{2}H^\dagger H S^2 + \left(\frac{\delta_1 m^2}{2\lambda}\right)S + \frac{\kappa_2}{2}S^2 + \frac{\kappa_3}{3}S^3 + \frac{\kappa_4}{4}S^4, \quad (17)$$

where S is the scalar-singlet field. A discrete Z_2 symmetry stabilizes S, making it a dark matter candidate. For the purpose of code validation we omit the details of spontaneous symmetry breaking and leave the mass of S a free parameter. Thus, the only relevant parameters for the relic density calculation are the mass m_S and the coupling of S to H which we denote as δ. The κ_n couplings only contribute to the masses and are omitted from the analysis.

The RXSM model is useful for MadDM validation as it contains a resonant $SS \to h \to SM$ channel in the list of annihilation diagrams shown in Fig. 3. Fig. 4 shows the result for the relic density calculation with an excellent agreement between our code and `micrOMEGAs`,[8] except in the region where the mass of dark matter $m_h/2 - 5\ \text{GeV} < m_S < m_h/2$, where we find a significant discrepancy of $O(1)$.

The large discrepancy is a result of known precision issues when dealing with ultra-narrow resonances close to threshold. The calculation of thermally averaged cross section in this region involves a convolution of a narrow Breit–Wigner function with a velocity distribution which can be approximated as $\beta^2 e^{-\beta^2/2\beta_0^2}$. If the resonance is close to and above threshold, the integral involves a numerically complex saddle point as the slopes of both the Breit–Wigner and the velocity distribution can be extremely steep. Note that the discrepancy does not occur for $m_S > m_h/2$ even close to the resonance. Since the resonance is below threshold in this case, the dark matter distribution convolves only the smooth tail of the Breit–Wigner function and no saddle point numerical issues arise.

In order to improve the numerical accuracy when calculating relic density in models with very narrow resonances close to threshold (*i.e.* Higgs), we recommend to increase the value of `iter_wij` and `iter_taacs` in the `maddm_card.inc` file. Note that for any change of this file to take effect you must recompile the Fortran module of the MadDM code.

4.2. Complex Scalar Extension of the Standard Model (CXSM)

Models with additional scalars offer more complex BSM sectors. In this section we consider a Complex Scalar Extension of the Standard Model (CXSM) [15,16] as a benchmark for comparisons between `micrOMEGAs` and MadDM.

We begin by considering a general renormalizable scalar potential obtained by the addition of a complex scalar singlet to the SM Higgs sector:

$$\begin{aligned} V(H,\mathbb{S}) = {} & \frac{m^2}{2}H^\dagger H + \frac{\lambda}{4}(H^\dagger H)^2 \\ & + \left(\frac{|\delta_1|e^{i\phi_{\delta_1}}}{4}H^\dagger H\mathbb{S} + c.c.\right) + \frac{\delta_2}{2}H^\dagger H|\mathbb{S}|^2 \\ & + \left(\frac{|\delta_3|e^{i\phi_{\delta_3}}}{4}H^\dagger H\mathbb{S}^2 + h.c\right) + \left(|a_1|e^{i\phi_{a_1}}\mathbb{S} + c.c.\right) \\ & + \left(\frac{|b_1|e^{i\phi_{b_1}}}{4}\mathbb{S}^2 + c.c.\right) \\ & + \frac{b_2}{2}|\mathbb{S}|^2 + \left(\frac{|c_1|e^{i\phi_{c_1}}}{6}\mathbb{S}^3 + c.c.\right) \\ & + \left(\frac{|c_2|e^{i\phi_{c_2}}}{6}\mathbb{S}|\mathbb{S}|^2 + c.c.\right) \end{aligned}$$

[8] The relic density calculation is very sensitive to the running b mass and the running b yukawa coupling in the resonant region. For the purpose of the comparison, we use a fixed $m_b = 4.7$eV and a vacuum expectation value (vev) of $v = 246$eV in both codes.

