DarkBit: a GAMBIT module for computing dark matter observables and likelihoods

The GAMBIT Dark Matter Workgroup; Bringmann, T.; Conrad, J.; Cornell, J. M.; Dal, L.A.; Edsjö, J.; Farmer, B.; Kahlhoefer, F.; Kvellestad, A.; Putze, A.; Savage, C.; Scott, P.; Weniger, C.; White, M.; Wild, S.

DOI
10.1140/epjc/s10052-017-5155-4

Publication date
2017

Document Version
Final published version

Published in
European Physical Journal C

License
CC BY

Citation for published version (APA):
The GAMBIT Dark Matter Workgroup, Bringmann, T., Conrad, J., Cornell, J. M., Dal, L. A., Edsjö, J., Farmer, B., Kahlhoefer, F., Kvellestad, A., Putze, A., Savage, C., Scott, P., Weniger, C., White, M., & Wild, S. (2017). DarkBit: a GAMBIT module for computing dark matter observables and likelihoods. European Physical Journal C, 77(12), [831]. https://doi.org/10.1140/epjc/s10052-017-5155-4

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: https://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.
DarkBit: a GAMBIT module for computing dark matter observables and likelihoods

The GAMBIT Dark Matter Workgroup: Torsten Bringmann\textsuperscript{1,a}, Jan Conrad\textsuperscript{2,3}, Jonathan M. Cornell\textsuperscript{4,5}, Lars A. Dal\textsuperscript{5}, Joakim Edsjö\textsuperscript{2,3}, Ben Farmer\textsuperscript{2,3}, Felix Kahlhoefer\textsuperscript{5}, Anders Kvellestad\textsuperscript{6}, Antje Putze\textsuperscript{7}, Christopher Savage\textsuperscript{6}, Pat Scott\textsuperscript{8,9}, Christoph Weniger\textsuperscript{9,10}, Martin White\textsuperscript{10,11}, Sebastian Wild\textsuperscript{5}

\textsuperscript{1}Department of Physics, University of Oslo, 0316 Oslo, Norway
\textsuperscript{2}Oskar Klein Centre for Cosmoparticle Physics, AlbaNova University Centre, 10691 Stockholm, Sweden
\textsuperscript{3}Department of Physics, Stockholm University, 10691 Stockholm, Sweden
\textsuperscript{4}Department of Physics, McGill University, 3600 rue University, Montreal, QC H3A 2T8, Canada
\textsuperscript{5}DESY, Notkestraße 85, 22607 Hamburg, Germany
\textsuperscript{6}NORDITA, Roslagstullsbacken 23, 10691 Stockholm, Sweden
\textsuperscript{7}LAPTh, Université de Savoie, CNRS, 9 chemin de Bellevue B.P.110, 74941 Annecy-le-Vieux, France
\textsuperscript{8}Department of Physics, Blackett Laboratory, Imperial College London, Prince Consort Road, London SW7 2AZ, UK
\textsuperscript{9}GRAPPA, Institute of Physics, University of Amsterdam, Science Park 904, 1098 XH Amsterdam, The Netherlands
\textsuperscript{10}Department of Physics, University of Adelaide, Adelaide, SA 5005, Australia
\textsuperscript{11}Australian Research Council Centre of Excellence for Particle Physics at the Tera-scale, Australia, http://www.coepp.org.au/

Received: 15 March 2017 / Accepted: 22 August 2017
© The Author(s) 2017. This article is an open access publication

Abstract We introduce DarkBit, an advanced software code for computing dark matter constraints on various extensions to the Standard Model of particle physics, comprising both new native code and interfaces to external packages. This release includes a dedicated signal yield calculator for gamma-ray observations, which significantly extends current tools by implementing a cascade decay Monte Carlo, as well as a dedicated likelihood calculator for current and future experiments (\texttt{gamLike}). This provides a general solution for studying complex particle physics models that predict dark matter annihilation to a multitude of final states. We also supply a direct detection package that models a large range of direct detection experiments (\texttt{DDCalc}), and provides the corresponding likelihoods for arbitrary combinations of spin-independent and spin-dependent scattering processes. Finally, we provide custom relic density routines along with interfaces to DarkSUSY and micrOMEGAs, and the neutrino telescope likelihood package nulike. DarkBit is written in the framework of the Global And Modular Beyond the Standard Model Inference Tool (GAMBIT), providing seamless integration into a comprehensive statistical fitting framework that allows users to explore new models with both particle and astrophysics constraints, and a consistent treatment of systematic uncertainties. In this paper we describe its main functionality, provide a guide to getting started quickly, and show illustrative examples for results obtained with DarkBit (both as a standalone tool and as a GAMBIT module). This includes a quantitative comparison between two of the main dark matter codes (DarkSUSY and micrOMEGAs), and application of DarkBit's advanced direct and indirect detection routines to a simple effective dark matter model.

Contents

1 Introduction ........................................
2 Module overview ....................................
3 Halo modeling ......................................
   3.1 Background ...................................
   3.1.1 Density profiles ............................
   3.1.2 Velocity distribution ......................
   3.2 Halo model implementation in GAMBIT ...  
   3.2.1 Halo models and associated capabilities 
   3.2.2 Likelihoods .................................
4 Relic density ......................................
   4.1 Background ...................................
   4.2 Interfaces to DarkSUSY and micrOMEGAs ...
   4.3 Relic density implementation in DarkBit ...
5 Direct detection ...................................
1 Introduction

The identity of dark matter (DM) remains one of the most vexing puzzles of fundamental physics. After decades of intense effort, its cosmological abundance has been determined at a precision of better than one percent [1], but so far no experiment has reported any clear evidence of its non-gravitational interactions. Despite these null searches, the leading hypothesis remains that DM consists of a new type of elementary particle [2]. Out of the many possibilities [3–5], weakly interacting massive particles (WIMPs) are often argued to be particularly appealing candidates, both because they almost inevitably appear in well-motivated extensions of the Standard Model – like supersymmetry [6] or universal extra dimensions [7] – but also because their thermal production in the early Universe naturally results in a relic abundance in broad agreement with the observed DM density today.

Traditionally, the particle identity of DM has been tested with three different strategies: (i) by trying to directly produce it in accelerator searches, (ii) by performing direct searches for recoiling nuclei caused by collisions with passing DM particles in large underground detectors, or (iii) by indirect searches for the debris from DM annihilation or decay in the Sun or outer space. Although these approaches are particularly suitable for WIMPs, several other candidates can be probed by some of these methods as well. More recently, another approach has emerged that is particularly relevant for DM scenarios beyond the standard WIMP case, e.g. for self-interacting DM, namely to (iv) use astrophysical probes related to the distribution of matter on galactic and cosmological scales [8,9]. For each of these methods, an immense amount of experimental data is expected during the next decade(s). In order to extract the maximal amount of information and narrow down the properties of a given DM candidate, or exclude it, it is mandatory to combine these measurements in a statistically rigorous way.

With this article we introduce DarkBit, a new numerical tool for tackling this task. DarkBit calculates DM observables and likelihoods in a comprehensive and flexible way, making them available for both phenomenological DM studies and broader Beyond-the-Standard Model (BSM) global fits. In particular, the first release of DarkBit contains up-to-date limits and likelihoods for indirect DM searches with gamma rays and neutrinos, for the spin-dependent and spin-independent cross-sections relevant to direct detection, and for the relic density. In order to increase the efficiency of observable and likelihood calculations by reusing as much code as possible, DarkBit relies on highly flexible data structures that can easily accommodate the specific needs of most particle models. Examples include the Process Catalogue (Sect. 6.3.1), which contains all relevant particles and interaction rates, a general halo model, and a fully model-independent framework to calculate the relic density.

DarkBit is designed as a module within the GAMBIT framework [10–14]. Where we introduce key terms with specific meanings in the context of GAMBIT, we highlight and link them to the glossary of standard GAMBIT terms at the end of this paper. GAMBIT defines a series of physics modules, each consisting of a collection of module functions. Each module function is able to compute an observable, a likelihood or some intermediate quantity required in the calculation of other observables or likelihoods. At runtime, the user informs GAMBIT of the observables they want to com-
pute, the theoretical model and parameter ranges over which those observables should be calculated, and how they would like GAMBIT to sample the parameter space. GAMBIT then identifies the module functions necessary for delivering the requested observables and arranges them into a dependency tree, describing which module functions must be run, and in what order. It then chooses parameter combinations, passes them to the module functions, and outputs the resulting samples. Module functions can call on additional functions from external backend codes, which GAMBIT also sorts into the dependency tree. Rules that dictate how different module functions, models and backends may rely on each other can be defined in either the source code or input file, allowing the user to force the resulting dependency tree to obey arbitrarily detailed physical conditions and relations.

One of the main design features of DarkBit, as compared to other DM codes, is its extremely modular structure. This allows users to interface essentially any external code in a straightforward way, allowing them to extract any given functionality for use within DarkBit. Examples of such backends used in the first release include DarkSUSY [15] and micrOMEGAs [16] (for various direct and indirect rates, as well as Boltzmann solvers), DDCalc (introduced in Sect. 5.2; for direct detection rates and likelihoods), gamLike (introduced in Sect. 6.2; for gamma-ray likelihoods) and nullike [17] (for neutrino likelihoods). In a situation where several backends provide the same functionality, the user can easily switch between them – or instead use powerful DarkBit internal routines, like an on-the-fly cascade decay spectrum generator (which we have implemented from scratch). On the technical side, DarkBit makes use of dynamic C++ function objects to facilitate the manipulation and exchange of real-valued functions between different backends and GAMBIT; these are implemented in the daFunk library (Appendix B).

In standalone mode, DarkBit can be used for directly computing observables and likelihoods, for any combination of parameter values in some underlying particle model. When employed as a GAMBIT module, it can be used to do this over an entire parameter space of a chosen BSM model, providing various independent likelihoods for automatic combination with those from other GAMBIT modules in a statistically consistent way. This usage mode makes it possible to not only simultaneously include all possible constraints from different observation channels, but also to incorporate the full uncertainties arising from poorly-constrained astrophysical or nuclear parameters in the scan, by treating them as nuisance parameters. Typical examples include nuclear form factors and halo model uncertainties.

This article is organised as follows. In Sect. 2 we give an overview of the physics, observables and likelihoods contained in DarkBit, along with the basic module structure. Section 3 details the DM halo models that we employ in DarkBit. In Sects. 4, 5 and 6 we go through DarkBit’s abilities in relic density calculations, direct and indirect detection, respectively. In particular, Sect. 5.2 introduces the new direct detection pheno-likelihood code DDCalc and Sect. 6.2 introduces the new gamma-ray indirect detection likelihood code gamLike. We show validation tests and illustrative examples of typical DarkBit usage in Sect. 7. We continue with an outlook on planned code expansions in Sect. 8, and conclude in Sect. 9. In Appendix A we provide a quick-start guide to installing DarkBit and running a simple test example. Appendix B introduces the new dynamic functions library daFunk. Appendix C provides a glossary containing the GAMBIT terms used in this paper.

The source code for DarkBit is available from gambit.hepforge.org, and is released under the terms of the standard 3-clause BSD license.1

2 Module overview

GAMBIT is built around the ideas of modularity and reusability. All calculations required to get from experimental data and model parameters to likelihood values and observables are performed in separate GAMBIT module functions. Each module function is able to calculate exactly one quantity, its capability. The type of this capability can be just about anything, from a simple integer to any complex C++ structure required to carry the result of the calculation. Examples of capabilities are: model parameters, particle spectra, experimental data, and the values of likelihood functions. Most module functions will also have dependencies on capabilities that were calculated by other module functions. These dependencies will be automatically resolved at run time, based on choices of the user about what particle physics model to analyse, what observables to include etc. For details, we refer the reader to the main GAMBIT paper [10].

DarkBit is one of the central GAMBIT modules, and essentially a collection of module functions that compute dark matter observables and likelihoods. Some of the basic elements of DarkBit and their relations are sketched in Fig. 1 (the full dependency tree with all module functions is far more complex than this sketch). Based on the model parameters of a particular point in the scan, DarkBit sets up several central computational structures (like the Process Catalogue, the effective annihilation rate, and WIMP-nucleon couplings), which are used for relic density calculation (see Sect. 4), the calculation of direct detection constraints (Sect. 5), and the calculation of gamma-ray and neutrino yields (Sect. 6).

1 http://opensource.org/licenses/BSD-3-Clause. Note that fjcore [18] and some outputs of FlexibleSUSY [19] (incorporating routines from SOFTSUSY [20]) are also shipped with GAMBIT 1.0 and 1.1. These code snippets are distributed under the GNU General Public License (GPL; http://opensource.org/licenses/GPL-3.0), with the special exception, granted to GAMBIT by the authors, that they do not require the rest of GAMBIT to inherit the GPL.
Predicted event rates in direct detection experiments and the rate of DM annihilation in the Sun depend on the local density of dark matter in the Milky Way. Indirect detection signals from annihilations to gamma rays and neutrinos (and charged cosmic rays) depend on the spatial DM distribution in the source being observed. In order to assure consistency in the calculations of these observables, GAMBIT contains halo models that describe the density and velocity of DM in the Milky Way and other astronomical objects.

3.1.1 Density profiles

Multiple forms of halo density profiles exist in the literature. Early analytic calculations of infalling dark matter onto collapsed density perturbations showed that the dark matter density should approximately scale like $r^{-2}$ [21], the same behaviour that one would expect for a halo with a constant velocity dispersion (or equivalently, constant temperature). This led to the modelling of the dark matter distribution by the modified isothermal profile

$$
\rho(r) = \frac{2\rho_s}{1 + (r/r_s)^2},
$$

where $r_s$ is a scale radius, and $\rho_s$ is the density at $r = r_s$.

Subsequent $N$-body simulations of the gravitational interactions of dark matter lead Navarro, Frenk, and White (NFW) to conclude that the structure of dark matter halos could be described by a cusped profile of the form [22]

$$
\rho(r) = \frac{4\rho_s}{(r/r_s)[1 + (r/r_s)]^{3/2}}.
$$

They showed that the dark matter density profile appears to be roughly universal, meaning that for halos of a range of sizes from dwarf galaxies to galaxy clusters, the form of the profile is the same. Other groups [23,24] found better fits to simulation data could be obtained by slightly modifying the NFW profile. To take these modifications into account, it is common to write the halo profile in the following form:

$$
\rho(r) = \frac{2^{(\beta - \gamma)/\alpha} \rho_s}{(r/r_s)^{\gamma} [1 + (r/r_s)^{\alpha}]^{(\beta - \gamma)/\alpha}}.
$$

Here, $\gamma$ describes the inner slope of the profile, $\beta$ the outer slope, and $\alpha$ the shape in the transition region around $r \sim r_s$.

More recently, it has been pointed out that a better fit to $N$-body simulations can be given by what is known as an Einasto profile [25], named after Einasto’s use of the profile to model the mass distribution of galaxies [26]. In this model the logarithm of the slope varies continuously with radius, leading to a density profile of the form

3 Halo modeling

3.1 Background

All of the direct and indirect detection observables that can be calculated in DarkBit are strongly dependent on the spatial distribution of DM particles, and often velocities as well.
\[ \rho(r) = \rho_s \exp \left\{ -\frac{2}{\alpha} \left[ \left( \frac{r}{r_s} \right)^\alpha - 1 \right] \right\} \]  

(4)

### 3.1.2 Velocity distribution

The velocities \( v \) of dark matter particles in a halo are usually taken to follow the Maxwell–Boltzmann distribution for an ideal gas at constant temperature. This distribution, truncated to reflect the fact that any particle with a speed beyond the escape velocity \( v_{\text{esc}} \) will leave the halo, takes the form

\[ f(v) = \frac{1}{N_{\text{esc}}} (\pi v_0^2)^{3/2} e^{-v^2/v_0^2} \]  

(5)

where \( v_0 \) is the most probable speed. Note that the above formula is only valid when \( |v| < v_{\text{esc}} \); we assume that the probability of a speed higher than \( v_{\text{esc}} \) is 0. The normalisation that corrects for the truncation, \( N_{\text{esc}} \), is given by

\[ N_{\text{esc}} = \text{erf} \left( \frac{v_{\text{esc}}}{v_0} \right) - \frac{2}{\sqrt{\pi}} \exp \left( -\frac{v_{\text{esc}}^2}{v_0^2} \right). \]  

(6)

Our Milky Way galaxy rotates in a dark matter halo that is essentially stationary. The velocity of the Earth in the halo is given by the sum

\[ v_{\text{obs}} = v_{\text{LSR}} + v_{\odot,\text{pec}} + V_\oplus(t). \]  

(7)

Here \( v_{\text{LSR}} = (0, v_{\text{rot},0}, 0) \) is the motion of the Local Standard of Rest in Galactic coordinates, \( v_{\odot,\text{pec}} = (11, 12, 7) \text{ km s}^{-1} \) is the well-known peculiar velocity of the Sun [27], and \( V_\oplus(t) \) is the velocity of the Earth relative to the Sun. The magnitude of \( V_\oplus \) is well measured at 29.78 km s\(^{-1} \) [28], and its changing direction is expected to give rise to an annual modulation of scattering rates in direct detection experiments [29]. The distribution of velocities \( u \) of dark matter particles in the Earth’s frame is given by

\[ f(u, t) = f(v_{\text{obs}}(t) + u). \]  

(8)

Assuming that the density profile of the halo surrounding the Milky Way is smooth and spherical like those discussed above, \( v_0 \) is approximately the same as the rotation speed \( v_{\text{rot}} \) of the galactic disk (for an isothermal density profile it is exactly the same, whereas for the NFW profile of Eq. 2, the two values can vary by over 10% [30]).

### 3.2 Halo model implementation in GABBIT

#### 3.2.1 Halo models and associated capabilities

In GABBIT, the radial distribution \( \rho(r) \) of dark matter in the Milky Way, the local density \( \rho_0 \), the distance from the Sun to the Galactic centre \( r_{\odot} \), as well as the local velocity distribution \( f(u) \) are simultaneously described by a given halo model. In the first release, we provide two main halo models: Halo\_gNFW and Halo\_Einasto. The former corresponds to the generalised NFW profile with parameters \( \rho_0, r_s, \alpha, \beta \) and \( \gamma \) as defined in Eq. 3, together with the Maxwell–Boltzmann velocity distribution given by Eqs. 5–8, specified by the model parameters \( v_{\odot}, v_{\text{rot}} \) and \( v_{\text{esc}} \). Lastly, the model contains \( r_{\odot} \) and \( \rho_0 \) as additional free parameters. Analogously, the Halo\_Einasto model describes the density profile given by Eq. 4, with free parameters \( \rho_0, r_s, \alpha, \beta \), and otherwise is identical to the Halo\_gNFW model. Note that with appropriate choices of \( \alpha, \beta, \) and \( \gamma \), the density profile in the Halo\_gNFW model is equivalent to the isothermal profile of Eq. 1 (\( \alpha = 2, \beta = 2, \gamma = 0 \)) or the NFW profile of Eq. 2 (\( \alpha = 1, \beta = 3, \gamma = 1 \)).

In these two halo models, the density profile \( \rho(r) \) relevant for calculating the gamma-ray flux induced by dark matter annihilations is completely decoupled from the local properties of dark matter (in particular from the local density \( \rho_0 \)), which set the event rate in direct detection experiments and neutrino telescopes. However, it is also possible to directly link the density profile to the local density by enforcing the relation \( \rho(r_{\odot}) \equiv \rho_0 \). For the case of the generalised NFW profile, this is realised by two child models of Halo\_gNFW, denoted Halo\_gNFW\_rho0 and Halo\_gNFW\_rhos. When employing the former model, the user need only specify the value of the local density \( \rho_0 \), which is then internally converted to the corresponding value of the scale density \( \rho_s \) using Eq. 3. Conversely, in the latter model one specifies \( \rho_s \), and \( \rho_0 \) is determined by GABBIT. A completely analogous choice is possible for the Einasto profile via the halo models Halo\_Einasto\_rho0 and Halo\_Einasto\_rhos.

In order to communicate the astrophysical properties of the Galactic dark matter population to the module and background functions relevant for direct and indirect searches, GABBIT employs two central capabilities: (1) GalacticHalo, which is of type GalacticHaloProperties, a data structure containing (i) a daFunk::Funk object (Appendix B), which describes the radial density profile as a function of the radius \( r^\ast \), and (ii) the distance \( r_{\odot} \) from the Sun to the Galactic centre. This capability is required in particular by the gamLike backend for the computation of gamma-ray fluxes from dark matter annihilations within the Milky Way. The other capability describing the dark matter halo is (2) LocalHalo, which is of type LocalMaxwellianHalo. This capability is required to ensure that the gamma-ray fluxes from dark matter annihilations within the Milky Way can be correctly correlated with the local dark matter density.

---

2 The remaining velocity parameters \( v_{\odot,\text{pec}} \) and \( |V_\oplus(t)| \) are much better known. Hence, instead of being part of the halo models, \( v_{\odot,\text{pec}} \) simply defaults to the value assumed by the DDCalc backend, \( 11, 12, 7 \) km s\(^{-1} \), and \( |V_\oplus| \) to 29.78 km s\(^{-1} \). The magnitude of \( V_\oplus \) can however be overridden in module functions that use it, by setting the YAML option \( v_{\text{earth}} \).
appears to be a growing consensus that \( \rho \) on assumptions about the shape of the halo [37]. There curve. The latter method often leads to results with smaller function that provides each likelihood. The capabilities associated with or \( \text{GalacticHalo}_{\text{gNFW}} \) can be provided by the module functions \( \rho \) direct detection and capture in the Sun, i.e. the local density object is simply a container for the parameters relevant to each likelihood are the same as the function name without the likelihood form, e.g. the capability for the \( \rho_0 \) likelihood is \( \text{lnl}_\rho \) \( \bar{\rho} \) so by default we set \( \rho_0 \) to this value. We take \( \sigma_\rho \) to be roughly categorised into two approaches (see [31] for a review). Local measures (e.g. [32]) use the kinematics of nearby stars to find \( \rho_0 \), whereas global measures (e.g. [33–36]) extrapolate the local density from the galactic rotation curve. The latter method often leads to results with smaller errors than the former, but they are very much dependent on assumptions about the shape of the halo [37]. There appears to be a growing consensus that \( \rho_0 \approx 0.4 \text{ GeV/cm}^3 \), so by default we set \( \bar{\rho}_0 \) to this value. We take \( \sigma_{\rho_0} \) to be 0.15 GeV/cm\(^3\), to represent the range of determinations of the local density in the literature (see e.g. [38]).

### Table 1

| Parameter                                      | Units   | Likelihood form | Central value | Uncertainty | Function                                      |
|------------------------------------------------|---------|-----------------|---------------|-------------|-----------------------------------------------|
| Local dark matter density (\( \rho_0 \))      | GeV/cm\(^3\) | Log-normal      | 0.4           | 0.15        | \text{lnl}_\rho \_lognormal                  |
| Maxwellian most-probable speed (\( v_0 \))    | km/s\(^{-1}\) | Gaussian       | 235           | 20          | \text{lnl}_v0 \_gaussian                     |
| Local disk rotation speed (\( v_{\text{rot}} \)) | km/s\(^{-1}\) | Gaussian       | 235           | 20          | \text{lnl}_vrot \_gaussian                   |
| Local galactic escape speed (\( v_{\text{esc}} \)) | km/s\(^{-1}\) | Gaussian       | 550           | 35          | \text{lnl}_vesc \_gaussian                   |

Depending on the halo model in use, the capability \( \text{GalacticHalo} \) can be provided by the module functions \( \text{GalacticHalo}_{\text{gNFW}} \) or \( \text{GalacticHalo}_{\text{Einasto}} \); for all halo models, the \( \text{LocalHalo} \) capability is obtained through the module function \( \text{ExtractLocalMaxwellianHalo} \) (see also Table 15). The rest of the \( \text{GAMBIT} \) code is designed such that only the module functions providing the capabilities \( \text{GalacticHalo} \) or \( \text{LocalHalo} \) explicitly depend on a halo model, while all other module and backend functions requiring access to the astrophysical properties of dark matter instead depend on these capabilities. This setup allows \( \text{GAMBIT} \) to be straightforwardly extended to incorporate new halo models, in particular to density profiles different from the generalised NFW and Einasto parameterisations.

#### 3.2.2 Likelihoods

\( \text{GAMBIT} \) provides several likelihood functions for the (typically quite large) uncertainties of the parameters included in the halo models. These are summarised in Tables 1 and 15. For the local dark matter density \( \rho_0 \), we implement the likelihood as a log-normal function:

\[
\mathcal{L}_{\rho_0} = \frac{1}{\sqrt{2\pi} \sigma'_{\rho_0}} \exp \left( -\ln(\rho_0/\bar{\rho}_0)^2 \right),
\]

(9)

where \( \sigma'_{\rho_0} = \ln(1 + \sigma_{\rho_0}/\rho_0) \).

The methods that have been used to determine \( \rho_0 \) can be roughly categorised into two approaches (see [31] for a review). Local measures (e.g. [32]) use the kinematics of nearby stars to find \( \rho_0 \), whereas global measures (e.g. [33–36]) extrapolate the local density from the galactic rotation curve. The latter method often leads to results with smaller errors than the former, but they are very much dependent on assumptions about the shape of the halo [37]. There appears to be a growing consensus that \( \rho_0 \approx 0.4 \text{ GeV/cm}^3 \), so by default we set \( \bar{\rho}_0 \) to this value. We take \( \sigma_{\rho_0} \) to be 0.15 GeV/cm\(^3\), to represent the range of determinations of the local density in the literature (see e.g. [38]).

We also provide likelihood functions for \( v_0 \), \( v_{\text{rot}} \), and \( v_{\text{esc}} \). For these parameters, we assume that the likelihood follows a standard Gaussian distribution

\[
\mathcal{L}_x = \frac{1}{\sqrt{2\pi} \sigma_x} \exp \left( -\frac{(x - \bar{x})^2}{2\sigma_x^2} \right).
\]

(10)

For the disk rotational velocity, by default the central value and error of the distribution are set to 235 ± 20 km/s\(^{-1}\), based on measurements of galactic masers [39, 40]. To take into account the possible discrepancies between \( v_0 \) and \( v_{\text{rot}} \) due to variances in the density profile away from the simple isothermal model, we have implemented an independent Gaussian likelihood for \( v_0 \) with the same parameters as \( v_{\text{rot}} \). Finally, for the escape velocity we use \( v_{\text{esc}} = 550 \pm 35 \text{ km/s}^{-1} \) based on measurements of high velocity stars in the RAVE survey [41]. The likelihood functions discussed in this section are listed in Table 1, along with their corresponding capabilities. The central values and errors for these likelihoods can be adjusted by setting the YAML options \texttt{param}_obs and \texttt{param}_obserr respectively, where \texttt{param} is the name of the parameter (e.g. to override the default likelihood for \( \rho_0 \) one would set \texttt{rho0}_obs and \texttt{rho0}_obserr).

