Eur. Phys. J. C manuscript No.
(will be inserted by the editor)

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

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

[1]Department of Physics, University of Oslo, N-0316 Oslo, Norway
[2]Oskar Klein Centre for Cosmoparticle Physics, AlbaNova University Centre, SE-10691 Stockholm, Sweden
[3]Department of Physics, Stockholm University, SE-10691 Stockholm, Sweden
[4]Department of Physics, McGill University, 3600 rue University, Montréal, Québec H3A 2T8, Canada
[5]DESY, Notkestraße 85, D-22607 Hamburg, Germany
[6]NORDITA, Roslagstullsbacken 23, SE-10691 Stockholm, Sweden
[7]LAPTh, Université de Savoie, CNRS, 9 chemin de Bellevue B.P.110, F-74941 Annecy-le-Vieux, France
[8]Department of Physics, Imperial College London, Blackett Laboratory, Prince Consort Road, London SW7 2AZ, UK
[9]GRAPPA, University of Amsterdam, Science Park 904, 1098 XH Amsterdam, Netherlands
[10]Department of Physics, University of Adelaide, Adelaide, South Australia 5005, Australia
[11]Australian Research Council Centre of Excellence for Particle Physics at the Tera-scale

Received: date / Accepted: date

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 (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 (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, 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.

[a]torsten.bringmann@fys.uio.no
[b]cornellj@physics.mcgill.ca
[c]p.scott@imperial.ac.uk
[d]c.weniger@uva.nl

Contents

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 (Sec. 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 compute, 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

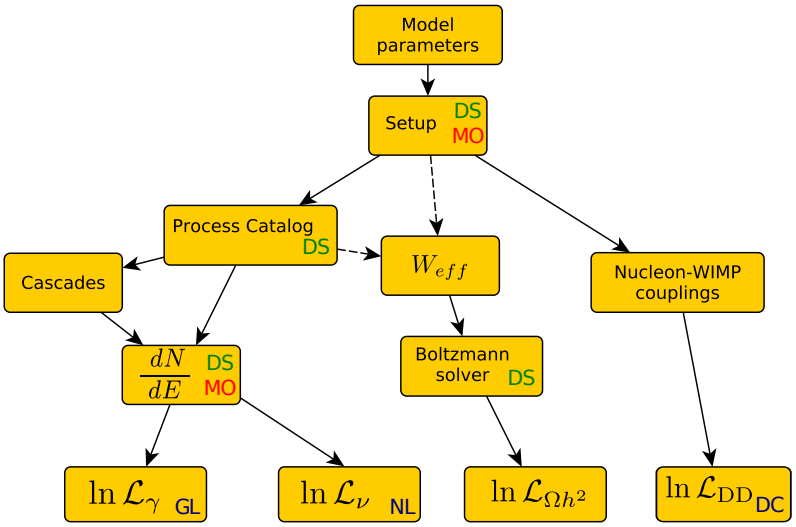

Fig. 1: Schematic overview over different DarkBit components. Based on the parameters of the scanned model(s), a Process Catalogue is initialised. This contains the relevant particle physics processes to infer dark matter annihilation spectra. The effective annihilation rate relevant for relic density calculations can (in simple cases) be inferred from the process catalogue, or it can be set directly. WIMP-nucleon couplings are also set depending on the model. All information is channelled into various likelihood routines. The two-letter insets indicate what backend codes can be used: DarkSUSY (DS), micrOMEGAs (MO), GamLike (GL), nulike (NL) and DDCalc (DC).

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 Sec. 5.2; for direct detection rates and likelihoods), GamLike (introduced in Sec. 6.2; for gamma-ray likelihoods) and nulike [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 Section 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 Sections 4, 5 and 6 we go through DarkBit's abilities in relic density calculations, direct and indirect detection, respectively. In particular, Section 5.2 introduces the new direct detection phenolikelihood code DDCalc and Section 6.2 introduces the new gamma-ray indirect detection likelihood code GamLike. We show validation tests and illustrative examples of typical DarkBit usage in Section 7. We continue with an outlook on planned code expansions in Section 8, and conclude in Section 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 re-usability. *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 annihilaton rate, and WIMP-nucleon couplings), which are used for relic density calculation (see Sec. 4), the calculation of direct detection constraints (Sec. 5), and the calculation of gamma-ray and neutrino yields (Sec. 6).

In many cases, and as indicated in Fig. 1, the calculations performed by DarkBit module functions build on functionality of external independent codes like DarkSUSY or micrOMEGAs. These codes provide a lot of functionality that can often be used beyond their original scope (examples are Boltzmann solvers, tabulated particle yields, routines to calculate J-values, etc). From the perspective of GAMBIT, these codes are **backends**. Technically, they are coupled to GAMBIT by compiling them as shared libraries, which are loaded by GAMBIT at runtime. The interface to these backends is provided by convenient **frontends**, which specify the form and subset of functionality of the backend that is accessible by GAMBIT. For details we again refer to Ref. [10].

Although the main purpose of DarkBit is to provide dark matter-related functionality for global scans with GAMBIT, it can also be used as a standalone code. In fact, most of the functionality of DarkBit could also be used outside of scans, e.g. implemented in a command line tool, if so desired. We will show a few examples for this below in Sec. 7.2.

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. 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} [18], 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

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

where r_s is a scale radius, and ρ_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 [19]:

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

[1] http://opensource.org/licenses/BSD-3-Clause

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 [20, 21] 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:

$$\rho(r) = \frac{2^{(\beta-\gamma)/\alpha}\rho_s}{(r/r_s)^\gamma\left[1+(r/r_s)^\alpha\right]^{(\beta-\gamma)/\alpha}}\,, \tag{3}$$

Here, γ describes the inner slope of the profile, β the outer slope, and α 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 [22], named after Einasto's use of the profile to model the mass distribution of galaxies [23]. In this model the logarithm of the slope varies continuously with radius, leading to a density profile of the form

$$\rho(r) = \rho_s \exp\left\{-\frac{2}{\alpha}\left[\left(\frac{r}{r_s}\right)^\alpha - 1\right]\right\}. \tag{4}$$

3.1.2 Velocity distribution

The velocities $\mathbf{v}$ 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 $v_{\rm esc}$ will leave the halo, takes the form

$$\tilde{f}(\mathbf{v}) = \frac{1}{N_{\rm esc}}(\pi v_0^2)^{-3/2} e^{-\mathbf{v}^2/v_0^2}\,, \tag{5}$$

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

$$N_{\rm esc} = \mathrm{erf}\left(\frac{v_{\rm esc}}{v_0}\right) - \frac{2v_{\rm esc}}{\sqrt{\pi}v_0}\exp\left(-\frac{v_{\rm esc}^2}{v_0^2}\right). \tag{6}$$

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

$$\mathbf{v}_{\rm obs} = \mathbf{v}_{\rm LSR} + \mathbf{v}_{\odot,\rm pec} + \mathbf{V}_\oplus(t)\,. \tag{7}$$

Here $\mathbf{v}_{\rm LSR} = (0, v_{\rm rot}, 0)$ is the motion of the Local Standard of Rest in Galactic coordinates, $\mathbf{v}_{\odot,\rm pec} = (11, 12, 7)\,\mathrm{km\,s^{-1}}$ is the well known peculiar velocity of the Sun [24], and $\mathbf{V}_\oplus(t)$ is the velocity of the Earth relative to the Sun. Although its magnitude is well measured at $29.78\,\mathrm{km\,s^{-1}}$ [25], the changing direction of $\mathrm{V}_\oplus$ is expected to give rise to an annual modulation of scattering rates in direct detection experiments [26]. The distribution of velocities $\mathbf{u}$ of dark matter particles in the Earth's frame is given by

$$f(\mathbf{u}, t) = \tilde{f}(\mathbf{v}_{\rm obs}(t) + \mathbf{u})\,. \tag{8}$$

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 $v_{\rm rot}$ 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% [27]).

3.2 Halo model implementation in GAMBIT

3.2.1 Halo models and associated capabilities

In GAMBIT, the radial distribution $\rho(r)$ of dark matter in the Milky Way, the local density ρ_0, the distance from the Sun to the Galactic centre $r_{\rm sun}$, as well as the local velocity distribution $f(\mathbf{u})$ 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 ρ_s, r_s, α, β and γ as defined in Eq. 3, together with the Maxwell-Boltzmann velocity distribution given by Eqs. 5–8, specified by the model parameters v_0, $v_{\rm rot}$ and $v_{\rm esc}$[2]. Lastly, the model contains $r_{\rm sun}$ and ρ_0 as additional free parameters. Analogously, the `Halo_Einasto` model describes the density profile given by Eq. 4, with free parameters ρ_s, r_s and α, and otherwise is identical to the `Halo_gNFW` model. Note that with appropriate choices of α, β, and γ, 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 ρ_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 $\rho(r_{\rm sun}) \equiv$

[2]The remaining velocity parameters $v_{\odot,\rm pec}$ and $|\mathbf{V}_\oplus(t)|$ are much better known. Hence, instead of being part of the halo models, $v_{\odot,\rm pec}$ simply defaults to the value assumed by the DDCalc backend, $(11, 12, 7)\,\mathrm{km\,s^{-1}}$, and $|\mathbf{V}_\oplus|$ to $29.78\,\mathrm{km\,s^{-1}}$. The magnitude of $\mathbf{V}_\oplus$ can however be overridden in module functions that use it, by setting the YAML option `v_earth`.

ρ_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 ρ_0, which is then internally converted to the corresponding value of the scale density ρ_s using Eq. 3. Conversely, in the latter model one specifies ρ_s, and ρ_0 is determined by GAMBIT. 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 backend functions relevant for direct and indirect searches, GAMBIT 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"`, and (ii) the distance $r_{\rm sun}$ 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 object is simply a container for the parameters relevant to direct detection and capture in the Sun, i.e. the local density ρ_0 as well as the velocity parameters v_0, $v_{\rm rot}$ and $v_{\rm esc}$.

Depending on the halo model in use, the capability `GalacticHalo` can be provided by the module functions `GalacticHalo_gNFW` or `GalacticHalo_Einasto`; for all halo models, the `LocalHalo` capability is obtained through the module function `ExtractLocalMaxwellianHalo` (see also Table A3). The rest of the GAMBIT code is designed such that only the module functions providing the capabilities `GalacticHalo` or `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 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

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 A3. For the local dark matter density ρ_0, we implement the likelihood as a log-normal function:

$$\mathcal{L}_{\rho_0} \propto \frac{\bar{\rho}_0}{\rho_0} \exp\left(-\frac{{\rho_0}^2 \ln(\rho_0/\bar{\rho}_0)^2}{2\sigma^2_{\rho_0}}\right). \tag{9}$$

The methods that have been used to determine ρ_0 can be roughly categorised into two approaches (see [28] for a review). Local measures (*e.g* [29]) use the kinematics of nearby stars to find ρ_0, whereas global measures (*e.g* [30–33]) 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 [34]. There appears to be a growing consensus that $\rho_0 \approx 0.4$ GeV/cm^3, so by default we set $\bar{\rho}_0$ to this value. We take σ_{ρ_0} to be 0.15 GeV/cm^3, to represent the range of determinations of the local density in the literature (see e.g. [35]).

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

$$\mathcal{L}_x \propto \exp\left(-\frac{(x-\bar{x})^2}{2{\sigma_x}^2}\right). \tag{10}$$

For the disk rotational velocity, by default the central value and error of the distribution are set to $235 \pm 20\,\mathrm{km\,s^{-1}}$, based on measurements of galactic masers [36, 37]. To take into account the possible discrepancies between v_0 and $v_{\rm rot}$ 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 $v_{\rm rot}$. Finally, for the escape velocity we use $v_{\rm esc} = 550 \pm 35\,\mathrm{km\,s^{-1}}$ based on measurements of high velocity stars in the RAVE survey [38]. The likelihood functions discussed in this section are listed in Tab. 1, along with their corresponding capabilities. The central values and errors for these likelihoods can be adjusted by setting the YAML options *param*`_obs` and *param*`_obserr` respectively, where *param* is the name of the parameter (e.g. to override the default likelihood for ρ_0 one would set `rho0_obs` and `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 GAMBIT initialisation file. For typical standard values we refer the reader to Ref. [39]; for recent kinematical constraints we refer to Ref. [33].

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

Parameter	Units	Likelihood Form	Central Value	Uncertainty	Function
Local dark matter density (ρ_0)	GeV/cm^3	log-normal	0.4	0.15	`lnL_rho0_lognormal`
Maxwellian most-probable speed (v_0)	km s^{-1}	Gaussian	235	20	`lnL_v0_gaussian`
Local disk rotation speed ($v_{\rm rot}$)	km s^{-1}	Gauusian	235	20	`lnL_vrot_gaussian`
Local galactic escape speed ($v_{\rm esc}$)	km s^{-1}	Gaussian	550	35	`lnL_vesc_gaussian`

Table 1: Milky Way halo parameters used in DarkBit, the form of their likelihood functions, their central values and 1 σ uncertainties, and the function that provides each likelihood. The capabilities associated with each likelihood are the same as the function name without the likelihood form, e.g. the capability for the ρ_0 likelihood is `lnL_rh0`.

particles, which in general can be written as [40]

$$\frac{dn}{dt} = -3Hn - \langle\sigma_{\rm eff} v\rangle \left(n^2 - n_{\rm eq}^2\right) \ , \tag{11}$$

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

$$\langle\sigma_{\rm eff} v\rangle = \frac{\int_0^\infty dp_{\rm eff} p_{\rm eff}^2 W_{\rm eff} K_1\left(\frac{\sqrt{s}}{T}\right)}{m_1^4 T\left[\sum_i \frac{g_i}{g_1}\frac{m_i^2}{m_1^2} K_2\left(\frac{m_i}{T}\right)\right]^2} \ , \tag{12}$$

where $p_{\rm eff} = \frac{1}{2}\sqrt{s - 4m_1^2}$ 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, $W_{\rm eff}$ is the effective annihilation rate, given by

$$\begin{aligned} W_{\rm eff} &= \sum_{ij} \frac{p_{ij}}{p_{11}} \frac{g_i g_j}{g_1^2} W_{ij} \\ &= \sum_{ij} \sqrt{\frac{[s - (m_i - m_j)^2][s - (m_i + m_j)^2]}{s(s - 4m_1^2)}} \frac{g_i g_j}{g_1^2} W_{ij} \ . \end{aligned} \tag{13}$$

Here,

$$p_{ij} \equiv \sqrt{\frac{(p_i \cdot p_j)^2 - m_i^2 m_j^2}{s}} \tag{14}$$

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

$$W_{ij} = 4p_{ij}\sqrt{s}\sigma_{ij} = 4E_i E_j \sigma_{ij} v_{ij} \ , \tag{15}$$

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 $W_{\rm eff}(p_{\rm eff})$. 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

$$\Omega_\chi = m_\chi n_0 / \rho_{\rm crit} \ . \tag{16}$$

Here, n_0 is the asymptotic value of $n(t \to \infty)$ as expected today and $\rho_{\rm crit} = 3H_0^2/8\pi G$. 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_χ – like for example for Dirac particles where there would be both DM particles and *anti*particles. 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 $W_{\rm eff}$ 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 $x \equiv m_\chi/T$ and $Y \equiv n/s$, where s now denotes the entropy density of the heat bath [41]. 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 $W_{\rm eff}$ 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 stepsize; see [15] for further details. s

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] [42] 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

[3] http://www.darksusy.org

backends. DarkSUSY is used by GAMBIT to obtain multiple theoretical quantities, including $W_{\rm eff}$, 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_PointInit` (see Table A13). It is then the responsibility of the function that provides this capability to initialise DarkSUSY correctly. A separate capability `DarkSUSY_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 A4 and A5.