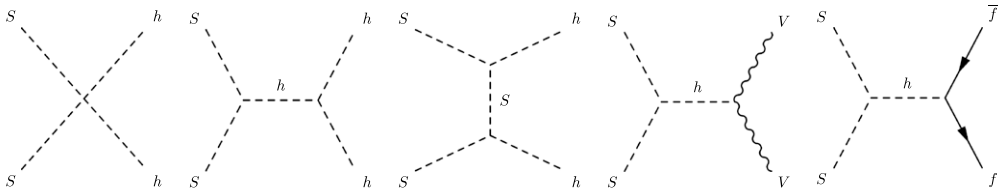

Fig. 3. Annihilation processes that contribute to the thermally averaged cross section.
Source: Taken from Ref. [14].

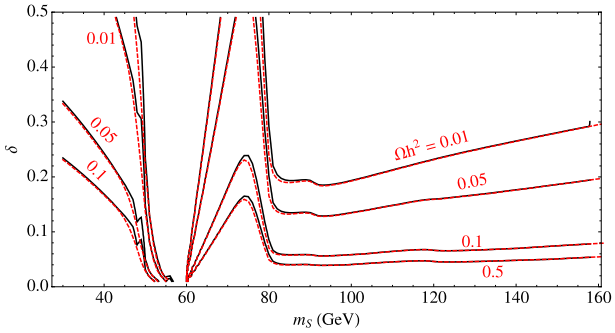

Fig. 4. Comparison between the relic density results obtained with MadDM and micrOMEGAs. The black, solid contours are the results obtained from MadDM. The red, dashed contours are obtained with micrOMEGAs. The agreement is within 5% for most of the parameter space, except very close to the resonance where precision issues are expected to arise. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

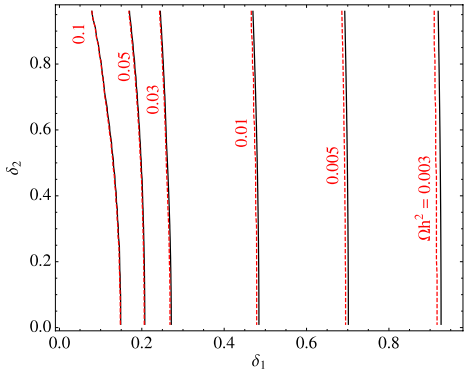

Fig. 5. Comparison between the relic density results obtained with MadDM and micrOMEGAs in the CXSM model. The black, solid curves are contours of constant Ωh^2, which are obtained from MadDM. The red, dashed contours are obtained with micrOMEGAs. The agreement is within 5% for the entire region of the parameter space. For the purpose of the comparison, we used $m_{\chi_1} = 400$ GeV, $m_{\chi_2} = 420$ GeV and $\delta_{12} = 0.05$. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

$$+\left(\frac{|d_1|e^{i\phi_{d_1}}}{8}\mathbb{S}^4+c.c.\right)+\left(\frac{|d_3|e^{i\phi_{d_3}}}{8}\mathbb{S}^2|\mathbb{S}|^2+c.c.\right)$$
$$+\frac{d_2}{4}|\mathbb{S}|^4, \tag{18}$$

where H is the $SU(2)$ doublet field that acquires a vev:

$$\langle H\rangle = \begin{pmatrix} 0 \\ v/\sqrt{2}\end{pmatrix}. \tag{19}$$

Here, m^2 and λ are the usual parameters of the SM Higgs potential, while we again adopt the SM vacuum expectation value of $v = 246$ GeV.

A discrete Z_2 symmetry on $\mathbb{S}$ serves to eliminate all the terms containing odd powers of the field, thus preventing preventing the decay of $\mathbb{S}$. The remaining scalar potential takes the form:

$$V_{\rm CXSM} = \frac{m^2}{2}H^\dagger H+\frac{\lambda}{4}(H^\dagger H)^2+\frac{\delta_2}{2}H^\dagger H|\mathbb{S}|^2+\frac{b_2}{2}|\mathbb{S}|^2+\frac{d_2}{4}|\mathbb{S}|^4$$
$$+\left(\frac{|b_1|}{4}e^{i\phi_{b_1}}\mathbb{S}^2+|a_1|\,e^{i\phi_{a_1}}\mathbb{S}+c.c.\right). \tag{20}$$

The result is a BSM sector which contains a stable dark matter candidate and a heavier state which dark matter can co-annihilate with, making CXSM an excellent framework for testing the ability of `MadDM` to perform relic density calculations in models with co-annihilations. Relevant annihilation diagrams of CXSM are similar to the ones in Fig. 3, replacing S with A. H_1 and H_2 are also possible in the place of h in Fig.3.