Currently, no specific likelihoods are included to constrain the Galactic halo profile parameters. This can, however, easily be done by specifying appropriate parameter ranges and priors in the \( \text{GAMBIT} \) initialisation file. For typical standard values we refer the reader to Ref. [42]; for recent kinematical constraints we refer to Ref. [36].

#### 4 Relic density

##### 4.1 Background

To calculate the relic density we need to solve the Boltzmann equation for the number density \( n \) of dark matter particles, which in general can be written as [43]

\[
\frac{dn}{dt} = -3Hn - \langle \sigma_{\text{eff}} v \rangle \left( n^2 - n_{eq}^2 \right),
\]

(11)
where $n_{eq}$ denotes the equilibrium density, and $H$ the Hubble constant. Furthermore, the thermal average of the effective annihilation cross section is defined as

$$\langle \sigma v \rangle = \int_0^{\infty} dp_\text{eff} p_\text{eff}^2 W_\text{eff} K_1 \left( \frac{s^{1/2}}{T} \right) \frac{m_1^4 T}{\sum_i \frac{g_i}{N_i} m_i^2 K_2 \left( \frac{m_i}{T} \right)}.$$

(12)

where $p_\text{eff} = \sqrt{s - 4m_i^2}$ is the effective momentum in the centre-of-momentum frame of the lightest DM species (assumed to be particle 1). $K_1$ and $K_2$ are modified Bessel functions, $g_i$ is the number of internal degrees of freedom of co-annihilating particle $i$, $m_i$ is its mass, $T$ is the temperature, and $s$ is the centre-of-momentum energy squared. Finally, $W_\text{eff}$ is the effective annihilation rate, given by

$$W_\text{eff} = \sum_{ij} \frac{p_{ij}}{p_{11}} \frac{g_i g_j}{g_1} W_{ij}$$

$$= \sum_{ij} \frac{[s - (m_i - m_j)^2][s - (m_i + m_j)^2]}{s(s - 4m_i^2)} \frac{g_i g_j}{g_1^2} W_{ij}. \quad \text{(13)}$$

Here,

$$p_{ij} \equiv \sqrt{(p_i \cdot p_j)^2 - m_i^2 m_j^2}$$

is the effective momentum for $ij$ annihilation, with $p_{11} = p_\text{eff}$, and $W_{ij}$ is related to the annihilation cross section by

$$W_{ij} = 4p_{ij} \sqrt{\sigma_{ij}} = 4E_i E_j \sigma_{ij} v_{ij}, \quad \text{(15)}$$

where $E_i$ is the energy of particle $i$. The main particle physics-specific quantity, to be provided by the respective particle model, is thus the invariant rate $W_\text{eff} (p_\text{eff})$. While its computation can be computationally expensive for complex models, it is independent of temperature; it is thus usually advantageous to tabulate this function (as done in e.g. DarkSUSY).

The integration of the Boltzmann equation, Eq. 11, can then proceed in a model-independent way and the final relic density of the DM particle is given by

$$\Omega_\chi = m_\chi n_0 / \rho_{\text{crit}}. \quad \text{(16)}$$

Here, $n_0$ is the asymptotic value of $n(t \to \infty)$ as expected today and $\rho_{\text{crit}} = 3H_0^2/8\pi G$. Note that there are two equivalent ways of dealing with a situation where there is more than one DM particle with the same mass $m_\chi$ – like for example for Dirac particles where there would be both DM particles and antiparticles. The first option is to treat the DM particles as separate species; the relic density given in Eq. 16 then only refers to the density of one of the species. Alternatively, all DM particles can be treated as a single effective species with a correspondingly larger value of $g_1$ in the definition of $W_\text{eff}$ in Eq. 15; for the case of Dirac DM, e.g., one would have to replace $g_1 \to 2g_1$. In this case, the expression in Eq. 16 will refer to the total DM density.

Numerically, the integration of the Boltzmann equation is simplified by changing variables from $n$ and $t$ to the dimensionless quantities $x = m_\chi / T$ and $Y = n/s$, where $s$ now denotes the entropy density of the heat bath [44]. In DarkSUSY, the thermal average in Eq. 12 is done by using an adaptive Gaussian method, employing splines to interpolate between the tabulated points of $W_\text{eff}$ and taking special care around the known locations of thresholds and resonances. The actual integration of the Boltzmann equation is then performed via an implicit trapezoidal method with adaptive step-size; see [15] for further details.

4.2 Interfaces to DarkSUSY and micrOMEGAs

Here we discuss the general features of the GAMBIT interface to DarkSUSY and micrOMEGAs, two of the most important backends for DarkBit.

DarkSUSY$^3$ has been fully implemented into the modular framework of GAMBIT, with the calculation of all observables broken down into discrete parts that can be easily replaced with calculations from other backends. DarkSUSY is used by GAMBIT to obtain multiple theoretical quantities, including $W_{\text{eff}}$, the DM relic density (through a fully numerical solution of the Boltzmann equation), effective couplings between nucleons and WIMPs, the rate of dark matter capture in the Sun, and the spectra of gamma rays from DM annihilation.

If DarkSUSY is used as a backend for DarkBit, it is important that it be correctly initialised at each point in the scan. The most basic model-independent initialisation happens in the DarkSUSY backend initialisation function. However, in order to ensure that MSSM observables are also calculated correctly, DarkBit module functions that rely on the model-dependent capabilities of DarkSUSY have an auxiliary dependency on the capability DarkSUSY$^4$ PointInit (see Table 25). It is then the responsibility of the function that provides this capability to initialise DarkSUSY correctly. A separate capability DarkSUSY$^4$ PointInit LocalHalo is provided to initialise the DM Halo model in DarkSUSY for those backend functions where it is necessary. The full set of relic density capabilities, functions and dependencies in DarkBit can be found in Tables 16 and 17.

MicrOMEGAs$^4$ can be used by DarkBit to obtain many of the same quantities as DarkSUSY, includ-

---

$^3$ http://www.darksusy.org.

$^4$ https://lapth.cnrs.fr/micromegas.
ing the relic density, WIMP-nucleon effective couplings, and gamma-ray spectra. However, the interface of micrOMEGAs with the GAMBIT framework is currently more coarse-grained than the one for DarkSUSY. An illustrative example of this is the calculation of the relic density: in the case of DarkSUSY, GAMBIT calls different DarkSUSY functions for each part of the calculation, including the calculation of \( W_{\text{eff}} \), and the solution of the Boltzmann equation, whereas with micrOMEGAs, the entire calculation is done by calling one function.

As micrOMEGAs is not set up to take information from the DarkBit Process Catalog, a dark matter model must be implemented in micrOMEGAs following the normal method for the code. This consists of writing a compatible set of CalcHEP \([50]\) model files and then compiling micrOMEGAs with these files. All of the relevant objects, both model specific and then generic, are then combined into a shared library that is used by the micrOMEGAs frontend. As there are model-specific functions in the library, separate libraries are needed for each particle physics model. GAMBIT comes with two micrOMEGAs frontends: one for the MSSM \([51,52]\) and one for the scalar singlet DM model. The latter is not included by default with micrOMEGAs, so we have included the CalcHEP files needed to implement it with the GAMBIT distribution.

The initialisation functions for these frontends, `MicrOMEGAs_MSSM_3_6_9_2_init` and `MicrOMEGAs_SingletDM_3_6_9_2_init`, both have the YAML options `VZdecay` and `VMDecay`. These control how micrOMEGAs treats annihilations to 3-body final states via virtual \( W \) and \( Z \) bosons. If these options are set to 0 these processes are ignored, while a value of 1 causes them to be included in the case of DM self-annihilations, and with a value of 2 they are taken into account for all coannihilation processes as well. To initialise the backend, `MicrOMEGAs_SingletDM_3_6_9_2_init` passes information about couplings, masses, and decays directly to micrOMEGAs, while `MicrOMEGAs_MSSM_3_6_9_2_init` writes an SLHA1 file to disk, which is subsequently read by the `lesHinput` function of MicrOMEGAs (this function is not compatible with the more general SLHA2 format). The latter function also has the YAML option `internal_decays`, which when set to `true` causes information from the GAMBIT decay table to be passed via the DECAY blocks of the SLHA1 file to micrOMEGAs. By default this is set to `false` and decay widths are calculated internally by micrOMEGAs. This ability to pass MSSM decay information to micrOMEGAs was a late addition to DarkBit; future versions will employ this option by default.

### 4.3 Relic density implementation in DarkBit

The general structure and main capabilities of relic density calculations in DarkBit are summarised in Fig. 2. As explained above, the most important particle physics input needed to calculate the DM relic density is the effective invariant annihilation rate \( W_{\text{eff}} \). In DarkBit, this is represented by the capability `RD_eff_annrate`.

Currently, there are two functions that provide this capability (Table 16), though further user-defined functions can easily be added. The first function, `RD_eff_annrate_DSprep`, returns \( W_{\text{eff}} \) for SUSY models using DarkSUSY as a backend. It depends on the capability `RD_eff_annrate_DSprep`, which ensures that DarkSUSY is correctly set up and configured to provide \( W_{\text{eff}} \) for neutralino annihilation. In this case two boolean options can be provided in the YAML file: `CoannCharginosNeutralinos` and `CoannSfermions`. These specify whether chargino, neutralino or sfermion coannihilations are taken into account. These options default to `true`, but the relevant coannihilations can be disabled to speed up the relic density calculation (at the expense of accuracy).\(^5\) Another important option to steer the numerical performance is `CoannMaxMass` (default 1.6), which specifies up to what mass, in units of the DM mass, coannihilating particles are taken into account when calculating \( \langle \sigma v \rangle \). The second alternative is to determine \( W_{\text{eff}} \) directly from the Process Catalogue (Sect. 6.3.1). The corresponding function obviously has `TH_ProcessCatalogue` as a dependency, as well as `DarkMatter_ID`, and can be used for models where coannihilations are not important. Here, the identity of the dark matter particle, as it exists in the GAMBIT particle database (see Ref. [10]), is provided by the capability `DarkMatter_ID` (Table 14).

The main capability of the relic density part of DarkBit is `RD_oh2`, which returns \( \Omega_\chi h^2 \) for a given model point. If the user wishes, the result can be expressed as a fraction of the total, measured DM density. This capability is `RD_fraction`, and is provided by three different mod-

---

\(^5\) DarkSUSY does not currently include gluino coannihilations.
ule functions: RD_fraction_one, RD_fraction_leq_one and RD_fraction_rescale. The choice of function dictates whether indirect and direct detection routines should use the observed relic density or the one calculated for the particular parameter point. RD_fraction_one simply returns 1, and causes the observed relic density to be used in all direct and indirect detection routines. RD_fraction_leq_one returns the lesser of 1 and the ratio of the computed relic density to the observed value. RD_fraction_rescale always returns the computed-to-reference ratio, regardless of whether or not it exceeds 1. The latter two functions accept an option "oh2_obs", which defaults to 0.1188; this is the observed value of $\Omega h^2$ to use for rescaling.

DarkBit currently includes three functions that provide RD_oh2 (Table 17). Two of those (RD_oh2_DarkSUSY and RD_oh2_MicrOmegas) are direct calls to the unmodified relic density routines of DarkSUSY and micrOMEGAs, with all particle model parameters initialised as per default in the respective backend code. RD_oh2_DarkSUSY is only compatible with the MSSM, while RD_oh2_MicrOmegas can be used with any model implemented in micrOMEGAs. This latter function has two options: fast, which determines the numerical accuracy of the calculation (int: 0: accurate [default], 1: fast), and Beps (double; default $10^{-5}$) corresponds to a minimum value for the parameter

$$B = \exp\left(-\frac{T}{m_1} (m_i + m_j - 2m_1)\right).$$

if $B$ is less than the Beps value for a certain process, that process will not be included in the calculation of the effective annihilation rate (this serves the same purpose as CoannMaxMass for DarkSUSY). The third function (RD_oh2_general) is a general Boltzmann solver, which again largely relies on various subroutines provided by the DarkSUSY backend, but not on any DarkSUSY-specific initialisation of particle parameters (e.g. the setting of particle masses and couplings): the option fast (int; 0: accurate, 1: fast [default]) again steers the numerical performance of the backend code. In order to calculate $\langle \sigma v \rangle$, c.f. Eq. 12, the Boltzmann solver needs not only the invariant rate, but also the internal degrees of freedom and masses of all (co)annihilating particles. For a high-precision result of this integral, one will in general also need to know the exact location of thresholds and resonances in $W_{eff}$. RD_oh2 therefore depends on the capability RD_spectrum_ordered which contains all this information, ordered by increasing $P_{dy}$. RD_spectrum_ordered in turn depends on the capability RD_spectrum which contains the same information, except for coannihilation thresholds, but not necessarily in an ordered form. Presently, RD_spectrum can be provided in two ways, either by the Process Catalogue (if coannihilations are not important; see Sect. 6.3.1) or by DarkSUSY.

Lastly, the capability of the likelihood function constraining the relic density is lnl_oh2. This capability is provided by two module functions. First, lnl_oh2_Simple is a Gaussian likelihood that implements the observational limits from Ref. [1] ($\Omega h^2 = 0.1188 \pm 0.0010$ at 1σ). In addition to the experimental error, we also take into the account possible theoretical errors in the calculation of the relic density. We assume that the likelihood for the prediction for $\Omega h^2$ is a Gaussian distribution centred around the calculated value with a standard deviation that is taken, by default, to be 5% of that calculated value. This error is conservative for most parameter combinations, and underestimates the $O(50\%)$ corrections that can occur due to loop corrections in a few specific scenarios [53–57]. The choice is thus a pragmatic compromise that represents the best that can be done with a single uncertainty. The user is free to change it, and we emphasise the importance of the user choosing an error appropriate for the particular model he or she is studying. The exact form of the Gaussian likelihood for the model parameters is determined by the YAML option profile_systematics, which determines whether the prediction for $\Omega h^2$ is profiled or marginalised over (by default this option is set to False, corresponding to marginalisation). The mean value, the experimental error and the theoretical error can be changed with the YAML file options oh2_obserr, oh2_obserr, and oh2_fractional_theory_err, respectively. For the exact form of the likelihood function and the details of its derivation, we refer the reader to sections 8.3.1 and 8.3.2 of [10].

Alternatively, lnl_oh2_upperlimit implements the observational constraint as a one-sided limit, leaving open the possibility that a certain DM candidate makes up only a fraction of the total abundance. The likelihood function is roughly a Gaussian similar to the one described in the previous paragraph for predictions greater than the observed relic density and flat for predictions below the observed value. Here the exact form is again dependent on the YAML parameter profile_systematics (see section 8.3.3 and 8.3.4 of [10] for many more details). The other three YAML parameters for this function are the same as before.

5 Direct detection

5.1 Background

Given that the solar system sits within a DM halo, DM particles are expected to pass through Earth continuously. If DM has any ability to interact with regular matter at all, it will occasionally scatter on terrestrial nuclei. Direct detection experiments [58] search for these scattering events, by looking for nuclear recoils in large volumes of inert target material placed in ultra-clean environments deep underground.
In natural units, the differential rate of recoil events in a direct detection experiment is

\[ \frac{dR}{dE} = \frac{2\rho_0}{m_\chi} \int v f(v,t) \frac{d\sigma}{dq^2}(q^2, v) d^3v, \quad (18) \]

where \( m_\chi \) is the WIMP mass, \( \rho_0 \) is the local DM mass density, \( f(v,t) \) is the three-dimensional, time-dependent WIMP velocity distribution, \( \frac{d\sigma}{dq^2}(q^2, v) \) is the velocity-dependent differential cross-section, and \( q^2 = 2m_{\text{nuc}} E \) is the momentum exchanged in the scattering process (for a nucleon mass \( m_{\text{nuc}} \) and recoil energy \( E \)). Numerical values of this differential rate are typically expressed in cpd (counts per day), per kg of target material, per keV recoil energy. Most direct search detectors contain more than one isotope, in which case the differential rate is given by a sum over Eq. 18 for each isotope, weighted according to the mass fraction of the isotope in the detector.

The expected number of signal events in an analysis by a direct search experiment is given by

\[ N_p = MT \int_0^\infty \phi(E) \frac{dR}{dE}(E) dE, \quad (19) \]

where \( M \) is the detector mass and \( T \) is the exposure time. The detector response function \( \phi(E) \) describes the fraction of recoil events of energy \( E \) that will be observed within some pre-defined analysis region. The precise definition of such an analysis region depends on the experiment under consideration. In the simplest case it would be given by a lower and an upper bound on the reconstructed energy, so that the response function \( \phi(E) \) can be calculated in terms of the energy resolution of the detector and the various trigger efficiencies. The energy range over which the experiment is sensitive is then encoded within \( \phi(E) \), so that there is no need to impose a finite upper or lower cutoff in the integral in Eq. 19.

Some experiments implement more elaborate analyses, imposing further cuts on observables that depend on the recoil energy in a more complicated way. All of these possibilities can be captured by an appropriate function \( \phi(E) \), because the detector response is always independent of the nature of the particle interaction of the WIMP with nuclei, which is contained in the differential event rate. The detector response \( \phi(E) \) can therefore be tabulated in advance for different analyses and re-used for any WIMP model. Equation 19 can be generalised to experiments with more than one analysis region (e.g. binned event rates) by defining a separate function \( \phi_i(E) \) for each analysis region.

For many WIMP candidates (which we refer to as \( \chi \)), the dominant WIMP-quark interactions arise from a combination of

- a scalar (\( \bar{\chi}\chi q \)) or vector (\( \bar{\chi}\gamma_{\mu}\chi q\gamma^\mu q \)) coupling, which gives rise to a spin-independent (SI) cross-section, and
- an axial-vector coupling (\( \bar{\chi}Y_{\mu}\gamma_{\nu}\chi q\gamma^\nu \gamma^\mu q \)), which gives rise to a spin-dependent (SD) cross-section

(for a complete list of possible operators see Ref. [59]). In both of these cases, the matrix element involved has no intrinsic dependence upon either the momentum exchanged in the collision, nor on the relative velocity of the DM and the nucleus. In such cases, it is convenient to write the scattering cross-section as a simple product of a cross-section \( \sigma_0 \) defined at zero-momentum-transfer, and a form factor \( F^2(q) \) that accounts for the finite size of the nucleus. For such velocity and momentum-independent interactions, the differential cross-section becomes

\[ \frac{d\sigma}{dq^2}(q^2, v) = \frac{\sigma_0}{4\mu^2v^2} F^2(q) \Theta(q_{\text{max}} - q), \quad (20) \]

where \( \Theta \) is the Heaviside step function, \( \mu \) is the WIMP-nucleon reduced mass, \( q_{\text{max}} = 2\mu v \) is the maximum momentum transfer in a collision at a relative velocity \( v \), and the velocity dependence is entirely due to kinematics rather than the interaction. The requirement that \( q \leq q_{\text{max}} \) for an interaction to be kinematically possible translates into a lower limit \( v \geq v_{\text{min}} = \sqrt{m_{\text{nuc}}E/2\mu^2} \) in the integral over the WIMP velocity distribution (Eq. 18). The total WIMP-nucleus differential cross-section is the sum over the SI and SD contributions, each with its own form factor.

The zero-momentum cross-section \( \sigma_0 \) for SI WIMP-nucleus interactions is

\[ \sigma_{\text{SI}} = \frac{\mu^2}{\pi} \left[ ZG_{\text{SI}}^p + (A - Z)G_{\text{SI}}^n \right]^2 \]

\[ = \frac{4\mu^2}{\pi} \left[ Zf_p + (A - Z)f_n \right]^2 \quad (21) \]

where \( Z \) is the atomic number and \( A \) is the atomic mass number. \( Z \) and \( A - Z \) are respectively the number of protons and neutrons in the nucleus, and \( f_p \) and \( f_n \) are the effective couplings to them. The latter depend on both the precise nature of the interaction between WIMPs and quarks (and/or gluons), and on the contents of the proton and neutron. We note that the alternative normalisation involving \( G_{\text{SI}}^p = 2f_p \) and \( G_{\text{SI}}^n = 2f_n \) is often found in the literature, where \( G_{\text{SI}}^N \) with \( N = p, n \) are the \( G_F \)-like effective four-fermion coupling constants in the case of scalar interactions. The micrOMEGAs manual, meanwhile, uses \( \lambda_N = \frac{1}{4}G_{\text{SI}}^N \). The nucleon contents are described by the nuclear hadronic matrix elements, discussed in detail in Sect. 5.3.2 below.

For most DM candidates with scalar couplings, the proton and neutron SI cross-sections are roughly the same, so \( f_n \simeq f_p \). For identical couplings \( (f_n = f_p) \), the SI cross-section...
reduces to
\[ \sigma_{SI} = \frac{\mu_p^2}{\mu_p^2} A^2 \sigma_{SI,p}, \]  
(22)
where \( \mu_p \) is the WIMP-proton reduced mass. Direct detection experiments are often designed to use heavy nuclei, as the SI cross-section grows rapidly with \( A \).

The SI form factor is essentially a Fourier transform of the mass distribution of the nucleus, and it is reasonably approximated by the Helm form factor \([60,61]\),
\[ F(q) = 3e^{-q^2 s^2 / 2} \frac{\sin(q r_n) - q r_n \cos(q r_n)}{(q r_n)^3}, \]  
(23)
where \( s \simeq 0.9 \) fm and \( r_n^2 = c^2 + 2 \pi^2 a^2 - 5 s^2 \) is an effective nuclear radius with \( a \simeq 0.52 \) fm and \( c \simeq 1.23 A^{1/3} - 0.60 \) fm. Further details on SI form factors can be found in Refs. \([61,62]\).

SD scattering is only present for detectors that contain isotopes with net nuclear spin. This generally requires the nucleus to possess an unpaired proton and/or neutron in its shell structure. The relevant WIMP-nucleon cross-section is
\[ \sigma_{SD} = \frac{4 \mu_p^2}{\pi^2} \frac{(J + 1)}{J} \left[ G_{SD}^p (S_p) + G_{SD}^n (S_n) \right]^2 \]  
\[ = \frac{4 \mu_p^2}{\pi^2} \frac{(J + 1)}{J} \left[ a_p (S_p) + a_n (S_n) \right]^2, \]  
(24)
where \( G_F \) is the Fermi constant, \( J \) is the spin of the nucleus, \( \langle S_p \rangle \) and \( \langle S_n \rangle \) are the average spin contributions from the proton and neutron groups respectively, and \( a_p \) and \( a_n \) are the effective couplings to the proton and the neutron in units of \( 2 \sqrt{2} G_F \). Similarly to \( f_p \) and \( f_p \), \( a_p \) and \( a_n \) depend on both the WIMP-quark interaction and on the relative contributions of different quark flavours to the nucleon spin; the latter is discussed further in Sect. 5.3.2. As in the spin-independent case, alternative normalisations can be found in the literature. The DarkSUSY manual, for example, refers to \( G_{SD}^N = 2 \sqrt{2} G_F a_N \), while the micrOMEGAs manual uses \( \xi_N = \frac{1}{2} G_{SD}^N \). In addition, whilst we here use \( a_N \) and \( G_{SD}^N \) to distinguish the two notations, \( a_N \) is frequently used within the literature for both cases.

Unlike the SI case, the two SD couplings \( a_p \) and \( a_n \) differ substantially in many theories. Individual experiments typically only strongly constrain either \( a_p \) or \( a_n \), as most detector materials do not contain isotopes with both unpaired neutrons and protons; experimental results on the SD cross-section are therefore generally presented in terms of \( \sigma_{SD,p} = \sigma_{SD}(a_n = 0) \) or \( \sigma_{SD,n} = \sigma_{SD}(a_p = 0) \).

The SD form factor is given in terms of the structure function \( S(q) \) normalised so that \( F^2(0) = 1 \),
\[ F^2(q) = S(q)/S(0), \]  
(25)
with
\[ S(q) = a_p^2 S_{pp}(q) + a_n^2 S_{nn}(q) + a_p a_n S_{pn}(q). \]  
(26)
In the limit \( q \to 0 \), the functions \( S_{nn}(0) \) and \( S_{pp}(0) \) are proportional to the expectation values of spins of the proton and neutron subsystems \([63,64]\),
\[ S_{pp}(0) = \frac{(J + 1)(2J + 1)}{\pi J} \langle S_p \rangle^2, \]  
\[ S_{nn}(0) = \frac{(J + 1)(2J + 1)}{\pi J} \langle S_n \rangle^2. \]  
(27)

5.2 DDCalc

The traditional presentation of results from direct searches for dark matter is an exclusion curve for the SI or SD WIMP-nucleon scattering cross-section, as a function of the WIMP mass. This invariably comes with some rather specific restrictions:

1. the exclusion is given at only a single confidence level (CL; traditionally 90%),
2. \( f_p = f_n \) (often somewhat loosely referred to as ‘isospin conservation’) is assumed,
3. either \( a_p = a_n = 0 \) (pure SI coupling), or \( f_p = f_n = (a_p \text{ or } a_n) = 0 \) (pure SD\(_p \) or SD\(_n \) coupling) is assumed,
4. it is assumed that the local density and velocities of DM follow the Standard Halo Model (cf. Sect. 3),
5. a specific set of nuclear form factors \( F^2(q) \) is adopted, and
6. the nuclear parameters are fixed to assumed values when calculating \( f_p, f_n, a_p \) and \( a_n \) for any comparison against theory predictions.

These restrictions are all problematic when trying to recast direct search results, as one must go from the idealised effective WIMP frameworks in which they are presented to real constraints on actual theories. Ultimately, we are interested in the overall degree to which a model with some arbitrary combination of couplings \( f_p, f_n, a_p \) and \( a_n \) agrees or disagrees with data, not merely which side of a 90\% CL curve it lies on under different limiting approximations about \( f_p, f_n, a_p \) and \( a_n \). Ideally, this would also include the impacts of systematic errors on that exclusion, due to uncertainties from the halo, nuclear and form-factor models that one must assume in order to obtain that result.

For these reasons, here we present DDCalc: a new, general solution for recasting direct search limits. DDCalc is released and maintained as a standalone backend code by the GAMBIT Dark Matter Workgroup. It can be obtained from http://ddcalc.hepforge.org under an academic use license. A development version of the code including only the first run of the
LUX experiment was previously released as LUXCalc [65], and has been used in a number of analyses (e.g. [66–68]).