MicrOMEGAs[4] [43–46] can be used by DarkBit to obtain many of the same quantities as DarkSUSY, including 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 $W_{\rm eff}$, 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 CalcHEP model file and then compiling micrOMEGAs with this file. 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 [47, 48] and one for the scalar singlet DM model. The latter is not included by default with micrOMEGAs, so we have included the CalcHEP file needed to implement it with the GAMBIT distribution.

The intialisation 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 `VWdecay`. 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 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. The behaviour of `MicrOmegas_MSSM_3_6_9_2_init` is also controlled by the YAML option `internal_decays`, which when set to `true` causes information from the GAMBIT decay table to be used to initialise micrOMEGAs. By default this is set to `false` and decay widths are calculated internally by micrOMEGAs. This ability to pass decay information to micrOMEGAs was a late addition to DarkBit; future versions will employ this option by default.

[4] https://lapth.cnrs.fr/micromegas

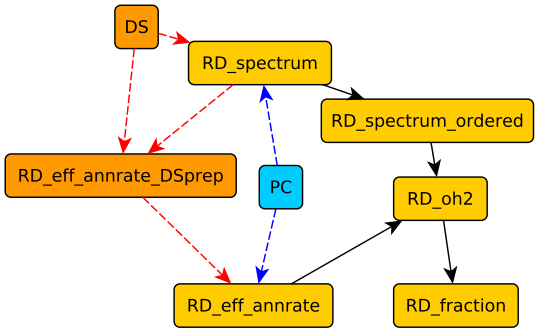

Fig. 2: Relic density overview plot. The capabilities required for relic density calculations are the effective annihilation rate $W_{\rm eff}$ and the ordered spectrum of coannihilating particles. Currently, these quantities can be set up either directly with DarkSUSY (DS) or in simpler cases without coannihilation via the Process Catalogue (PC). However, each of these functions can be easily replaced with a user-defined function.

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 $W_{\rm eff}$. In DarkBit, this is represented by the capability `RD_eff_annrate`.

Currently, there are two functions that provide this capability (Table A4), though further user-defined functions can easily be added. The first function, `RD_eff_annrate_SUSY`, returns $W_{\rm eff}$ 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 $W_{\rm eff}$ 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). 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 $\langle\sigma_{\rm eff}v\rangle$. The second alternative is to determine $W_{\rm eff}$ directly from the Process Catalogue (Sec. 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 A2).

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 function `RD_fraction_from_oh2`. The behaviour of this function depends on two options, each with default values shown in square brackets:

`oh2_obs`[`0.1188`]: the reference (observed) value of Ωh^2

`mode`[`"one"`]: return either the default fraction of `1` (`"one"`), the lesser of `1` and the computed-to-reference relic density ratio (`"leq_one"`), or the computed-to-reference ratio regardless of whether or not it exceeds `1` (`"any"`).

This last option specifies whether indirect and direct detection routines should use the observed relic density or the one calculated for the particular particle point.

DarkBit currently includes three functions that provide `RD_oh2` (Table A5). Two of those 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. 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 sparticle masses and couplings): the option `fast` (`int`; `0`: accurate, `1`: fast [default]) steers the numerical performance of the backend code. In order to calculate $\langle\sigma_{\rm eff}\rangle$, 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 $W_{\rm eff}$. `RD_oh2` therefore depends on the capability `RD_spectrum_ordered` which contains all this information, ordered by increasing $p_{\rm eff}$. `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 Sec. 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 simple Gaussian likelihood that implements the limits from Ref. [1] ($\Omega_\chi h^2 = 0.1188 \pm 0.0010$ at 1σ). In addition to the experimental error, a 5% systematic theory error is included in the likelihood by default. This value is conservative for most parameter combinations, and underestimates the $\mathcal{O}(50\%)$ corrections that can occur due to loop corrections in a few very specific scenarios. [49, 50]. The choice is thus a pragmatic compromise that represents the best that can be done with a single uncertainty. The theory error is added to the experimental error in quadrature. The mean value, the experimental error and the theoretical error can be changed with the YAML file options `oh2_obs`, `oh2_obserr`, and `oh2_fractional_theory_err`, respectively. Second, `lnL_oh2_upperlimit` implements the same observational constraint as a one-sided limit. It has the same three YAML file options as above. The likelihood function is a simple half-sided Gaussian.

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 [51] 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

$$\frac{dR}{dE} = \frac{2\rho_0}{m_\chi}\int v f(\mathbf{v},t)\frac{d\sigma}{dq^2}(q^2,v)\,d^3v, \tag{17}$$

where m_χ is the WIMP mass, ρ_0 is the local DM mass density, $f(\mathbf{v},t)$ 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 $q^2 = 2m_{\rm nuc}E$ is the momentum exchanged in the scattering process (for

a nucleon mass $m_{\rm nuc}$ 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. (17) 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

$$N_{\rm p} = MT \int_0^\infty \phi(E) \frac{dR}{dE}(E)\, dE, \tag{18}$$

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. 18.

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. Eq. 18 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 χ), the dominant WIMP-quark interactions arise from a combination of

- a scalar ($\bar\chi\chi\bar q q$) or vector ($\bar\chi\gamma_\mu\chi\bar q\gamma^\mu q$) coupling, which give rise to a spin-independent (SI) cross-section, and
- an axial-vector coupling ($\bar\chi\gamma_\mu\gamma_5\chi\bar q\gamma^\mu\gamma_5 q$), which gives rise to a spin-dependent (SD) cross-section

(for a complete list of possible operators see Ref. [52]). 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 σ_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

$$\frac{d\sigma}{dq^2}(q^2, v) = \frac{\sigma_0}{4\mu^2 v^2} F^2(q)\, \Theta(q_{\rm max} - q)\ , \tag{19}$$

where Θ is the Heaviside step function, μ is the WIMP-nucleon reduced mass, $q_{\rm max} = 2\mu v$ 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 $q \leq q_{\rm max}$ for an interaction to be kinematically possible translates into a lower limit $v \geq v_{\rm min} = \sqrt{m_{\rm nuc} E / 2\mu^2}$ in the integral over the WIMP velocity distribution (Eq. 17). 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 σ_0 for SI WIMP-nucleus interactions is

$$\begin{aligned} \sigma_{\rm SI} &= \frac{\mu^2}{\pi}\Big[Z G^{\rm p}_{\rm SI} + (A-Z) G^{\rm n}_{\rm SI}\Big]^2 \\ &= \frac{4\mu^2}{\pi}\Big[Z f_{\rm p} + (A-Z) f_{\rm n}\Big]^2\ , \end{aligned} \tag{20}$$

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 $f_{\rm p}$ and $f_{\rm n}$ 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 $G^{\rm p}_{\rm SI} = 2f_{\rm p}$ and $G^{\rm n}_{\rm SI} = 2f_{\rm n}$ is often found in the literature, where $G^N_{\rm SI}$ with $N = n, p$ are the G_F-like effective four-fermion coupling constants in the case of scalar interactions. The micrOMEGAs manual, meanwhile, uses $\lambda_N = \frac{1}{2} G^N_{\rm SI}$. The nucleon contents are described by the nuclear hadronic matrix elements, discussed in detail in Sec. 5.3.2 below.

For most DM candidates with scalar couplings, the proton and neutron SI cross-sections are roughly the same, so $f_{\rm n} \simeq f_{\rm p}$. For identical couplings ($f_{\rm n} = f_{\rm p}$), the SI cross-section reduces to

$$\sigma_{\rm SI} = \frac{\mu^2}{\mu_{\rm p}^2} A^2\, \sigma_{\rm SI,p}\ , \tag{21}$$

where $\mu_{\rm p}$ 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 [53, 54],

$$F(q) = 3e^{-q^2 s^2/2}\,\frac{\sin(qr_n) - qr_n\cos(qr_n)}{(qr_n)^3}\,, \tag{22}$$

where $s \simeq 0.9$ fm and $r_n^2 = c^2 + \frac{7}{3}\pi^2 a^2 - 5s^2$ is an effective nuclear radius with $a \simeq 0.52$ fm and $c \simeq 1.23A^{1/3} - 0.60$ fm. Further details on SI form factors can be found in Refs. [54, 55].

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

$$\begin{aligned}\sigma_{\rm SD} &= \frac{4\mu^2}{\pi}\,\frac{(J+1)}{J}\left[G^{\rm p}_{\rm SD}\langle S_{\rm p}\rangle + G^{\rm n}_{\rm SD}\langle S_{\rm n}\rangle\right]^2 \\ &= \frac{32\mu^2\,G_F^2}{\pi}\,\frac{(J+1)}{J}\left[a_{\rm p}\langle S_{\rm p}\rangle + a_{\rm n}\langle S_{\rm n}\rangle\right]^2,\end{aligned} \tag{23}$$

where G_F is the Fermi constant, J is the spin of the nucleus, $\langle S_{\rm p}\rangle$ and $\langle S_{\rm n}\rangle$ are the average spin contributions from the proton and neutron groups respectively, and $a_{\rm p}$ and $a_{\rm n}$ are the effective couplings to the proton and the neutron in units of $2\sqrt{2}G_F$. Similarly to $f_{\rm p}$ and $f_{\rm p}$, $a_{\rm p}$ and $a_{\rm n}$ 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 Sec. 5.3.2. As in the spin-independent case, alternative normalisations can be found in the literature. The DarkSUSY manual, for example, refers to $G^N_{\rm SD} = 2\sqrt{2}G_F a_N$, while the micrOMEGAs manual uses $\xi_N = \frac{1}{2}G^N_{\rm SD}$. In addition, whilst we here use a_N and $G^N_{\rm SD}$ to distinguish the two notations, a_N is frequently used within the literature for *both* cases.

Unlike the SI case, the two SD couplings $a_{\rm p}$ and $a_{\rm n}$ differ substantially in many theories. Individual experiments typically only strongly constrain either $a_{\rm p}$ or $a_{\rm n}$, 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 $\sigma_{\rm SD,p} = \sigma_{\rm SD}(a_{\rm n} = 0)$ or $\sigma_{\rm SD,n} = \sigma_{\rm SD}(a_{\rm p} = 0)$.

The SD form factor is given in terms of the structure function $S(q)$ normalised so that $F^2(0) = 1$,

$$F^2(q) = S(q)/S(0), \tag{24}$$

with

$$S(q) = a_{\rm p}^2\,S_{\rm pp}(q) + a_{\rm n}^2\,S_{\rm nn}(q) + a_{\rm p}\,a_{\rm n}\,S_{\rm pn}(q)\,. \tag{25}$$

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 the proton and neutron subsystems [56, 57],

$$\begin{aligned}S_{\rm pp}(0) &= \frac{(J+1)(2J+1)}{\pi J}\langle S_{\rm p}\rangle^2, \\ S_{\rm nn}(0) &= \frac{(J+1)(2J+1)}{\pi J}\langle S_{\rm n}\rangle^2.\end{aligned} \tag{26}$$

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. $f_{\rm p} = f_{\rm n}$ (often somewhat loosely referred to as 'isospin conservation') is assumed,
3. either $a_{\rm p} = a_{\rm n} = 0$ (pure SI coupling), or $f_{\rm p} = f_{\rm n} = (a_{\rm p}$ or $a_{\rm n}) = 0$ (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. Sec. 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 $f_{\rm p}$, $f_{\rm n}$, $a_{\rm p}$ and $a_{\rm n}$ 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 $f_{\rm p}$, $f_{\rm n}$, $a_{\rm p}$ and $a_{\rm n}$ agrees or disagrees with data, not merely which side of a 90% CL curve it lies on under different limiting approximations about $f_{\rm p}$, $f_{\rm n}$, $a_{\rm p}$ and $a_{\rm n}$. 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 [58], and has been used in a number of analyses (e.g. [59–61]).

Yet another issue is that the limits presented by experimental collaborations almost always assume Eq. 19, 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. 17), and extra momenta must be included in the final integral over the differential rate when calculating the total event yield (Eq. 18). Furthermore, these models often probe 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. 18, as implemented in the source file `DDRates.f90`. Similarly, velocity-dependent cross-sections can be implemented in the integrand of Eq. 17, 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. 22). 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. [62]. 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_esc, local speed relative to the halo v_obs and local density ρ_χ are all individually configurable; v_obs can also be constructed automatically from the local standard of rest $\mathbf{v}_\mathrm{LSR}$ or disk rotation speed v_loc, along with the Sun's peculiar velocity $\mathbf{v}_{\odot,\mathrm{pec}}$ 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 $N_{\mathrm{p},i}$ for the ith experimental analysis region (Eq. 18), DDCalc calculates a Poisson likelihood for the model as

$$\mathcal{L}_i(N_{\mathrm{p},i}|N_{\mathrm{o},i}) = \frac{(b_i + N_{\mathrm{p},i})^{N_{\mathrm{o},i}}\, e^{-(b_i+N_{\mathrm{p},i})}}{N_{\mathrm{o},i}!}, \tag{27}$$

where $N_{\mathrm{o},i}$ 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, $\{\mathcal{L}_i\}$ 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:

$$b_i = \begin{cases} N_{\mathrm{o},i} - N_{\mathrm{p},i}\,, & N_{\mathrm{o},i} > N_{\mathrm{p},i} \\ 0\,, & N_{\mathrm{o},i} \leq N_{\mathrm{p},i}\,. \end{cases} \tag{28}$$

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 σ-m_χ plane. A parameter point is considered to be excluded at 90% confidence level if

$$2\log\mathcal{L}(\sigma = 0) - 2\log\mathcal{L}(\sigma, m_\chi) > 2.7\,, \tag{29}$$

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

Feldman-Cousins

DDCalc implements the standard Feldman-Cousins method [63] for generation of one- or two-sided confidence intervals, based on the Poisson likelihood in Eq. 27. 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 [64] 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.

Experiment	Analysis
XENON100	2012 [65]
SuperCDMS	2014 [66]
SIMPLE	2014 [67]
LUX (run 1)	2013 [68], 2015 [69]
LUX (run 2)	2016 [70]
PandaX	2016 [71]
PICO-60	2016 [72]
PICO-2L	2016 [73]

Table 2: Experimental analyses included in DDCalc 1.0.0.

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 [58] 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 $\phi_k(E)$ for each of these intervals, one can then use Eq. (18) 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 [64].

5.2.2 Experiments

Event rate calculations in DDCalc rely on the availability of experimental response functions $\phi(E)$, 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 [68–70], PandaX [71], XENON100 [65] and SuperCDMS [66] analyses are most useful for constraining SI scattering and SD scattering on neutrons, whereas the PICO-2L [73], PICO-60 [72] and SIMPLE [67] 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 $\phi(E)$ directly as described in Ref. [67]. 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 $\phi(E)$ for flourine is provided in Ref. [74]. 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 $\phi(E)$ 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 $\mathcal{L} = e^{-N_\mathrm{P}}$, 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 presecription [54]. The first is based directly on the published efficiencies and event energies of all detectors included in the experimental run [66]. 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.[5] This alternative parameterisation is the default within DDCalc for this analysis.