Fig. 5 shows a numerical comparison between `MadDM` and `micrOMEGAs` in the context of the CXSM model. We find that the results agree within 5% over a wide range of parameter space. The agreement with `micrOMEGAs` can be further improved by decreasing the `eps_wij`, `eps_taacs` and `eps_ode` precision parameters of the `MadDM` code. In cases where the dominant annihilation channel is through a narrow s-channel resonance, additional handle on precision is the minimum step in dark matter velocity `beta_step_min` (used in the integration of $\langle\sigma v\rangle$ of Eq. (2)). Finally, the minimum number of iterations `iter_wij` and `iter_taacs` can be increased to ensure that the possible strongly varying features of cross sections are accurately captured by the Romberg integration routines.

4.3. MSSM

One of the more complicated models which offers a dark matter candidate is the MSSM. The parameter space of the MSSM is vast enough to require a dedicated study, and is beyond the scope of this paper. We instead choose to test `MadDM` only against several Snowmass Points and Slopes (SPS) parameter choices [17]. The benchmark points are characterized in Table 1.

For the purpose of comparison we used the SUSYHIT [12] package to calculate the MSSM mass spectra and decay rates in `MadDM`. Table 2 shows the results against `micrOMEGAs` v.3.6.7. Note that in `MadDM` we used the MSSM implementation from the FeynRules [18] website, rather than the default MSSM implementation provided in `MadGraph`. We obtain very good agreement for all tested SPS points, except in the case of SPS 4 where we find a discrepancy of a factor of 25%. The SPS4 point is relevant to the so called "resonance region" of the MSSM parameter space, where the resonant annihilation through the Higgs dominates the velocity averaged cross section. We find that in this region, the calculation of relic density is highly sensitive to the details of running b quark masses, the problem which is beyond the scope of this paper.

4.4. Minimal Universal Extra Dimensions (MUED)

Models of extra dimensions offer a rich particle phenomenology which often provides a viable dark matter candidate (given

Table 1
SPS points.

SPS 1a:	
Point:	$m_0 = 100$ GeV, $m_{1/2} = 250$ GeV, $A_0 = -100$ GeV, $\tan\beta = 10$, $\mu > 0$.
Slope:	$m_0 = -A_0 = 0.4m_{1/2}$, $m_{1/2}$ varies.
SPS 1b:	
Point:	$m_0 = 200$ GeV, $m_{1/2} = 450$ GeV, $A_0 = 0$, $\tan\beta = 30$, $\mu > 0$.
Slope:	$m_0 = -A_0 = 0.4m_{1/2}$, $m_{1/2}$ varies.
SPS 2:	
Point:	$m_0 = 1450$ GeV, $m_{1/2} = 300$ GeV, $A_0 = 0$ GeV, $\tan\beta = 10$, $\mu > 0$.
Slope:	$m_0 = 2m_{1/2} + 850$ GeV, $m_{1/2}$ varies.
SPS 3:	
Point:	$m_0 = 90$ GeV, $m_{1/2} = 400$ GeV, $A_0 = 0$ GeV, $\tan\beta = 10$, $\mu > 0$.
SPS 4:	
Point:	$m_0 = 400$ GeV, $m_{1/2} = 300$ GeV, $A_0 = 0$ GeV, $\tan\beta = 50$, $\mu > 0$.
SPS 5:	
Point:	$m_0 = 150$ GeV, $m_{1/2} = 300$ GeV, $A_0 = 0$ GeV, $\tan\beta = 10$, $\mu > 0$. at GUT scale: $M_1 = 480$ GeV, $M_2 = M_3 = 300$,
SPS 9:	
Point:	$m_0 = 450$ GeV, $m_{\text{aux}} = 60$ TeV, $\tan\beta = 10$, $\mu > 0$.
Slope:	$m_0 = 0.0075m_{\text{aux}}$, m_{aux} varies.

Table 2
Values for Ωh^2 for different *SPS* points of MSSM between micrOMEGAs and MadDM.