Yet another issue is that the limits presented by experimental collaborations almost always assume Eq. 20, i.e. that the scattering matrix element has no explicit velocity or momentum dependence. Many DM models involve non-trivial momentum- or velocity-dependences in their cross-sections. This means that extra velocity factors must be incorporated into the integral over the WIMP velocity distribution (Eq. 18), and extra momenta must be included in the final integral over the differential rate when calculating the total event yield (Eq. 19). Furthermore, these models often probe the properties of the target nuclei not captured by the standard SI and SD nuclear form factors. Although the first release of DDCalc does not include such generalised couplings out of the box, its structure is designed to easily accommodate them (and they will be explicitly included in a future release).

5.2.1 Methods

DDCalc calculates predicted signal rates and likelihoods for various experiments, given input WIMP and halo models.

A DDCalc WIMP model consists of the DM mass and the four couplings $f_p, f_n, a_p$ and $a_n$, specifiable also directly as SI/SD proton/neutron cross-sections. DDCalc does not deal directly with nuclear uncertainties; users (or GAMBIT as the case may be) are expected to vary these externally in the calculation of the couplings. Momentum-dependent couplings can be implemented by adding the requisite additional power of $q$ to the integrand of Eq. 19, as implemented in the source file DDRates.f90. Similarly, velocity-dependent cross-sections can be implemented in the integrand of Eq. 18, found in DDHalo.f90. Some more explicit tips for the brave are provided in the DDCalc README. SI form factors in the first release default to Helm (Eq. 23). SD form factors are included for $^{19}$F, $^{23}$Na, $^{27}$Al, $^{29}$Si, $^{73}$Ge, $^{127}$I, $^{129}$Xe and $^{131}$Xe from Ref. [69]. Alternative form factors can be encoded in DDNuclear.f90.

The local halo model consists of a constant local density and a truncated Maxwell–Boltzmann velocity distribution (Eq. 5). The most-probable speed $v_0$, truncation speed $v_{\text{esc}}$, local speed relative to the halo $v_{\text{obs}}$ and local density $\rho_X$ are all individually configurable; $v_{\text{obs}}$ can also be constructed automatically from the local standard of rest $v_{\text{LSR}}$ or disk rotation speed $v_{\text{rot}}$, along with the Sun’s peculiar velocity $v_{\odot,\text{pec}}$ relative to it.

A particular set of WIMP and halo model parameters will produce a set of predicted yields in various direct search experiments. Having calculated the expected number of signal events $N_{p,i}$ for the $i$th experimental analysis region (Eq. 19), DDCalc calculates a Poisson likelihood for the model as

$$L_i(N_{p,i}|N_o,i) = \frac{(b_i + N_{p,i})^{N_o,i} e^{-(b_i+N_{p,i})}}{N_o,i!},$$

where $N_{o,i}$ is the number of observed events in the analysis region, and $b_i$ is the expected number of background events in that region. Note that a single experiment may comprise more than one analysis region (for example bins in reconstructed energy). Unbinned analyses can in principle also be implemented, provided sufficient information from the experiment is available. The likelihoods for each experiment and analysis region, $\{L_i\}$ can then be combined to form a composite direct search likelihood, which can itself be used in combination with other likelihood terms from other experiments.

Some direct detection experiments do not provide explicit background estimates or prefer not to perform a background subtraction in order to test the WIMP hypothesis. In this case, $b_i$ should be set to the value that maximises the likelihood:

$$b_i = \begin{cases} 
N_{o,i} - N_{p,i}, & N_{o,i} > N_{p,i} \\
0, & N_{o,i} \leq N_{p,i}.
\end{cases}$$

This leads to a one-sided likelihood, i.e. a non-zero WIMP signal can only be disfavoured but not preferred relative to the background-only hypothesis.

The likelihood functions can be used directly to calculate constraints in the $\sigma$–$m_\chi$ plane. A parameter point is considered to be excluded at 90% confidence level if

$$2 \log \mathcal{L}(\sigma = 0) - 2 \log \mathcal{L}(\sigma, m_\chi) > 1.64,$$

where $\mathcal{L}(\sigma = 0)$ denotes the likelihood of the background-only hypothesis. Alternatively, one can also use DDCalc to obtain constraints in the $\sigma, m_\chi$ plane using one of two $p$-value methods:

**Feldman–Cousins**

DDCalc implements the standard Feldman–Cousins method [70] for generation of one- or two-sided confidence intervals, based on the Poisson likelihood in Eq. 28. Note that the likelihood in this case uses the total expected signal and background yields across the entire analysis region, and hence does not incorporate spectral information that might lead to a stronger result.

**Maximum gap**

Yellin’s maximum gap method [71] was proposed as a way of handling spectral information in the case that the magnitude and shape of the background are unknown and a background subtraction is therefore not possible. The method assumes that all of the observed events could in principle be signal events, leading to a conservative exclusion limit on the WIMP scattering cross-section. Nevertheless, exploiting the spectral information of the observed events will typically yield...
stronger results than the Feldman-Cousins method; see [65] for an example.

The idea of the maximum gap method is to break the signal region into a number of intervals bounded by the observed events. Given an efficiency functions $\phi_k(\mathcal{E})$ for each of these intervals, one can then use Eq. 19 to calculate the expected number of events between any two observed events. The “maximum gap” is the interval where this number is largest. By calculating the probability that a gap as large as the observed one could arise from random fluctuations it is then possible to quantify the $p$-value of the assumed model [71].

5.2.2 Experiments

Event rate calculations in DDCalc rely on the availability of experimental response functions $\phi(\mathcal{E})$, the predicted number of background events in an analysis $b$, the total number of observed events $N_o$, and the experimental exposure $MT$.

Version 1.0.0 of DDCalc ships with this data already implemented for eight experimental analyses, shown in Table 2 (see also Fig. 8 for a comparison of our analyses with the published bounds).

The LUX [75–77], PandaX [78], XENON100 [72] and SuperCDMS [73] analyses are most useful for constraining SI scattering and SD scattering on neutrons, whereas the PICO-2L [80], PICO-60 [79] and SIMPLE [74] analyses provide good sensitivity to SD scattering on protons. In the following we provide additional details on the implementation of the experimental details.

**SIMPLE.** We implement the efficiency curve $\phi(\mathcal{E})$ directly as described in Ref. [74]. We do not include the contribution from carbon, due to its high threshold energy for nuclear recoils. SIMPLE expected 12.7 events and observed 8.

**PICO-2L.** The efficiency curve $\phi(\mathcal{E})$ for flourine is provided in Ref. [83]. Again, we do not include the contribution from carbon. The background expectation of 1.0 events agrees well with the one event observed.

**PICO-60.** This experiment has a time-dependent energy threshold, so the efficiency curve $\phi(\mathcal{E})$ is obtained by convolving the exposure as a function of threshold with the (appropriately rescaled) efficiency curve for fixed threshold. The total exposure is reduced by a trial factor of 1.8 as proposed by the collaboration. Because the efficiency curves for fluorine and iodine differ, we implement the two target materials as independent experiments. This is possible only because PICO-60 has observed no signal events and does not perform a background subtraction, in which case the likelihood reduces to $L = e^{-N_o}$, so that the contributions for the individual target nuclei factorise. Again, we do not include the contribution from carbon.

**SuperCDMS.** We implement two different efficiency curves including gap information, with nuclear recoil energies converted from phonon energies assuming the Lindhard prescription [61]. The first is based directly on the published efficiencies and event energies of all detectors included in the experimental run [73]. During this run, the ionisation guard of one detector (T5Z3) was inoperative, allowing additional background events to enter the analysis region and reduce the overall sensitivity of the experiment. We therefore implement a second efficiency curve and corresponding set of analysis parameters, where the T5Z3 detector is excluded from the analysis. This alternative parameterisation is the default within DDCalc for this analysis.

**PandaX.** We implement the efficiency curve $\phi(\mathcal{E})$ provided by the collaboration [78], with an additional factor of 2 to account for the fact that only events below the mean of the nuclear recoil band are considered. Three events were observed in this search window compared to a background expectation of 4.8.

**XENON100.** While XENON100 does provide efficiency curves as a function of the nuclear recoil energy [72], this information is insufficient to make use of the spectral information, i.e. the energy of the observed events. This spectral information is however very helpful, because events at high recoil energies ($\mathcal{E} > 30$ keV) as well as events close to the threshold have a higher probability to result from backgrounds than from WIMP scattering. To be able to use this spectral information (for example in the context of the maximum gap method), we have simulated the detector using the Time Projection Chamber Monte Carlo (TPCMC) code [84], which in turn relies on NEST for modelling the physics of recoiling heavy nuclei [85]. The TPCMC output is included in the public release of DDCalc, and provides the full gap information for XENON100.

---

Table 2 Experimental analyses included in DDCalc

| Experiment       | Analysis |
|------------------|----------|
| Included in DDCalc 1.0.0 |          |
| XENON100         | 2012 [72]|
| SuperCDMS        | 2014 [73]|
| SIMPLE           | 2014 [74]|
| LUX (run 1)      | 2013 [75], 2015 [76]|
| LUX (run 2)      | 2016 [77]|
| PandaX           | 2016 [78]|
| PICO-60          | 2016 [79]|
| PICO-2L          | 2016 [80]|
| Added in DDCalc 1.1.0 |          |
| Xenon1T          | 2017 [81]|
| PICO-60          | 2017 [82]|

---

6 The additional efficiency information for T5Z3 was kindly provided by the SuperCDMS Collaboration.
compared to an expected background of 0.36 events. We therefore use the same acceptance functional factor of 0.851 to account for the selection efficiency neglecting the contribution from carbon. We include an additional factor of 0.851 to account for the selection efficiency for single scatters. No such events have been observed and we do not perform a background subtraction.

We emphasise that the accurate implementation of likelihood functions for WIMP masses below 10 GeV is very challenging, as the experimental sensitivity results largely from upward fluctuations and is furthermore very sensitive to astrophysical uncertainties. A precision study of the experimental constraints on low-mass WIMPs will likely require the implementation of additional experimental information in order to refine the likelihood functions.

5.2.3 Command-line usage

By default, compiling DDCalc produces static and shared libraries as well as a default executable DDCalc_run. This latter can be run via either a command-line interface, or via an interactive mode that is entered automatically if the user supplies no arguments to the executable.

A complete list of arguments for DDCalc_run can be obtained with:

```
DDCalc_run --help
```

The command-line signature of the program is:

```
DDCalc_run [mode] [options] [WIMP parameters]
```

The `mode` flag switches between a variety of run modes, summarised in Table 3. WIMP parameters is a list of arguments that can take one of four forms (in units of GeV for \( m \), and pb for the cross-sections):

```
m m sigmaSI m sigmaSD m sigmapSI sigmanSI sigmapSD sigmanSD
```

In the first case, all WIMP-nucleon cross-sections are set to 1 pb. In the second case, only spin-independent couplings are turned on, and \( \sigma_{\text{SI}} \) and \( \sigma_{\text{SD}} \) are used as common cross-sections for both WIMP-proton and WIMP-neutron interactions (in the SI and SD cases, respectively). Negative cross-sections can be given in order to indicate that the corresponding coupling should be negative (the actual cross-section will be taken as the magnitude of the value supplied).

Various general options for DDCalc are summarised in Table 4. In Table 5, we list the options that can be used to change the dark matter halo distribution.

The DDCalc package includes a number of advanced detector options for defining the precise isotopic composition of a detector, and the efficiency functions. These are listed in Table 6, but should not be necessary for the general user, as the existing analysis flags set the defaults correctly for the experimental analyses that are included in the release.
Table 3  The various run modes of the DDCalc default executable DDCalc_run

| Mode                     | Summary                                                                                                                                                                                                 |
|--------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| DDCalc_run               | Calculates the expected number of signal events in the analysis region, the likelihood of the WIMP parameters, and the \( p \)-value. The logarithm of the latter two are given                                                                 |
| DDCalc_run --log-likelihood | Calculates the logarithm of the Poisson-based likelihood for the specified WIMP parameters                                                                                                               |
| DDCalc_run --log-pvalue  | Calculates the logarithm of the \( p \)-value for the specified WIMP parameters. If the analysis includes interval information, this returns the maximum gap \( p \) value. If the analysis does not include interval information, the \( p \)-value calculated from the Poisson likelihood but without background subtraction (i.e. setting \( b = 0 \)) to give a conservative upper bound |
| DDCalc_run --spectrum    | Tabulates the raw recoil spectrum \( \frac{dR}{dE} \) for the detector material and given WIMP parameters, by energy. The tabulation of the energies can be modified using the \(--E-tabulation\) option                                                      |
| DDCalc_run --events-by-mass | Tabulates the expected signal events for fixed WIMP-nucleon cross-sections, by WIMP mass. The tabulation of masses can be modified using the \(--m-tabulation\) option                                                   |
| DDCalc_run --constraints-SI | Tabulates the cross-section lower and upper constraints in the spin-independent case, by mass. Constraints are 1D confidence intervals at each mass, determined using a Poisson likelihood with signal plus background and Feldman-Cousins ordering. The confidence level is given using the \(--confidence-level\) option, with the default set to 0.9 (90% CL). The ratio of the WIMP–neutron and WIMP–proton couplings is held fixed, with the ratio defined by an angle \( \theta \) such that \( \tan \theta = G_n/G_p \). The angle can be specified via the \(--theta-SI\) option or, more conveniently, given in units of \( \pi \) via \(--theta-SI-pi\). The default is \( G_p = G_n \) (\( \theta = \pi/4 \)). The tabulation of the masses can be modified using the \(--m-tabulation\) option |
| DDCalc_run --limits-SI   | Tabulates the cross-section upper limits in the spin-independent case, by mass. Limits are determined using the maximum gap \( p \)-value method. Excluded \( p \)-values are given using the \(--p\) or \(--lnp\) options, with the default being \( p = 0.10 \) (90% CL exclusion limits). The ratio of the WIMP–neutron and WIMP–proton couplings is held fixed, with the ratio defined by an angle \( \theta \) such that \( \tan \theta = G_n/G_p \). The angle can be specified via the \(--theta-SI\) option or, more conveniently, given in units of \( \pi \) via \(--theta-SI-pi\). The default is \( G_p = G_n \) (\( \theta = \pi/4 \)). The tabulation of the masses can be modified using the \(--m-tabulation\) option |
| DDCalc_run --limits-SD   | As above, but for the spin-dependent case                                                                                                                                                              |

5.2.4 Library interface (API)

For reference, here we include a summary of the main DDCalc functions. These can be accessed from a Fortran calling program with

```
USE DDCalc
```

and from a C/C++ program with

```
#include "DDCalc.hpp"
```

Usage of these functions by GAMBIT is documented in the following subsection.

Derived types

DDCalc defines three types of object. These are the bedrock of the code; almost every calculation must be provided with an instance of each of these to do its job.

- **WIMPStruct**: A WIMP model, containing values of the WIMP mass and couplings.
- **HaloStruct**: A halo model, containing values of the local DM density and the parameters of the truncated Maxwell-Boltzmann velocity distribution.

```
DetectorStruct A detector/analysis object, containing efficiencies, the background model, energy threshold, exposure and observed events.
```

WIMP model

Initialisation

```
TYPE(WIMPStruct) FUNCTION DDCalc_InitWIMP()
```

Creates a new WIMPStruct and initialises it with a mass of 100 GeV and couplings of 1 pb.

Parameter setting

```
SUBROUTINE DDCalc_SetWIMP_mfa(WIMP,m,fp,fn, ap, an)
SUBROUTINE DDCalc_SetWIMP_mG(WIMP,m,GpSI, GnSI,GpSD,GnSD)
SUBROUTINE DDCalc_SetWIMP_msigma(WIMP,m, sigmapSI,sigmanSI,sigmapSD,sigmanSD)
```

Sets the internal parameters of the WIMP object. Here \( m \) is \( m_\chi \) in GeV, \( f_p \) and \( f_n \) are \( f_p \) and \( f_n \) in GeV\(^{-2}\. \)
Table 4 Options of the DDCalc default executable DDCalc_run

| General options | Mode-specific options | Statistical options | Experiment-specific options |
|-----------------|-----------------------|---------------------|----------------------------|
| `--help` | Displays help information | | Sets the detector and analysis according to the XENON100 2012 result [72] |
| `--verbosity` | Sets the verbosity level (a higher level gives more output). The data output at different levels is mode-specific. Selection of verbosity level 0 is also available via the flag `--quiet`; `--verbose` gives verbosity level 2 | | Sets the detector and analysis according to the LUX run 1 result from 2013 [75] |
| `--E-tabulation=<min>,<max>[,<N>][0.1,1000,-50]` | Specifies the tabulation for the recoil energy $E$. Energies will be tabulated from $<\text{min}>$ to $<\text{max}>$ inclusive, with $N$ logarithmically spaced intervals between tabulation points. A negative $<\text{N}>$ value indicates $|<\text{N}|$ points per decade. The $<\text{N}>$ argument is optional | | Sets the detector and analysis according to the reanalysis of the LUX run 1 result from 2015 [76] |
| `--interactive` | Turns on the interactive mode, which will prompt the user for the WIMP parameters | | Sets the detector and analysis according to our implementation of the LUX run 2 result from 2016 [77] |
| `--m-tabulation=<min>,<max>[,<N>][1,1000,-20]` | Specifies the tabulation for the WIMP mass $m$. Masses will be tabulated from $<\text{min}>$ to $<\text{max}>$ inclusive, with $N$ intervals between tabulation points. The points will be logarithmically spaced unless the fourth comma-separated value is F or 0, in which case the spacing will be linear. A negative $<\text{N}>$ value indicates $|<\text{N}|$ points per decade in the logarithmic case. The $<\text{N}>$ and $<\text{log}>$ arguments are optional | | Sets the detector and analysis according to the PandaX 2016 result [78] |
| `--theta-SD=<val>[π/4]` | Fixes the ratio of the WIMP-nucleon couplings to $\tan \theta = G_p/G_n$ in the spin-dependent case. This option is only used in run modes where the absolute couplings cannot be specified. The similar option `--theta-SD-pi` allows the ratio to be specified in units of $\pi$ | | Sets the detector and analysis according to the SuperCDMS 2014 low-energy result [73] |
| `--theta-SI=<val>[π/4]` | As above, but for the spin-independent case | | Sets the detector and analysis according to the SIMPLE 2014 C$_2$ClF$_5$ result [74] |
| `--confidence-level=<val>[0.9]` | The confidence level to use in determining constraints/limits | | Sets the detector and analysis according to the PICO-2L 2016 result [80] |
| `--confidence-level-sigma=<val>` | The confidence level corresponding to the fraction of the normal distribution within the given number of standard deviations (symmetric). That is, a value of 3 here would result in a $3\sigma$ CL constraint/limit. Equivalent to `--p-value-sigma` | | Sets the detector and analysis according to the PICO-60 2016 result [79], using only the contribution from fluorine |
| `--p-value=<val>[0.1]` | The $p$-value to use in determining constraints/limits. The logarithm of the $p$-value can be specified via `log-p-value` | | Sets the detector and analysis according to the PICO-60 2016 result [79], using only the contribution from iodine |
| `--LUX-2015` | Sets the detector and analysis according to the LUX run 1 result from 2015 [75] | | Sets the detector and analysis according to the Xenon1T 2017 result (DDCalc 1.1.0 only) [81] |
| `--XENON100-2012` | Sets the detector and analysis according to the XENON100 2012 result [72] | | Sets the detector and analysis according to the Xenon1T 2017 result (DDCalc 1.1.0 only) [81] |
| `--LUX-2013` | Sets the detector and analysis according to the LUX run 1 result from 2013 [75] | | Sets the detector and analysis according to the Xenon1T 2017 result (DDCalc 1.1.0 only) [81] |
| `--LUX-2016` | Sets the detector and analysis according to the LUX run 1 result from 2015 [76] | | Sets the detector and analysis according to the XENON100 2012 result [72] |
| `--PandaX-2016` | Sets the detector and analysis according to the PandaX 2016 result [78] | | Sets the detector and analysis according to the PICO-60 2016 result [79], using only the contribution from fluorine |
| `--SuperCDMS-2014` | Sets the detector and analysis according to the SuperCDMS 2014 low-energy result [73] | | Sets the detector and analysis according to the Xenon1T 2017 result (DDCalc 1.1.0 only) [81] |
| `--SIMPLE-2014` | Sets the detector and analysis according to the SIMPLE 2014 C$_2$ClF$_5$ result [74] | | Sets the detector and analysis according to the PICO-60 2016 result [79], using only the contribution from iodine |
| `--PICO-2L` | Sets the detector and analysis according to the PICO-2L 2016 result [80] | | Sets the detector and analysis according to the PICO-60 2016 result [79], using only the contribution from fluorine |
| `--PICO-60_F` | Sets the detector and analysis according to the PICO-60 2016 result [79], using only the contribution from fluorine | | Sets the detector and analysis according to the Xenon1T 2017 result (DDCalc 1.1.0 only) [81] |
| `--PICO-60_I` | Sets the detector and analysis according to the PICO-60 2016 result [79], using only the contribution from iodine | | Sets the detector and analysis according to the PICO-60 2016 result [79], using only the contribution from fluorine |
| `--Xenon1T-2017 (DDCalc 1.1.0 only)` | Sets the detector and analysis according to the Xenon1T 2017 result [81] | | Sets the detector and analysis according to the PICO-60 2016 result [79], using only the contribution from iodine |
| `--PICO-60_2017` | Sets the detector and analysis according to the PICO-60 2017 result [79] | | Sets the detector and analysis according to the PICO-60 2016 result [79], using only the contribution from fluorine |

and $a_p$ and $a_n$, the $G$ parameters are $G_{SI}^p$, $G_{SI}^n$, $G_{SD}^p$, and $G_{SD}^n$, and the $\sigma$ parameters are $\sigma_{SI,p}$, $\sigma_{SI,n}$, $\sigma_{SD,p}$, and $\sigma_{SD,n}$ in pb. Negative cross-sections indicate that the corresponding coupling should be negative. In all cases, ‘$p$’ refers to proton and ‘$n$’ to neutron.
Table 5  Dark matter halo distribution options of the DDCalc default executable DDCalc_run

| Option          | Description                                                                                     |
|-----------------|-------------------------------------------------------------------------------------------------|
| --rho=<val>     | [0.4]                                                                                           |
| --v0=<val>      | [vrot]                                                                                          |
| --vesc=<val>    | [550]                                                                                           |
| --vlsr=<val>,<val>,<val> | [0,vrot,0]                                                                                   |
| --vobs=<val>    | [|vsun|]                                                                                         |
| --vpec=<val>,<val>,<val> | [11,12,7]                                                                                   |
| --vsun=<val>,<val>,<val> | [vlsr+vpec]                                                                                   |
| --vrot=<val>    | [235]                                                                                           |

The local dark matter density (in units of GeV cm$^{-3}$)

The most probable speed of the Maxwell-Boltzmann distribution (in km s$^{-1}$). Related to the rms speed by $v_{\text{rms}} = \sqrt{\frac{3}{2}v_0}$ and the mean speed by $\bar{v} = \sqrt{\frac{4}{\pi}v_0}$. For the isothermal sphere of the Standard Halo Model, this is equal to the galactic circular velocity (disk rotation speed), so this is set equal to the value supplied with --vrot (or its default) unless explicitly set here

The local Galactic escape speed (in km s$^{-1}$). This is used to truncate the Maxwell-Boltzmann distribution as the highest speed particles should be depleted due to escape from the Galactic potential

The Local Standard of Rest (in km s$^{-1}$), i.e. the velocity of the Galactic disk relative to the Galactic rest frame in Galactic coordinates $UVW$, where $U$ is the anti-radial direction (towards the Galactic centre), $V$ is the direction of disk rotation, and $W$ is in the direction of the galactic pole. Set to $(0,-v_{\text{rot}},0)$ unless specified here

The speed of the observer (detector) relative to the galactic rest frame (in km s$^{-1}$), where the galactic rest frame is the frame in which the dark matter exhibits no bulk motion. This is set equal to the magnitude of --vsun, unless explicitly set here

The velocity of the Sun relative to the Local Standard of Rest in galactic coordinates $UVW$

The motion of the Sun relative the galactic rest frame (in km s$^{-1}$) in galactic coordinates $UVW$. Set equal to --vlsr + --vpec unless explicitly set here

The local disk rotation speed (in km s$^{-1}$)

Sets the internal parameters of the Halo object. Here $\rho_\chi$ is $\rho_\chi$ in GeV cm$^{-3}$, $v_{\text{rot}}$ is $v_{\text{rot}}$ in km s$^{-1}$, $v_0$ is $v_0$ in km s$^{-1}$, and $v_{\text{esc}}$ is $v_{\text{esc}}$ in km s$^{-1}$.

Advanced setters and getters

See DDCalc_SetHalo() and DDCalc_GetHalo() in DDHalo.f90.

Experiments and analysis

Initialisation

TYPE(DetectorStruct) FUNCTION DDCalc_InitDetector(intervals)

The first of these functions initialises an object carrying the information about the default experimental analysis; in DDCalc 1.0.0 this is the LUX 2013 analysis [75]. Here intervals is a flag indicating whether calculations should be performed for intervals between observed events or not. This is only necessary for maximum gap calculations and can be set to FALSE for likelihood analyses. Non-default analyses can be obtained with the specific functions listed second, where analysis_name is one of the analyses given in Table 2, e.g. SuperCDMS_2014. See DDExperiments.f90 and analyses/analysis_name.f90 for more details. Note that these specific analysis constructors are only available directly by declaring

USE DDExperiments
Table 6  Advanced detector options of the DDCalc default executable DDCalc_run

General detector options

--background=<val> [0.64]
  The average expected number of background events in the analysis region

--events=<N> [1]
  The number of observed events in the analysis region

--exposure=<val> [118×85.3]
  The detector’s fiducial exposure (in kg day). Set equal to --mass × --time unless explicitly set here

--mass=<val> [118]
  The detector’s fiducial mass (in kg)

--time=<val> [85.3]
  The detector’s exposure time (in days)

Isotopic composition options

--argon
  Sets the isotopic composition to the naturally occurring isotopes of argon (Ar), germanium (Ge), sodium iodide (NaI), silicon (Si), or xenon (Xe)

--germanium

--sodium-iodide

--silicon

--xenon

--element-Z=<Z1>,<Z2>,<Z3>,...
  Sets the isotopic composition to the naturally occurring isotopes of the compound with the given set of elements and stoichiometry. For example, CF3Cl would have Z=6, 9, 17 and a stoichiometry 1, 3, 1. If the stoichiometry is not given, it is assumed to be 1 for each element

--stoichiometry=<N1>,<N2>,<N3>,...
  Provide an explicit list of isotopes to use. The list of atomic numbers (Z), mass numbers (A), and mass fractions (f) must all be provided and be of the same length or these options will be ignored

--isotope-Z=<Z1>,<Z2>,<Z3>,...
  Applies a minimum energy threshold for rate calculations (in keV). This effectively takes: \( \phi(E) \rightarrow \Theta(E - E_{min}) \phi(E) \), where \( \Theta(x) \) is the Heaviside step function. This allows easy removal of low-energy scattering from the signal calculations without having to modify efficiency input files

--isotope-A=<A1>,<A2>,<A3>,...