PandaX. We implement the efficiency curve $\phi(E)$ provided by the collaboration [71], 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 [65], 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 ($E > 30$ 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 [75], which in turn relies on NEST for modelling the physics of recoiling heavy nuclei [76]. The TPCMC output is

[5] The additional efficiency information for T5Z3 was kindly provided by the SuperCDMS Collaboration.

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 θ 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 π 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 --constraints-SD`	As above, but for the spin-dependent case.
`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 θ 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 π 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.

Table 3: The various run modes of the DDCalc default executable `DDCalc_run`.

included in the public release of DDCalc, and provides the full gap information for XENON100.

LUX. The TPCMC code has also been used to obtain the efficiencies for the first run of LUX [68] and these are included in DDCalc. For the reanalysis of the first run [69], which improves the sensitivity to low-mass WIMPs, only the total efficiency curve provided by the collaboration is included (again scaled down by a factor of 2). In its second run, LUX saw a total of six events (compared to a background expectation of 3.5 events). This makes the use of spectral information for both signal and background an important part of the analysis. The maximum gap method is not suitable for this task, as it includes only the spectral information of the signal but treats the background as unknown. To approximately reproduce the LUX analysis, we exclude those parts of the $S1$-$S2$ plane where events are likely to have arisen from the background. Specifically we impose the requirement $3\,\text{phe} \leq S1 \leq 33\,\text{phe}$ as well as an $S2$ signal below the mean of the nuclear recoil band. We calculate the efficiency curve $\phi(E)$ for this search window using similar methods as for the first LUX run. To estimate the expected background, we assume that backgrounds due to leakage from the electron recoil band are equally distributed in $S1$, while all other backgrounds have a shape that resembles neutron recoils from calibration data. Using these assumptions we predict 2.3 background events in the reduced search window, whereas LUX observed one event. While the definition of this reduced search window contains some degree of arbitrariness and the background estimation is rather crude, this approach enables us to reproduce the published LUX bound to good accuracy (see Fig. 8). Once the LUX collaboration has released more details on the analysis, the run 2 implementation can be updated accordingly.

General options	
`--help`	Displays help information.
`--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.
Mode-specific options	
`--E-tabulation=<min>,<max>[,<N>] [0.1,1000,-50]`	Specifies the tabulation for the recoil energy E. Energies will be tabulated from `<min>` to `<max>` inclusive, with N logarithmically spaced intervals between tabulation points. A negative `<N>` value indicates `|<N>|` points per decade. The `<N>` argument is optional.
`--interactive`	Turns on the interactive mode, which will prompt the user for the WIMP parameters.
`--m-tabulation=<min>,<max>[,<N>] [1,1000,-20]`	Specifies the tabulation for the WIMP mass m. Masses will be tabulated from `<min>` to `<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 `<N>` value indicates `|<N>|` points per decade in the logarithmic case. The `<N>` and `<log>` arguments are optional.
`--theta-SD=<val>` [$\pi/4$]	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 π.
`--theta-SI=<val>` [$\pi/4$]	As above, but for the spin-independent case.
Statistical options	
`--confidence-level=<val>` [0.9]	The confidence level to use in determining constraints/limits.
`--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 3σ CL constraint/limit. Equivalent to `--p-value-sigma`.
`--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`.
Experiment-specific options	
`--XENON100-2012`	Sets the detector and analysis according to the XENON100 2012 result [65].
`--LUX-2013`	Sets the detector and analysis according to the LUX run 1 result from 2013 [68].
`--LUX-2015`	Sets the detector and analysis according to the reanalysis of the LUX run 1 result from 2015 [69].
`--LUX-2016_prelim`	Sets the detector and analysis according to our preliminary implementation of the LUX run 2 result from 2016 [70].
`--PandaX-2016`	Sets the detector and analysis according to the PandaX 2016 result [71].
`--SuperCDMS-2014`	Sets the detector and analysis according to the SuperCDMS 2014 low-energy result [66].
`--SIMPLE-2014`	Sets the detector and analysis according to the SIMPLE 2014 C_2ClF_5 result [67].
`--PICO-2L`	Sets the detector and analysis according to the PICO-2L 2016 result [73].
`--PICO-60`	Sets the detector and analysis according to the PICO-60 2016 result [72], using only the contribution from fluorine.

Table 4: Options of the DDCalc default executable `DDCalc_run`.

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:

`--rho=<val> [0.4]`	The local dark matter density (in units of GeV cm^{-3}).
`--v0=<val> [vrot]`	The most probable speed of the Maxwell-Boltzmann distribution (in km s^{-1}). Related to the rms speed by $v_{rms} = \sqrt{3/2}v_0$ and the mean speed by $\bar{v} = \sqrt{4/\pi}v_0$. 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.
`--vesc=<val> [550]`	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.
`--vlsr=<val>,<val>,<val> [0,vrot,0]`	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,`--vrot`,0) unless specified here.
`--vobs=<val> [|vsun|]`	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.
`--vpec=<val>,<val>,<val> [11,12,7]`	The velocity of the Sun relative to the Local Standard of Rest in galactic coordinates UVW.
`--vsun=<val>,<val>,<val> [vlsr+vpec]`	The motion of the Sun relative the the galactic rest frame (in km s^{-1}) in galactic coordinates UVW. Set equal to `--vlsr` + `--vpec` unless explicitly set here.
`--vrot=<val> [235]`	The local disk rotation speed (in km s^{-1}).

Table 5: Dark matter halo distribution options of the DDCalc default executable `DDCalc_run`.

```
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 sigmaSI 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 `sigmaSI` and `sigmaSD` 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.

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.

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 com-
`--stoichiometry=<N1>,<N2>,<N3 ↪>,...`	pound with the given set of elements and stoichiometry. For example, CF_3Cl 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.
`--isotope-Z=<Z1>,<Z2>,<Z3>,...`	Provide an explicit list of isotopes to use. The list of atomic numbers (`Z`),
`--isotope-A=<A1>,<A2>,<A3>,...`	mass numbers (`A`), and mass fractions (`f`) must all be provided and be of
`--isotope-f=<f1>,<f2>,<f3>,...`	the same length or these options will be ignored.
Detector efficiency options	
`--Emin=<val> [0]`	Applies a minimum energy threshold for rate calculations (in keV). This effectively takes: $\phi(E) \to \Theta(E - E_{min})\phi(E)$, 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.
`--file=<file>`	A file containing the efficiency $\phi(E)$ tabulated by energy, where $\phi(E)$ is defined as the fraction of events at an energy E that will be observed in the analysis region after factoring in trigger efficiencies, energy resolution, data cuts, etc. The first column is taken to be the energy (in keV). The next column should contain only numbers between 0 and 1; this is taken to be $\phi(E)$; note that this allows TPCMC output to be used, as columns of `<S1>` and `<S2>` will be safely ignored. Blank lines and lines beginning with a hash character will be ignored. If the analysis observed any events, the file can optionally include additional columns beyond the $\phi(E)$ column representing the efficiencies $\phi_k(E)$ for detecting events in the sub-intervals for use with the maximum gap method.
`--no-intervals`	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.

Table 6: Advanced detector options of the DDCalc default executable `DDCalc_run`.

`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_χ in GeV, `fp` and `fn` are f_p and f_n in GeV^{-2}, `ap` and `an` are (dimensionless) $a_\mathrm{p}$and a_n, the `G` parameters are G^p_SI, G^n_SI, G^p_SD and G^n_SD, and the `sigma` parameters are $\sigma_\mathrm{SI,p}$, $\sigma_\mathrm{SI,n}$, $\sigma_\mathrm{SD,p}$ and $\sigma_\mathrm{SD,n}$ in pb. Negative cross-sections indicate that the corresponding coupling should be negative. In all cases, 'p' refers to proton and 'n' to neutron.

Parameter retrieval

```
SUBROUTINE DDCalc_GetWIMP_mfa(WIMP,m,fp,fn,ap,an)
SUBROUTINE DDCalc_GetWIMP_mG(WIMP,m,GpSI,GnSI,GpSD,
 GnSD)
SUBROUTINE DDCalc_GetWIMP_msigma(WIMP,m,sigmapSI,
 sigmanSI,sigmapSD,sigmanSD)
```

As per `SetWIMP`, but retrieves the WIMP parameters

from WIMP. The only difference here is that returned cross-sections are always positive, regardless of the sign of the corresponding coupling.

Advanced setters and getters

See DDCalc_SetWIMP() and DDCalc_GetWIMP() in DDWIMP.f90.

Halo model

Initialisation

```
TYPE(HaloStruct)FUNCTION DDCalc_InitHalo()
```

Creates a new HaloStruct and initialises it to the Standard Halo Model ($\rho_\chi = 0.4\,\mathrm{GeV\,cm^{-3}}$, $v_{\mathrm{rot}} = 235\,\mathrm{km\,s^{-1}}$, $v_0 = 235\,\mathrm{km\,s^{-1}}$, $v_{\mathrm{esc}} = 550\,\mathrm{km\,s^{-1}}$).

Parameter setting

```
SUBROUTINE DDCalc_SetSHM(Halo,rho,vrot,v0,vesc)
```

Sets the internal parameters of the Halo object. Here rho is ρ_χ in $\mathrm{GeV\,cm^{-3}}$, vrot is v_{rot} in $\mathrm{km\,s^{-1}}$, v0 is v_0 in $\mathrm{km\,s^{-1}}$, and vesc is v_{esc} in $\mathrm{km\,s^{-1}}$.

Advanced setters and getters

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

Experiments and analysis

Initialisation

```
TYPE(DetectorStruct)FUNCTION
                    DDCalc_InitDetector(intervals)
                    analysis_name_Init(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 [68]. 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
```

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, but note that the efficiency curves already account for detector and analysis thresholds regardless of this setting. Setting this to 0 keV therefore still does not allow very low energy recoils to actually contribute to the signal.

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 saved internally in the Detector analysis object, and 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_Signal(Detector)`
   Alternatively, for the 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. 27.
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. 27 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)`
   Calculates the factor x by which the cross-sections must be scaled ($\sigma \to x\sigma$) 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