SPS Pt.	MadDM	micrOMEGAs	Difference (%)	Description
SPS1a	0.197	0.195	1	neutralino DM
SPS1b	0.390	0.374	5	coan. with stau
SPS2	7.914	7.860	1	focus pt, higgsino DM
SPS3	0.118	0.116	2	conn. with sleptons
SPS4	0.0596	0.0474	22	resonance region
SPS5	0.332	0.338	−2	neutralino DM
SPS9	0.00111	0.00117	−4	AMSB, coan. w/chargino

a symmetry which stabilizes the lightest Kaluza–Klein mode). Minimal Universal Extra Dimensions (MUED) is the simplest of such models, where the lightest Kaluza–Klein particle (LKP) is the Kaluza–Klein (KK) photon γ_1. Dark matter in the context of MUED has been extensively studied in Refs. [19–22]. For the purpose of MadDM validation, we compared the results from our code to the results obtained in Refs. [19,21]. Fig. 6 shows comparison of the relic density of the LKP as a function of R^{-1}, the inverse of the size of the extra dimension. Four curves (both solid and dotted) are reproduced following Ref. [21], while circles on each curve (with 100 GeV interval on R^{-1}) are obtained with our code, following the same assumptions for each case. The (red) line marked "a" is the result from considering $\gamma_1\gamma_1$ annihilation only and the (blue) line marked "b" repeats the same analysis, but uses a temperature-dependent g_* and includes the relativistic correction to the b-term [21]. The (black) line marked "c" relaxes the assumption of KK mass degeneracy, and uses the actual MUED mass spectrum. Finally the dotted line is the result from the full calculation in MUED, including all co-annihilation processes, with the proper choice of masses. The green horizontal band denotes the preferred WMAP region for the relic density. We find good agreement between our calculation and existing results in literature, where the slight difference is due to the fact that existing results rely on non-relativistic velocity expansion, while we solve full Boltzmann equation.[9] For the same reason, we are not able to reproduce the red curve (marked as "a"), because it is impossible to undo the relativistic corrections in our code.

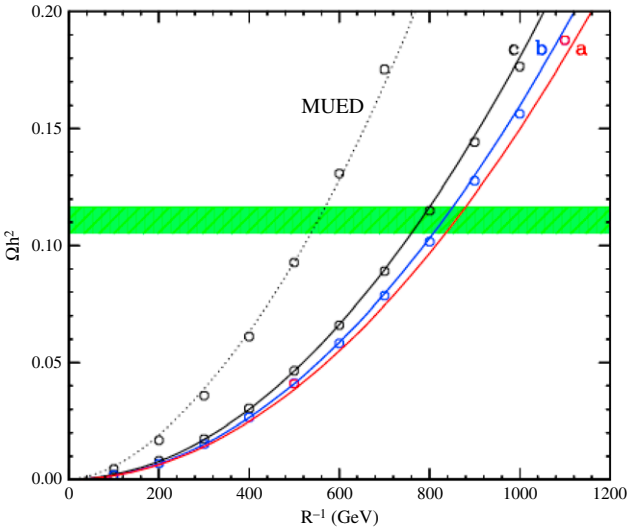

Fig. 6. Comparison between the relic density results in a MUED model between MadDM and the existing literature. The plot shows the relic density Ωh^2 as a function of the extra dimension radius R, the only parameter in the model.

5. Summary and conclusions

We presented MadDM v.1.0, a numerical code to calculate dark matter relic density in a generic MadGraph model. The project is a pioneering effort to provide the MadGraph framework with the ability to calculate cosmological and astro-physical signatures of dark matter. The aim of the project is to bridge the existing gap between collider and dark matter numerical tools and thus provide the experimental collaborations with an "all in one" dark matter phenomenology package which can be easily incorporated into the future dark matter searches at the LHC.

We envisioned MadDM as an easy-to-use, easy-to-install plugin for MadGraph. MadDM can be operated in a modular mode, whereby a user can use the existing Python libraries to perform a customized dark matter calculation or parameter scans. For convenience, we also provided a simple user interface which guides the user through the necessary steps to perform the relic density calculation.

Much like MadGraph 5 itself, MadDM code is split into two modules. The Python module of the code utilizes the existing MadGraph 5 architecture to generate dark matter annihilation diagrams in any Universal FeynRules Output (UFO) model. A FORTRAN module then calculates the corresponding relic density.