--isotope-f=<f1>,<f2>,<f3>,...

Detector efficiency options

--Emin=<val> [0]
  Disables the use of sub-interval calculations, even if efficiencies for the sub-intervals are available. This is implied by certain program modes where the sub-intervals cannot be used

--no-intervals

or

include "DDExperiments.hpp"

in the calling program, in addition to the regular USE DDCalc/include "DDCalc.hpp". Both versions of these functions return a DetectorStruct containing the analysis details.

Set threshold for nuclear recoils

SUBROUTINE DDCalc_SetDetectorEmin(Detector, Emin)
Manually sets the minimum nuclear recoil energy discernible by a given Detector to Emin, in keV. The default value is 0, meaning that the detector response is determined by the pre-calculated efficiency curves, which account for detector and analysis thresholds. Setting Emin to a non-zero value allows to obtain more conservative exclusion limits or to study the dependence of the experimental results on the assumed low-energy cut-off.

Do rate and likelihood calculations

SUBROUTINE DDCalc_CalcRates(Detector, WIMP, Halo)
Perform the rate calculations used for likelihood and confidence intervals, using the analysis Detector on the specified WIMP and Halo models. The results are
These use CDDCalc to the main interface described above is defined within a C/C++ namespace DDCalc. The routines can be accessed using the following routines.

Retrieve results of calculations

1. Number of observed events in the analysis:

   ```
   INTEGER FUNCTION DDCalc_Events(Detector)
   ```

2. Expected number of background events:

   ```
   REAL*8 FUNCTION DDCalc_Background(Detector)
   ```

3. Expected number of signal events:

   ```
   REAL*8 FUNCTION DDCalc_SignalSI(Detector)
   ```

   Alternatively, for the separate spin-independent and spin-dependent contributions:

   ```
   REAL*8 FUNCTION DDCalc_SignalSI(Detector)
   REAL*8 FUNCTION DDCalc_SignalSD(Detector)
   ```

4. Log-likelihood:

   ```
   REAL*8 FUNCTION DDCalc_LogLikelihood (Detector)
   ```

   Uses the Poisson distribution Eq. 28.

5. Logarithm of the p-value:

   ```
   REAL*8 FUNCTION DDCalc_LogPValue(Detector)
   ```

   Uses the maximum gap method if Detector was initialised with `intervals = TRUE` and the analysis contains the necessary interval information to allow such a method; otherwise uses a Poisson distribution in the number of observed events, assuming zero background contribution (Eq. 28 with $b = 0$).

6. Factor by which the WIMP cross-sections must be multiplied to achieve a given p-value:

   ```
   REAL*8 FUNCTION DDCalc_ScaleToPValue(lnp) 8
   ```

   Calculates the factor $x$ by which the cross-sections must be scaled ($\sigma \rightarrow x\sigma$) to achieve the desired $p$-value, given via `lnp` as $\ln(p)$. See DDCalc_LogPValue above for a description of the statistics.

C/C++ interface

For ease of use in linking to these routines from C/C++ code, a second (wrapper) version of each of the interface routines described above is defined within a C namespace DDCalc. These use C-compatible types only, and access interfaces to the main DDCalc library through explicitly-specified symbol names, to get around name-mangling inconsistencies between different compilers when dealing with Fortran modules. These functions work just like the ones above, but neither accept nor return WIMPStruct, HaloStruct nor DetectorStruct objects directly. Instead, they return and accept integers corresponding to entries in an internal array of Fortran objects held in trust for the C/C++ calling program by DDCalc. The routines can be used to delete the objects held internally by DDCalc.

5.3 Direct detection implementation in DarkBit

5.3.1 WIMP-nucleon couplings

As shown in Table 13, GAMBIT contains three functions capable of calculating effective WIMP-nucleon couplings: `DD_couplings_DarkSUSY`, `DD_couplings_MicrOmegas`, and `DD_couplings_SingletDM`. Each has capability `DD_couplings_SI` and returns a `dm_couplings_SI` object, which contains the parameters $G_{SI}$, $G_{ST}$, $G_{SD}$ and $G_{SD}$. `DD_couplings_DarkSUSY` calculates these couplings for a generic MSSM model using DarkSUSY, while `DD_couplings_SingletDM` calculates them for the scalar singlet model internally. `DD_couplings_MicrOmegas` calculates the couplings using micrOMEGAs for either model – the appropriate version of the micrOMEGAs backend is chosen by GAMBIT depending on which model is being scanned.

For the MSSM `DD_couplings_DarkSUSY` and `DD_couplings_MicrOmegas` compute the couplings by passing MSSM Spectrum information and nuclear parameters (described in more detail in the next section) to the external codes. By default, DarkSUSY does not take into account loop corrections to the DM-nucleon scattering process, aside of course from the one-loop coupling of the Higgs to gluons via the triangle diagram involving heavy quarks. On the other hand, certain classes of one-loop corrections are taken into account by default when using micrOMEGAs, such as SUSY QCD corrections to the Higgs-exchange diagrams or box diagrams involving external gluons (for details we refer the reader to [48]). Similar corrections have been implemented in DarkSUSY (see [86] for details), and in GAMBIT 1.1.0, we have added the ability to turn these on by setting the YAML option `loop` (default: false) for the function `DD_couplings_DarkSUSY` to true. Also added to this function is the option `pole` (default: false), which when set to false causes DarkSUSY to approximate the squark propagators in the calculation of the SI and SD cou-
plings as $1/m_q^2$ to avoid poles (this only applies when loop is false, otherwise this option is ignored). A final change in GAMBIT 1.1.0 is addition of the box option (default: true) to DD_couplings_MicrOmegas, which determines whether the box diagrams for DM-gluon scattering are calculated for MSSM-like models.

In the case of the scalar singlet model, DD_couplings_SingletDM calculates the effective Higgs-nucleon coupling $f_N$, internally, using the nuclear matrix elements relevant for SI scattering, and combines it with the scalar mass and Higgs-portal coupling as described in Ref. [87] to determine the effective SI couplings (there is no SD scattering in this model). DD_couplings_MicrOmegas uses micrOEGAs with our CalcHEP implementation of the scalar singlet model for a similar calculation.

5.3.2 Nuclear uncertainties

When the interaction between DM and quarks can be described by a scalar operator, the spin independent effective couplings $G^p_{\text{SI}}$ and $G^n_{\text{SI}}$ depend heavily on both the sea and valence quark content of the proton and neutron respectively. These are parameterised by the 6 nuclear matrix elements

$$m_N f_{Tq}^{(N)} \equiv \langle N | m_q q | N \rangle,$$

where $N \in \{p, n\}$ and $q \in \{u, d, s\}$. The equivalent quantities for the heavy quarks $Q \in \{c, b, t\}$ are related to these parameters via the formula [87,88]

$$f_{TQ}^{(N)} = \frac{2}{27} \left(1 - \sum_{q=u,d,s} f_{Tq}^{(N)}\right).$$

The 6 lighter quark matrix elements are part of the nuclear_params_fnq model in GAMBIT. They can be calculated from other quantities more closely related to lattice and experimental results. These are often the light and strange quark contents of the nucleon, defined as

$$\sigma_l \equiv m_l \langle N | \bar{u}u + \bar{d}d | N \rangle$$

$$\sigma_s \equiv m_s \langle N | \bar{s}s | N \rangle,$$

where $m_l \equiv (1/2)(m_u + m_d)$. In the nuclear_params_sigmasignal model, these 2 parameters replace the 6 $f_{Tq}^{(N)}$ values, with the conversion between the two parameter sets described in Ref. [87]. An equivalent parameterisation replaces $\sigma_s$ with the parameter $\sigma_0$:

$$\sigma_0 \equiv m_l \langle N | \bar{u}u + \bar{d}d - 2\bar{s}s | N \rangle.$$

By comparing the forms of the above equations, we see that $\sigma_0$ and $\sigma_l$ are related by the formula

$$\sigma_0 = \sigma_l - \sigma_s \left(\frac{2m_l}{m_s}\right).$$

The GAMBIT model nuclear_params_sigmasignal contains $\sigma_0$ and $\sigma_l$.

For the spin-dependent effective couplings $G^p_{\text{SD}}$ and $G^n_{\text{SD}}$, the relevant nuclear parameters are $\Delta_q^{(N)}$, which describe the spin content of the nucleons, where again $N \in \{p, n\}$ and $q \in \{u, d, s\}$. Here there are only three parameters since the values for the proton and neutron are directly related: $\Delta_u^{(p)} = \Delta_d^{(p)}$, $\Delta_u^{(n)} = \Delta_d^{(n)}$, and $\Delta_s^{(p)} = \Delta_s^{(n)}$. All of the nuclear_params models contain the three $\Delta_q^{(p)}$.

All of the nuclear parameters are set in the appropriate backend when calculating the WIMP-nucleon couplings using the functions DD_couplings_DarkSUSY and DD_couplings_MicrOmegas. DD_couplings_SingletDM, which does not make use of an external backend, also uses the nuclear parameters in its calculation of the couplings.

A combined likelihood for $\sigma_l$ and $\sigma_0$ is implemented in DarkBit as the capability lnl_SI_nuclear_parameters. This capability is provided by the function lnl_sigmassignal (Table 18), in which the two likelihoods take the form of simple Gaussian distributions (Eq. 10), with the default values $\sigma_l = 43 \pm 8$ MeV, based on a global fit of lattice calculations [89], and $\sigma_0 = 58 \pm 9$ MeV. This last value is based on the range of recent determinations of $\sigma_0$ in the literature from analyses of pion-nucleon scattering data [90–92], $\sigma_l$ has also been extracted from lattice QCD results [93–96], and the more recent analyses based on this approach point to a lower preferred value of $\sigma_l \sim 40$ MeV.

For the $\Delta_q^{(N)}$, we provide likelihoods for the following 2 combinations of parameters:

$$a_3 = \Delta_u^{(p)} - \Delta_d^{(p)}$$

$$a_8 = \Delta_u^{(p)} + \Delta_d^{(p)} - 2\Delta_s^{(p)}$$

and $\Delta_s^{(p)}$ itself. A combined likelihood for all of these parameters is given by the capability lnl_SD_nuclear_parameters, which can currently only be fulfilled by the function lnl_deltaq (Table 18). The likelihoods again take the form of Gaussian distributions, with $a_3 = 1.2723 \pm 0.0023$, determined via analysis of measurements of neutron $\beta$ decays [97], and $a_8 = 0.585 \pm 0.023$ based on hyperon $\beta$ decay results [98]. $\Delta_s^{(p)} = -0.09 \pm 0.03$, based on a measurement of the spin-dependent structure function of the deuteron from the COMPASS fixed target experiment [99].

For both the SI and SD nuclear parameters, the central values and widths of the Gaussians can be adjusted with the YAML options param_obs and param_obser respectively, where param refers to the parameter name (see Table 18 for the list of options).
5.3.3 Event rates and likelihoods

**GAMBIT 1.0.0** includes rate and likelihood functions for the eight direct detection experiments supported by **DDCalc 1.0.0**: LUX (run 1) [75,76], LUX (run 2) [77], PandaX [78], XENON100 [72], SuperCDMS [73], PICO-2L [80], PICO-60 [79] and SIMPLE [74]. As discussed above, the two dominant target elements of PICO-60 (fluorine and iodine) are implemented as independent experiments, but should always be used together. **GAMBIT 1.1.0** includes in addition rate and likelihood functions for the two new experiments included in **DDCalc 1.1.0**: Xenon1T [81] and the 2017 analysis of PICO-60 [82].

At the beginning of a scan, **GAMBIT** creates **DDCalc** WIMP and Halo objects, as well as a separate Detector object for each supported experimental analysis (see Sect. 5.2.4 for explanation of these classes). For each point in a scan, it updates the WIMP object with newly-calculated nuclear couplings from DarkBit, and the Halo object with any new halo parameters. DarkBit provides a series of ‘getter’ routines for different quantities that the user may be interested in, for each supported analysis: the observed and predicted number of events, the predicted number of background and signal events (broken down into SI and SD components if desired), and the final Poissonian log-likelihood for the given model and experiment. These functions are detailed in Table 18. Each of them depends on a single calculation routine, which passes the WIMP, Halo and relevant Detector object (or rather, their internal indices) to **DDCalc**’s own master **DDCalc_CalcRates** function (cf. Sect. 5.2.4). That function then computes the rates and likelihoods for the given analysis and model point, and provides them to the getter functions.

6 Indirect detection

6.1 Background

Besides collider and direct searches, the third traditional way of looking for DM is to test the particle hypothesis *in situ*, by identifying the (Standard Model) products that result from DM annihilation or decay at places with large DM densities. Locally, the injection rate of a Standard Model particle type \( f \), per volume and energy, is given by

\[
\frac{dQ(E_f, x)}{dE_f} = \frac{1}{N_x} \frac{\rho^2(x)}{m_\chi} \sum_i \left( \sigma_i v \right) \frac{dN_i}{dE_f}.
\]  

(39)

Here, \( \sigma_i \) is the annihilation cross section into final state \( i \), \( v \) the relative velocity of the annihilating DM particle pairs, (...) denotes an ensemble average over the DM velocities, and \( dN_i/dE_f \) is the (differential) number of particles \( f \) that result per annihilation into final state \( i \). The dark matter mass density is given by \( \rho \) and its particle mass by \( m_\chi \). \( N_x \) depends on the nature of the DM particle, e.g. \( N_x = 2 \) for Majorana fermions, and \( N_x = 4 \) for Dirac fermions. For decaying DM, one simply would have to replace \( \langle \sigma_i v \rangle \rho^2 / N_f \rightarrow m_\chi \Gamma_i \rho \) in the above expression, where \( \Gamma_i \) is the partial decay width.

The yields (i.e. the number and energy distribution of final state particles) are typically not significantly affected by the ensemble average, allowing to write \( \langle \sigma_i v \rangle dN_i/dE_f = \langle \sigma_i v \rangle dN_i/dE_f = \langle \sigma_i v \rangle dN_i/dE_f \) (and correspondingly for the decaying case), and therefore to tabulate \( dN_i/dE_f \), \( v = 0 \) for a pre-defined set of centre-of-mass energies for e.g. annihilation into quarks (given by the DM mass for highly non-relativistic DM). Interpolating between these tables rather than running event generators such as **Pythia** [100] for every model point constitutes a significant gain in performance.

The DM density enters quadratically into Eq. 39. This implies that substructures in the DM distribution (usually in form of subhalos within larger halos) in general enhance the observable annihilation flux significantly with respect to what one would expect in absence of substructures. In practice, one hence often replaces the DM density squared as follows

\[
\rho^2 \chi (x) = \left[ 1 + B_x (x) \right] \rho^2_{\chi, \text{no sub}} (x),
\]  

(40)

where \( \rho^2_{\chi, \text{no sub}} (x) \) is the DM distribution smoothed over substructures (describing the general distribution of DM in the main halo), whereas \( B_x (x) \) is the boost factor that parameters the enhancement due to substructure.

Once produced, those particles \( f \) then propagate through a often significant part of the Galaxy, before they reach the observer. The details of this process depend strongly on the type of messenger, as well as on the type of the source. **Gamma rays** (Section 6.1.1) play a pronounced role in this context as they propagate completely unperturbed through the Galaxy, for energies below a few TeV, thus offering distinct spatial and spectral features to look for [101]. While much harder to detect, this property is shared by **neutrinos** (Section 6.1.2); they are furthermore unique in that they can easily leave even very dense environments and hence are the only probes of expected high DM concentrations in celestial bodies like the Sun and the Earth [102].

**Charged cosmic ray particles**, on the other hand, are deflected by randomly distributed inhomogeneities in the Galactic magnetic field such that (almost) all directional information is lost. In particular, antiprotons provide a powerful tool to constrain DM annihilating or decaying to hadronic channels [103–105], while cosmic-ray positron data strongly constrain leptonic channels [106]. While charged cosmic rays are not included in this first release of **DarkBit**, the implementation of both propagation and relevant experimental likelihoods for these channels is high on the priority list for planned extensions of the code (see Section 8).
6.1.1 Gamma rays

Gamma-ray spectra from dark matter annihilation or decay can be broadly separated in two components (see Ref. [107] for a review). (1) Continuous gamma-ray spectra are generated in annihilations into quarks, massive gauge bosons and τ-leptons. The gamma-ray photons come here mostly from the decay of neutral pions (π⁰ → γγ), which are produced in fragmentation and hadronisation processes. Most of the gamma-ray energy is deposited into photons with energies about an order of magnitude below the dark matter mass. (2) Prompt photons are directly produced in the hard part of the annihilation process and lead to much sharper features in the gamma-ray spectrum, typically with energies close to the DM mass. The most extreme example is the annihilation into photon pairs (χχ → γγ), which gives rise to mono-energetic gamma rays [108]. Virtual internal bremsstrahlung [109] or box-like features from cascade-decays [110] can also play a significant role. Such sharp spectral features are usually much simpler to discern from astrophysical backgrounds, and hence play a central role in indirect dark matter searches.

Various target objects are interesting for dark matter searches with gamma rays. The predicted annihilation flux is largest from the centre of the Milky Way. However, the diffuse gamma-ray emission caused by cosmic rays along the line-of-sight towards the Galactic bulge makes detecting a signal from the Galactic centre subject to large systematic uncertainties. Simpler and basically background-free targets are dwarf spheroidal galaxies, which are dark matter-dominated satellite galaxies of the Milky Way. We will discuss the results from various targets and instruments that we use in DarkBit in Sect. 6.2.2.

In all cases, the morphology and intensity of the gamma-ray annihilation signal depends on the spatial distribution and clumping of dark matter in the target object, according to Eq. 40. For the various likelihoods that we discuss below, in most cases the user can choose to either employ the spatial distribution of dark matter in the Milky Way as defined in the halo model used in the corresponding scan (see Sect. 3), or to make the same assumptions on the target halos as in the corresponding publications from which the likelihoods were extracted. In both cases, we neglect the substructure boost, which can be of O(1) for reasonable assumptions [111]. It is however straightforward to change the halo properties, if so desired.

6.1.2 Neutrinos

Like other indirect probes, searches for high-energy astrophysical neutrinos can be used to constrain the DM annihilation cross-section, by looking in directions with high DM densities such as the Galactic centre and dwarf galaxies. Unlike other indirect searches however, neutrinos can also probe nuclear scattering cross-sections. This is because a cross-section with nuclei implies that DM can scatter on nuclear matter in stars and other compact objects, losing sufficient energy to become gravitationally bound to the object’s potential well [112–115]. Being on a bound orbit, the DM then returns to scatter repeatedly, eventually settling to an equilibrated, concentrated distribution within the stellar body. If it is of a variety that is able to annihilate, DM therefore does so in the stellar core, producing high-energy annihilation products. Many of these products are stopped very quickly by interactions with nuclei, forming unstable intermediaries such as B mesons, which go on to decay to other SM particles including high-energy neutrinos [116].

Neutrinos produced this way, and any produced directly in the annihilation, are able to escape the stellar body because they interact so weakly with nuclei. They can then propagate to Earth, and be detected by neutrino telescopes. The most promising target for this type of search by far is the Sun, owing to its proximity and deep potential well, allowing it to accumulate large amounts of DM.⁷

At present, neutrino telescope limits on the total DM annihilation cross-section are weak [124,125], and cannot compete with gamma rays. In DarkBit we therefore currently focus exclusively on solar searches for high-energy neutrinos, and their implications for annihilating DM models with significant nuclear scattering cross-sections. Here ‘high energy’ means GeV-scale neutrino energies; MeV-scale neutrinos are more difficult to distinguish from regular solar (fusion) neutrinos, and neutrinos with energies in the TeV range and above suffer from significant nuclear opacities in the Sun, making their escape difficult and substantially lowering their fluxes at Earth.

For scattering cross-sections that do not depend on the relative velocity or momentum transfer in the DM-nucleus system, the capture rate in the Sun is given by [114]

$$C(t) = 4\pi \int_0^{R_\odot} \frac{f_\odot(u)}{u} \frac{w}{\Omega_w(w)} \int_{u}^{\infty} \frac{f_\odot(u)}{u} \frac{w}{\Omega_w(w)} \, du \, dr,$$

where \(r\) is the height from the solar centre, \(u\) is the DM velocity before being influenced by the Sun’s gravitational potential, and \(f_\odot(u)\) is the distribution of \(u\) in the Sun’s rest frame. The quantity \(\Omega_w(w)\) is the scattering rate of DM particles from velocity \(w\) to less than \(v\), where \(v\) is the local escape velocity at height \(r\) in the Sun, and \(w = w(u, r, t) = \sqrt{u^2 + v(r, t)^2}\) is the velocity an infalling DM particle obtains by the time it collides with a nucleus at height \(r\). \(\Omega_w(w)\) thus describes the local capture rate at height \(r\) in

⁷ We point out that such a population of DM in the Sun and other stars can have a raft of other observable consequences beyond high-energy neutrinos, which can be highly relevant for some models [117–123]; these are slated for inclusion in future releases of DarkBit.
the Sun, from the part of the velocity distribution corresponding to incoming velocity \( u \). The total population \( N_\chi \) of DM in the Sun can then be determined at any point in its lifetime by solving the differential equation

\[
\frac{dN_\chi(t)}{dt} = C(t) - A(t) - E(t), \tag{42}
\]

where \( A \) and \( E \) are the annihilation and evaporation rates, respectively. Except where DM is lighter than a few GeV, \( E \) is generally negligible [126–128]. Assuming \( E = 0 \), and that \( C \) and \( A \) are constant, the solution to Eq. 42 approaches a steady state on characteristic timescale \( t_\chi \), the equilibration time between capture and annihilation. When this steady state is reached, the rate-limiting step in the whole process is capture rather than annihilation. In this regime, the annihilation rate is identical to the capture rate, and the annihilation cross-section has no further bearing upon the number of neutrinos coming from the Sun. Many previous analyses have assumed that capture and annihilation are in equilibrium in the Sun, so that the annihilation rate can be obtained directly from the capture rate; in \textsc{DarkBit} we instead solve Eq. 42 and determine \( N_\chi \) explicitly for each model.

Knowing the DM population (and therefore annihilation rate) by solving Eq. 42, the annihilation branching fractions can then be used to determine the spectra of high-energy particles injected into the Sun, on a model-by-model basis. The stopping of these annihilation products, ensuing particle production and decay, and subsequent propagation and oscillation of neutrinos through the Sun, space and Earth, have been studied by extensive Monte Carlo simulations [116, 129]. The resulting neutrino yield tables at Earth are included in \textsc{DarkSUSY} [15], \textsc{PPPC4DMID} [42] and \textsc{micrOMEGAs} [16]. In \textsc{DarkBit}, the canonical method to obtain these fluxes is to compute the capture and annihilation rates using \textsc{DarkSUSY}, and to then obtain neutrino yields at Earth from the \textsc{WimpSim} [130] tables contained therein (although getting the same from \textsc{PPPC4DMID} or \textsc{micrOMEGAs} would also be straightforward).

Although SI scattering of DM by e.g. oxygen, helium and iron can dominate the capture rate for some models, the differing strength of direct limits on SI and SD scattering cross-sections, and the fact that hydrogen possesses nuclear spin, mean that typically, solar neutrinos are most interesting for SD scattering.

Neutrino telescopes are presently responsible for the strongest limits on the SD scattering cross-section with protons, with IceCube providing the tightest limits above masses of \( \sim 100 \text{ GeV} \) [17, 131], Super Kamiokande (Super-K) dominating at lower masses [132], and ANTARES and Baksan providing weaker constraints. We implement the IceCube search likelihood on a model-by-model basis, using the nulike package [17, 133] to compute the likelihood function for each model, given its predicted neutrino spectrum at Earth. \textsc{Nulike} computes a fully unbinned likelihood based on the event-level energy and angular information contained in the three independent event selections of the 79-string IceCube dataset [134]. We do not implement a Super-K likelihood for now, as a) unlike IceCube, Super-K have not publicly released their event data, b) IceCube does have some sensitivity at low mass, and c) Super-K data only become more constraining for relatively light DM particles (at least in the context of SUSY and scalar singlet models, given other constraints).

6.2 \textsc{gamLike}

6.2.1 Overview

Constraints on dark matter annihilation come from gamma-ray observations of various targets using various instruments. The experimental collaborations usually present their results as constraints on particular annihilation channels, using particular dark matter profiles. This makes the limits not only often difficult to compare, but also makes it hard to directly use the experimental results in scans over dark matter models with complex final states. In order to simplify and unify the adoption of gamma-ray indirect detection results in global scans and beyond, we present \textsc{gamLike}.

The \textsc{gamLike} code is released under the terms of the MIT license\(^9\) and maintained as a standalone backend by the GAMBIT Dark Matter Workgroup. It can be obtained from http://gamlike.hepforge.org.

The present version of \textsc{gamLike} ships with the likelihood functions discussed in Sect. 6.1.1, which are also listed in Table 7. It is written in C++ and can be linked either as shared library (this is how it is used in \textsc{GAMBIT}), or just as a static library. All experimental likelihoods are called in the same way: with a function that takes as its argument a tabulated version of the quantity

\[
\frac{d\Phi}{dE} = \frac{\sigma v}{8\pi m_\chi^2} \frac{dN_\nu}{dE}, \tag{43}
\]

which is the differential version of Eq. 45. The integration over energy bins happens within \textsc{gamLike} according to the energy bins used in the various analyses. Eq. 43 holds for self-conjugate dark matter particles, but can be easily adapted to e.g. Dirac fermion dark matter by using the appropriate prefactors as discussed in Sect. 6.1.

\(^8\) We note that a tool with a similar purpose, \textsc{LikeDM} [141], which deals with both gamma rays and charged cosmic rays, has recently been released.

\(^9\) http://opensource.org/licenses/MIT.
Various options for the so-called J-factors, which describe the effective DM content of the targets, are included in gamLike as well. These make it possible to marginalise or profile over J-factor uncertainties. The implementation of the combined dwarf limits from Ref. [136], for example, performs a profiling over the J-factors of all 15 adopted dwarfs separately for determining the combined likelihood value. The various implemented treatments are listed in Table 7.