```
void DDCalc::FreeWIMPs();
void DDCalc::FreeHalos();
void DDCalc::FreeDetectors();
void DDCalc::FreeAll();
```

can be used to delete the objects held internally by DDCalc.

5.3 Direct detection implementation in DarkBit

5.3.1 WIMP-nucleon couplings

GAMBIT contains multiple functions capable of calculating WIMP-nucleon couplings. These are given in Table A1. Each has capability DD_couplings, and returns a DM_nucleon_couplings object, which contains the parameters $G^{\rm p}_{\rm SI}$, $G^{\rm n}_{\rm SI}$, $G^{\rm p}_{\rm SD}$ and $G^{\rm n}_{\rm SD}$, see Table A1.

The DD_couplings_DarkSUSY and DD_couplings_MicrOmegas functions are capable of computing the couplings for generic MSSM models by passing MSSM Spectrum information and nuclear parameters to external codes; one retrieves the couplings from DarkSUSY, the other obtains them from micrOMEGAs. The actual calculation of the WIMP-nucleon couplings from the nuclear parameters and neutralino-quark scattering matrix elements in both these cases closely follows the descriptions in Refs. [6, 35].

The DD_couplings_SingletDM and DD_couplings_MicrOmegas functions both have the ability to calculate the nucleon couplings for the scalar singlet model. The former function calculates the effective Higgs-nucleon coupling f_N internally, from the nuclear matrix elements relevant for SI scattering (described in more detail in the next section), and combines it with the scalar mass and Higgs-portal coupling as described in Ref. [77]. The latter uses micrOMEGAs with our CalcHEP implementation of the scalar singlet model to determine the couplings.

5.3.2 Nuclear uncertainties

When the interaction between DM and quarks can be described by a scalar operator, the spin independent effective couplings $G^{\rm p}_{\rm SI}$ and $G^{\rm n}_{\rm SI}$ 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

$$m_N f^{(N)}_{T_q} \equiv \langle N|m_q \bar{q} q|N\rangle \,, \tag{30}$$

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 [77, 78]

$$f^{(N)}_{TQ} = \frac{2}{27}\left(1 - \sum_{q=u,d,s} f^{(N)}_{T_q}\right) \tag{31}$$

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

$$\sigma_l \equiv m_l \langle N|\bar{u}u + \bar{d}d|N\rangle \tag{32}$$

$$\sigma_s \equiv m_s \langle N|\bar{s}s|N\rangle \,, \tag{33}$$

where $m_l \equiv (1/2)(m_u + m_d)$. In the **nuclear_params_sigmas_sigmal** model, these 2 parameters replace the 6 $f^{(N)}_{T_q}$ values, with the conversion between the two parameter sets described in Ref. [77]. An equivalent parameterisation replaces σ_s with the parameter σ_0:

$$\sigma_0 \equiv m_l \langle N|\bar{u}u + \bar{d}d - 2\bar{s}s|N\rangle \,. \tag{34}$$

By comparing the forms of the above equations, we see that σ_0 and σ_l are related by the formula

$$\sigma_0 = \sigma_l - \sigma_s \left(\frac{2m_l}{m_s}\right) . \tag{35}$$

The GAMBIT model **nuclear_params_sigma0_sigmal** contains σ_0 and σ_l.

For the spin-dependent effective couplings $G^{\rm p}_{\rm SD}$ and $G^{\rm n}_{\rm SD}$, the relevant nuclear parameters are $\Delta^{(N)}_q$, 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: $\Delta^{(n)}_u = \Delta^{(p)}_d$, $\Delta^{(n)}_d = \Delta^{(p)}_u$, and $\Delta^{(n)}_s = \Delta^{(p)}_s$. All of the **nuclear_params** models contain the three $\Delta^{(p)}_q$.

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 σ_s and σ_l is implemented in DarkBit as the capability lnL_SI_nuclear_parameters. This capability is provided by the function

lnL_sigmas_sigmal (Table A6), in which the two likelihoods take the form of simple Gaussian distributions (Eq. 10), with $\sigma_s = 43 \pm 8$ MeV, based on a global fit of lattice calculations [79], and $\sigma_l = 58 \pm 9$ MeV. This last value is based on the range of determinations of σ_l in the literature. This parameter has been determined using multiple methods, including extrapolation of pion-nucleon scattering results (e.g. [80]), as well as fits of the parameters of baryon chiral perturbation theory to scattering [81] and lattice QCD [82] data.

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

$$a_3 = \Delta_u^{(p)} - \Delta_d^{(p)} \tag{36}$$

$$a_8 = \Delta_u^{(p)} + \Delta_d^{(p)} - 2\Delta_s^{(p)} \tag{37}$$

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 A6). The likelihoods again take the form of Gaussian distributions, with $a_3 = 1.2723 \pm 0.0023$, determined via analysis of measurements of neutron β decays [83], and $a_8 = 0.585 \pm 0.023$ based on hyperon β decay results [84]. $\Delta_s^{(p)} = -0.09 \pm 0.03$, based on a a measurement of the spin-dependent structure function of the deuteron from the COMPASS fixed target experiment [85]. For all the nuclear parameter likelihoods, the central values and widths of the Gaussians can be adjusted with the YAML options *param*_obs and *param*_obserr respectively, where *param* refers to the parameter name.

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) [68, 69], LUX (run 2) [70], PandaX [71], XENON100 [65], SuperCDMS [66], PICO-2L [73], PICO-60 [72] and SIMPLE [67]. 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 1.0.0 also includes routines for the proposed xenon and argon phases of the (now split) DARWIN detector [86], in preparation for future support by DDCalc.

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 Sec. 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 A6. 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. Sec. 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

$$\frac{d\mathcal{Q}(E_f, \mathbf{x})}{dE_f} = \frac{1}{N_\chi} \frac{\rho_\chi^2(\mathbf{x})}{m_\chi^2} \sum_i \left\langle \sigma_i v \frac{dN_i}{dE_f} \right\rangle . \tag{38}$$

Here, σ_i is the annihilation cross section into final state i, v the relative velocity of the annihilating DM particle pairs, $\langle ... \rangle$ 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 ρ_χ and its particle mass by m_χ. N_χ depends on the nature of the DM particle, e.g. $N_\chi = 2$ for Majorana fermions, and $N_\chi = 4$ for Dirac fermions. For *decaying* DM, one simply would have to replace $\langle \sigma_i v \rangle \rho_\chi^2 / N_f \to m_\chi \Gamma_i \rho_\chi$ in the above expression, where Γ_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 $\langle \sigma_i v \; dN_i/dE_f \rangle = \langle \sigma_i v \rangle \, dN_i/dE_f$ (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 [87] for every model point constitutes a significant gain in performance.

The DM density enters quadratically into Eq. (38). 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

$$\rho^2_\chi(\mathbf{x}) = [1 + B_\chi(\mathbf{x})]\rho^2_{\chi,\mathrm{nosub}}(\mathbf{x}) \ , \tag{39}$$

where $\rho^2_{\chi,\mathrm{nosub}}(\mathbf{x})$ is the DM distribution smoothed over substructures (describing the general distribution of DM in the main halo), whereas $B_\chi(\mathbf{x})$ is the boost factor that parameterises 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 [88]. 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 [89].

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 [90–92], while cosmic-ray positron data strongly constrain leptonic channels [93]. 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. [94] 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 ($\pi^0 \to \gamma\gamma$), 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 ($\chi\chi \to \gamma\gamma$), which gives rise to mono-energetic gamma rays [95]. Virtual internal bremsstrahlung [96] or box-like features from cascade-decays [97] 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 Sec. 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. 39. 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 Sec. 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 $\mathcal{O}(1)$ for reasonable assumptions [98]. 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 indrect 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 [99–102]. 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 [103].

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.[6]

At present, neutrino telescope limits on the total DM annihilation cross-section are weak [110, 111], 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 [101]

$$C(t) = 4\pi \int_0^{R_\odot} r^2 \int_0^\infty \frac{f_\odot(u)}{u} w \Omega_v^-(w)\, \mathrm{d}u\, \mathrm{d}r, \tag{40}$$

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 $\Omega_v^-(w)$ 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 $w = w(u, r, t) \equiv \sqrt{u^2 + v(r,t)^2}$ is the velocity an infalling DM particle obtains by the time it collides with a nucleus at height r. $\Omega_v^-(w)$ thus describes the local capture rate at height r in the Sun, from the part of the velocity distribution corresponding to incoming velocity u. The total population N_χ of DM in the Sun can then be determined at any point in its lifetime by solving the differential equation

$$\frac{dN_\chi(t)}{dt} = C(t) - A(t) - E(t), \tag{41}$$

where A and E are the annihilation and evaporation rates, respectively. Except where DM is lighter than a few GeV, E is generally negligible [112–114]. Assuming $E = 0$, and that C and A are constant, the solution to Eq. 41 approaches a steady state on characteristic timescale t_χ, 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 DarkBit we instead solve Eq. 41 and determine N_χ explicitly for each each model.

Knowing the DM population (and therefore annihilation rate) by solving Eq. 41, 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 [103, 115]. The resulting neutrino yield tables at Earth are included in DarkSUSY [15], PPPC4DMID [39] and micrOMEGAs [16]. In DarkBit, the canonical method to obtain these fluxes is to compute the capture and annihilation rates using DarkSUSY, and to then obtain neutrino yields at Earth from the WimpSim [116] tables contained therein (although getting the same from PPPC4DMID or 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 ∼100 GeV [17, 117], Super Kamiokande (Super-K) dominating at lower masses [118], and ANTARES and Baksan providing weaker constraints. We implement the IceCube search likelihood on a model-by-model basis, using the nulike package [17, 119] to compute the likelihood function for each

[6] 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 [104–109]; these are slated for inclusion in future releases of DarkBit.

model, given its predicted neutrino spectrum at Earth. 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 [120]. 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 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 GamLike. GamLike is released under the terms of the MIT license[7] and maintained as a standalone backend code by the GAMBIT Dark Matter Workgroup. It can be obtained from `http://gamlike.hepforge.org`.

The present version of GamLike ships with the likelihood functions discussed in Sec. 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 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

$$\frac{d\Phi}{dE} = \frac{\sigma v}{8\pi\, m_\chi^2}\frac{dN_\gamma}{dE}\,, \tag{42}$$

which is the differential version of Eq. (44). The integration over energy bins happens within GamLike according to the energy bins used in the various analyses. Eq. (42) 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 Sec. 6.1.

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 a marginalise or profile over J-factor uncertainties. The implementation of the combined dwarf limits from Ref. [122], 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. [122], 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. [122] are available as tabulated binned Poisson likelihoods.[8] The composite likelihood from the dSph analysis is given by

$$\ln\mathcal{L}_{\rm exp} = \sum_{k=1}^{N_{\rm dSph}}\sum_{i=1}^{N_{\rm ebin}} \ln\mathcal{L}_{ki}(\Phi_i\cdot J_k)\,, \tag{43}$$

where $N_{\rm dSph}$ and $N_{\rm ebin}$ are the number of considered dSphs and the number of energy bins, respectively. The partial likelihoods $\mathcal{L}_{ki}$ depend on the predicted signal flux $\Phi_i\cdot J_k$. It is a product of the particle physics factor

$$\Phi_i = \frac{(\sigma v)_0}{8\pi m_\chi^2}\int_{\Delta E_i} dE\,\frac{dN_\gamma}{dE}\,, \tag{44}$$

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

$$J_k = \int_{\Delta\Omega_k} d\Omega \int_{\rm l.o.s.} ds\,\rho_\chi^2\,. \tag{45}$$

Here, ΔE_i and $\Delta\Omega_k$ denote the energy bin i and solid angle k over which the signal is integrated, m_χ is the mass of dark matter particles and ρ_χ 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. [122, 127], to a good approximation the corresponding uncertainties for the

[7] http://opensource.org/licenses/MIT

[8] `https://www-glast.stanford.edu/pub_data/1048/`

Instrument	Target(s)	Notes	Energy range [GeV]	Reference
Fermi-LAT	15 dSphs pass7	composite likelihood; J-factor profiling	0.5 GeV – 500 GeV	[121]
Fermi-LAT	15 dSphs pass8	composite likelihood; J-factor profiling	0.5 GeV – 500 GeV	[122]
H.E.S.S.	Galactic Halo	single E-bin; J-factor fixed; NFW	265 GeV – 30 TeV	[123]
H.E.S.S.	Galactic Halo	single E-bin; J-factor from halo model	265 GeV – 30 TeV	[123]
H.E.S.S.	Galactic Halo	multiple E-bins; J-factor fixed; NFW	230 GeV – 30 TeV	[123]
H.E.S.S.	Galactic Halo	multiple E-bins; J-factor from halo model	230 GeV – 30 TeV	[123]
Fermi-LAT	GCE	J-factor fixed; contr. NFW	0.3 GeV – 500 GeV	[124]
Fermi-LAT	GCE	J-factor marginalised	0.3 GeV – 500 GeV	[124, 125]
Fermi-LAT	GCE	J-factor marginalised + 10% HEP syst.	0.3 GeV – 500 GeV	[124, 125]
Fermi-LAT	GCE	J-factor from halo model	0.3 GeV – 500 GeV	[124]
CTA	Galactic Halo	Morphological analsiys; Einasto	25 GeV – 10 TeV	[126]
CTA	Galactic Halo	Morphological analsiys; J-factor from halo model	25 GeV – 10 TeV	[126]

Table 7: Likelihoods and J-factor treatment currently implemented in GamLike, for dwarf spheroidals, the Galactic halo, the *Fermi* Galactic centre GeV excess (GCE). J-factors are either calculated for the halo model employed in the scan ("J-factor from halo model") or derived based on the dark matter profiles in the respective references (usually regular NFW profiles, and a contracted NFW in the case of the GCE). The CTA likelihood is a future projection.

J-factors can be modelled by a log-normal distribution,

$$\ln \mathcal{L}_J = \sum_{k=1}^{N_{\rm dSph}} \ln \mathcal{N}(\log_{10} J_k | \log_{10} \hat{J}_k, \sigma_k) \,, \tag{46}$$

with parameters taken from Ref. [122], and $\mathcal{N}(x|\mu,\sigma)$ being a normal distribution with mean μ and standard deviation σ.

The halo and gamma-ray likelihoods can be combined by profiling over the nuisance parameters J_k. The corresponding profile likelihood is given by

$$\ln \mathcal{L}^{\rm prof.}_{\rm dwarfs}(\Phi_i) = \max_{J_1 \ldots J_k} (\ln \mathcal{L}_{\rm exp} + \ln \mathcal{L}_J) \,. \tag{47}$$

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

$$\mathcal{L}^{\rm marg.}_{\rm dwarfs}(\Phi_i) = \int dJ_1 \ldots dJ_k \, \mathcal{L}_{\rm exp} \cdot \mathcal{L}_J \,. \tag{48}$$

The main results from Ref. [122] are derived using the composite profile likelihood for 15 dwarfs, and this is what we implemented in GamLike. Furthermore, for comparison, we also implemented the older results from Ref. [121], 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 *Fermi*-LAT line search results is high on the priority list for future versions of GamLike (see Sec. 8).

The '*Fermi* Galactic centre excess'

Gamma-ray observations of the Galactic centre with the *Fermi*-LAT identified an extended excess emission at GeV energies, which can be interpreted as a dark matter annihilation signal (see e.g. [124, 128–133]). Although the case for millisecond pulsars as explanation for the excess emission was strengthened in recent analyses [134, 135], it remains interesting to consider the consequences of various DM explanations.

Ref. [124] 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

$$\ln \mathcal{L}_{\rm GC} = -\frac{1}{2} \sum_{ij} (J_{\rm GC} \Phi_i - f_i) \Sigma^{-1}_{ij} (J_{\rm GC} \Phi_j - f_j) \,, \tag{49}$$

where f_i denotes the measured flux in energy bin i, Σ_{ij} is the covariance matrix, and $J_{\rm GC}$ 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 Sec. 3). Alternatively, one can choose a fixed J-factor, $J = 2.07 \times 10^{23}\,{\rm GeV^2 cm^{-5}}$, corresponding to a contracted NFW profile with inner slope $\gamma = 1.2$ (see Ref. [124] for details), or derive a marginalised likelihood function assuming a log-normal distribution of J-factors with mean $1.96 \times 10^{23}\,{\rm GeV^2 cm^{-5}}$ and standard deviation of 0.37 (as done in Ref. [125], motivated by Ref. [136]).

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. [125], 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. [123], 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,

$$\ln \mathcal{L}_{\rm HESS} = -\frac{1}{2}\sum_{i=1}^{N_{\rm ebins}} \frac{(\Phi_i(J_{\rm sig} - J_{\rm bg}R) - f_i^{\rm sig} - f_i^{\rm bg}R)^2}{\Delta f_i^2} \tag{50}$$

where $J_{\rm sig(bg)}$ denotes the J-factors in the signal (background) regions, $f_i^{\rm sig(bg)}$ the corresponding fluxes in energy bin i, and $R \equiv \Omega_{\rm sig}/\Omega_{\rm bg}$ is a geometrical rescaling factor that depends on the angular size $\Omega_{\rm sig(bg)}$ 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. [123] (local dark matter density of $0.39\,{\rm GeV\,cm^{-3}}$ at 8.5 kpc distance from the Sun).

In Ref. [123] 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. [123] 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 cases where the DM signal has a pronounced spectral structure (like the annihilation spectrum of τ-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. [126], 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. [126] 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 [137].

The corresponding likelihood function takes essentially the form of Eq. (43) above, and covers energies between 25 GeV and 10 TeV. As for the Galactic centre and H.E.S.S. likelihoods, the default behaviour is to calculate the J-factor based on the halo model defined in the scan. Alternatively, one can also choose a fixed Einasto profile with local density $\rho_\chi = 0.4\,{\rm GeV\,cm^{-3}}$. Further details about the adopted detector response, Galactic diffuse emission model and DM profile can be found in [126].

6.2.3 Library Interface (API)

The GamLike library interface is the same for all implemented experiments. There are three 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 $\ln\mathcal{L}$ for `experiment_tag`, given the tabulated $d\Phi/dE$ as in Eq. (42)