MadDM takes into account the full effects of resonant-annihilation and co-annihilations. In addition, an important feature of MadDM is that it is also able to calculate relic density in a general dark matter annihilation scenario, with any number of dark matter candidates. We performed detailed validation of the code against

[9] We used MUED model from FeynRules web page: http://feynrules.irmp.ucl.ac.be/wiki/MUED.

micrOMEGAs against several benchmark dark matter models. The real singlet extension of the Standard Model served to test the ability of MadDM to handle resonant annihilation channels, while the complex singlet extension of the Standard Model offered a framework to test models with co-annihilations. In both cases we found excellent agreement between the codes. The exception is the region very close and above the resonance in the real singlet model where we noticed a discrepancy due to precision issues. The agreement between micrOMEGAs and MadDM can be improved by readjusting the precision parameters of the MadDM numerical module.

In addition to simplified models, we also tested MadDM in the framework of MSSM and MUED. The existing literature on Kaluza–Klein dark matter agrees well with our results. The benchmark points of the MSSM show a good agreement with the results obtained with micrOMEGAs in most cases, while a discrepancy of up to 30% appears is parts of the parameter space where the details of the running b mass produce significant effects. The large discrepancy is due to the differences in the treatment of running b masses and the Higgs width in the spectrum generator used by micrOMEGAs and the version of SUSPECT we used for comparison.

In addition to relic density, future renditions of the MadDM code will also provide the ability to compute direct detection and indirect detection rates, as well as relic density in models which allow for semi-annihilations.

Acknowledgments

This work is supported in part by a US Department of Energy Grant Number DE-FG02-12ER41809, by the National Science Foundation under Award No. EPS-0903806 and by matching funds from the State of Kansas through Kansas Technology Enterprise Corporation. We are grateful to Olivier Mattelaer and Johan Alwall for helpful discussions and advice on the MadGraph 5 code and John Ralston for useful insight and suggestions.

Appendix. Example MadDM python script

To illustrate the use of the MadDM code, we proceed with an example. The following code illustrates a possible implementation of the MadDM code which omits the user interface, yet uses most of the important features of the code. The program loads the model labeled by rsxSM-coann (Two Real Singlet Extension of the Standard Model) and calculates the relic density. The dark matter candidate x1 and the co-annihilating particle x2 are hard-coded in this case. Removing these arguments would prompt the function to search for the dark matter / co-annihilation particles automatically. Finally, the code outputs the resulting relic density on the screen:

```
#! /usr/bin/env python
from init import *
from darkmatter import *

#Create the relic density object.
dm=darkmatter()
#Initialize it from the rsxSM-coann model in the MadGraph model
 folder,
#and store all the results in the Projects/rsxSM-coann
 subfolder.
dm.init_from_model('rsxSM-coann', 'rsxSM-coann', new_proj
 = True)

# Determine the dark matter candidate...
dm.FindDMCandidate('x1')

#...and all the co-annihilation partners.
dm.FindCoannParticles('x2')

#Get the project name with the set of DM particles and see
#if it already exists.
dm.GetProjectName()

#Generate all 2-2 diagrams.
dm.GenerateDiagrams()

#Print some dark matter properties in the mean time.
print "------ Testing the darkmatter object properties ------"
print "DM name: "+dm._dm_particles[0].get('name')
print "DM spin: "+str(dm._dm_particles[0].get('spin'))
print "DM mass var: "+dm._dm_particles[0].get('mass')
print "Mass: "+ str(dm.GetMass(dm._dm_particles[0].
 get('pdg_code')))
print "Project: "+dm._projectname

#Output the FORTRAN version of the matrix elements
#and compile the numerical code.
dm.CreateNumericalSession()

#Calculate relic density.
omega = dm.CalculateRelicAbundance()
print "-----------------------------------------------"
print "Relic Density: "+str(omega),
print "-----------------------------------------------"
```