### 6.2.2 gamLike targets

#### Dwarf spheroidal galaxies with Fermi-LAT

Some of the most stringent and robust limits on the dark matter annihilation cross-section come from the non-observation of gamma-ray emission from dwarf spheroidal Galaxies (dSphs). The most recent and stringent constraints on gamma-ray emission from dSphs from six years of data of the Fermi Large Area Telescope (LAT) were derived in Ref. [136], based on the new Pass 8 event-level analysis. They performed a search for gamma-ray emission from the sources and presented upper limits that in general disfavour s-wave dark matter annihilation into hadronic final states at the thermal rate for dark matter masses below around 100 GeV.

The results from Ref. [136] are available as tabulated binned Poisson likelihoods. The composite likelihood from the dSph analysis is given by

\[
\ln \mathcal{L}_{\text{exp}} = \sum_{k=1}^{N_{\text{dSph}}} \sum_{i=1}^{N_{\text{ebin}}} \ln \mathcal{L}_{ki}(\Phi_i \cdot J_k).
\]

Here \(N_{\text{dSph}}\) is the number of dwarf spheroidal galaxies included in the analysis, and \(N_{\text{ebin}}\) is the number of energy bins to be considered. The partial likelihoods \(\mathcal{L}_{ki}\) depend on the predicted signal flux \(\Phi_i \cdot J_k\). This is a product of the particle physics factor

\[
\Phi_i = \frac{(\sigma v)_0}{8 \pi m_\chi^2} \int_{E_i} dE \frac{dN_\gamma}{dE},
\]

which depends only on the gamma-ray yield per annihilation \(dN_\gamma/dE\) and the zero-velocity limit of the annihilation cross-section \((\sigma v)_0 \equiv \sigma v|_{v\to 0}\), and the astrophysics factor

\[
J_k = \int_{\Delta E_i} d\Omega \int_{\text{i.o.s.}} ds \rho_\chi^2.
\]

Here, \(\Delta E_i\) and \(\Delta \Omega_k\) denote the energy bin \(i\) and solid angle \(k\) over which the signal is integrated, \(m_\chi\) is the mass of dark matter particles and \(\rho_\chi\) is the dark matter mass density in the target object.

Our knowledge of the distribution of dark matter in dSphs relies on Jeans analyses of the kinematics of member stars. Following Refs. [136, 142], to a good approximation the corresponding uncertainties for the J-factors can be modelled by a log-normal distribution,

\[
\ln \mathcal{L}_J = \sum_{k=1}^{N_{\text{dSph}}} \ln \mathcal{N}(\log_{10} J_k | \log_{10} \tilde{J}_k, \sigma_k),
\]

with parameters taken from Ref. [136], and \(\mathcal{N}(x|\mu, \sigma)\) being a normal distribution with mean \(\mu\) and standard deviation \(\sigma\).

The halo and gamma-ray likelihoods can be combined by profiling over the nuisance parameters \(J_k\). The corresponding profile likelihood is given by
\[ \ln L_{\text{dwarfs}}^{\text{prof}}(\Phi_1) = \max_{J_1 \ldots J_k} (\ln L_{\text{exp}} + \ln L_J). \] (48)

An alternative is to marginalise over the nuisance parameters, which yields the marginalised likelihood function

\[ L_{\text{dwarfs}}^{\text{marg}}(\Phi_1) = \int dJ_1 \ldots dJ_k L_{\text{exp}} \cdot L_J. \] (49)

The main results from Ref. [136] are derived using the composite profile likelihood for 15 dwarfs, and this is what we implemented in \texttt{gamLike}. Furthermore, for comparison, we also implemented the older results from Ref. [135], which are based on four years of pass 7 data. In both cases, the energy range spans from 500 MeV to 500 GeV. The implemented likelihoods are listed in Table 7.

Note that the effects of energy dispersion are neglected when evaluating the constraints. Given that current constraints from a dedicated line search are much more constraining than the line limits that one can derive from dSph observations, this limitation is of little practical consequence. Implementing \texttt{Fermi}-LAT line search results is high on the priority list for future versions of \texttt{gamLike} (see Sect. 8).

**The `Fermi` Galactic centre excess**

Gamma-ray observations of the Galactic centre with the \texttt{Fermi}-LAT identified an extended excess emission at GeV energies, which can be interpreted as a dark matter annihilation signal (see e.g. [138,143–148]). Although the case for millisecond pulsars as explanation for the excess emission was strengthened in recent analyses [149,150], it remains interesting to consider the consequences of various DM explanations.

Ref. [138] presented a spectral characterisation of the GeV excess that included estimates of systematic uncertainties, which were derived from residuals along the Galactic disk. These uncertainties are correlated over the different energy bins. The corresponding likelihood function was approximated to be Gaussian, and has the form

\[ \ln L_{\text{GC}} = -\frac{1}{2} \sum_{ij} (J_{\text{GC}} \Phi_i - f_i) \Sigma_{ij}^{-1} (J_{\text{GC}} \Phi_j - f_j), \] (50)

where \( f_i \) denotes the measured flux in energy bin \( i \), \( \Sigma_{ij} \) is the covariance matrix, and \( J_{\text{GC}} \) denotes the \( J \)-factor of the Galactic centre Region-Of-Interest (ROI). The considered energy range spans from 300 MeV to 500 GeV, and the ROI covers Galactic latitudes \( 20^\circ > |b| > 2^\circ \) and longitudes \( |\ell| < 20^\circ \).

For the \( J \)-factors, the default behaviour is to employ the value calculated from the halo model used in the corresponding scan (see Sect. 3).\(^{11}\) Alternatively, one can choose a fixed \( J \)-factor, \( J = 2.07 \times 10^{23} \text{GeV}^2 \text{cm}^{-5} \), corresponding to a contracted NFW profile with inner slope \( \gamma = 1.2 \) (see Ref. [138] for details), or derive a marginalised likelihood function assuming a log-normal distribution of \( J \)-factors with mean \( 1.96 \times 10^{23} \text{GeV}^2 \text{cm}^{-5} \) and standard deviation of 0.37 (as done in Ref. [139], motivated by Ref. [151]).

Finally, we also included a likelihood function that (on top of astrophysical uncertainties) includes an estimated 10% uncorrelated systematic accounting for possible uncertainties in the modelling of the DM signal spectrum. (This scenario was considered in the work of Ref. [139], and we include it here for completeness.) All the available likelihoods are listed in Table 7.

**Galactic centre observations with H.E.S.S.**

For dark matter masses above several hundred GeV, the current best limits on dark matter annihilation come from observations of the Galactic centre with the Air Cherenkov Telescope H.E.S.S. [137], based on 112 hours of data. The limits are derived from a comparison of the measured gamma-ray fluxes in a search region within 1° of the Galactic centre, and a background region just outside the inner 1°. Limits are derived from the non-observation of an excess of gamma-ray emission in the signal region over the flux measured in the background region.

We model the corresponding likelihood function as a Gaussian,

\[ \ln L_{\text{HESS}} = -\frac{1}{2} \sum_{i=1}^{N_{\text{bin}}} (\Phi_j (J_{\text{sig}} - J_{\text{bg}}) R - f_i^{\text{sig}} - f_i^{\text{bg}} R)^2 \Delta f_i^2, \] (51)

where \( J_{\text{sig(bg)}} \) denotes the \( J \)-factors in the signal (background) regions, \( f_i^{\text{sig(bg)}} \) the corresponding fluxes in energy bin \( i \), and \( R = \Omega_{\text{sig}}/\Omega_{\text{bg}} \) is a geometrical rescaling factor that depends on the angular size \( \Omega_{\text{sig(bg)}} \) of the signal (background) region. For the dark matter profile, we assume by default the halo model employed in the corresponding scan. Alternatively, one can use the NFW profile used in Ref. [137] (local dark matter density of 0.39 GeV cm\(^{-3} \) at 8.5 kpc distance from the Sun).

In Ref. [137] limits are derived from a combination of all energy bins into one single wide energy bin from 265 GeV to 30 TeV. Using this same energy bin, we can reproduce the results from that work. However, Ref. [137] also provides enough information for a spectral analysis with 35 energy bins in the range 230 GeV to 30 TeV, which we implemented as well (see Table 7). It provides more accurate results in

\(^{11}\) Note that here only the overall flux within the region \( |\ell| < 20^\circ \) and \( 2^\circ < |b| < 20^\circ \) is rescaled. Variations in the signal morphology are not taken into account in the current treatment.
cases where the DM signal has a pronounced spectral structure (like the annihilation spectrum of \(\tau\)-leptons). However, because the effects of energy dispersion are not included in the present version of the likelihood, results obtained with this likelihood should be interpreted with care. Spectral features like gamma-ray lines should be constrained by results from a dedicated line search.

**Projected Galactic centre searches with CTA**

As discussed in Ref. [140], the sensitivity of the future Cherenkov Telescope Array (CTA) to diffuse emission from DM annihilation will be not limited by statistics, but mainly by systematic uncertainties in the differential detector acceptance within the Field-of-View (FoV), and by the modelling of the Galactic diffuse emission. Ref. [140] addressed these issues and proposed a combined morphological analysis of the fluxes measured in different segments of the FoV. To this end, it is (optimistically) assumed that the Galactic diffuse emission can be modelled well up to an overall unconstrained normalisation. The results of that work can be represented as tabulated likelihood functions [152].

The corresponding likelihood function takes essentially the form of Eq. 44 above, and covers energies between 25 GeV and 10 TeV. The halo model is fixed to the Einasto profile with local density \(\rho_\chi = 0.4\text{ GeV cm}^{-3}\), used in the original analysis (since this analysis was taking into account morphological information, it is for the projected CTA limits currently not possible to adopt the general halo model that is used in the scan). Further details about the adopted detector response, Galactic diffuse emission model and DM profile can be found in [140].

**6.2.3 Library interface (API)**

The `gamLike` library interface is the same for all implemented experiments. There are four relevant functions that can currently be called:

- `void set_data_path(const std::string & path)`
  Sets the path to the data files
- `void init(experiment_tag)`
  Initialise the likelihood for `experiment_tag`
- `double lnL(experiment_tag, const std::vector<double>& E, const std::vector<double>& dPhidE)`
  Retrieve log-likelihood \(\mathcal{L}\) for `experiment_tag`, given the tabulated \(d\Phi/dE\) as in Eq. 43
- `void set_halo_profile(int mode, const std::vector<double>& r, const std::vector<double>& rho, double dist)`
  Initialise the Galactic halo model

The `experiment_tag` is a C++ enum that corresponds to one of the likelihoods listed in Table 7. The relevant functions and the enum are declared in `gamLike.hpp`, and a C-compliant API for e.g. linkage with Fortran code is available in `gamLike_C.hpp`.

The range over which \(d\Phi/dE\) is tabulated should cover the energy range of the activated experiments as shown in Table 7. Outside of the tabulated range it is assumed to be zero. The integration over energy bins is a simple trapezoidal integration based on the tabulation grid. This has the advantage that spectral features can be arbitrarily well resolved, provided the user chooses a fine enough grid in the critical energy range (but we note again that energy dispersion effects are not currently included in `gamLike`).

**6.3 Implementation of indirect detection in DarkBit**

**6.3.1 The Process Catalogue**

One of the central structures in `DarkBit` is the 'Process Catalogue'. This is an object of C++ type `DarkBit::TH_ProcessCatalog`. Functions able to generate the Process Catalog (Table 13) have capability `TH_ProcessCatalog`. The Process Catalogue carries all relevant information about the decay and annihilation of particles. This information is mainly used to calculate dark matter annihilation rates, gamma-ray and neutrino yields for indirect searches. It can also be used for relic density calculations, although in this case coannihilation processes are currently not supported. The information from the Process Catalogue is also used in the cascade annihilation Monte Carlo, which we discuss in Sect. 6.3.4.

The Process Catalogue is a simple C++ struct. The internal structure is most easily summarised using the following nested list.

**Process Catalogue:**

- **Processes**
  - Initial state: \(\chi\chi\)
    - Channels
      - \(bb\): genRate
      - \(\mu^+\mu^-\gamma\): genRate
      - ...
    - genRateMisc
  - Initial state: \(h^0\)
    - ...
  - ...
- **Particle properties**
  - \(\chi\): mass, spin
  - \(h^0\), mass, spin
  - ...

For a detailed example of how to set up the catalogue and access data within, we refer the reader to the source
code of the DarkBit_standalone_WIMP example program described in Sect. 7.2.

The Process Catalogue carries a list of annihilation and decay processes. Each process has one (decays) or two (annihilations) initial states, and a list of decay/annihilation channels. Each channel consists of a list of two or three final state particles (more than three final state particles are currently not supported), as well as the specific rate for that channel given by genRate, which has type daFunk::Funk (see Appendix B for details). It provides information about the decay width or the annihilation cross section of the described process.

In the case of two-body decay, genRate is simply a constant that equals the decay width $\Gamma$ in GeV (cf. Table 8). In the case of two-body annihilation, it is the annihilation cross-section $\langle \sigma v \rangle$, given as a function of the relative velocity $v$. For three-body decays, genRate refers to the differential decay width, which is a function of the two kinematic variables $E$ and $E_1$, corresponding to the energy of the first and second final state particles, respectively. In the case of three-body annihilation final states, genRate refers to the differential annihilation rate, and has an additional dependence on the relative velocity $v$. For three-body decays, genRate refers to the differential decay width, which is a function of the two kinematic variables $E$ and $E_1$, corresponding to the energy of the first and second final state particles, respectively. In the case of three-body annihilation final states, genRate refers to the differential annihilation rate, and has an additional dependence on the relative velocity $v$.

Lastly, each process also has a genRateMisc, which describes additional invisible contributions to the decay width or annihilation cross-section that are not associated with a specific channel, but can affect relic density calculations. The different roles of genRate and genRateMisc are summarised in Table 8. Note that genRateMisc enters the calculation of $W_{\text{eff}}$ from the Process Catalogue (if this how the user has chosen to obtain $W_{\text{eff}}$), but does not directly affect annihilation yields.

Besides the list of processes, the catalogue also comes with a list of particle properties relevant for its processes. This section of the catalogue maps particle IDs onto particle masses and spin. Masses are required for calculating decay or annihilation kinematics. The remaining information is currently unused, but has obvious potential future applications.

We stress that channels involving three-body final states are conceptually different from those with two-body final states, because they cannot be implemented independently from the two-body states, unless the contribution from any of the associated two-body processes is absent or strongly suppressed. (An example of the latter situation is virtual internal bremsstrahlung from neutralinos annihilating to light fermions [109].) In general, three-body final states provide a correction to the tree-level result, and genRate hence returns the difference between the full NLO spectrum and the spectrum at tree level. The output can therefore be positive or negative. This implies that setting up many-body final states in the Process Catalogue requires detailed knowledge of how the tree-level annihilation or decay spectrum is defined.12

While the structure of the Process Catalogue in principle allows one to take into account all possible radiative corrections (with the above caveats in mind), currently only three-body final states involving hard photons are included explicitly in the Process Catalogue. Contributions from the decay and/or fragmentation of (on-shell) final state particles can be obtained either from tabulated yield tables (Sect. 6.3.2) or via DarkBit’s own Fast Cascade Monte Carlo (FCMC; Sect. 6.3.4).

6.3.2 Gamma rays

As discussed above, the calculation of annihilation or decay spectra often involves tabulated results from event generators like Pythia [156]. In order to allow maximal flexibility with the adopted yield tables, DarkBit provides access to tabulated yields using a general structure called SimYieldTable. Currently, this structure makes it possible to import spectra from the DarkSUSY and micrOMEGAs backends,13 but can eas-

---

12 Final state radiation of photons and gluons, for example, is often argued to contribute to the ‘model-independent part’ of the three-body spectrum, and to therefore typically be included already in tabulated yields from two-body final states obtained by Monte Carlo simulation. Sometimes even electroweak final state radiation is added following a similar philosophy [42]. In general, however, all these contributions are highly model-dependent and can differ substantially from the ‘model-independent’ results [153–155]. For $qq$ final states, for example, the change in the normalisation of $qq$ final states due to loop corrections at the same order in $\alpha_s$ must also be included for consistency [153]. For three-body final states involving Higgs or electroweak gauge bosons, it is challenging to even define the contribution from a single annihilation or decay channel in a consistent way [154,155]. Although this can of course always be done formally for the sake of fitting into the structure of the Process Catalogue, no particular physical significance should be associated with any individual channel in that case. Rather, only the sum over all three-body channels provides a meaningful correction (to the total tree-level yield resulting from the sum over all associated two-body channels).

13 If the YAML option allow_yield_extrapolation is set to true, the spectra from these backends are extrapolated to dark matter masses that have not been covered by the corresponding Pythia runs. By default, extrapolation is not performed, in which case the DM mass is limited to values below 5 TeV both for DarkSUSY and micrOMEGAs, while there is a lower limit of 10 GeV for DarkSUSY and 2 GeV for Footnote 13 continued micrOMEGAs. Setting the flag to true allows for DM masses of
ily be extended to other sources. The information carried by a `SimYieldTable` is summarised in the following.

| SimYieldTable: |
|----------------|
| • Channel list |
|   – \( \bar{b}b \): dNdE, Ecm\_min, Ecm\_max |
|   – \( \mu^+ \mu^- \): dNdE, Ecm\_min, Ecm\_max |
|   – ... |

Each one- or two-particle spectrum is defined by the ID(s) of the initial particle(s), the ID of the tabulated final-state particle (currently only gamma-rays are implemented), and the centre-of-mass energy range for which the tabulated spectrum is available. The spectrum itself is conveniently wrapped into a `daFunk::Funk` object (like `genRate` above, see Appendix B for more details). We emphasise that this object does not, in most cases, directly carry the tables from DarkSUSY or micrOMEGAs, but is merely a convenient and flexible wrapper that directly calls the corresponding backend functions.

In extreme cases, the tabulated yields implemented in different dark matter codes can differ substantially – mostly due to being based on different (versions of) event generators, but also due to different methods (or the absence of a method) for including contributions from higher-order processes. With the above structure, DarkBit offers a flexible and convenient way to select the desired yields and switch between them for detailed comparisons.

The gamma-ray yield is calculated by module functions with the capability `GA_AnnYield`, based on information from both the ProcessCatalog and the SimYieldTable. These are outlined in Table 19. Note that the resulting spectra are in general only partially based on the SimYieldTable, and can also make use of analytic expressions for e.g. three body final states, or include results from the FCMC (Sect. 6.3.4). The result of `GA_AnnYield` is a `daFunk::Funk` object. It refers to the physical expression \( m_\chi^2 \cdot \sigma v \cdot dN/dE \), which is equivalent to Eq. 43 up to a factor of \( 8\pi \). It is a function of the photon energy "E" and of the relative velocity "v" (currently, only the \( v = 0 \) case is used for actual likelihood evaluations; adding velocity-dependent effects is planned for the near future). Note that the function object is in general a composite object that wraps various DarkSUSY and micrOMEGAs functions providing e.g. tabulated yields or differential cross-sections for three-body final states. Calling this generalised function at different values of \( E \) or \( v \) calls all of these backend functions behind the scenes, sums up and rescales their results, and performs phase space integrations if necessary.

Note that the calculation of gamma-ray spectra often involves internal integrations (over 3-body phase space, or for boosting spectra into the lab frame). In cases where the integrations fail, a warning is issued, and the integration returns zero (see Appendix B). This implies that derived upper limits are in all cases conservative.

If results from the FCMC are required, the Monte Carlo simulation is automatically run before the `GA_AnnYield` module function (see Sect. 6.3.4). It is the job of module functions with capability `GA_missingFinalStates` to determine, by comparing Process Catalogue entries and the SimYieldTable, for which final states the cascade annihilation Monte Carlo is necessary.

In Fig. 3, we show a number of annihilation spectra generated with `GA_AnnYield`, comparing the yields obtained from various backends, including line features. The processes are indicated in the legend of the figure. One of the spectra shown in Fig. 3 corresponds to annihilation into \( Z^0 \gamma \) final states. The actual spectrum is calculated as combination of a monochromatic \( \gamma \) and the gamma-ray yield from decay of a single \( Z^0 \). The latter is approximated by the tabulated spectrum from threshold decay into \( Z^0 Z^0 \) final states, with the photon yield divided by two.

Gamma-ray likelihood functions in DarkBit make use of the backend `gamLike` (see Sect. 6.2 and Table 7). As detailed in Table 20, the resulting likelihoods are wrapped in various DarkBit module functions, with one capability for each experiment-target pair. The different options concerning \( J \) factors or the version of a measurement can be selected with the run-time option `version` in the YAML file. The list of available capabilities and modes is:

- up to 1 PeV and down to the mass of the final state particle both for DarkSUSY and micrOMEGAs.
6.3.3 Neutrinos

The different neutrino indirect detection capabilities of DarkBit are summarised in Table 21. The capabilities that describe the relevant WIMP properties are listed in Table 14.

The neutrino routines in DarkBit use the DM mass and nuclear scattering cross-sections to first calculate the DM capture rate in the Sun (capability capture_rate_Sun). The canonical way to do this is to call the corresponding function from DarkSUSY, which (at least in v5.1) assumes the AGSS09ph solar density profile [157, 158], and does not distinguish between scattering on protons and neutrons. DarkBit uses the SI and SD cross-sections on protons for this purpose.

The capture rate is then used together with the late-time annihilation cross-section to solve Eq. 42 for the annihilation rate (capabilities annihilation_rate_Sun and annihilation_rate_Sun). Together with the contents of the ProcessCatalog, the annihilation rate is used to prime the calculation of the neutrino spectrum at Earth (essentially by setting appropriate common blocks in DarkSUSY with annihilation and Higgs decay information), producing a pointer to a function in DarkSUSY that can return the neutrino yield at the IceCube detector (capability nuyield_ptr).14

This pointer is passed to nulike [17], which uses it to convolve the predicted differential neutrino flux with the various IceCube detector response functions, and evaluate the overall neutrino telescope likelihood for the model. DarkBit provides individual likelihoods from each of the three independent event selections included in the original 79-string IceCube analysis [134]: winter high-energy (WH), winter low-energy (WL) and summer low-energy (SL). The combined likelihood from all three of these searches is provided under the capability IC79_loglike; the individual likelihoods correspond to IC79WH_loglike, IC79WL_loglike and IC79SL_loglike. The earlier 22-string likelihood [133, 159] is also available as IC22_loglike, and combined with all 79-string likelihoods as simply IceCube_likelihood. More information about these capabilities is available in Table 22.

14 The neutrino yield can also be calculated using routines in micrOMEGAs, but these functions are currently not backedend in GAMBIT. We plan to add the ability to use them as an alternative to the DarkSUSY calculation in a future version of DarkBit.

When combining different search regions like this, we work with an effective log-likelihood defined as the difference between the actual log-likelihood and the log-likelihood of the background-only model.15 We also impose a hard upper limit of zero to this effective likelihood, to prevent overfitting of spurious features at low energies, where the IceCube limits degrade steeply and the instrumental response functions (especially in energy) are least well understood. This has the impact of preventing exclusion of the background hypothesis, in much the same way as the $\mathcal{CL}_s$ method [160, 161].

The nulike speed parameter can be chosen from the master initialisation file, by setting the module function option nulike_speed. The default is 3, but the speed setting is configurable from the master YAML file via the module function option nulike_speed.

![Comparison of different limits obtained from 79-string IceCube data using nulike, with different speed settings. The default is 3, but the speed setting is configurable from the master YAML file via the module function option nulike_speed.](image)

15 This is the same strategy as employed by ColliderBit in combining different LHC searches; more information on the procedure can be found in that paper [11]. This case is somewhat simpler than the ColliderBit one, as we do not allow different signal region combinations for different model parameter values, given that the IceCube event selections are statistically independent by construction. Also unlike the LHC searches implemented in ColliderBit, here the different signal regions do not come with different numbers of counting bins – each has exactly one such ‘bin’, the Poisson counting term at the front of its likelihood function – but different event selections do bring different numbers of event terms into the unbinned part of the likelihood function [17].
Following the original nulike paper [17], we assume a flat theoretical error on the predicted neutrino yield of 5% for DM masses below 100 GeV, rising logarithmically to 50% at masses of 10 TeV, and onward at even higher masses. This form of the error term is designed to account for neglected higher-order contributions and round off errors, present at all masses, and the increasing error introduced by DarkSUSY’s interpolation in its WimpSim masses, and the increasing error introduced by DarkSUSY’s interpolation in its WimpSim tables with increasing mass above ~100 GeV.

As side products of the various likelihood calculations, DarkBit also provides module function access to various related nulike outputs for the different analyses (Table 22). For \( X \in \{ \text{IC22}, \text{IC79WH}, \text{IC79WL}, \text{IC79SL} \} \), these are:

- \( X_{\text{signal}} \): The predicted number of signal events (from DM annihilation in the Sun).
- \( X_{\text{bg}} \): The predicted number of background events.
- \( X_{\text{nobs}} \): The total number of observed events.
- \( X_{\text{bgloglike}} \): The likelihood for the background-only model.
- \( X_{\text{pvalue}} \): A simple \( p \)-value for the model, based only on the number of events observed in the individual analysis, i.e. discarding event-level information.

These are all extracted from nudata objects, which are returned by functions with capabilities \( X_{\text{data}} \).

### 6.3.4 Fast Cascade Monte Carlo (FCMC)

In DarkBit, the calculation of gamma-ray yields from cascade decays is implemented as a Monte Carlo, as the kinematics of long decay chains are too complicated to handle analytically. Codes like DarkSUSY, for example, take simplified angular averages over decay phase spaces.

The cascade decay code has two main parts: a decay chain Monte Carlo and an accompanying analysis framework. The Monte Carlo code generates random decay chains based on relative branching fractions for individual decays. Currently, all particles in the decay chain are assumed to be on-shell, and only two-body decays are allowed in each step of the chain. Furthermore, spin correlations are neglected. The analysis framework interpolates the resulting histograms and creates wrapper functions for these interpolated spectra, which can be used in e.g. GA_AnnYield in order to derive the total annihilation yield. The generation of decay chains, and the following analysis steps are fully OpenMP-enabled. Each available CPU core independently generates and analyses events.

As mentioned in Sect. 6.3.2, it is the responsibility of each module function with capability GA_missingFinalStates to determine, by comparing Process Catalogue entries with the SimYieldTable, for which initial state particles gamma-ray yields are required. The FCMC then generates decay chains for each of the identified initial states by Monte Carlo.