- `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 A1) 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 Sec. 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
      - $\bar{b}b$: `genRate`
      - $\mu^+\mu^-\gamma$: `genRate`
      - ...
    - `genRateMisc`
  - Initial state: h_2^0
    - ...
  - ...
- Particle properties
  - χ: mass, spin
  - h_2^0: 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 Sec. 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 Sec. B for details). It provides information about the decay width or the annihilation cross section of the described process.

`genRate`	Units	Parameters	Process
Γ	GeV^{-1}	–	2-body decay
$\frac{d^2\Gamma}{dEdE_1}$	GeV^{-3}	`"E","E1"`	3-body decay
$\langle\sigma v\rangle$	$\mathrm{cm}^3\,\mathrm{s}^{-1}$	`"v"`	2-body ann.
$\frac{d\langle\sigma v\rangle}{dEdE_1}$	$\mathrm{cm}^3\,\mathrm{s}^{-1}\,\mathrm{GeV}^{-2}$	`"E","E1","v"`	3-body ann.

Table 8: Overview of the various definitions of the generalised rate `genRate`, for different possible processes in the Process Catalog. Note that `genRate` is a `daFunk::Funk` object with the indicated parameters. See main text for details.

In the case of two-body decay, `genRate` is simply a constant that equals the decay width Γ 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 `"E1"`, 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"` (as in the two-body case).

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` are summarised in Table 8. Note that `genRateMisc` enters the calculation of W_{eff} from the Process Catalogue (if this how the user has chosen to obtain W_{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 [96].) In gen-

eral, 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.[9]

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 (Sec. 6.3.2) or via DarkBit's own Fast Cascade Monte Carlo (FCMC; Sec. 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 [141]. 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[10], but can easily 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 A7. 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 (Sec. 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. (42) up to a factor of 8π . 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 evalulations; 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

[9] 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 [39]. In general, however, all these contributions are highly model-dependent and can differ substantially from the 'model-independent' results [138–140]. For $\bar{q}qg$ final states, for example, the change in the normalisation of $\bar{q}q$ final states due to loop corrections at the same order in α_s must also be included for consistency [138]. 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 [139, 140]. 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 assocaited two-body channels).

[10] 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 dark matter mass is limited to values below 5 TeV both for DarkSUSY and micrOMEGAs.

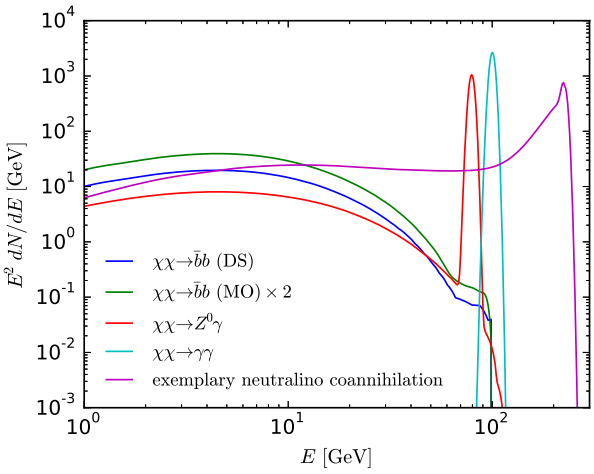

Fig. 3: Example spectra generated with DarkBit, using tabulated 2-body final states from DarkSUSY and micrOMEGAs, line-like spectra assuming an energy resolution of 3%, and a benchmark case from MSSM neutralino annihilation in the coannihilation region. The DM mass is $m_\chi = 226\,\text{GeV}$ (neutralino) or $m_\chi = 100\,\text{GeV}$ (all other cases).

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 Sec. 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 γ and the gamma-ray yield from decay of a single Z^0. We use tabulated spectra from on-shell Z^0 decays, in order to avoid unphysical super-threshold photons that would occur when constructing single Z^0 spectra from tabulated Z^0Z^0 yields.

Gamma-ray likelihood functions in DarkBit make use of the backend GamLike (see Sec. 6.2 and Table 7). As detailed in Table A8, 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:

- `lnL_FermiLATdwarfs` (`version` = `"pass7"`, `"pass8"`),
- `lnL_FermiGC` (`version` = `"fixedJ"`, `"margJ"`, `"margJ_HEP"`, `"externalJ"`),
- `lnL_HESSGC` (`version` = `"integral_fixedJ"`, `"spectral_fixedJ"`, `"integral_externalJ"`, `"spectral_externalJ"`),
- `lnL_CTAGC` (`version` = `"fixedJ"`, `"externalJ"`).

6.3.3 Neutrinos

The different neutrino indirect detection capabilities of DarkBit are summarised in Table A9. The capabilities that describe the relevant WIMP properties are listed in Table A2.

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 [142, 143], 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. 41 for the equilibration time t_χ and annihilation rate (capabilities `equilibration_time_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`).

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 [120]: 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 [119, 144] 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 A10.

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

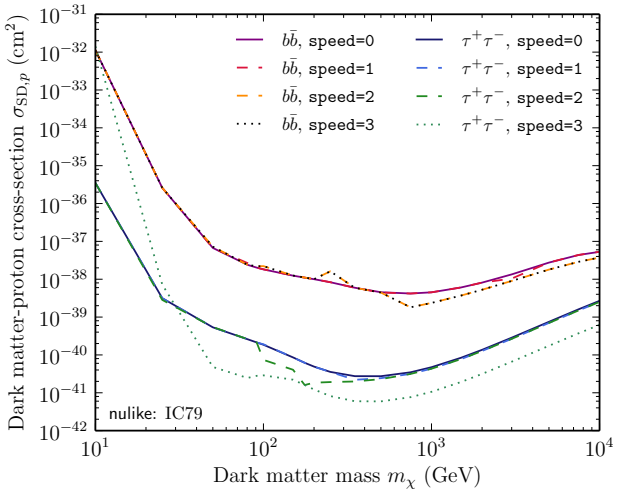

Fig. 4: Comparison of different limits obtained from 79-string IceCube data using nulike, with different `speed` settings. The default is 3, but the `speed` setting is configurable from the master YAML file via the module function option `nulike_speed`.

log-likelihood of the background-only model.[11] 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 CL_s method [145, 146].

The nulike `speed` parameter can be chosen from the master initialisation file, by setting the module function option `nulike_speed`. The default is `3`. For production scans, we run nulike with this default, as this allows OpenMP-enabled neutrino likelihood evaluations for models with appreciable signal fractions to be achieved in walltimes of order one second. Parameter combinations resulting in negligible signal fractions run much faster, so the mean runtime is well below a second. This does come with an accuracy cost; Fig. 4 compares the accuracy of some example limits obtained with the different speed settings.

[11] 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 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 A10). For $X \in \{$`IC22`, `IC79WH`, `IC79WL`, `IC79SL`$\}$, these are:

- X`_signal` The predicted number of signal events (from DM annihilation in the Sun).
- X`_bg` The predicted number of background events.
- X`_nobs` The total number of observed events.
- X`_bgloglike` The likelihood for the background-only model.
- X`_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`_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 Sec. 6.3.2, it is the responsibility of each module function with capability `GA_missingFinalStates` to deterime, 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.[12]

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[13] 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 A11 and A12. 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` → `cascadeMC_ChainEvent` → `cascadeMC_Histograms` → `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`.

Various options are available to tune the FCMC. We list them here, with default values in square brackets.

`cascadeMC_LoopManagement`:
- `int cMC_maxEvents[20000]`: sets the maximum number of MC runs per point.

`cascadeMC_GenerateChain`:
- `int cMC_maxChainLength[-1]`: the maximum number of decay steps to consider. A value of `-1` indicates that no maximum should be applied.
- `double cMC_Emin[0.0]`: the minimum lab-frame energy of particles to be tracked, in GeV.

`cascadeMC_Histograms`:

Histogramming options:
- `int cMC_NhistBins[140]`: number of logarithmic bins of the generated histogram.
- `double cMC_binLow[0.001]`: the lowest energy in GeV of the generated histogram.
- `double cMC_binHigh[10000]`: the highest energy in GeV of the generated histogram.
- `int cMC_numSpecSamples[25]`: the number of samples to draw from tabulated spectra.

Convergence criteria:
- `double cMC_gammaRelError[0.20]`: the maximum allowed relative error in the bin with the highest expected signal-to-background ratio.
- `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.
- `int cMC_endCheckFrequency[25]`: the number of events to wait between successive checks of the convergence criteria.

Note that if `cMC_maxEvents` is exceeded in `cascadeMC_LoopManagement` before convergence is reached in `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 which were set up using the DarkBit WIMP standalone discussed in Sec. 7.2. Specifically, we set up a pair of DM particles annihilating to two scalars, $\chi\chi \to \phi_1\phi_2$, where ϕ_1 decays to a pair of photons and ϕ_2 to $\bar{b}b$. In the rest frame of ϕ_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. [97])

$$\frac{dN_{\phi\to\gamma\gamma}}{dE_\gamma} = \frac{2}{\Delta E}\theta(E_\gamma - E_{\rm min})\theta(E_{\rm max} - E_\gamma)\,. \tag{51}$$

[12] Relative, as the branching ratios of the two-body decays may not always sum up to one.

[13] 'Lab frame' means the rest frame of the initial state in the cascade.

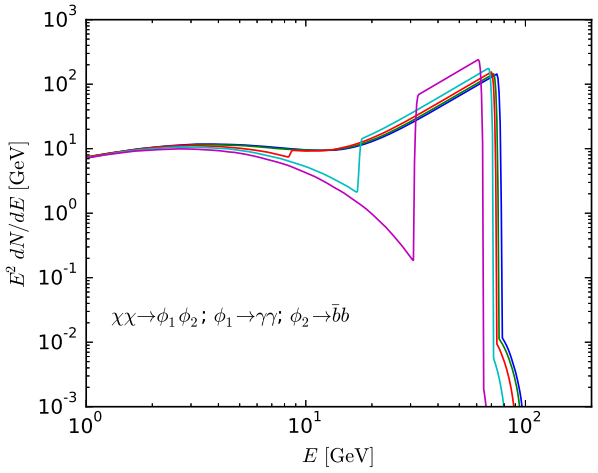

Fig. 5: Example spectra generated with the DarkBit FCMC, for $m_\chi = 100\,\text{GeV}$, $m_{\phi_2} = 100\,\text{GeV}$ and $m_{\phi_1} = 10, 30, 50, 70, 90\,\text{GeV}$.

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

$$E_{\text{max,min}} = (E_{\phi_1}/2)\left(1 \pm \sqrt{1 - m_{\phi_1}^2/E_{\phi_1}^2}\right), \tag{52}$$

where

$$E_{\phi_1} = m_\chi\left[1 + (m_{\phi_1}^2 - m_{\phi_2}^2)/(4m_\chi^2)\right] \tag{53}$$

is the energy of ϕ_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 $\phi_2 \to \bar{b}b$ produces a continuum spectrum of photons from the tabulated yields of $\bar{b}b$ final states. Compared to the direct annihilation of DM to $\bar{b}b$, as in $\chi\chi \to \bar{b}b$, 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 [147] – 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 ϕ_1 (for constant $m_{\phi_2} = m_\chi = 100\,\text{GeV}$). The reason is that the largest photon energy kinematically available from $\phi_2 \to \bar{b}b$ is given by E_{max} as provided in the expression after Eq. (51), with ϕ_1 and ϕ_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.

Model	Parameter	Value
Singlet DM	λ_{hS}	0.03
	m_S [GeV]	90
CMSSM	M_0 [GeV]	3075
	$M_{1/2}$ [GeV]	465
	$\tan\beta$	51
	A_0 [GeV]	1725
	sign(μ)	+
MSSM 7	M_2 [GeV]	690
	$m_{H_d}^2$ [GeV^2]	9.86×10^7
	$m_{H_u}^2$ [GeV^2]	1.4×10^4
	$\tan\beta$	23
	m_f^2 [GeV^2]	3.8×10^6
	A_d [GeV]	1000
	A_u [GeV]	2680
	sign(μ)	+

Table 9: Central points of spokes plotted in Fig. 6. All of the parameters of the MSSM 7 are defined at an energy scale of 1 TeV.

Model	Parameter	Range
Singlet DM	λ_{hS}	[0.01, 0.05]
	m_S [GeV]	[70, 110]
CMSSM	M_0 [GeV]	[2840, 3310]
	$M_{1/2}$ [GeV]	[330, 600]
MSSM 7	M_2 [GeV]	[450, 850]
	$m_{H_d}^2$ [GeV^2]	[9.2×10^7, 1.03×10^8]

Table 10: Ranges that parameters are varied over in Fig. 6. All of the parameters of the MSSM 7 are defined at an energy scale of 1 TeV.

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 [148], MSSM7 [149] and scalar singlet DM [150] 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).

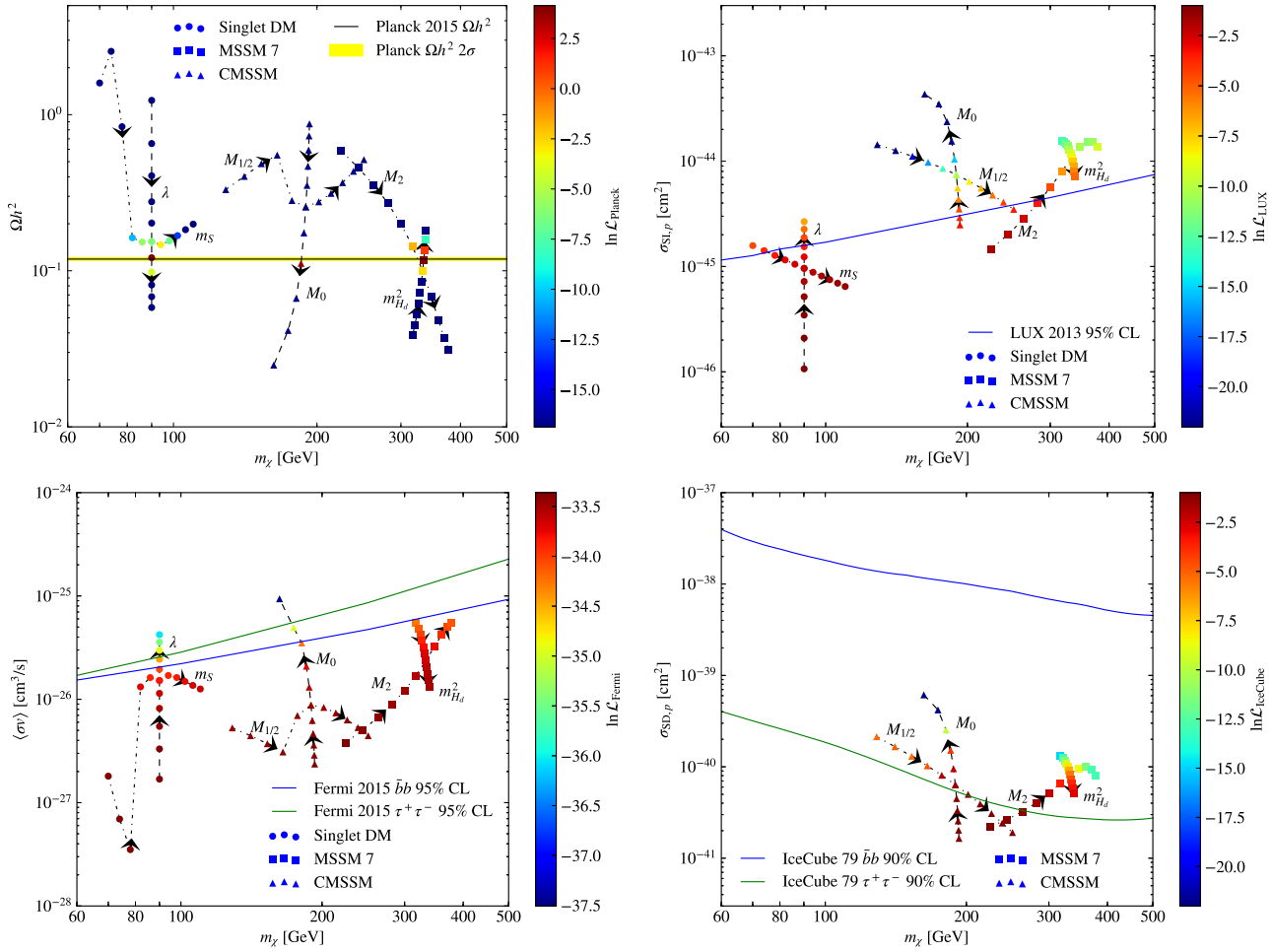

Fig. 6: Tracks in observable space corresponding to spokes in model parameter space. The arrows point in the direction of increasing parameter values. Clockwise from the top-let, panels show the DM relic density, spin-independent and spin-dependent DM-proton scattering cross sections, and the velocity-averaged 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 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 Ωh^2 and its 2σ (experimental) uncertainty.

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 (Sec. 5.2), the IceCube 79-string likelihood for WIMP annihilation in the Sun from nulike (Sec. 6.3.3), and the the stacked dwarf spheroidal likelihood based on six years of *Fermi* data from GamLike (Sec. 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 Sec. 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 [68], IceCube [17], and *Fermi* [122] collaborations. For the relic density, we plot the Planck best fit value and 2 σ uncertainty on it. 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,

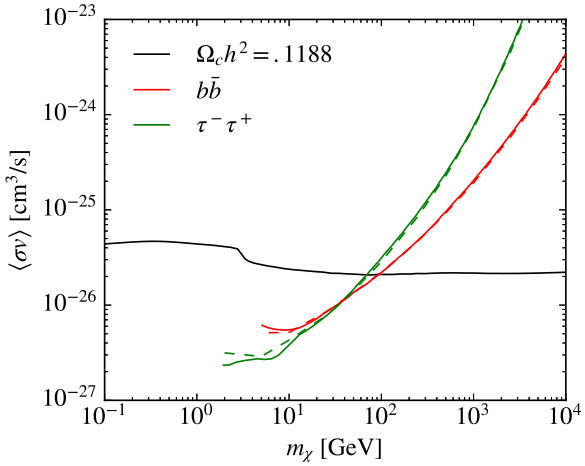

Fig. 7: Simple WIMP results obtained with DarkBit. The solid lines represent 95% CL upper limits on $b\bar{b}$ and $\tau^+\tau^-$ final states from *Fermi* `pass 8` observations of dwarf spheroidal galaxies as calculated by DarkBit, while the dashed lines represent the corresponding limits reported by the *Fermi* collaboration [122]. The solid black line represents the values of σv that reproduce (for s-wave annihilation) a DM density of $\Omega_c h^2 = 0.1188$.

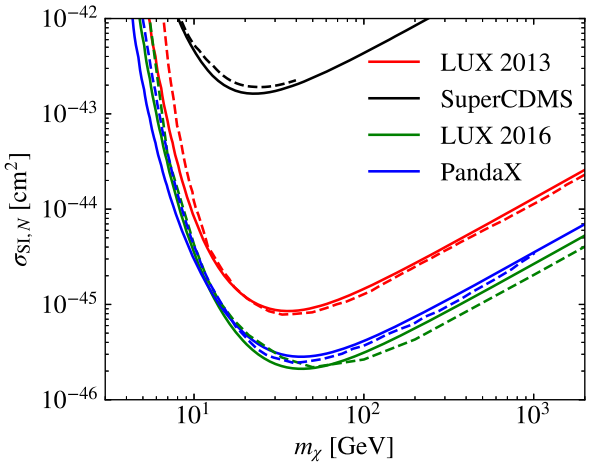

Fig. 8: Limits on the spin independent DM-nucleon scattering cross section from SuperCDMS, LUX, and PandaX at 90% CL. The solid curves are the limit determined using DarkBit and the dashed curves are the official limits from the collaborations [66, 68, 69, 71].

`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.

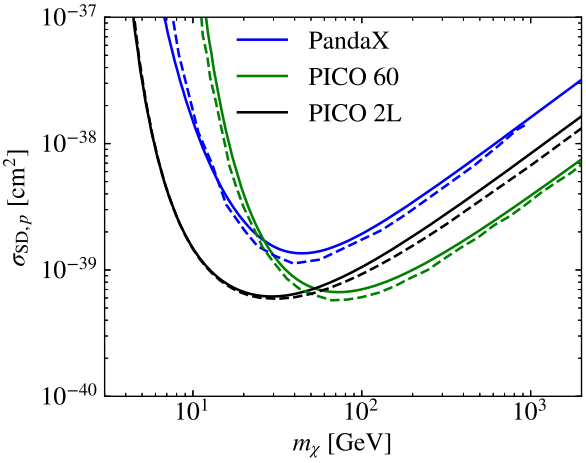

Fig. 9: Limits on the spin dependent DM-proton scattering cross section from PICO-2L, PICO-60L, and PandaX at 90% CL. The solid curves are the limits determined using DarkBit and the dashed curve are the official limits from the collaborations [72, 73, 151].

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.

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, one can use the standalone example to calculate tables for *Fermi*-LAT, spin-independent, and spin-dependent direct detection likelihoods as functions of the dark matter mass and the annihilation or scattering cross section; we show the corresponding upper limits in Figs. 7–9 respectively. These are at 95% CL (obtained at $\Delta 2\ln\mathcal{L} = 3.84$) in Fig. 7, as is customary for indirect searches, and at 90% CL ($\Delta 2\ln\mathcal{L} = 2.71$) in Figs. 8 and 9, as is typical in direct detection. 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 $\Omega_\chi h^2 = 0.1188$, 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 [152]. 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 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 likelihoods 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.

If `DarkBit_standalone_MSSM` is given an SLHA1 file it calculates observables using both DarkSUSY and micrOMEGAs when possible; however, if given an SLHA2 file the program only uses DarkSUSY, as micrOMEGAs is currently incompatible with SLHA2. If present, information in the SLHA `DECAY` block is passed to the backends, otherwise decay widths and branching ratios are determined by the backends using their default methods (note that for DarkSUSY the `DECAY` block contents are only used when the input file is in SLHA2 format).

As a demonstration of the sorts of comparisons possible, we have chosen sample 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`. Results of the calculations can be seen in Table 12, demonstrating some interesting differences worthy of a dedicated future study.

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 nulike).

The present treatment of thermal freeze-out cannot easily handle semi-annihilation [153] nor asymmetric DM scenarios [154]. Another natural extension would be 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 [155, 156].

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 [157], for example. Helio- and astroseismological probes of DM-nucleon couplings (see e.g. [109]) 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 [93], and constraints from antiprotons can likely be improved considerably [90, 91]. 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 [158, 159]. These can be relevant for e.g. DM annihilation in subhalos [160], or close to the black hole at the Galactic centre [161, 162].

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

Model	Description	M_2 [GeV]	$m^2_{H_d}$ [10^6 GeV^2]	$m^2_{H_u}$ [10^6 GeV^2]	$\tan\beta$	$m^2_{\tilde{f}}$ [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	$\tilde{\tau}$ 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 11: MSSM-7 points used as benchmarks for comparisons between DarkSUSY and micrOMEGAs. The sign of μ 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	Ωh^2		$\sigma_{\mathrm{SI},p}$ [10^{-46}cm^2]		$\sigma_{\mathrm{SD},p}$ [10^{-43}cm^2]	
	DarkSUSY	micrOMEGAs	DarkSUSY	micrOMEGAs	DarkSUSY	micrOMEGAs
1	0.1396	0.08322	9.815	10.88	1.335	1.263
2	0.01479	0.007722	166.5	186.6	61.14	58.01
3	0.05507	0.03119	189.4	211.1	32.89	31.21
4	0.002217	0.001421	25.54	26.29	5270	5001
5	0.1103	0.1093	0.4713	7.011	1.118	0.06170
6	0.02212	0.02272	3.181	3.745	0.09488	0.8305
7	0.004588	0.003339	189.1	218.7	46.74	39.00
8	0.005815	0.004286	230.5	273.9	62.08	52.61
9	0.003000	0.003148	8.757	8.877	191.6	181.8

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. Note that information is passed to DarkSUSY in SLHA2 format, whereas micrOMEGAs accepts only SLHA1 information.

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 [75], 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 nulike 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 Research Council (621-2014-5772), the Australian Research Council (CE110001004, FT130100018, FT140100244, FT160100274), The University of Sydney (Australia; IRCA-G162448), PLGrid Infrastructure (Poland), Polish National Science Center (Sonata UMO-2015/17/D/ST2/03532), the Swiss National Science Foundation (PP00P2-144674), 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).

Appendix A: Getting started

As described in Sec. 2, DarkBit is a standalone and complimentary module of the GAMBIT software, which can be downloaded from the official GAMBIT website[14]. 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.

A.1: Content of DarkBit download & 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[15]. 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:

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

[14] gambit.hepforge.org
[15] 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`.

A.2: Running the example program

To demonstrate how DarkBit can be used in a fully-fledged scan, we show a fully annotated example of a DarkBit `yaml` file in `gambit/gambit_DarkBit_Example.yaml`. The file demonstrates how to specify the CMSSM model parameters and priors, choose and configure a sampler, choose a printer (either hdf5 or ASCII), and 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 example requires the micrOMEGAs, SuperIso, 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