References

[1] A. Belyaev, N.D. Christensen, A. Pukhov, CalcHEP 3.4 for collider physics within and beyond the standard model, Comput. Phys. Commun. 184 (2013) 1729–1769.
[2] G. Belanger, F. Boudjema, A. Pukhov, A. Semenov, MicrOMEGAs 2.0: a program to calculate the relic density of dark matter in a generic model, Comput. Phys. Commun. 176 (2007) 367–382.
[3] J. Alwall, M. Herquet, F. Maltoni, O. Mattelaer, T. Stelzer, MadGraph 5: hoing beyond, J. High Energy Phys. 1106 (2011) 128.
[4] T. Gleisberg, S. Hoeche, F. Krauss, M. Schonherr, S. Schumann, et al., Event generation with SHERPA 1.1, J. High Energy Phys. 0902 (2009) 007.
[5] P. Gondolo, J. Edsjo, P. Ullio, L. Bergstrom, M. Schelke, et al., DarkSUSY: computing supersymmetric dark matter properties numerically, JCAP 0407 (2004) 008.
[6] M. Aoki, M. Duerr, J. Kubo, H. Takano, Multi-component dark matter systems and their observation prospects, Phys. Rev. D 86 (2012) 076015.
[7] K. Griest, D. Seckel, Three exceptions in the calculation of relic abundances, Phys. Rev. D 43 (1991) 3191–3203.
[8] M. Ibe, H. Murayama, T. Yanagida, Breit–Wigner enhancement of dark matter annihilation, Phys. Rev. D 79 (2009) 095009.
[9] W.-L. Guo, Y.-L. Wu, Enhancement of dark matter annihilation via Breit–Wigner resonance, Phys. Rev. D 79 (2009) 055012.
[10] M. Srednicki, R. Watkins, K.A. Olive, Calculations of relic densities in the early universe, Nucl. Phys. B 310 (1988) 693.
[11] A. Djouadi, J.-L. Kneur, G. Moultaka, SuSpect: a Fortran code for the supersymmetric and Higgs particle spectrum in the MSSM, Comput. Phys. Commun. 176 (2007) 426–455.
[12] A. Djouadi, M. Muhlleitner, M. Spira, Decays of supersymmetric particles: the program SUSY-HIT (SUspect-SdecaY-Hdecay-InTerface), Acta Phys. Polon. B 38 (2007) 635–644.
[13] G. Belanger, F. Boudjema, P. Brun, A. Pukhov, S. Rosier-Lees, et al., Indirect search for dark matter with micrOMEGAs2.4, Comput. Phys. Commun. 182 (2011) 842–856.
[14] V. Barger, P. Langacker, M. McCaskey, M.J. Ramsey-Musolf, G. Shaughnessy, LHC phenomenology of an extended standard model with a real scalar singlet, Phys. Rev. D 77 (2008) 035005.
[15] V. Barger, P. Langacker, M. McCaskey, M. Ramsey-Musolf, G. Shaughnessy, Complex singlet extension of the standard model, Phys. Rev. D 79 (2009) 015018.
[16] V. Barger, M. McCaskey, G. Shaughnessy, Complex scalar dark matter vis-à-vis CoGeNT, DAMA/LIBRA and XENON100, Phys. Rev. D 82 (2010) 035019.
[17] B. Allanach, M. Battaglia, G. Blair, M.S. Carena, A. De Roeck, et al., The Snowmass points and slopes: benchmarks for SUSY searches, Eur. Phys. J. C 25 (2002) 113–123.
[18] C. Degrande, C. Duhr, B. Fuks, D. Grellscheid, O. Mattelaer, et al., UFO - The universal FeynRules output, Comput. Phys. Commun. 183 (2012) 1201–1214.
[19] G. Servant, T.M. Tait, Elastic scattering and direct detection of Kaluza–Klein dark matter, New J. Phys. 4 (2002) 99.
[20] S. Arrenberg, L. Baudis, K. Kong, K.T. Matchev, J. Yoo, Kaluza–Klein dark matter: direct detection vis-a-vis LHC, Phys. Rev. D 78 (2008) 056002.
[21] K. Kong, K.T. Matchev, Precise calculation of the relic density of Kaluza–Klein dark matter in universal extra dimensions, J. High Energy Phys. 0601 (2006) 038.
[22] S. Arrenberg, L. Baudis, K. Kong, K.T. Matchev, J. Yoo, Kaluza–Klein dark matter: direct detection vis-a-vis LHC (2013 update), 2013. arxiv.1310.1921.