The decay chains in DarkBit are implemented as a doubly-linked tree: each particle in the decay chain is represented by an instance of a class named ChainParticle, which contains pointers to its parent particle, and any child particles (decay products). A decay chain is generated by first creating an initial state ChainParticle, which is initialised using a decay table containing relevant masses and decays for all particles that can occur in the cascade. The ChainParticle class features a member function named generateDecayChainMC, which recursively generates a decay chain. The function uses the FCMC-internal decay table (see below) to select a decay from the list of possible processes, using probabilities given by the relative branching fractions for the allowed decays.\(^{16}\)

The recursive decay continues until particles that are stable or have pre-computed decay spectra are reached, or until one of the pre-defined cutoff conditions is reached. These cutoff conditions are the maximum number of allowed decay steps, as specified by the YAML option cMC_maxChainLength, and a cut on the lab frame\(^{17}\) energy of the decaying particle, specified by the YAML option cMC_Emin. The cutoff is triggered by whichever of these two conditions is reached first.

Once a full decay chain has been generated, the final state particles of the chain are collected and analysed, and final states of interest are histogrammed. Tabulated final state spectra are boosted to the lab frame. To this end, we sample photon energies from the tabulated spectra, boost these to the lab frame, and add the corresponding box spectra to the result histogram.

In the remaining part of this section, we provide information about the implementation in DarkBit. The relevant functions and capabilities are summarised in Tables 23 and 24. The module function cascadeMC_FinalStates provides a list of string identifiers ("gamma", "e+", "pbar", etc) that indicate which final states need to be calculated. This list can be set in the master YAML file using the option cMC_finalStates. The default (and currently only supported) option is a list with the single entry "gamma". The function cascadeMC_DecayTable generates an FCMC-internal list of all relevant decays (based on the content of the Process Catalogue). Next, the loop manager cascadeMC_LoopManagement runs a sequence of module functions with capabilities cascadeMC_InitialState \( \rightarrow \) cascadeMC_ChainEvent \( \rightarrow \) cascadeMC_Histograms \( \rightarrow \) cascadeMC_EventCount that takes care of generating the Monte Carlo samples and histograms. Finally, cascadeMC_gammaSpectra uses the histogram results to generate interpolating daFunk::Funk objects, which are used in GA_AnnYield.

\(^{16}\) Relative, as the branching ratios of the two-body decays may not always sum up to one.

\(^{17}\) 'Lab frame' means the rest frame of the initial state in the cascade.
Various options are available to tune the FCMC. We list them here, with default values in square brackets.

\textbf{cascadeMC\_LoopManagement:}

- \texttt{int cMC\_maxEvents[20000]}: sets the maximum number of MC runs per point.

\textbf{cascadeMC\_GenerateChain:}

- \texttt{int cMC\_maxChainLength[-1]}: the maximum number of decay steps to consider. A value of -1 indicates that no maximum should be applied.
- \texttt{double cMC\_Emin[0.0]}: the minimum lab-frame energy of particles to be tracked, in GeV.

\textbf{cascadeMC\_Histograms:}

\textbf{Histogramming options:}

- \texttt{int cMC\_NhistBins[140]}: number of logarithmic bins of the generated histogram.
- \texttt{double cMC\_binLow[0.001]}: the lowest energy in GeV of the generated histogram.
- \texttt{double cMC\_binHigh[10000]}: the highest energy in GeV of the generated histogram.
- \texttt{int cMC\_numSpecSamples[25]}: the number of samples to draw from tabulated spectra.

\textbf{Convergence criteria:}

- \texttt{double cMC\_gammaBGPower[-2.5]}: power-law slope to assume for astrophysical background when calculating the position of the bin with the highest expected signal-to-background ratio.
- \texttt{double cMC\_endCheckFrequency[25]}: the number of events to wait between successive checks of the convergence criteria.

Note that if \texttt{cMC\_maxEvents} is exceeded in \texttt{cascadeMC\_LoopManagement} before convergence is reached in \texttt{cascadeMC\_Histograms}, the Monte Carlo will terminate before any convergence criteria are met.

We show example spectra generated with the FCMC in Fig. 5. These were set up using the DarkBit WIMP standalone discussed in Sect. 7.2. Specifically, we set up a pair of DM particles annihilating to two scalars, $\chi \chi \rightarrow \phi_1 \phi_2$, where $\phi_1$ decays to a pair of photons and $\phi_2$ to $b\bar{b}$. In the rest frame of $\phi_1$, the resulting photons are monochromatic; in the galactic rest-frame of the annihilating DM particles, this leads to a flat “box” feature with the following spectrum (see e.g. [110]):

\begin{equation}
\frac{dN_{\phi \rightarrow \gamma \gamma}}{dE_{\gamma}} = \frac{2}{\Delta E} \theta(E_{\gamma} - E_{\text{min}}) \theta(E_{\text{max}} - E_{\gamma}).
\end{equation}

Fig. 5 Gamma-ray spectra from DM cascade annihilation, generated with the DarkBit FCMC. DM particles, $\chi$, annihilate into bosons $\phi_1$ and $\phi_2$, which decay, respectively, in photon pairs and bottom quark pairs. For all lines, we assume $m_\chi = 100$ GeV and $m_{\phi_1} = 100$ GeV. The mass of $\phi_1$ varies between $m_{\phi_1} = 10, 30, 50, 90$ GeV (with the lowest mass corresponding to the magenta line with the pronounced box-like feature on the right).

Here, $\theta$ is a step function, $\Delta E \equiv E_{\text{max}} - E_{\text{min}}$ and

\begin{equation}
E_{\text{max, min}} = (E_{\phi_1}/2) \left( 1 \pm \sqrt{1 - m_{\phi_1}^2/E_{\phi_1}^2} \right),
\end{equation}

where

\begin{equation}
E_{\phi_1} = m_\chi \left[ 1 + (m_{\phi_1}^2 - m_{\phi_2}^2)/(4m_\gamma^2) \right]
\end{equation}

is the energy of $\phi_1$. These boxes are clearly seen in Fig. 5, with endpoints and normalisation of the numerical results agreeing nicely with the above analytical expression.

The decay $\phi_2 \rightarrow b\bar{b}$ produces a continuum spectrum of photons from the tabulated yields of $b\bar{b}$ final states. Compared to the direct annihilation of DM to $b\bar{b}$, as in $\chi \chi \rightarrow b\bar{b}$, the form of the resulting photon spectrum is roughly retained but the peak (in $E_{\gamma}^2 dN/dE_{\gamma}$) is shifted down by a factor of about 2 in energy [162] – essentially because each of the quarks now has on average only half the kinetic energy at its disposal. This part of the spectrum is clearly visible in Fig. 5 as the “background” of the box feature discussed above. At high energies, this part of the FCMC-produced spectrum is also seen to be affected by the mass of $\phi_1$ (for constant $m_{\phi_1} = m_\chi = 100$ GeV). The reason is that the largest photon energy kinematically available from $\phi_2 \rightarrow b\bar{b}$ is given by $E_{\text{max}}$ as provided in the expression after Eq. 52, with $\phi_1$ and $\phi_2$ interchanged. Again, this is in agreement with the location of the cutoff visible in the figure.

We finally note that the cascade code in the current form does not handle off-shell decay, and neglects the finite widths of particles in the kinematics.
7 Examples

In this Section we present a few selected examples that illustrate the scope and potential applications of DarkBit. At the same time, these examples serve as validation tests of the code.

7.1 CMSSM, MSSM and Singlet DM

To demonstrate the ability of DarkBit to calculate observables and likelihoods, we undertake a number of simple grid scans using the grid scanner [14]. For these demonstrations, we consider the parameter spaces of the CMSSM [163], MSSM7 [164] and scalar singlet DM [165] models. For each model, we choose 2 parameters that are particularly relevant for dark matter phenomenology. The parameters and their ranges are shown in Table 10. We vary only one parameter at a time, whilst keeping all others fixed at the values shown in Table 9, leading to a scan over two “spokes” in parameter space for each model (Fig. 6).

For each point in the scan, we calculate spin-independent and spin-dependent DM-proton scattering cross sections, velocity-averaged DM annihilation cross sections at late times, and the DM relic density. For all of these observables, we also calculate a corresponding experimental likelihood using an appropriate backend code included with DarkBit: the LUX 2013 likelihood from DDCalc (Sect. 5.2), the IceCube 79-string likelihood for WIMP annihilation in the Sun from nullike (Sect. 6.3.3), and the stacked dwarf spheroidal likelihood based on six years of Fermi data from gamLike (Sect. 6.2.2). For the relic density, we calculate the simple Gaussian likelihood based on the best fit value from the Planck analysis [1] described in Sect. 4.3. The results of these scans are shown in Fig. 6, where the colour-coding of the points represents the likelihood value. For comparison, we plot the limits from corresponding analyses from the LUX [75], IceCube [17], and Fermi [136] collaborations. For the relic density, we plot the Planck best fit value and its 2σ uncertainty. In all cases, the likelihoods calculated by DarkBit and its associated backends agree with the results from the experimental collaborations.

7.2 Effective WIMPs

In order to further illustrate some of the functionality of DarkBit, and to show how DarkBit can be used without a full scan in GAMBIT, DarkBit ships with three example standalone programs. One of them, DarkBit_standalone_WIMP, shows how to set up and calculate various observables for a simple WIMP model, in which the three parameters are the WIMP mass and the cross sections for WIMP self-annihilation and WIMP-nucleon scattering. We will discuss this example here in some detail. Further examples specific to singlet dark matter and the MSSM can be found as well; the MSSM example will be discussed in the next subsection.

All of the model specifics for the standalone example are specified in only three module functions. These are defined as QUICK_FUNCTIONS at the beginning of the source file of the example. One function, DarkMatter_ID_WIMP, simply returns the string identifier for the WIMP particle. Another function, DD_couplings_WIMP, sets up the direct detection couplings. In the present example, these are entirely determined by module function options. The most complex function is the function that sets up the Process Catalog for the given example, TH_ProcessCatalog_WIMP. Besides the relevant processes, the masses and spins of the participating particles also have to be defined. Furthermore, we define a few functions to dump gamma-ray annihilation spectra into ASCII tables.

| **Model** | **Parameter** | **Range** |
|----------|--------------|-----------|
| Singlet DM | $\lambda_{hS}$ | [0.01, 0.05] |
| | $m_S$ [GeV] | [70, 110] |
| CMSSM | $M_0$ [GeV] | [2840, 3310] |
| | $M_{1/2}$ [GeV] | [340, 590] |
| MSSM 7 | $M_2$ [GeV] | [450, 850] |
| | $m^2_{H_u}$ [GeV$^2$] | [9.2 × 10$^7$, 1.03 × 10$^8$] |
Fig. 6 Tracks in observable space corresponding to spokes in model parameter space. The arrows point in the direction of increasing parameter values. Clockwise from the top-left, panels show the DM relic density, spin-independent and spin-dependent DM-proton scattering cross sections, and the velocity-averaged DM annihilation cross section. Points are colour-coded with a Gaussian likelihood based on the Planck 2015 analysis, the LUX 2013 likelihood from DDCalc, the IceCube 79-string likelihood from nulike, or the Fermi dwarf spheroidal likelihood from gamLike respectively. The darkest blue points correspond to likelihoods below the smallest value shown in the colour bars. For comparison, we also plot the limit from the corresponding analyses done by the experimental collaborations. For the relic density, we show the Planck best-fit value for $\Omega h^2$ and its $2\sigma$ (experimental) uncertainty.

In the main part of the code, different options are available that illustrate how to calculate annihilation yields for various final states and DM masses. Furthermore, the standalone example has the ability to calculate tables of likelihoods for Fermi-LAT/HESS/CTA indirect detection and direct detection experiments as functions of the dark matter mass and the annihilation or scattering cross section; we show the corresponding upper limits in Figs. 7, 8 and 9. These are 95% CL upper limits (obtained at $\Delta 2 \ln \mathcal{L} = 2.71$) in Fig. 7, as is customary for indirect searches, and at 90% CL in Figs. 8 and 9, as is typical in direct detection. For the direct detection limits, we determine the 90% CL value for $\Delta 2 \ln \mathcal{L}$ from the Poisson upper limit on the expected number of events. For low statistics this value can be substantially larger than $\Delta 2 \ln \mathcal{L} = 1.64$, the value obtained in the asymptotic limit. The agreement between the official limits and those calculated with the standalone show that known results can be easily reproduced. The standalone example can also be used to calculate similar tables for the relic density. With this output, in Fig. 7 we indicate the cross-section for which the relic density reaches $\Omega h^2 = 0.1188$, the preferred value from Planck [1].

7.3 Comparing DarkSUSY and micrOMEGAs

DarkBit offers the unique possibility to easily compare different numerical codes for the computation of DM properties in a well-defined and consistent way. For illustration, here we focus on DarkSUSY [15] and micrOMEGAs [167]. We
stress, however, that it is straightforward for users to perform similar comparisons for essentially any other numerical code, simply by adding it as a backend to GAMBIT.

The ability of DarkBit to facilitate these comparisons for the MSSM is demonstrated in the example program DarkBit_standalone_MSSM. This program takes an SLHA file (including a decay block, if present) as input and calculates the relic density and DM-nucleon scattering cross sections using both DarkSUSY and micrOMEGAs. Analogously to DarkBit_standalone_WIMP, it also calculates

![Fig. 7](image_url) Simple WIMP results obtained with DarkBit. The green and red solid lines represent 95% CL upper limits on $bb$ and $τ+τ−$ final states from Fermi pass 8 observations of dwarf spheroidal galaxies as calculated by DarkBit, while the dashed lines represent the corresponding limits reported by the Fermi collaboration [136]. The magenta and yellow solid lines show the HESS and projected CTA GC limits, and the dashed lines the published results from Refs. [137,140] (differences are due to differences in the adopted photon yields). The blue solid line shows the 99.7% CL contour for our Fermi GC likelihood. Finally, the solid black line represents the values of $σv$ that reproduce (for $s$-wave annihilation) a DM density of $Ω/b^2 = 0.1188$

![Fig. 8](image_url) Limits on the spin independent DM-nucleon scattering cross section from SuperCDMS, LUX, and PandaX at 90% CL. The solid curves are the limits determined using DarkBit and the dashed curves are the official limits from the collaborations [73,75,76,78].

![Fig. 9](image_url) Limits on the spin dependent DM-proton scattering cross section from PICO-2L, PICO-60L, and PandaX at 90% CL. The solid curves are the limits determined using DarkBit and the dashed curve are the official limits from the collaborations [79,80,166].

likelyhoods for the relic density, direct detection experiments, and indirect searches in neutrinos and gamma rays. The standalone shows how it is possible to change the source of the theoretical inputs for these likelihood calculations (such as the DM-nucleon coupling in the case of direct detection) by just changing a single line of code.

As a demonstration of the sorts of comparisons possible, we have chosen some benchmark MSSM model points that can cause difficulties in the calculation of the relic density due to coannihilations or resonances. Details of the points are given in Table 11. We generated SLHA files for each of these model points (including decay blocks) using SpecBit and the standalone example 3-BIT-HIT [13], which we then fed into DarkBit_standalone_MSSM. These SLHA files can be found in the DarkBit/data/benchmarks/ directory of the GAMBIT distribution. In GAMBIT 1.1.0, which was used to find the results presented here, DarkBit_standalone_MSSM by default calculates the relic density with the YAML option fast set to set to 0 (corresponding to a more accurate calculation) in both RD_oh2_MicrOmegas and RD_oh2_DarkSUSY. The options loop and box are set to true in DD_couplings_DarkSUSY and DD_couplings_MicrOmegas respectively, so that all available loop corrections are used in each backend.

Results of the calculations can be seen in Table 12. While the values of $Ωb^2$ from the two backends agree well for some of the benchmarks, in others there are significant differences. In particular for benchmarks 1-4, where there is resonant annihilation via the $A^0$ or $h$, the relic density is substantially higher when the calculation is done using DarkSUSY. This can be traced to the fact that the micrOMEGAs [$σv eff$ at

---

18 Here we use the MSSM7; see Ref. [10] for the definition of this model, and Ref. [164] for a thorough treatment of its phenomenology and present status.
Table 11  MSSM7 points used as benchmarks for comparisons between DarkSUSY and micrOMEGAs (see Ref. [10] for the definition of the model). The sign of $\mu$ is positive for all points and the parameters are defined at an energy scale of 1 TeV. As shown in the description column, the points were chosen to have different types of processes contribute to the relic density calculation.

| Model | Description | $M_2$ (GeV) | $m_{H_d}$ (10^6 GeV^2) | $m_{H_u}$ (10^6 GeV^2) | $m_f$ (10^6 GeV^2) | $\tan\beta$ | $m_{\tilde{g}}$ (10^6 GeV^2) | $A_d$ (GeV) | $A_u$ (GeV) |
|-------|-------------|-------------|----------------|----------------|----------------|-------------|----------------|-------------|-------------|
| 1     | Resonant annihilation via $A^0$, gaugino-like neutralino | 3442 | -10.86 | 10.07 | 17.25 | 71.45 | 9588 | -5886 |
| 2     | Resonant annihilation via $A^0$, mixed neutralino | 2224 | -0.007416 | -9.361 | 42.63 | 87.23 | 3019 | -3716 |
| 3     | Resonant annihilation via $A^0$, Higgsino-like neutralino | 3283 | 6.904 | -8.602 | 39.22 | 73.11 | -5453 | -2963 |
| 4     | Resonant annihilation via $h$ | -659.0 | 27.52 | -0.4085 | 21.68 | 4.309 | 9870 | 4565 |
| 5     | $\tilde{t}$ coannihilations | -681.8 | 94.43 | -1.667 | 9.798 | 0.4103 | 69.60 | -1471 |
| 6     | $\tilde{t}$ coannihilations, gaugino-like neutralino | 2631 | 4.369 | -4.448 | 7.760 | 11.52 | 9993 | -5103 |
| 7     | $\tilde{t}$ coannihilations, mixed neutralino | 2323 | 4.169 | -2.222 | 9.283 | 8.899 | 9617 | -4472 |
| 8     | $\tilde{t}$ coannihilations, Higgsino-like neutralino | 2316 | 4.164 | -2.072 | 11.44 | 8.560 | 229.2 | -3976 |
| 9     | Chargino coannihilations | 1582 | 8.029 | -2.938 | 45.01 | 42.07 | -125.5 | -768.3 |

Table 12  The dark matter relic density and proton-scattering cross-sections, both spin-independent and spin-dependent, for a range of MSSM model points. The model points are defined in Table 11. All quantities were calculated with DarkBit_standalone_MSSM using both the DarkSUSY and micrOMEGAs backends.

| Model | $\Omega h^2$ | $\sigma_{SI, p}$ [10^{-46}cm^2] | $\sigma_{SD, p}$ [10^{-43}cm^2] |
|-------|--------------|---------------------------------|---------------------------------|
|   |     | DarkSUSY | micrOMEGAs | DarkSUSY | micrOMEGAs | DarkSUSY | micrOMEGAs |
| 1   | 0.1545 | 0.09471 | 9.817 | 10.88 | 1.332 | 1.263 |
| 2   | 0.01617 | 0.008546 | 166.5 | 186.6 | 61.14 | 58.01 |
| 3   | 0.05888 | 0.03355 | 189.5 | 211.1 | 32.88 | 31.21 |
| 4   | 0.002318 | 0.001480 | 25.50 | 26.28 | 5272. | 5002. |
| 5   | 0.1110 | 0.1094 | 10.45 | 7.011 | 0.06781 | 0.06170 |
| 6   | 0.02290 | 0.02410 | 3.367 | 3.745 | 0.7975 | 0.8305 |
| 7   | 0.004982 | 0.003623 | 198.0 | 218.7 | 41.34 | 39.00 |
| 8   | 0.006317 | 0.004722 | 250.0 | 273.9 | 55.69 | 52.61 |
| 9   | 0.003008 | 0.003131 | 8.749 | 8.869 | 192.4 | 182.6 |

temperatures around freeze-out for these points is consistently larger than the DarkSUSY result. The fractional differences are largest on the resonance; adjusting $m_{A_0}$ or $m_h$ away from $2m_{\chi_0}$ leads to much better agreement between the two codes. The ultimate source of this discrepancy should be tracked down, and to this end we are planning a follow-up study using the GAMBIT framework in which we investigate the reasons behind the differences and look into the effects of adding loop corrections currently not included in both backends.

For some of the benchmarks, there are also significant differences between the nuclear scattering cross-sections computed with DarkSUSY and micrOMEGAs, specifically in cases where the cross-sections are small. The discrepancies between the two codes are almost certainly linked to the fact that at tree level, small nuclear scattering cross-sections in the MSSM arise due to cancellations between different diagrams.
The cancellations can easily be spoilt by small changes in the input parameters, or equivalently, different choices of spectrum generator and/or treatments of running parameters in the calculation of the matrix elements, as well as different treatments of loop corrections in the calculations of scattering amplitudes. As in the case of the relic density, we are planning a future study to more precisely understand the source of these differences.

8 Outlook

As detailed in the preceding sections, DarkBit is equipped with sophisticated tools for calculating observables and likelihoods for the DM relic density, direct detection experiments and indirect searches with neutrinos and gamma rays. Each of these cases demonstrates the modularity of the code, and the ease with which external codes can interfaced with DarkBit. This modularity also implies that extensions of DarkBit in all possible directions are straight-forward to implement, and do not in general require the expertise of GAMBIT Collaboration members or highly experienced external users of the code. The focus of future developments will thus be steered largely by the needs (and indeed, contributions) of the community. Nevertheless, here we list a few obvious code extensions that we expect to include in future releases (aside from obvious additions of new experimental likelihoods to existing components like gamLike, DDCalc and nullike).

The combination of the process catalogue and the DarkSUSY Boltzmann solver currently allows us to calculate the relic density for simple arbitrary DM models. We intend to expand this framework so that coannihilations can be included in the process catalogue and the relic density can be calculated in the case of semi-annihilating [168] and asymmetric DM [169]. This would complement the existing capabilities of micrOMEGAs to handle these scenarios. We also plan to backend MadDM [170,171] in a future version of DarkBit, which with its interface to MadGraph [172] will be a useful alternative to micrOMEGAs for quickly implementing new DM models. To enhance the accuracy of our relic density calculations, we also intend to add the effects of Sommerfeld enhancement on the relevant (co)annihilation cross sections [173,174]. Moving beyond the standard relic density calculation, we have plans to include the ability to deal with non-standard cosmological expansion histories. This capability is available in SuperIso Relic [175], to which we will provide a frontend. A final natural extension in this area is to calculate kinetic freeze-out of DM from the thermal bath rather than only chemical freeze-out as is done now. This would lead to an additional observable: a cutoff in the power spectrum of matter density perturbations [176,177].

For direct detection experiments, the implementation of velocity- and momentum-dependent cross sections in DDCalc will be a high priority extension, allowing one to systematically study the full set of available non-relativistic operators [178], for example. Helio- and astroseismological probes of DM-nucleon couplings (see e.g. [123]) are another expected extension.

The most important extension relevant for indirect DM searches, given the high precision expected from the AMS-02 experiment on board the international space station, is charged cosmic rays. Indeed, positron data already put one of the most stringent limits on leptophilic DM models [106], and constraints from antiprotons can likely be improved considerably [103,104]. Another extension that we aim for in the near future is to fully allow for velocity-dependent annihilation cross sections, such as in the case of resonances or the Sommerfeld effect [179,180]. These can be relevant for e.g. DM annihilation in subhalos [181], or close to the black hole at the Galactic centre [182,183].

9 Conclusions

The particle nature of DM is one of the most perplexing puzzles in present-day particle physics and cosmology. Despite decades of research, only non-gravitational signals of DM have been identified so far. However, rapid experimental developments in recent years have provided a wealth of new data that can be exploited in the search for DM signals, and used to constrain DM models. In order to facilitate a systematic study of a large number of DM scenarios, in this paper we presented DarkBit, a new numerical tool for DM calculations.

DarkBit is designed to allow DM observables to be included in global scanning tools like GAMBIT. It can also be used as standalone module. The first release of DarkBit ships with a large number of likelihood functions for various experiments. These are implemented more accurately than what is usually done in the literature. The overarching design goals are reusability, self-consistency and modularity, which are achieved in a number of ways. First, by allowing seamless integration of existing numerical tools like DarkSUSY and micrOMEGAs, using the GAMBIT method of abstracting backend function-handling for cross-language coding environments. Second, by providing internal structures for particle and astrophysical DM properties that are consistently used in all calculations, and passed to external codes if necessary. Third, by splitting up calculations into their most elemental building blocks wherever possible.

The modular implementation of the DM relic density calculations in DarkBit allows it to solve the Boltzmann equation independently of the actual particle model chosen for DM (Fig. 2). Alternatively, the user can directly call relic density routines provided by backend codes for specific models,
allowing, for example, a systematic comparison between the results of DarkSUSY and micrOMEGAs.

The new backend code DDCalc provides a general solution to the problem of testing DM models against direct detection data, including detailed likelihoods for many of the most important experiments. This allows both spin-dependent and spin-independent signals of the same model to be calculated and combined self-consistently across the full range of relevant experiments. For liquid noble gas detectors, the sensitivities in DDCalc are based on the output of TPCMC [84], a dedicated detector Monte Carlo simulation.

We have implemented likelihood functions for gamma-ray indirect DM searches in the new backend gamLike. We included Fermi-LAT and HESS observations of dwarf spheroidal galaxies and the Galactic centre, as well as projections for the future with CTA. The likelihood functions in gamLike are pre-tabulated for fast evaluation, but based on event-level (mock) data where possible. DarkBit also includes a new Monte Carlo code for the fast simulation of cascade annihilation spectra, and an interface to the event-level neutrino telescope likelihood tool nullike for calculating neutrino indirect detection likelihoods.

The first release of DarkBit ships with the essentials of DM indirect searches. Extensions planned for the near future include charged cosmic rays, accurate treatment of velocity-dependent annihilation cross-sections and Sommerfeld enhancement, and inclusion of new experimental analyses in gamLike. In direct detection, we plan to implement velocity- and momentum-dependent cross-sections in DDCalc, as well as new experimental results as they become available. Furthermore, new classes of likelihoods, like helio/astroseismological probes for DM and limits from radio and CMB observations, will be included in future releases.