```
make micromegas
make superiso
make gamlike
make ddcalc
make darksusy
make nulike
make feynhiggs
make susyhit
```

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

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

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:

```
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); //f(x,y) ≡ 2x+3cos(y)
```

The name of a variable is specified as a `std::string`, and `daFunk::Funk` objects can be combined into new `daFunk::Funk` objects using normal arithmetic or common (appropriately overloaded) `std::math` operations. In the above example, `x` is a function of variable list `["x"]`, `y` is a function of the variable list `["y"]`, `f` is a function of the variable list `["x", "y"]`, and `c` is a function of the variable list `[]`.

For performance reasons, the evaluation of `daFunk::Funk` objects is split into two steps. First, the positions of the variables are 'bound', using the `bind` member function. This generates an object of the type `daFunk::FunkBound`, which can then be evaluated using the `eval` member function. This is shown in the following example:

```
  // bind variable positions
  daFunk::FunkBound fb = f->bind("y", "x");
  // return f(4,3) = 2×4+3cos(3)
  fb->eval(3., 4.);
```

This two-step procedure separates the overhead related to the dynamical construction of nested functions with tagged variables (which in general includes string matching, various consistency checks and needs to be done only once) from the possibly large number of actual function evaluations.

A more complex situation that involves (1) the wrapping of plain C functions into daFunk objects, (2) the wrapping of daFunk objects in plain C functions, and (3) variable substitution, is shown in the following example:

```
// Declaration of a plain C function
double dNdE(double E, double m, double v);

// Define traits class for daFunk function pointer
DEF_FUNKTRAIT(T)

// daFunk variables
int main()
{
  daFunk::Funk v = daFunk::var("v");
  daFunk::Funk E = daFunk::var("E");
  daFunk::Funk m = daFunk::var("m");
  daFunk::Funk Ecm = daFunk::var("Ecm");

  // Wrap plain C function
  daFunk::Funk f = daFunk::func(dNdE, E, m, v);

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

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

  // Returns ann. yield for E_cm = 100 at E = 10
  double Ecm_d = 100;
  double E_d = 10;
  std::cout << (*h)(Ecm_d, E_d) << std::endl;
}
```

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 $m = E_{\rm cm}/2$. 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.

```
daFunk::Funk x = var("x");  // variable x
daFunk::Funk a = var("a");  // variable a
daFunk::Funk f = x*x;       // f(x) = x^2

// g(a) ≡ ∫_0^a f(x) dx = ∫_0^a x^2 dx
daFunk::Funk g = f->gsl_integration("x", 0., "a");
daFunk::FunkBound gb = g->bind("a");
// print g(5/2) to stdout
std::cout << gb->eval(2.5) << std::endl;
```

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

Here we explain some terms that have specific technical definitions in GAMBIT.

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 **back-**

end 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 A1), some simple informative capabilities (Table A2), capabilities related to DM halo properties (Table A3), relic density capabilities (Tables A4 and A5), direct detection capabilities (Table A6), gamma-ray yield capabilities (Table A7), gamma-ray likelihoods (Table A8), neutrino capabilities (Tables A9 and A10), cascade decay capabilities (Tables A11 and A12), and various miscellaneous capabilities (Table A13).

References

1. Planck Collaboration, *Planck 2015 results. XIII. Cosmological parameters*, 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** (2009) 557–588, [arXiv:0811.3347].
4. J. L. Feng, *Dark Matter Candidates from Particle Physics and Methods of Detection*, *ARA&A* **48** (2010) 495–545, [arXiv:1003.0904].
5. H. Baer, K.-Y. Choi, J. E. Kim, and L. Roszkowski, *Dark matter production in the early Universe: beyond the thermal WIMP paradigm*, *Phys. Rept.* **555** (2015) 1–60, [arXiv:1407.0017].
6. G. Jungman, M. Kamionkowski, and K. Griest, *Supersymmetric dark matter*, *Phys.Rept.* **267** (1996) 195–373, [hep-ph/9506380].
7. D. Hooper and S. Profumo, *Dark matter and collider phenomenology of universal extra dimensions*, *Phys. Rep.* **453** (2007) 29–115, [hep-ph/0701197].
8. S. Tulin, H.-B. Yu, and K. M. Zurek, *Beyond Collisionless Dark Matter: Particle Physics Dynamics for Dark Matter Halo Structure*, *Phys. Rev. D* **87** (2013) 115007, [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.

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 MSSM_spectrum decay_rates DarkMatter_ID	setMassesForIB dssigmav dsIBffdxdy dsIBhhdxdy dsIBwhdxdy dsIBwwdxdy IBintvars	ignore_three_body (bool) ProcessCatalog_MinBranching(double)
	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	dsddgpgn mspctm ddcom	rescale_couplings (double)
	DD_couplings_MicrOmegas (DM_nucleon_couplings): Determine the WIMP mass and couplings using micrOMEGAs.		nucleonAmplitudes FeScLoop MOcommon	
	DD_couplings_SingletDM (DM_nucleon_couplings): Determine the WIMP mass and couplings for scalar singlet DM.	SingletDM_spectrum		

Table A1: Central DarkBit capabilities that store details about annihilation and scattering processes.

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_late_universe (double): Retrieve the total thermally-averaged annihilation cross-section for indirect detection ($\mathrm{cm}^3\,\mathrm{s}^{-1}$), 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 A2: 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.

10. GAMBIT Collaboration: P. Athron, C. Balázs, *et. al.*, *GAMBIT: The Global and Modular Beyond-the-Standard-Model Inference Tool*, *Eur. Phys. J. C submitted* (2017) [arXiv:1703.xxxxx].
11. GAMBIT Collider Workgroup: C. Balázs, A. Buckley, *et. al.*, *ColliderBit: A GAMBIT module for the calculation of high energy collider observables and likelihoods*, *Eur. Phys. J. C submitted* (2017) [arXiv:1703.xxxxx].
12. GAMBIT Flavour Workgroup: F. U. Bernlochner, M. Chrząszcz, *et. al.*, *FlavBit: A GAMBIT module for computing flavour observables and likelihoods*, *Eur. Phys. J. C, to be submitted* (2017) [arXiv:1703.xxxxx].
13. GAMBIT Models Workgroup: P. Athron, C. Balazs, *et. al.*, *SpecBit, DecayBit and PrecisionBit: GAMBIT modules for computing mass spectra, particle decay rates and precision observables*, *Eur. Phys. J. C submitted* (2017) [arXiv:1703.xxxxx].
14. GAMBIT Scanner Workgroup: G. D. Martinez, J. McKay, *et. al.*, *Comparison of statistical sampling methods with ScannerBit, the GAMBIT scanning module*, *Eur. Phys. J. C submitted* (2017) [arXiv:1703.xxxxx].
15. P. Gondolo, J. Edsjö, *et. al.*, *DarkSUSY: computing supersymmetric dark matter properties numerically*, *JCAP* **7** (2004) 8, [astro-ph/0406204].

Capability	Function (Return Type): Brief Description	Dependencies	Options (Type)
GalacticHalo	GalacticHalo_gNFW (GalacticHaloProperties): Provides the generalised NFW density profile $\rho(r)$ and $r_{\rm sun}$.		
	GalacticHalo_Einasto (GalacticHaloProperties): Provides the Einasto density profile $\rho(r)$ and $r_{\rm sun}$.		
LocalHalo	ExtractLocalMaxwellianHalo (LocalMaxwellianHalo): Provides the local density ρ_0 as well as the velocity parameters v_0, $v_{\rm rot}$ and $v_{\rm esc}$.		
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 A3: Capabilities connected to the Milky Way halo parameters.

16. G. Bélanger, J. Da Silva, T. Perrillat-Bottonet, and A. Pukhov, *Limits on dark matter proton scattering from neutrino telescopes using micrOMEGAs*, *JCAP* **12** (2015) 036, [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** (2016) 022, [arXiv:1601.00653].
18. E. Bertschinger, *Self - similar secondary infall and accretion in an Einstein-de Sitter universe*, *Astrophys. J. Suppl.* **58** (1985) 39.
19. J. F. Navarro, C. S. Frenk, and S. D. M. White, *The Structure of cold dark matter halos*, *Astrophys. J.* **462** (1996) 563–575, [astro-ph/9508025].
20. A. V. Kravtsov, A. A. Klypin, J. S. Bullock, and J. R. Primack, *The Cores of dark matter dominated galaxies: Theory versus observations*, *Astrophys. J.* **502** (1998) 48, [astro-ph/9708176].
21. B. Moore, T. R. Quinn, F. Governato, J. Stadel, and G. Lake, *Cold collapse and the core catastrophe*, *MNRAS* **310** (1999) 1147–1152, [astro-ph/9903164].
22. A. W. Graham, D. Merritt, B. Moore, J. Diemand, and B. Terzic, *Empirical models for Dark Matter Halos. I. Nonparametric Construction of Density Profiles and Comparison with Parametric Models*, *Astron. J.* **132** (2006) 2685–2700, [astro-ph/0509417].
23. U. Haud and J. Einasto, *Galactic models with massive corona I. Method*, *A&A* **223** (1989) 89–94.
24. R. Schoenrich, J. Binney, and W. Dehnen, *Local Kinematics and the Local Standard of Rest*, *MNRAS* **403** (2010) 1829, [arXiv:0912.3693].
25. K. Freese, M. Lisanti, and C. Savage, *Annual Modulation of Dark Matter: A Review*, *Rev.Mod.Phys.* **85** (2013) 1561–1581, [arXiv:1209.3339].
26. A. K. Drukier, K. Freese, and D. N. Spergel, *Detecting Cold Dark Matter Candidates*, *Phys. Rev. D* **33** (1986) 3495–3508.
27. P. D. Serpico and G. Bertone, *Astrophysical limitations to the identification of dark matter: indirect neutrino signals vis-a-vis direct detection recoil rates*, *Phys. Rev. D* **82** (2010) 063505, [arXiv:1006.3268].
28. J. I. Read, *The Local Dark Matter Density*, *J. Phys.* **G41** (2014) 063101, [arXiv:1404.1938].
29. J. Bovy and S. Tremaine, *On the local dark matter density*, *Astrophys. J.* **756** (2012) 89, [arXiv:1205.4033].
30. J. A. R. Caldwell and J. P. Ostriker, *The Mass distribution within our Galaxy: A Three component model*, *Astrophys. J.* **251** (1981) 61–87.
31. R. Catena and P. Ullio, *A novel determination of the local dark matter density*, *JCAP* **1008** (2010) 004, [arXiv:0907.0018].
32. P. Salucci, F. Nesti, G. Gentile, and C. F. Martins, *The dark matter density at the Sun's location*, *A&A* **523** (2010) A83, [arXiv:1003.3101].
33. M. Pato, F. Iocco, and G. Bertone, *Dynamical constraints on the dark matter distribution in the Milky Way*, *JCAP* **1512** (2015) 001, [arXiv:1504.06324].
34. M. Pato, O. Agertz, G. Bertone, B. Moore, and R. Teyssier, *Systematic uncertainties in the determination of the local dark matter density*, *Phys. Rev. D* **82** (2010) 023531, [arXiv:1006.1322].

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 pacodes particle_code	CoannCharginos Neutralinos (bool) CoannSfermions (bool) CoannMaxMass (double)
	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		
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		
RD_eff_annrate_DSprep	RD_annrate_DSprep_func (int): Initializes DarkSUSY to be able to provide effective invariant rate $W_{\rm eff}$.	RD_spectrum	rdmgev	
RD_eff_annrate	RD_eff_annrate_SUSY (fptr_dd): Returns the effective invariant rate $W_{\rm eff}$ as provided by DarkSUSY.	RD_eff_annrate_DSprep	dsanwx	
	RD_eff_annrate_from_ProcessCatalog (fptr_dd): Returns the effective invariant rate $W_{\rm eff}$ as calculated from the information contained in the process catalogue.	TH_ProcessCatalog DarkMatter_ID		

Table A4: General 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 RD_eff_annrate	dsrdthlim dsrdtab dsrdeqn dsrdwintp particle_code widths rdmgev rdpth rdpars rdswitch rdlun rdpadd rddof rderrors	fast (int)
	RD_oh2_DarkSUSY (double): Routine for directly obtaining results from DarkSUSY.	DarkSUSY_PointInit	dsrdomega	omtype (int) fast (int)
	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): Basic Gaussian log-likelihood for the relic density.	RD_oh2		oh2_obs (double) oh2_obserr (double) oh2_fractional_theory_err (double)
	lnL_oh2_upperlimit (double): A half-Gaussian log-likelihood (see Secs. 8.3.3 and 8.3.4 of [10]) for the relic density, treating the measured value as a smeared upper bound.	RD_oh2		oh2_cental (double) oh2_obserr (double) limit_method (str) oh2_fractional_theory_err (double)

Table A5: The main relic density capabilities provided by DarkBit.

Capability	Function (Return Type): Brief Description	Dependencies	Backend requirements	Options (Type)
*X*_Calculate	*X*_Calculate (bool): Perform rate calculations for direct detection analysis *X*.		DD_CalcRates DD_Experiment	
*X*_Events	*X*_Events_DDCalc (int): The number of observed events for direct detection analysis *X*.	*X*_Calculate	DD_Events DD_Experiment	
*X*_Background	*X*_Background_DDCalc (double): The number of background events for direct detection analysis *X*.	*X*_Calculate	DD_Background DD_Experiment	
*X*_Signal	*X*_Signal_DDCalc (double): The number of signal events for direct detection analysis *X*.	*X*_Calculate	DD_Signal DD_Experiment	
*X*_SignalSI	*X*_SignalSI_DDCalc (double): The number of spin-independent signal events for direct detection analysis *X*.	*X*_Calculate	DD_SignalSI DD_Experiment	
*X*_SignalSD	*X*_SignalSD_DDCalc (double): The number of spin-dependent signal events for direct detection analysis *X*.	*X*_Calculate	DD_SignalSD DD_Experiment	
*X*_LogLikelihood	*X*_LogLikelihood_DDCalc (double): Calculate the log-likelihood for direct detection analysis *X*.	*X*_Calculate	DD_LogLikelihood DD_Experiment	
lnL_SI_nuclear_parameters	lnL_sigmas_sigmal (double): The log-likehood for the nuclear parameters relevant for spin-independent scattering.			sigmas_obs (double) sigmas_obserr (double) sigmal_obs (double) sigmal_obserr (double)
lnL_SD_nuclear_parameters	lnL_deltaq (double): The log-likehood for the nuclear parameters relevant for spin-dependent scattering.			a3_obs (double) a3_obserr (double) a8_obs (double) a8_obserr (double) deltas_obs (double) deltas_obserr (double)

Table A6: DarkBit capabilities for direct detection log-likelihoods and related observables. Possible values for *X* are XENON100_2012, LUX_2013, LUX_2015, LUX_2016_prelim, PandaX_2016, SuperCDMS_2014, SIMPLE_2014, PICO_2L, PICO_60_F, PICO_60_I, DARWIN_Ar_2015 and DARWIN_Xe_2015.

Capability	Function (Return Type): Brief Description	Dependencies	Backend requirements	Options (Type)
SimYieldTable	SimYieldTable_DarkSUSY (DarkBit::SimYieldTable): Provides access to tabulated yields in DarkSUSY		dshayield	allow_yield_extrapolation (bool)
	SimYieldTable_MicrOmegas (DarkBit::SimYieldTable): Provides access to tabulated yields in MicrOmegas.		dNdE	allow_yield_extrapolation (bool)
	SimYieldTable_PPPC (DarkBit::SimYieldTable): Provides access to tabulated yields in PPPC.			
GA_missingFinalStates	GA_missingFinalStates (std::vector<std::string>): Determines final states that are not available with tabulated spectra.	TH_ProcessCatalog SimYieldTable DarkMatter_ID		ignore_all (bool) ignore_two_body (bool) ignore_three_body (bool)
GA_AnnYield	GA_AnnYield_General (Funk::Funk): General function for calculating gamma-ray yields from the process catalogue.	TH_ProcessCatalog SimYieldTable DarkMatter_ID cascadeMC_gammaSpectra		line_width (double)

Table A7: General gamma-ray capabilities provided by DarkBit.

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 set_gamLike_GC_halo	GamLike	version (str)
lnL_HESSGC	lnL_HESSGC_gamLike (double): Log-likelihood for the HESS Galactic halo searches, using GamLike.	GA_AnnYield RD_fraction set_gamLike_GC_halo	GamLike	version (str)
lnL_CTAGC	lnL_CTAGC_gamLike (double): Log-likelihood for projected dark matter searches with CTA, using GamLike.	GA_AnnYield RD_fraction set_gamLike_GC_halo	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 $r_{\rm sun}$.			
	GalacticHalo_Einasto (GalacticHaloProperties): Provides the Einasto density profile $\rho(r)$ and $r_{\rm sun}$.			

Table A8: DarkBit capabilities for gamma-ray indirect detection likelihoods.

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 sigma_SI_p sigma_SD_p	cap_Sun_v0q0_isoscalar
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 mwimp sigmav sigma_SI_p sigma_SD_p DarkMatter_ID	nuyield_setup nuyield get_DS_neutral_h_decay_channels get_DS_charged_h_decay_channels

Table A9: General DarkBit capabilities for neutrino indirect detection processes.

35. Y. Akrami, C. Savage, P. Scott, J. Conrad, and J. Edsjö, *How well will ton-scale dark matter direct detection experiments constrain minimal supersymmetry?*, *JCAP* **1104** (2011) 012, [arXiv:1011.4318].
36. M. J. Reid *et. al.*, *Trigonometric Parallaxes of Massive Star Forming Regions: VI. Galactic Structure, Fundamental Parameters and Non-Circular Motions*, *Astrophys. J.* **700** (2009) 137–148, [arXiv:0902.3913].
37. J. Bovy, D. W. Hogg, and H.-W. Rix, *Galactic masers and the Milky Way circular velocity*, *Astrophys. J.* **704** (2009) 1704–1709, [arXiv:0907.5423].
38. M. C. Smith *et. al.*, *The RAVE Survey: Constraining the Local Galactic Escape Speed*, *MNRAS* **379** (2007) 755–772, [astro-ph/0611671].
39. M. Cirelli, G. Corcella, *et. al.*, *PPPC 4 DM ID: a poor particle physicist cookbook for dark matter indirect detection*, *JCAP* **3** (2011) 051, [arXiv:1012.4515].
40. J. Edsjö and P. Gondolo, *Neutralino relic density including coannihilations*, *Phys. Rev. D* **56** (1997) 1879–1894, [hep-ph/9704361].
41. P. Gondolo and G. Gelmini, *Cosmic abundances of stable particles: Improved analysis*, *Nucl. Phys.* **360** (1991) 145–179.

Capability	Function (Return Type): Brief Description	Dependencies	Backend requirements	Options (Type)
*X*_data	*X*_full (nudata): Do signal, likelihood and related calculations for neutrino indirect detection analysis *X*.	mwimp annihilation_rate_Sun nuyield_ptr	nubounds	nulike_speed (int)
*X*_signal	*X*_signal (double): Number of signal events for neutrino indirect detection analysis *X*.	*X*_data		
*X*_bg	*X*_bg (double): Number of background events for neutrino indirect detection analysis *X*.	*X*_data		
*X*_loglike	*X*_loglike (double): Log-likelihood for neutrino indirect detection analysis *X*.	*X*_data		
*X*_bgloglike	*X*_bgloglike (double): Background-only log-likelihood for neutrino indirect detection analysis *X*.	*X*_data		
*X*_pvalue	*X*_pvalue (double): *p*-value for neutrino indirect detection analysis *X*.	*X*_data		