Acknowledgements We wish to thank Lauren Hsu for contributing to the DDCalc treatment of SuperCDMS, and Ankit Beniwal and Andre Scaffidi for helpful conversations regarding DDCalc. We warmly thank the Casa Matemáticas Oaxaca, affiliated with the Banff International Research Station, for hospitality whilst part of this work was completed, and the staff at Cyfronet, for their always helpful supercomputing support. GAMBIT has been supported by STFC (UK; ST/K00414X/1, ST/P000762/1), the Royal Society (UK; UF110191), Glasgow University (UK; Leadership Fellowship), the Research Council of Norway (FRIPRO 230546/F20), NOTUR (Norway; NN9284K), the Knut and Alice Wallenberg Foundation (Sweden; Wallenberg Academy Fellowship), the Swedish National Science Center (Sonata UMO-2015/17/D/ST2/03532), the Swiss National Science Foundation (PP0OP2-144674), European Commission Horizon 2020 (Marie Skłodowska-Curie actions H2020-MSCA-RISE-2015-691164, European Research Council Starting Grant ERC-2014-STG-638528), the ERA-CAN+ Twinning Program (EU & Canada), the Netherlands Organisation for Scientific Research (NWO-Vidi 016.149.331), the National Science Foundation (USA; DGE-1339067), the FRQNT (Québec) and NSERC/The Canadian Tri-Agencies Research Councils (BPDF-424460-2012).

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made. Funded by SCOAP³.

Appendix A: Getting started

As described in Sect. 2, DarkBit is a standalone and complimentary module of the GAMBIT software, which can be downloaded from the official GAMBIT website. In the following, we describe the content of the DarkBit standalone download, the installation of the DarkBit software as standalone or GAMBIT module, and the running of the example program.

Appendix A.1: Content of DarkBit download and installation

Each GAMBIT module contains the

- Backends (utility functions used for backend interfaces)
- Models (predefined BSM models and utility functions used for model definitions)
- Logs (general GAMBIT logging system);
- Utils (GAMBIT utility functions);
- Elements (general GAMBIT macro and type definitions)

folders in addition to the specific module folder (here the DarkBit folder). A detailed description of the GAMBIT functionalities can be found in the GAMBIT main paper [10]. Each standalone module requires all of these folders to work. If the DarkBit module is used in conjunction with other GAMBIT modules, only the DarkBit folder is needed and should be placed into the main folder containing the other GAMBIT modules.

GAMBIT uses the open-source cross-platform build system CMake. CMake supports in-source and out-of-source builds, but we recommend the latter to keep the source directory unchanged and enable multiple builds. To do such a build, run the following commands in the directory that contains the GAMBIT module folders:

19 gambit.hepforge.org.
20 www.cmake.org.
For further details we refer to the GAMBIT main paper [10].

The DarkBit standalones can be found in Darkbit/examples, and can be built with

```
make DarkBit_standalone_X
```

where X can be MSSM, WIMP or SingletDM.

Appendix A.2: Running the example program

To demonstrate how DarkBit can be used in a fully-fledged scan, we provide 2 annotated examples of a DarkBit yaml file: yaml_files/DarkBit_SingletDM.yaml and yaml_files/DarkBit_MSSM7.yaml. These examples show how to specify a model, prior ranges over which to sample its parameters, a scanner and an output device (‘printer’), then run the relic density, gamma-ray, neutrino and direct search likelihoods. The user can also select the parameters of the halo model and the nuclear parameters relevant for direct detection. The examples require the micrOMEGAs, gamlike, DDCalc, DarkSUSY, nulike, FeynHiggs and SUSY-HIT backends to be present, which can be accomplished by running the following commands in the GAMBIT build directory

```
mkdir build
cd build
cmake ..
make
```

The yaml file is complete, i.e. all options of all module functions available in DarkBit are mentioned and documented there.

Appendix B: Handling Fortran/C/C++ functions with daFunk

Appendix B.1: Design goals and philosophy

One of the major technical challenges when combining a large number of different codes but trying to maintain maximum portability is wrapping and manipulating Fortran, plain C and C++ object member functions in a systematic and coherent way. In DarkBit, quite commonly functions of the type $\mathbb{R}^n \to \mathbb{R}$ (which describe e.g. annihilation yields, dark matter profiles, velocity distributions, differential cross sections, or effective annihilation rates) need to be wrapped in a generic structure so they can be shared amongst different backends.

Typically, the results of these functions need to be manipulated before they can be used, in order to comply with conventions and requirements in the subsequent codes. Sometimes this means using basic arithmetic operations, sometimes passing them through complex trigonometric functions or performing variable substitution. Further common operations are partial integrations, sometimes with non-constant boundaries and singularity handling, or checks of parameter domains. Often, these manipulated functions need to be wrapped back into plain C functions in order to be able to pass them back into the backend codes (like e.g. the DarkSUSY Boltzmann solver).

In order to facilitate all these operations, we present the new lightweight C++ header-only library daFunk (dynamisch allokierbare Funktionen). Despite the complex function handling that it allows, the daFunk API is relatively simple. This is achieved with recursive variadic templates, polymorphism and shared pointers. The most relevant features are:

- Interpolation in linear- and log-space
- Multidimensional integration with complex boundaries
- Handling of (1-dimensional) singularities
- Parameter substitution and chaining of functions
- Wrapping of Fortran functions, plain C/C++ functions and C++ object member functions
- Reverse wrapping of daFunk::Funk objects into Fortran and plain C/C++ functions
- Overloading of all basic arithmetic and trigonometric functions for easy manipulation and combination of daFunk::Funk objects
- Flexible function handling based on shared pointers
- Checks of parameter domains
- Basic if...else constructions
- OpenMP-enabled calculations

Appendix B.2: Selected examples

The atomic object in daFunk is daFunk::Funk, which is a shared pointer to an instance of the daFunk::FunkBase class. Importantly, the daFunk::FunkBase class is an abstract base class: it leaves the virtual member function responsible for all actual calculations, `virtual double value(...)`, undefined. The main purpose of daFunk::Funk is to provide a unified interface and a flexible C++ type, independent of whatever calculation it actually performs. The actual computations are implemented in classes derived from daFunk::FunkBase.

Each daFunk::Funk object provides a list of names of variables on which it depends. This list is simply a list of std::string tags. The specific content of this list depends
on the implementation of value. The most basic implementations are variables and constants, shown in the following simple example:

```cpp
// Variable substitution
daFunk::Funk g = f->set("y", 0.0001) -> set("m", Ecm/2);

// Wrap daFunk in plain function
double (*h)(double&, double&) = g->plain<T>(g, "Ecm", "E");
```

Here, dNdE is a plain C or Fortran function (e.g. the annihilation yield as function of final state energy \( E \), initial state relative velocity \( v \), and for dark matter mass \( m \)), which is wrapped into a daFunk object \( f \). The function \( g \) is derived from \( f \) by fixing \( v = 10^{-4} \), and substituting the dark matter mass by the centre-of-mass frame energy \( m = E_{\text{cm}}/2 \). The function \( g \) is then wrapped back into a plain C function \( h \) that can be evaluated as usual, or passed back into some of the backend codes (note that the bind step happens here behind the scenes when calling plain<T>). Note that the pointer to the daFunk::Funk object that is wrapped in \( h \) is stored in the traits class \( T \). Furthermore, the generated C function takes arguments by reference, which is an implicit convention in Fortran. This makes it possible to pass \( h \) directly back to a Fortran backend.

Lastly, we give an example for 1-dimensional integration with non-trivial boundaries.

```cpp
daFunk::Funk x = daFunk::var("x"); // variable x
daFunk::Funk y = daFunk::var("y"); // variable y
daFunk::Funk c = daFunk::cnst(2.); // constant 2
daFunk::Funk f = c*x+3*cos(y);
```

Note that in cases where the integration fails, a warning message is printed to stderr, and zero is returned. For more examples we refer the reader the DarkBit code.

### Appendix C: Glossary

- **backend** An external code containing useful functions (or variables) that one might wish to call (or read/write) from a module function.
backend function A function contained in a backend. It calculates a specific quantity indicated by its capability. Its capability and call signature are defined in the backend’s frontend header.

backend requirement A declaration that a given module function needs to be able to call a backend function or use a backend variable, identified according to its capability and type(s). Backend requirements are declared in module functions’ entries in rollcall headers.

backend variable A global variable contained in a backend. It corresponds to a specific quantity indicated by its capability. Its capability and type are defined in the backend’s frontend header.

capability A name describing the actual quantity that is calculated by a module or backend function. This is one possible place for units to be noted; the other is in the documented description of the capability (see Sec. 10.7 of Ref. [10]).

dependency A declaration that a given module function needs to be able to access the result of another module function, identified according to its capability and type. Dependencies are declared in module functions’ entries in rollcall headers.

dependency resolution The process by which GAMBIT determines the module functions, backend functions and backend variables needed and allowed for a given scan, connects them to each others’ dependencies and backend requirements, and determines the order in which they must be called.

dependency tree A result of dependency resolution; a directed acyclic graph of module functions connected by resolved dependencies. See Fig. 5 of Ref. [10] for an example.

frontend The interface between GAMBIT and a given backend, consisting of a frontend header plus optional source files and type headers.

frontend header The C++ header in which the frontend to a given backend is declared.

module A subset of GAMBIT functions following a common theme, able to be compiled into a standalone library. Although module often gets used as shorthand for physics module, this term technically also includes the GAMBIT scanning module ScannerBit.

module function A function contained in a physics module. It calculates a specific quantity indicated by its capability and type, as declared in the module’s rollcall header. It takes only one argument, by reference (the quantity to be calculated), and has a void return type.

physics module Any module other than ScannerBit, containing a collection of module functions following a common physics theme.

rollcall header The C++ header in which a given physics module and its module functions are declared.

type A general fundamental or derived C++ type, often referring to the type of the capability of a module function.

Appendix D: Capability overview

For reference, we provide a complete list of the DarkBit capabilities, dependencies and options. These include the complete process and coupling capabilities (Table 13), some simple informative capabilities (Table 14), capabilities related to DM halo properties (Table 15), relic density capabilities (Tables 16 and 17), direct detection capabilities (Table 18), gamma-ray yield capabilities (Table 19), gamma-ray likelihoods (Table 20), neutrino capabilities (Tables 21 and 22), cascade decay capabilities (Tables 23 and 24), and various miscellaneous capabilities (Table 25).
Table 13  Central DarkBit capabilities that store details about annihilation and scattering processes. The starred (*) options are only available in GAMBIT 1.1.0

| Capability                      | Function (return type): brief description                                      | Dependencies          | Backend requirements | Options (Type)               |
|---------------------------------|---------------------------------------------------------------------------------|-----------------------|----------------------|-----------------------------|
| **TH_ProcessCatalog**           | **TH_ProcessCatalog_MSSM** (DarkBit::TH_ProcessCatalog): Generates process catalogue for the MSSM, based on the DarkSUSY backend | DarkSUSY_PointInit    | setMassesForIB       | ignore_three_body           |
|                                 |                                                                                 | MSSM_spectrum         | dsigmv               | (bool)                      |
|                                 |                                                                                 | decay_rates           | dsIBffdxddy          | ProcessCatalog_             |
|                                 |                                                                                 | DarkMatter_ID         | dsIBhdxddy           | MinBranching(double)        |
|                                 |                                                                                 |                       | dsIBwmdxddy          |                             |
|                                 |                                                                                 |                       | IBintvars            |                             |
| **TH_ProcessCatalog_SingletDM** | **TH_ProcessCatalog_SingletDM** (DarkBit::TH_ProcessCatalog): Generates process catalogue for scalar singlet dark matter | SingletDM_spectrum   | decay_rates          | ProcessCatalog_             |
|                                 |                                                                                 |                       |                      | MinBranching(double)        |
| **DD_couplings**                | **DD_couplings_DarkSUSY** (DM_nucleon_couplings): Determine the WIMP mass and couplings using DarkSUSY | DarkSUSY_PointInit    | dsdddpgtn            | rescale_couplings(double)   |
|                                 |                                                                                 | mspctm                | ddcom                | loop*(bool)                 |
|                                 |                                                                                 |                       |                      | pole*(bool)                 |
|                                 |                                                                                 | nucleonAmplitudes     | FeScLoop             | box*(bool)                  |
|                                 |                                                                                 | MOcommon              |                      |                             |
| **DD_couplings_MicrOmegas**     | **DD_couplings_MicrOmegas** (DM_nucleon_couplings): Determine the WIMP mass and couplings using micrOMEGAs |                       |                      |                             |
|                                 |                                                                                 |                       |                      |                             |
| **DD_couplings_SingletDM**      | **DD_couplings_SingletDM** (DM_nucleon_couplings): Determine the WIMP mass and couplings for scalar singlet DM | SingletDM_spectrum    |                      |                             |
Table 14  DarkBit capabilities for WIMP-nucleon couplings ($N = p, n$ refers to the relevant nucleon), annihilation cross-section, dark matter mass and dark matter particle ID.

| Capability        | Function (return type): brief description                                                                 | Dependencies                  |
|-------------------|-----------------------------------------------------------------------------------------------------------|-------------------------------|
| mwimp             | mwimp.generic(double): Retrieve the DM mass in GeV for generic models                                     | TH_ProcessCatalog             |
|                   |                                                                                                           | DarkMatter_ID                 |
| sigmav            | sigmav晚期宇宙(double): Retrieve the total thermally-averaged annihilation cross-section for indirect detection ($cm^3 s^{-1}$) at $v = 0$ | TH_ProcessCatalog             |
|                   |                                                                                                           | DarkMatter_ID                 |
| sigma_SI_N        | sigma_SI_N.simple(double): Simple calculator of the spin-independent WIMP-proton or WIMP-neutron cross-section | DD_couplings                  |
|                   |                                                                                                           | mwimp                         |
| sigma_SD_N        | sigma_SD_N.simple(double): Simple calculator of the spin-dependent WIMP-proton or WIMP-neutron cross-section | DD_couplings                  |
|                   |                                                                                                           | mwimp                         |
| DarkMatter_ID     | DarkMatter_ID_SingletDM(std::string): Returns string ID for dark matter particle                           |                               |
|                   | DarkMatter_ID_MSSM30atQ(std::string): Returns string ID for dark matter particle                           |                               |

Table 15  Capabilities connected to the Milky Way halo parameters

| Capability        | Function (return type): brief description                                                                 | Dependencies                  | Options (type) |
|-------------------|-----------------------------------------------------------------------------------------------------------|-------------------------------|---------------|
| GalacticHalo      | GalacticHalo_gNFW(GalacticHaloProperties): Provides the generalised NFW density profile $\rho(r)$ and $r_\odot$ |                               |               |
|                   | GalacticHalo_Einasto(GalacticHaloProperties): Provides the Einasto density profile $\rho(r)$ and $r_\odot$ |                               |               |
| LocalHalo         | ExtractLocalMaxwellianHalo(LocalMaxwellianHalo): Provides the local density $\rho_0$ as well as the velocity parameters $v_0$, $v_{rot}$ and $v_{esc}$ |                               |               |
| lnL_rho0          | lnL_rho0_lognormal(double): Log of the log-normal likelihood for the local DM density                      | LocalHalo                    | rho0_obs (double) |
|                   |                                                                                                           |                               | rho0_obserr(double) |
| lnL_vrot          | lnL_vrot_gaussian(double): Log of the Gaussian likelihood for the local disk rotation speed                | LocalHalo                    | vrot_obs (double) |
|                   |                                                                                                           |                               | vrot_obserr (double) |
| lnL_v0            | lnL_v0_gaussian(double): Log of the Gaussian likelihood for the most-probable DM speed                   | LocalHalo                    | v0_obs (double)  |
|                   |                                                                                                           |                               | v0_obserr (double) |
| lnL_vesc          | lnL_vesc_gaussian(double): Log of the Gaussian likelihood for the escape velocity                        | LocalHalo                    | vesc_obs (double) |
|                   |                                                                                                           |                               | vesc_obserr (double) |
### Table 16 General relic density capabilities provided by DarkBit

| Capability               | Function (return type): brief description                                                                 | Dependencies                      | Backend requirements                   | Options (type)                      |
|--------------------------|-------------------------------------------------------------------------------------------------------------|------------------------------------|----------------------------------------|------------------------------------|
| RD_spectrum              | RD_spectrum_SUSY (DarkBit::RD_spectrum_type): Returns masses and d.o.f. for all (co-)annihilating particles, plus location of thresholds and resonances. Information retrieved from DarkSUSY | DarkSUSY_PointInit mspctm widths intdof pacades particle_code |                                        | CoannCharginos                      |
|                         | RD_spectrum_from_ProcessCatalog (DarkBit::RD_spectrum_type): Returns mass and d.o.f. of DM particles, plus location of thresholds and resonances. Information retrieved from Process catalogue | TH_ProcessCatalog DarkMatter_ID     |                                        | Neutralinos(bool)                   |
|                         | RD_spectrum_ordered RD_spectrum_ordered_func (DarkBit::RD_spectrum_type): Adds co-annihilation thresholds to the output from RD_spectrum, and orders all thresholds and resonances by energy | RD_spectrum                       |                                        | CoannSfermions(bool)                |
|                         | RD_eff_annrate_DSprep RD_annrate_DSprep_func(int): Initializes DarkSUSY to be able to provide effective invariant rate $\dot{W}_g$ | RD_spectrum rdmgev                |                                        | CoannMaxMass(double)                |
|                         | RD_eff_annrate_SUSY (fptr_dd): Returns the effective invariant rate $\dot{W}_g$ as provided by DarkSUSY   | RD_eff_annrate_DSprep dsanwx       |                                        |                                    |
|                         | RD_eff_annrate_from_ProcessCatalog (fptr_dd): Returns the effective invariant rate $\dot{W}_g$ as calculated from the information contained in the process catalogue | TH_ProcessCatalog DarkMatter_ID     |                                        |                                    |
Table 17 The main relic density capabilities provided by DarkBit

| Capability         | Function (return type): brief description                                                                 | Dependencies | Backend requirements | Options (type) |
|--------------------|----------------------------------------------------------------------------------------------------------|--------------|----------------------|-----------------|
| RD_oh2             | RD_oh2_general (double): The general dark matter relic density                                           | RD_spectrum_ordered | dsrdtlm dsrdtab dsrdeqn dsrdint particle_code widths rdmg ev rdpth rdpars rdswitch rdlun rdpadd rddof rddof          |
|                    |                                                                                                         | RD_eff_annrate | fast (int)           |                 |
| RD_oh2_DarkSUSY    | RD_oh2_DarkSUSY (double): Routine for directly obtaining results from DarkSUSY                          | DarkSUSY_PointInit | dsrdomega           | omtype (int)    |
|                    |                                                                                                         |              | fast (int)           |                 |
| RD_oh2_MicrOmegas  | RD_oh2_MicrOmegas (double): Routine for directly obtaining results from micrOMEGAs                     | oh2          |                      | fast (int)      |
|                    |                                                                                                         |              | Beps (double)        |                 |
| RD_fraction        | RD_fraction_from_oh2 (double): The relic density expressed as a fraction of the critical density         | RD_oh2       | oh2_obs (double)     | mode (std::string) |
| lnL_oh2            | lnL_oh2_Simple (double): Gaussian log-likelihood (see Secs. 8.3.1 and 8.3.2 of [10]) for the relic density | RD_oh2       | oh2_obs (double)     | profile_systematics (bool) |
|                    |                                                                                                         |              |                      |                 |
| lnL_oh2_upperlimit | lnL_oh2_upperlimit (double): A half-Gaussian log-likelihood (see Secs. 8.3.3 and 8.3.4 of [10]) for the relic density | RD_oh2       | oh2_obs (double)     | profile_systematics (bool) |
|                    |                                                                                                         |              |                      |                 |

Table 18 DarkBit capabilities for direct detection log-likelihoods and related observables. Possible values for $X$ are XENON100_2012, LUX_2013, LUX_2015, LUX_2016, PandaX_2016, SuperCDMS_2014, SIMPLE_2014, PICO_2L, PICO_60_P and PICO_60_I in version 1.0.0 and in addition XENON1T_2017 and PICO_60_2017 in version 1.1.0.

| Capability     | Function (return type): brief description | Dependencies | Backend requirements | Options (type) |
|----------------|--------------------------------------------|---------------|----------------------|----------------|
| $X_{\text{Calculate}}$ | $X_{\text{Calculate}}(\text{bool})$: Perform rate calculations for direct detection analysis $X$ | $X_{\text{Calculate}}$ | DD_CalcRates | |
| $X_{\text{Events}}$ | $X_{\text{Events}}_{\text{DDCalc}}(\text{int})$: The number of observed events for direct detection analysis $X$ | $X_{\text{Calculate}}$ | DD_Events | |
| $X_{\text{Background}}$ | $X_{\text{Background}}_{\text{DDCalc}}(\text{double})$: The number of background events for direct detection analysis $X$ | $X_{\text{Calculate}}$ | DD_Background | |
| $X_{\text{Signal}}$ | $X_{\text{Signal}}_{\text{DDCalc}}(\text{double})$: The number of signal events for direct detection analysis $X$ | $X_{\text{Calculate}}$ | DD_Signal | |
| $X_{\text{SignalSI}}$ | $X_{\text{SignalSI}}_{\text{DDCalc}}(\text{double})$: The number of spin-independent signal events for direct detection analysis $X$ | $X_{\text{Calculate}}$ | DD_SignalSI | |
| $X_{\text{SignalSD}}$ | $X_{\text{SignalSD}}_{\text{DDCalc}}(\text{double})$: The number of spin-dependent signal events for direct detection analysis $X$ | $X_{\text{Calculate}}$ | DD_SignalSD | |
| $X_{\text{LogLikelihood}}$ | $X_{\text{LogLikelihood}}_{\text{DDCalc}}(\text{double})$: Calculate the log-likelihood for direct detection analysis $X$ | $X_{\text{Calculate}}$ | DD_LogLikelihood | |
| $\ln L_{\text{SI nuclear parameters}}$ | $\ln L_{\text{sigmas signal}}(\text{double})$: The log-likelihood for the nuclear parameters relevant for spin-independent scattering | | | sigmas_obs (double) |
| $\ln L_{\text{SD nuclear parameters}}$ | $\ln L_{\text{deltaq}}(\text{double})$: The log-likelihood for the nuclear parameters relevant for spin-dependent scattering | | | a3_obs (double) a8_obs (double) deltas_obs (double) |

$\ln L_{\text{SI nuclear parameters}}$: lnL_SI_nuclear_parameters

$\ln L_{\text{SD nuclear parameters}}$: lnL_SD_nuclear_parameters
| Capability         | Function (return type): brief description                                                                 | Dependencies | Backend requirements              | Options (type)                           |
|-------------------|------------------------------------------------------------------------------------------------------------|--------------|-----------------------------------|------------------------------------------|
| SimYieldTable     | SimYieldTable_DarkSUSY (DarkBit::SimYieldTable)                                                           | dshayield    | allow_yield_extrapolation(bool)  |
|                   | Provides access to tabulated yields in DarkSUSY                                                           |              |                                   |                                          |
| SimYieldTable     | SimYieldTable_MicrOmegas (DarkBit::SimYieldTable)                                                         | dNDE         | allow_yield_extrapolation(bool)  |
|                   | Provides access to tabulated yields in MicrOmegas                                                         |              |                                   |                                          |
| GA_missingFinalStates | GA_missingFinalStates (std::vector<std::string>)                                                          | TH_ProcessCatalog | ignore_all (bool)          |
|                   | Determines final states that are not available with tabulated spectra                                      | SimYieldTable | ignore_two_body (bool)           |
|                   |                                                                                                           | DarkMatter_ID | ignore_three_body (bool)         |
| GA_AnnYield       | GA_AnnYield_General (Funk::Funk)                                                                          | TH_ProcessCatalog | line_width (double)               |
|                   | General function for calculating gamma-ray yields from the process catalogue                              | SimYieldTable |                                   |                                          |
|                   |                                                                                                           | DarkMatter_ID |                                   |                                          |
|                   |                                                                                                           | cascadeMC_    |                                   |                                          |
|                   |                                                                                                           | gammaSpectra |                                   |                                          |
| Capability          | Function (return type): brief description                                                                 | Dependencies                | Backend requirements | Options (type) |
|---------------------|-----------------------------------------------------------------------------------------------------------|------------------------------|----------------------|-----------------|
| lnL_FermiLATdwarfs  | lnL_FermiLATdwarfs_gamLike (double): Log-likelihood for the Fermi-LAT dwarf galaxy search, using gamLike    | GA_AnnYield, RD_fraction     | gamLike              | version (str)   |
| lnL_FermiGC         | lnL_FermiGC_gamLike (double): Log-likelihood for the Fermi-LAT GeV excess, using gamLike                 | GA_AnnYield, RD_fraction     | gamLike              | version (str)   |
| lnL_HESSGC          | lnL_HESSGC_gamLike (double): Log-likelihood for the HESS Galactic halo searches, using gamLike            | GA_AnnYield, RD_fraction     | gamLike              | version (str)   |
| lnL_CTA GC          | lnL_CTA GC_gamLike (double): Log-likelihood for projected dark matter searches with CTA, using gamLike   | GA_AnnYield, RD_fraction     | gamLike              | version (str)   |
| set_gamLike_GC_halo | set_gamLike_GC_halo (bool): Initialises the Galactic dark matter distribution in gamLike, based on the halo model used in the corresponding scan | GalacticHalo                 | gamLike              |                 |
| GalacticHalo        | GalacticHalo_gNFW (GalacticHaloProperties): Provides the generalised NFW density profile \( \rho(r) \) and \( r_{\text{sun}} \) |                             |                      |                 |
|                     | GalacticHalo_Einasto (GalacticHaloProperties): Provides the Einasto density profile \( \rho(r) \) and \( r_{\text{sun}} \) |                             |                      |                 |
| Capability               | Function (return type): brief description                                                                 | Dependencies                                           | Backend requirements    |
|--------------------------|----------------------------------------------------------------------------------------------------------------|--------------------------------------------------------|-------------------------|
| capture_rate_Sun         | capture_rate_Sun_const_xsec (double): Capture rate of regular dark matter in the Sun (no $v$-dependent or $q$-dependent cross-sections) ($s^{-1}$) | mwimp                                                   | cap_Sun_v0q0_isoscalar  |
|                          |                                                                                                              | sigma_SI_p                                              |                         |
|                          |                                                                                                              | sigma_SD_p                                              |                         |
| equilibration_time_Sun   | equilibration_time_Sun (double): Equilibration time for capture and annihilation of dark matter in the Sun ($s$) | mwimp                                                   |                         |
|                          |                                                                                                              | sigmav                                                  |                         |
|                          |                                                                                                              | capture_rate_Sun                                        |                         |
| annihilation_rate_Sun    | annihilation_rate_Sun (double): Annihilation rate of dark matter in the Sun ($s^{-1}$)                       | equilibration_time_Sun                                   |                         |
|                          |                                                                                                              | capture_rate_Sun                                        |                         |
| nuyield_ptr              | nuyield_from_DS (double): Neutrino yield function pointer and setup                                         | TH_ProcessCatalog                                       | nuyield_setup           |
|                          |                                                                                                              | mwimp                                                   |                         |
|                          |                                                                                                              | sigmav                                                  |                         |
|                          |                                                                                                              | sigma_SI_p                                              |                         |
|                          |                                                                                                              | sigma_SD_p                                              |                         |
|                          |                                                                                                              | DarkMatter_ID                                           |                         |
|                          |                                                                                                              | nuyield                                                |                         |
|                          |                                                                                                              | get_DS_neutral_h_                                       |                         |
|                          |                                                                                                              | decay_channels                                          |                         |
|                          |                                                                                                              | get_DS_charged_h_                                       |                         |
|                          |                                                                                                              | decay_channels                                          |                         |
Table 22  DarkBit capabilities for neutrino indirect detection likelihoods. Possible values for \( X \) are IC22, IC79WH, IC79WL, and IC79SL

| Capability      | Function (return type): brief description                                                                 | Dependencies                                      | Backend requirements | Options (type)     |
|-----------------|----------------------------------------------------------------------------------------------------------------|---------------------------------------------------|----------------------|--------------------|
| \( X_{\text{data}} \) | \( X_{\text{full}}(\text{nudata}) \): Do signal, likelihood and related calculations for neutrino indirect detection analysis \( X \) | \text{mwp} \text{imp} \text{annihilation\_rate\_Sun} \text{nubarounds} \text{nulike\_speed} (\text{int}) |                      |                    |
| \( X_{\text{signal}} \) | \( X_{\text{signal}}(\text{double}) \): Number of signal events for neutrino indirect detection analysis \( X \) | \( X_{\text{data}} \)                           |                      |                    |
| \( X_{\text{bg}} \) | \( X_{\text{bg}}(\text{double}) \): Number of background events for neutrino indirect detection analysis \( X \) | \( X_{\text{data}} \)                           |                      |                    |
| \( X_{\text{loglike}} \) | \( X_{\text{loglike}}(\text{double}) \): Log-likelihood for neutrino indirect detection analysis \( X \) | \( X_{\text{data}} \)                           |                      |                    |
| \( X_{\text{bgloglike}} \) | \( X_{\text{bgloglike}}(\text{double}) \): Background-only log-likelihood for neutrino indirect detection analysis \( X \) | \( X_{\text{data}} \)                           |                      |                    |
| \( X_{\text{pvalue}} \) | \( X_{\text{pvalue}}(\text{double}) \): \( p \)-value for neutrino indirect detection analysis \( X \) | \( X_{\text{data}} \)                           |                      |                    |
| \( X_{\text{nobs}} \) | \( X_{\text{nobs}}(\text{int}) \): Number of observed events for neutrino indirect detection analysis \( X \) | \( X_{\text{data}} \)                           |                      |                    |
| IC79_loglike    | IC79_loglike(\text{double}) \text{ The full 79-string IceCube log-likelihood} | \( Y_{\text{loglike}} \) \text{ for all } \( Y \in \{\text{IC79WH, IC79WL, IC79SL}\} \) |                      |                    |
| IceCube_likelihood | IC_loglike(\text{double}) \text{ The complete IceCube log-likelihood} | \( Y_{\text{loglike}} \) \text{ for all } \( Y \in \{\text{IC22, IC79WH, IC79WL, IC79SL}\} \) |                      |                    |
| Capability           | Function (return type): brief description                                      | Dependencies                        | Options (type)                     |
|----------------------|---------------------------------------------------------------------------------|--------------------------------------|------------------------------------|
| cascadeMC_FinalStates| cascadeMC_FinalStates (std::vector<std::string>): Function for retrieving list of final states for cascade decays | cMC_finalStates (std::vector<std::string>) |                                    |
| cascadeMC_DecayTable | cascadeMC_DecayTable (DarkBit::DecayChain::DecayTable): Function setting up the decay table used in decay chains | TH_ProcessCatalog SimYieldTable       |                                    |
| cascadeMC_gammaSpectra | cascadeMC_gammaSpectra (DarkBit::stringFunkMap): Function requesting and returning gamma ray spectra from cascade decays | GA_missingFinalStates cascadeMC_FinalStates cascadeMC_Histograms | cascadeMC_EventCount |
Table 24 The loop manager capability for the DarkBit cascade decay Monte Carlo, and the capabilities that are filled within the loop. Each of these depends on the `cascadeMC_LoopManager` capability.

| Capability                  | Function (return type): brief description                                                                 | Dependencies                     | Options (type)                        |
|-----------------------------|-----------------------------------------------------------------------------------------------------------|-----------------------------------|---------------------------------------|
| `cascadeMC_LoopManager`     | `cascadeMC_LoopManager(void):` Controls the loop for the cascade decay Monte Carlo simulation               | `GA_missingFinalStates`           | `cMC_maxEvents(int)`                  |
| `cascadeMC_InitialState`    | `cascadeMC_InitialState(std::string):` Function selecting the initial state for the cascade decay chain     |                                   |                                       |
| `cascadeMC_ChainEvent`     | `cascadeMC_GenerateChain(DarkBit::DecayChain::ChainContainer): Function for generating decay chains`      | `cascadeMC_DecayTable`            | `cMC_Emin(double)`                    |
| `cascadeMC_Histograms`     | `cascadeMC_Histograms(DarkBit::simpleHistContainer): Function responsible for histogramming and evaluating the end conditions for the event loop in the cascade decay Monte Carlo simulation | `TH_ProcessCatalog TH_HistProcessCatalog` | `cMC_numSpecSamples(int)`             |
### Table 25  Miscellaneous capabilities for DarkSUSY initialisation and debugging

| Capability | Function (return type): brief description | Dependencies | Backend requirements | Options (type) |
|------------|------------------------------------------|--------------|----------------------|----------------|
| DarkSUSY_PointInit | DarkSUSY_PointInit_MSSM (bool): Function to initialise DarkSUSY to a specific model point. The generic DarkSUSY initialisation is done in the backend initialisation; this here is only necessary for other capabilities that make use of model-specific DarkSUSY routines. | MSSM_spectrum | dswidth | use_DS_isasugra (bool) |
|              |                                          | decay_rates  | dsprep               | use_dsSLHAread (bool) |
|              |                                          |              | mssmpar              | debug_SLHA_filenames |
|              |                                          |              | dssusy               | (std::vector<str>) |
|              |                                          |              | dsSLHAread           |                     |
|              |                                          |              | dssusy_isasugra      |                     |
|              |                                          |              | dsgive_model_isasugra|                     |
|              |                                          |              | initFromSLHAea       |                     |
|              |                                          |              | And Decay Table      |                     |
| DarkSUSY_PointInit_LocalHalo | DarkSUSY_PointInit_LocalHalo_func (bool): Function to initialise Milky Way halo model parameters in DarkSUSY. Any GAMBIT function that uses a DarkSUSY function that depends on the structure of the Milky Way halo should have this as a dependency. | RD_fraction | dshmcom | v_earth (double) |
|              |                                          | LocalHalo    | dshmisodf |                     |
|              |                                          |              | dshnframevelcom      |                     |
|              |                                          |              | dshmoclue            |                     |
| dump_GammaSpectrum | dump_GammaSpectrum (double): Dumps gamma-ray yield into ASCII table | GA_AnnYield | filename (std::string) |                     |
| UnitTest_DarkBit | UnitTest_DarkBit (int): Prints various DarkBit results into YAML file | DD_couplings | fileroot (std::string) |                     |
|              |                                          | RD_oh2       | GA_AnnYield:Emin    |                     |
|              |                                          | GA_AnnYield:Emax |               |                     |
|              |                                          | TH_ProcessCatalog |              |                     |
|              |                                          | DarkMatter_ID | GA_AnnYield:nbins   |                     |
|              |                                          |              | (double)            |                     |
References

1. Planck Collaboration, P.A.R. Ade, et. al., Planck 2015 results. XIII. Cosmological parameters. A&A 594, A13 (2016). arXiv:1502.01589

2. J. Silk et. al., Particle Dark Matter: Observations, Models and Searches (2010)

3. F.D. Steffen, Dark matter candidates—axions, neutralinos, gravitinos, and axinos. Eur. Phys. J. C 59, 557–588 (2009). arXiv:0811.3347

4. J.L. Feng, Dark matter candidates from particle physics and methods of detection. ARA&A 48, 495–545 (2010). arXiv:1003.0904

5. H. Baer, K.-Y. Choi, J.E. Kim, L. Roszkowski, Dark matter production in the early Universe: beyond the thermal WIMP paradigm. Phys. Rep. 555, 1–60 (2015). arXiv:1407.0017

6. G. Jungman, M. Kamionkowski, K. Griest, Supersymmetric dark matter. Phys. Rep. 267, 195–373 (1996). arXiv:hep-ph/9506380

7. D. Hooper, S. Profumo, Dark matter and collider phenomenology of universal extra dimensions. Phys. Rep. 453, 29–115 (2007). arXiv:hep-ph/0701197

8. S. Tulin, H.-B. Yu, K.M. Zurek, Beyond collisionless dark matter: particle physics dynamics for dark matter Halo structure. Phys. Rev. D 87, 115007 (2013). arXiv:1302.3898

9. F.-Y. Cyr-Racine, K. Sigurdson, et. al., ETHOS—an effective theory of structure formation: from dark particle physics to the matter distribution of the Universe. arXiv:1512.05344

10. GAMBIT Collaboration, P. Athron, C. Baláz, et. al., GAMBIT: the global and modular beyond-the-standard-model inference tool. arXiv:arXiv:1705.07908

11. GAMBIT Collider Workgroup, C. Baláz, A. Buckley, et. al., ColliderBit: a GAMBIT module for the calculation of high-energy collider observables and likelihoods. arXiv:1705.07919

12. GAMBIT Flavour Workgroup, F.U. Bernlochner, M. Chrzaszcz, et. al., FlavBit: a GAMBIT module for computing flavour observables and likelihoods. arXiv:1705.07933

13. GAMBIT Models Workgroup, P. Athron, C. Baláz, et. al., SpecBit, DecayBit and PrecisionBit: GAMBIT modules for computing mass spectra, particle decay rates and precision observables. arXiv:1705.07936

14. GAMBIT Scanner Workgroup, G.D. Martinez, J. McKay, et. al., Comparison of statistical sampling methods with ScannerBit, the GAMBIT scanning module. arXiv:1705.07959

15. P. Gondolo, J. Edsjo et al., DarkSUSY: computing supersymmetric dark matter properties numerically. JCAP 7, 8 (2004). arXiv:astro-ph/0406204

16. G. Bélanger, J. Da Silva, T. Perrillat-Bottom, A. Pukhov, Limits on dark matter proton scattering from neutrino telescopes using micrOMEGAs. JCAP 12, 036 (2015). arXiv:1507.07987

17. IceCube Collaboration, M.G. Aartsen et. al., Improved limits on dark matter annihilation in the Sun with the 79-string IceCube detector and implications for supersymmetry. JCAP 04, 022 (2016). arXiv:1601.00653

18. M. Cacciari, G.P. Salam, G. Soyez, FastJet user manual. Eur. Phys. J. C 72, 1896 (2012). arXiv:1111.6097

19. P. Athron, J.-H. Park, D. Stöckinger, A. Voigt, FlexibleSUSY—a spectrum generator generator for supersymmetric models. Comput. Phys. Commun. 190, 139–172 (2015). arXiv:1406.2319

20. B.C. Allanach, SOFTSUSY: a program for calculating supersymmetric spectra. Comput. Phys. Commun. 143, 305–331 (2002). arXiv:hep-ph/0104145

21. E. Bertschinger, Self-similar secondary infall and accretion in an Einstein-de Sitter universe. ApJ Suppl. 58, 39 (1985)

22. J.P. Navarro, C.S. Frenk, S.D.M. White, The Structure of cold dark matter halos. ApJ 462, 563–575 (1996). arXiv:astro-ph/9508025

23. A.V. Kravtsov, A.A. Klypin, J.S. Bullock, J.R. Primack, The cores of dark matter dominated galaxies: theory versus observations. ApJ 502, 48 (1998). arXiv:astro-ph/9708176

24. B. Moore, T.R. Quinn, F. Governato, J. Stadel, G. Lake, Cold collapse and the core catastrophe. MNRAS 310, 1147–1152 (1999). arXiv:astro-ph/9903164

25. A.W. Graham, D. Merritt, B. Moore, J. Diemand, B. Terzic, Empirical models for dark matter Halos. I. Nonparametric construction of density profiles and comparison with parametric models. Astron. J. 132, 2685–2700 (2006). arXiv:astro-ph/0509417

26. U. Haud, J. Einasto, Galactic models with massive corona I. Method. A&A 223, 89–94 (1989)

27. R. Schoenrich, J. Binney, W. Dehnen, Local kinematics and the local standard of rest. MNRAS 403, 1829 (2010). arXiv:0912.3693

28. K. Freese, M. Lisanti, C. Savage, Annual modulation of dark matter: a review. Rev. Mod. Phys. 85, 1561–1581 (2013). arXiv:1209.3339

29. A.K. Drukier, K. Freese, D.N. Spergel, Detecting cold dark matter candidates. Phys. Rev. D 33, 3495–3508 (1986)

30. P.D. Serpico, G. Bertone, Astrophysical limitations to the identification of dark matter: indirect neutrino signals vis-a-vis direct detection recoil rates. Phys. Rev. D 82, 063505 (2010). arXiv:1006.3268

31. J.I. Read, The local dark matter density. J. Phys. G 41, 063101 (2014). arXiv:1404.1938

32. J. Bovy, S. Tremaine, On the local dark matter density. ApJ 756, 89 (2012). arXiv:1205.4033

33. J.A.R. Caldwell, J.P. Ostriker, The mass distribution within our galaxy: a three component model. ApJ 251, 61–87 (1981)

34. R. Catena, P. Ullio, A novel determination of the dark matter density. JCAP 1008, 004 (2010). arXiv:0907.0018

35. P. Salucci, F. Netti, G. Gentile, C.F. Martins, The dark matter density at the Sun’s location. A&A 523, A83 (2010). arXiv:1003.3101

36. M. Pato, F. Iocco, G. Bertone, Dynamical constraints on the dark matter distribution in the Milky Way. JCAP 1512, 001 (2015). arXiv:1504.06324

37. M. Pato, O. Agertz, G. Bertone, B. Moore, R. Teyssier, Systematic uncertainties in the determination of the local dark matter density. Phys. Rev. D 82, 023531 (2010). arXiv:1006.1322

38. Y. Akrami, C. Savage, P. Scott, J. Conrad, J. Edsjo, How well will ton-scale dark matter direct detection experiments constrain minimal supersymmetry? JCAP 1104, 012 (2011). arXiv:1101.4318

39. M.J. Reid et al., Trigonometric parallaxes of massive star forming regions: VI. Galactic structure, fundamental parameters and non-circular motions. ApJ 700, 137–148 (2009). arXiv:0902.3913

40. J. Bovy, D.W. Hogg, H.-W. Rix, Galactic masers and the Milky Way cycle. ApJ 704, 1704–1709 (2009). arXiv:0907.5423

41. M.C. Smith et al., The RAVE survey: constraining the local galactic escape speed. MNRAS 379, 755–772 (2007). arXiv:astro-ph/0611671

42. M. Cirelli, G. Corcella et al., PPPC 4 DM ID: a poor particle physicist cookbook for dark matter indirect detection. JCAP 3, 051 (2011). arXiv:1012.4515

43. J. Edsjo, P. Gondolo, Neutralino relic density including coannihilations. Phys. Rev. D 56, 1879–1894 (1997). arXiv:hep-ph/9704361

44. P. Gondolo, G. Gelmini, Cosmic abundances of stable particles: improved analysis. Nucl. Phys. A 360, 145–179 (1991)

45. P. Gondolo, J. Edsjö et al., DarkSUSY: computing supersymmetric dark matter properties numerically. JCAP 0407, 008 (2004). arXiv:astro-ph/0406204

46. G. Bélanger, F. Boudjema, A. Pukhov, A. Semenov, micrOMEGAs 3: a program for calculating dark matter
109. T. Bringmann, L. Bergström, J. Edsjö, New gamma-ray contribu-
107. T. Bringmann, C. Weniger, Gamma ray signals from dark mat-
105. M. Cirelli, D. Gaggero, G. Giesen, M. Taoso, A. Urbano, Antipro-
104. T. Bringmann, M. Vollmann, C. Weniger, Updated cosmic-ray and
101. T. Sjöstrand, S. Mrenna, P.Z. Skands, PYTHIA 6.4 physics and
99. COMPASS Collaboration, V. Yu. Alexakhin et. al., The deuteron
98. Y. Goto et al., Polarized parton distribution functions in the
96. P.E. Shanahan, Chiral Effective theory methods and their applica-
95. ETM, A. Abdel-Rehim, C. Alexandrou, et. al., Direct evaluation
93. L. Alvarez-Ruso, T. Ledwig, J. Martin Camalich, M.J. Vicente-
92. J. Ruiz de Elvira, M. Hoferichter, B. Kubis, U.-G. MeiSSner,
91. J.M. Alarcon, J. Martin Camalich, J.A. Oller, The chiral represen-
90. T. Bringmann, C. Weniger, Gamma ray signals from dark matter: concepts, status and prospects. Phys. Dark Univ. 1 (2012). arXiv:1208.5481
89. A. Ibarra, S. Lopez Gehier, M. Pato, Dark matter constraints from box-shaped gamma-ray features. JCAP 1207, 043 (2012). arXiv:1205.0007
88. L. Bergström, P. Ullio, J.H. Buckley, Observability of gamma-rays from dark matter neutralino annihilations in the Milky Way halo. Astropart. Phys. 9, 137–162 (1998). arXiv: astro-ph/9712318
87. T. Bringmann, L. Bergström, J. Edsjö, New gamma-ray contributions to supersymmetric dark matter annihilation. JHEP 01, 049 (2008). arXiv:0710.3169
86. L. Bergholm, T. Edsjö, J. Edsjö, P. Gondolo, Indirect detection of dark matter in km size neutrino telescopes. Phys. Rev. D 58, 103519 (1998). arXiv:hep-ph/9806293
85. M. Cirelli, G. Giesen, Antiprotons from dark matter: current constraints and future sensitivities. JCAP 1304, 015 (2013). arXiv:1301.7079
84. T. Bringmann, M. Vollmann, C. Weniger, Updated cosmic-ray and radio constraints on dark matter: implications for the GeV gamma-ray excess at the Galactic center. Phys. Rev. D 90, 123001 (2014). arXiv:1406.6027
83. M. Cirelli, D. Gaggero, G. Giesen, M. Taoso, A. Urbano, Antiproton constraints on the GeV gamma-ray excess: a comprehensive analysis. JCAP 1402, 045 (2014). arXiv:1407.2173
82. L. Bergström, T. Bringmann, I. Cholis, D. Hooper, C. Weniger, New limits on dark matter annihilation from AMS cosmic ray positron data. Phys. Rev. Lett. 111, 171101 (2013). arXiv:1306.3983
81. T. Bringmann, C. Weniger, Gamma ray signals from dark matter: concepts, status and prospects. Phys. Dark Univ. 1, 194–217 (2012). arXiv:1208.5481
80. L. Bergholm, J. Edsjö, P. Gondolo, Indirect detection of dark matter in km size neutrino telescopes. Phys. Rev. D 58, 103519 (1998). arXiv:hep-ph/9806293
79. Particle Data Group, K. A. Olive et. al., Review of Particle Physics. Chin. Phys. C 38, 090001 (2014)
78. Y. Goto et al., Polarized parton distribution functions in the nucleon. Phys. Rev. D 62, 034017 (2000). arXiv:hep-ph/0001046
77. COMPASS Collaboration, V.Y. Alexakhin et. al., The deuerton spin-dependent structure function g1(d) and its first moment. Phys. Rev. B 64, 8–17 (2007). arXiv:hep-ex/0609038
76. T. Sjöstrand, S. Mrenna, P.Z. Skands, PYTHIA 6.4 physics and manual. JHEP 05, 026 (2006). arXiv:hep-ph/0603175
75. T. Bringmann, C. Weniger, Gamma ray signals from dark matter: concepts, status and prospects. Phys. Dark Univ. 1, 194–217 (2012). arXiv:1208.5481
74. L. Bergström, J. Edsjö, P. Gondolo, Indirect detection of dark matter in km size neutrino telescopes. Phys. Rev. D 58, 103519 (1998). arXiv:hep-ph/9806293
73. M. Cirelli, G. Giesen, Antiprotons from dark matter: current constraints and future sensitivities. JCAP 1304, 015 (2013). arXiv:1301.7079
72. T. Bringmann, M. Vollmann, C. Weniger, Updated cosmic-ray and radio constraints on dark matter: implications for the GeV gamma-ray excess at the Galactic center. Phys. Rev. D 90, 123001 (2014). arXiv:1406.6027
71. M. Cirelli, D. Gaggero, G. Giesen, M. Taoso, A. Urbano, Antiproton constraints on the GeV gamma-ray excess: a comprehensive analysis. JCAP 1402, 045 (2014). arXiv:1407.2173
70. L. Bergström, T. Bringmann, I. Cholis, D. Hooper, C. Weniger, New limits on dark matter annihilation from AMS cosmic ray positron data. Phys. Rev. Lett. 111, 171101 (2013). arXiv:1306.3983
69. T. Bringmann, C. Weniger, Gamma ray signals from dark matter: concepts, status and prospects. Phys. Dark Univ. 1, 194–217 (2012). arXiv:1208.5481
the Fermi large area telescope. Phys. Rev. D 89, 042001 (2014). arXiv:1310.0828

136. Fermi-LAT Collaboration, M. Ackermann, A. Albert, et. al., Searching for dark matter annihilation from milky way dwarf spheroidal galaxies with six years of fermi large area telescope data. Phys. Rev. Lett. 115, 231301 (2015). arXiv:1503.02641

137. HESS Collaboration, A. Abramowski, et. al., Search for a Dark Matter annihilation signal from the Galactic Center halo with H.E.S.S. Phys. Rev. Lett. 106, 161301 (2011). arXiv:1103.3266

138. F. Calore, I. Cholis, C. Weniger, Background model systematics for the Fermi GeV excess. JCAP 1503, 038 (2015). arXiv:1409.0042

139. A. Achterberg, S. Amoroso et al., A description of the Galactic Center excess in the minimal supersymmetric standard model. JCAP 1508, 006 (2015). arXiv:1502.05703

140. H. Silverwood, C. Weniger, P. Scott, G. Bertone, A realistic assessment of the CTA sensitivity to dark matter annihilation. JCAP 1503, 055 (2015). arXiv:1408.4131

141. X. Huang, Y.-L.S. Tsai, Q. Yuan, LikeDM: likelihood calculator of dark matter detection. Comput. Phys. Commun. 213, 252–263 (2017). arXiv:1603.07119

142. A. Chiappi, J. Cohen-Tanugi, et. al., Dwarf spheroidal J-factors without priors: a likelihood-based analysis for indirect dark matter searches. arXiv:1608.07111

143. D. Hooper, L. Goodenough, Dark matter annihilation in the galactic center as seen by the Fermi gamma ray space telescope. Phys. Lett. B 697, 412–428 (2011). arXiv:1010.2752

144. O. Macias, C. Gordon, Contribution of cosmic rays interacting with molecular clouds to the Galactic Center gamma-ray excess. Phys. Rev. D 89, 063516 (2014). arXiv:1312.6671

145. K.N. Abazajian, N. Canac, S. Horiuchi, M. Kaplinghat, Astrophysical and dark matter interpretations of extended gamma-ray emission from the galactic center. Phys. Rev. D 90, 023526 (2014). arXiv:1402.4090

146. T. Daylan, D.P. Finkbeiner et al., The characterization of the Fermi gamma ray space telescope. Phys. Lett. B 697, 412–428 (2011). arXiv:1010.2752

147. B. Zhou, Y.-F. Liang et al., GeV excess in the Milky Way: the role of diffuse galactic gamma-ray emission templates. Phys. Rev. D 91, 123010 (2015). arXiv:1406.6948

148. Fermi-LAT Collaboration, M. Ajello et. al., Fermi-LAT observations of high-energy γ-ray emission toward the galactic center. ApJ 819, 44 (2016). arXiv:1511.02938

149. R. Bartels, S. Krishnamurthy, C. Weniger, Strong support for the millisecond pulsar origin of the Galactic center GeV excess. Phys. Rev. Lett. 116, 051102 (2016). arXiv:1506.05104

150. S.K. Lee, M. Lisanti, B.R. Safdi, T.R. Slatyer, W. Xue, Evidence for unresolved γ-ray point sources in the inner galaxy. Phys. Rev. Lett. 116, 051103 (2016). arXiv:1506.05124

151. F. Calore, I. Cholis, C. McCabe, C. Weniger, A tale of tails: dark matter interpretations of the Fermi GeV excess in light of background model systematics. Phys. Rev. D 91, 063003 (2015). arXiv:1411.4647

152. H. Silverwood, CTA morphological likelihood analysis. Private code

153. T. Bringmann, A.J. Galea, P. Wala, Leading QCD corrections for indirect dark matter searches: a fresh look. Phys. Rev. D 93, 043529 (2016). arXiv:1510.02473

154. T. Bringmann, F. Calore, Significant enhancement of neutralino dark matter annihilation from electronbeam Bremsstrahlung. Phys. Rev. Lett. 112, 071301 (2014). arXiv:1308.1089

155. T. Bringmann, F. Calore, A. Galea, M. Garny, Electrobeam and Higgs Boson Internal Bremsstrahlung: General considerations for Majorana dark matter annihilation and application to MSSM neutrinos. arXiv:1705.03466
ture from galactic center. Phys. Rev. D 71, 063528 (2005). arXiv:hep-ph/0412403
180. R. Iengo, Sommerfeld enhancement: general results from field theory diagrams. JHEP 05, 024 (2009). arXiv:0902.0688
181. J. Bovy, Substructure boosts to dark matter annihilation from Sommerfeld enhancement. Phys. Rev. D 79, 083539 (2009). arXiv:0903.0413
182. C. Arina, T. Bringmann, J. Silk, M. Vollmann, Enhanced line signals from annihilating Kaluza-Klein dark matter. Phys. Rev. D 90, 083506 (2014). arXiv:1409.0007
183. J. Choquette, J.M. Cline, J.M. Cornell, p-wave annihilating dark matter from a decaying predecessor and the galactic center excess. Phys. Rev. D 94, 015018 (2016). arXiv:1604.01039