*X*_nobs	*X*_nobs (int): Number of observed events for neutrino indirect detection analysis *X*.	*X*_data		
IC79_loglike	IC79_loglike (double): The full 79-string IceCube log-likelihood.	*Y*_loglike *Y*_bgloglike for all *Y* ∈ {IC79WH, IC79WL,IC79SL}		
IceCube_likelihood	IC_loglike (double): The complete IceCube log-likelihood.	*Y*_loglike *Y*_bgloglike for all *Y*∈ {IC22,IC79WH, IC79WL,IC79SL}		

Table A10: DarkBit capabilities for neutrino indirect detection likelihoods. Possible values for *X* are IC22, IC79WH, IC79WL, and 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 A11: Cascade decay capabilities provided by DarkBit that do not run inside the cascade decay Monte Carlo loop.

42. P. Gondolo, J. Edsjö, *et. al.*, *DarkSUSY: Computing supersymmetric dark matter properties numerically*, *JCAP* **0407** (2004) 008, [astro-ph/0406204].
43. G. Belanger, F. Boudjema, A. Pukhov, and A. Semenov, *micrOMEGAs_3: A program for calculating dark matter observables*, *Comp. Phys. Comm.* **185** (2014) 960–985, [arXiv:1305.0237].
44. G. Belanger, F. Boudjema, *et. al.*, *Indirect search for dark matter with micrOMEGAs2.4*, *Comp. Phys. Comm.* **182** (2011) 842–856, [arXiv:1004.1092].
45. G. Belanger, F. Boudjema, A. Pukhov, and A. Semenov, *Dark matter direct detection rate in a generic model with micrOMEGAs 2.2*, *Comp. Phys. Comm.* **180** (2009) 747–767, [arXiv:0803.2360].
46. G. Belanger, F. Boudjema, A. Pukhov, and A. Semenov, *MicrOMEGAs 2.0: A Program to calculate the relic density of dark matter in a generic model*, *Comp. Phys. Comm.* **176** (2007) 367–382, [hep-ph/0607059].
47. G. Belanger, F. Boudjema, A. Pukhov, and A. Semenov, *micrOMEGAs: Version 1.3*, *Comp. Phys. Comm.* **174** (2006) 577–604,

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.	GA_missingFinalStates	
cascadeMC_EventCount	cascadeMC_EventCount (DarkBit::stringIntMap): The event counter for cascade decays.	cascadeMC_InitialState	
cascadeMC_ChainEvent	cascadeMC_GenerateChain (DarkBit::DecayChain::ChainContainer): Function for generating decay chains.	cascadeMC_InitialState cascadeMC_DecayTable	cMC_maxChainLength (int) cMC_Emin (double)
cascadeMC_Histograms	cascadeMC_Histograms (DarkBit::simpleHistContainter): Function responsible for histogramming and evaluating the end conditions for the event loop in the cascade decay Monte Carlo simulation.	cascadeMC_InitialState cascadeMC_ChainEvent TH_ProcessCatalog SimYieldTable cascadeMC_FinalStates	cMC_numSpecSamples (int) cMC_NhistBins (int) cMC_binLow (double) cMC_binHigh (double) cMC_gammaBGPower (double) cMC_gammaRelError (double) cMC_endCheckFrequency (int)

Table A12: 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_LoopManagement capability.

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 decay_rates	dswwidth dsprep mssmpar dssusy dsSLHAread dssusy_isasugra dsgive_model_isasugra initFromSLHAeaAndDecayTable	use_DS_isasugra (bool) use_dsSLHAread (bool) debug_SLHA_filenames (std::vector<str>)
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 LocalHalo	dshmcom dshmisodf dshmframevelcom dshmnoclue	v_earth (double)
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 RD_oh2 GA_AnnYield TH_ProcessCatalog DarkMatter_ID		fileroot (std::string) GA_AnnYield:Emin (double) GA_AnnYield:Emax (double) GA_AnnYield:nbins (double)

Table A13: Miscellaneous capabilities for DarkSUSY initialisation and debugging.

[hep-ph/0405253].

48. G. Belanger, F. Boudjema, A. Pukhov, and A. Semenov, *MicrOMEGAs: A Program for calculating the relic density in the MSSM*, *Comp. Phys. Comm.* **149** (2002) 103–120, [hep-ph/0112278].
49. J. Harz, B. Herrmann, M. Klasen, K. Kovarik, and P. Steppeler, *Precise Prediction of the Dark Matter Relic Density within the MSSM*, *PoS* **EPS-HEP2015** (2015) 410, [arXiv:1510.06295].
50. J. Harz, B. Herrmann, M. Klasen, K. Kovarik, and P. Steppeler, *Theoretical uncertainty of the supersymmetric dark matter relic density from scheme and scale variations*, *Phys. Rev. D* **93** (2016) 114023, [arXiv:1602.08103].
51. M. W. Goodman and E. Witten, *Detectability of Certain Dark Matter Candidates*, *Phys.Rev.* **31** (1985) 3059.

52. J. Kumar and D. Marfatia, *Matrix element analyses of dark matter scattering and annihilation*, *Phys. Rev. D* **88** (2013) 014035, [arXiv:1305.1611].
53. R. H. Helm, *Inelastic and Elastic Scattering of 187-Mev Electrons from Selected Even-Even Nuclei*, *Phys.Rev.* **104** (1956) 1466–1475.
54. J. Lewin and P. Smith, *Review of mathematics, numerical factors, and corrections for dark matter experiments based on elastic nuclear recoil*, *Astropart.Phys.* **6** (1996) 87–112.
55. G. Duda, A. Kemper, and P. Gondolo, *Model Independent Form Factors for Spin Independent Neutralino-Nucleon Scattering from Elastic Electron Scattering Data*, *JCAP* **0704** (2007) 012, [hep-ph/0608035].
56. V. Bednyakov and F. Simkovic, *Nuclear spin structure in dark matter search: The Zero momentum transfer limit*, *Phys.Part.Nucl.* **36** (2005) 131–152, [hep-ph/0406218].
57. V. Bednyakov and F. Simkovic, *Nuclear spin structure in dark matter search: The Finite momentum transfer limit*, *Phys.Part.Nucl.* **37** (2006) S106–S128, [hep-ph/0608097].
58. C. Savage, A. Scaffidi, M. White, and A. G. Williams, *LUX likelihood and limits on spin-independent and spin-dependent WIMP couplings with LUXCalc*, *Phys. Rev. D* **92** (2015) 103519, [arXiv:1502.02667].
59. A. Berlin, S. Gori, T. Lin, and L.-T. Wang, *Pseudoscalar Portal Dark Matter*, *Phys. Rev. D* **92** (2015) 015005, [arXiv:1502.06000].
60. A. Beniwal, F. Rajec, *et. al.*, *Combined analysis of effective Higgs portal dark matter models*, arXiv:1512.06458.
61. S. Liem, G. Bertone, *et. al.*, *Effective Field Theory of Dark Matter: a Global Analysis*, arXiv:1603.05994.
62. P. Klos, J. Menéndez, D. Gazit, and A. Schwenk, *Large-scale nuclear structure calculations for spin-dependent WIMP scattering with chiral effective field theory currents*, *Phys. Rev. D* **88** (2013) 083516, [arXiv:1304.7684].
63. G. J. Feldman and R. D. Cousins, *A Unified approach to the classical statistical analysis of small signals*, *Phys.Rev.* **57** (1998) 3873–3889, [physics/9711021].
64. S. Yellin, *Finding an upper limit in the presence of unknown background*, *Phys.Rev.* **66** (2002) 032005, [physics/0203002].
65. XENON100 Collaboration: E. Aprile, M. Alfonsi, *et. al.*, *Dark Matter Results from 225 Live Days of XENON100 Data*, *Phys. Rev. Lett.* **109** (2012) 181301, [arXiv:1207.5988].
66. SuperCDMS Collaboration: R. Agnese *et. al.*, *Search for Low-Mass Weakly Interacting Massive Particles with SuperCDMS*, *Phys. Rev. Lett.* **112** (2014) 241302, [arXiv:1402.7137].
67. SIMPLE Collaboration: M. Felizardo *et. al.*, *The SIMPLE Phase II Dark Matter Search*, *Phys. Rev. D* **89** (2014) 072013, [arXiv:1404.4309].
68. LUX Collaboration: D. S. Akerib *et. al.*, *First results from the LUX dark matter experiment at the Sanford Underground Research Facility*, *Phys. Rev. Lett.* **112** (2014) 091303, [arXiv:1310.8214].
69. D. S. Akerib, H. M. Araújo, *et. al.*, *Improved Limits on Scattering of Weakly Interacting Massive Particles from Reanalysis of 2013 LUX Data*, *Phys. Rev. Lett.* **116** (2016) 161301, [arXiv:1512.03506].
70. D. S. Akerib, S. Alsum, *et. al.*, *Results from a Search for Dark Matter in the Complete LUX Exposure*, *Phys. Rev. Lett.* **118** (2017) 021303, [arXiv:1608.07648].
71. PandaX-II Collaboration, :, *et. al.*, *Dark Matter Results from First 98.7-day Data of PandaX-II Experiment*, *ArXiv e-prints* (2016) [arXiv:1607.07400].
72. C. Amole, M. Ardid, *et. al.*, *Dark matter search results from the PICO-60 CF_3 I bubble chamber*, *Phys. Rev. D* **93** (2016) 052014, [arXiv:1510.07754].
73. PICO: C. Amole *et. al.*, *Improved dark matter search results from PICO-2L Run 2*, *Phys. Rev. D* **93** (2016) 061101, [arXiv:1601.03729].
74. PICO: C. Amole *et. al.*, *Dark Matter Search Results from the PICO-2L C_3F_8 Bubble Chamber*, *Phys. Rev. Lett.* **114** (2015) 231302, [arXiv:1503.00008].
75. C. Savage, "TPCMC: a Time Projection Chamber Monte Carlo for dark matter searches." Private code.
76. http://nest.physics.ucdavis.edu/site/.
77. J. M. Cline, K. Kainulainen, P. Scott, and C. Weniger, *Update on scalar singlet dark matter*, *Phys. Rev. D* **88** (2013) 055025, [arXiv:1306.4710].
78. J. R. Ellis, K. A. Olive, and C. Savage, *Hadronic Uncertainties in the Elastic Scattering of Supersymmetric Dark Matter*, *Phys. Rev. D* **77** (2008) 065026, [arXiv:0801.3656].
79. H.-W. Lin, *Lattice QCD for Precision Nucleon Matrix Elements*, arXiv:1112.2435.
80. M. M. Pavan, I. I. Strakovsky, R. L. Workman, and R. A. Arndt, *The Pion nucleon Sigma term is*

definitely large: Results from a G.W.U. analysis of pi nucleon scattering data, *PiN Newslett.* **16** (2002) 110–115, [hep-ph/0111066].

81. J. M. Alarcon, J. Martin Camalich, and J. A. Oller, *The chiral representation of the πN scattering amplitude and the pion-nucleon sigma term*, *Phys. Rev. D* **85** (2012) 051503, [arXiv:1110.3797].
82. L. Alvarez-Ruso, T. Ledwig, J. Martin Camalich, and M. J. Vicente-Vacas, *Nucleon mass and pion-nucleon sigma term from a chiral analysis of lattice QCD data*, *Phys. Rev. D* **88** (2013) 054507, [arXiv:1304.0483].
83. Particle Data Group: K. A. Olive *et. al.*, *Review of Particle Physics*, *Chin. Phys. C* **38** (2014) 090001.
84. Asymmetry Analysis: Y. Goto *et. al.*, *Polarized parton distribution functions in the nucleon*, *Phys. Rev. D* **62** (2000) 034017, [hep-ph/0001046].
85. COMPASS: V. Yu. Alexakhin *et. al.*, *The Deuteron Spin-dependent Structure Function g1(d) and its First Moment*, *Phys. Rev. B* **647** (2007) 8–17, [hep-ex/0609038].
86. J. L. Newstead, T. D. Jacques, L. M. Krauss, J. B. Dent, and F. Ferrer, *Scientific reach of multiton-scale dark matter direct detection experiments*, *Phys. Rev. D* **88** (2013) 076011, [arXiv:1306.3244].
87. T. Sjöstrand, S. Mrenna, and P. Z. Skands, *PYTHIA 6.4 Physics and Manual*, *JHEP* **05** (2006) 026, [hep-ph/0603175].
88. T. Bringmann and C. Weniger, *Gamma Ray Signals from Dark Matter: Concepts, Status and Prospects*, *Phys. Dark Univ.* **1** (2012) 194–217, [arXiv:1208.5481].
89. L. Bergström, J. Edsjö, and P. Gondolo, *Indirect detection of dark matter in km size neutrino telescopes*, *Phys. Rev. D* **58** (1998) 103519, [hep-ph/9806293].
90. M. Cirelli and G. Giesen, *Antiprotons from Dark Matter: Current constraints and future sensitivities*, *JCAP* **1304** (2013) 015, [arXiv:1301.7079].
91. T. Bringmann, M. Vollmann, and C. Weniger, *Updated cosmic-ray and radio constraints on light dark matter: Implications for the GeV gamma-ray excess at the Galactic center*, *Phys. Rev. D* **90** (2014) 123001, [arXiv:1406.6027].
92. M. Cirelli, D. Gaggero, G. Giesen, M. Taoso, and A. Urbano, *Antiproton constraints on the GeV gamma-ray excess: a comprehensive analysis*, *JCAP* **1412** (2014) 045, [arXiv:1407.2173].
93. L. Bergström, T. Bringmann, I. Cholis, D. Hooper, and C. Weniger, *New limits on dark matter annihilation from AMS cosmic ray positron data*, *Phys. Rev. Lett.* **111** (2013) 171101, [arXiv:1306.3983].
94. T. Bringmann and C. Weniger, *Gamma ray signals from dark matter: Concepts, status and prospects*, *Physics of the Dark Universe* **1** (2012) 194–217, [arXiv:1208.5481].
95. L. Bergström, P. Ullio, and J. H. Buckley, *Observability of gamma-rays from dark matter neutralino annihilations in the Milky Way halo*, *Astropart. Phys.* **9** (1998) 137–162, [astro-ph/9712318].
96. T. Bringmann, L. Bergström, and J. Edsjö, *New Gamma-Ray Contributions to Supersymmetric Dark Matter Annihilation*, *JHEP* **01** (2008) 049, [arXiv:0710.3169].
97. A. Ibarra, S. Lopez Gehler, and M. Pato, *Dark matter constraints from box-shaped gamma-ray features*, *JCAP* **1207** (2012) 043, [arXiv:1205.0007].
98. M. A. SÃąnchez-Conde and F. Prada, *The flattening of the concentrationâĂŞmass relation towards low halo masses and its implications for the annihilation signal boost*, *MNRAS* **442** (2014) 2271–2277, [arXiv:1312.1729].
99. G. Steigman, H. Quintana, C. L. Sarazin, and J. Faulkner, *Dynamical interactions and astrophysical effects of stable heavy neutrinos*, *AJ* **83** (1978) 1050–1061.
100. T. K. Gaisser, G. Steigman, and S. Tilav, *Limits on cold-dark-matter candidates from deep underground detectors*, *Phys. Rev. D* **34** (1986) 2206–2222.
101. A. Gould, *Resonant enhancements in weakly interacting massive particle capture by the earth*, *ApJ* **321** (1987) 571–585.
102. M. Danninger and C. Rott, *Solar WIMPs unravelled: Experiments, astrophysical uncertainties, and interactive tools*, *Physics of the Dark Universe* **5** (2014) 35–44, [arXiv:1509.08230].
103. M. Blennow, J. Edsjö, and T. Ohlsson, *Neutrinos from WIMP annihilations obtained using a full three-flavor Monte Carlo approach*, *JCAP* **1** (2008) 21, [arXiv:0709.3898].
104. P. Scott, M. Fairbairn, and J. Edsjö, *Dark stars at the Galactic Centre - the main sequence*, *MNRAS* **394** (2009) 82–104, [0809.1871].
105. M. Taoso, F. Iocco, G. Meynet, G. Bertone, and P. Eggenberger, *Effect of low mass dark matter particles on the Sun*, *Phys. Rev. D* **82** (2010)

083509, [arXiv:1005.5711].
106. F. Iocco, M. Taoso, F. Leclercq, and G. Meynet, *Main Sequence Stars with Asymmetric Dark Matter*, *Phys. Rev. Lett.* **108** (2012) 061301, [arXiv:1201.5387].
107. A. C. Vincent, P. Scott, and A. Serenelli, *Possible Indication of Momentum-Dependent Asymmetric Dark Matter in the Sun*, *Phys. Rev. Lett.* **114** (2015) 081302, [arXiv:1411.6626].
108. A. C. Vincent, A. Serenelli, and P. Scott, *Generalised form factor dark matter in the Sun*, *JCAP* **8** (2015) 40, [arXiv:1504.04378].
109. A. C. Vincent, P. Scott, and A. Serenelli, *Updated constraints on velocity and momentum-dependent asymmetric dark matter*, *JCAP* **11** (2016) 007, [arXiv:1605.06502].
110. The IceCube Collaboration: M. G. Aartsen, K. Abraham, *et. al.*, *Search for dark matter annihilation in the Galactic Center with IceCube-79*, *Eur. Phys. J. C* **75** (2015) 492, [arXiv:1505.07259].
111. The ANTARES Collaboration: S. Adrián-Martínez *et. al.*, *Search of dark matter annihilation in the galactic centre using the ANTARES neutrino telescope*, *JCAP* **10** (2015) 068, [arXiv:1505.04866].
112. A. Gould, *Weakly interacting massive particle distribution in and evaporation from the sun*, *ApJ* **321** (1987) 560–570.
113. G. Busoni, A. De Simone, and W.-C. Huang, *On the minimum dark matter mass testable by neutrinos from the Sun*, *JCAP* **7** (2013) 010, [arXiv:1305.1817].
114. G. Busoni, A. De Simone, P. Scott, and A. Vincent. in preparation.
115. P. Baratella, M. Cirelli, *et. al.*, *PPPC 4 DMν: a Poor Particle Physicist Cookbook for Neutrinos from Dark Matter annihilations in the Sun*, *JCAP* **3** (2014) 053, [arXiv:1312.6408].
116. WimpSim is available at http://www.fysik.su.se/∼edsjo/wimpsim/.
117. IceCube Collaboration: M. G. Aartsen *et. al.*, *Search for annihilating dark matter in the Sun with 3 years of IceCube data*, *Eur. Phys. J. C* **77** (2017) 146, [arXiv:1612.05949].
118. K. Choi, K. Abe, *et. al.*, *Search for Neutrinos from Annihilation of Captured Low-Mass Dark Matter Particles in the Sun by Super-Kamiokande*, *Phys. Rev. Lett.* **114** (2015) 141301.
119. P. Scott, C. Savage, J. Edsjö, and the IceCube Collaboration: R. Abbasi et al., *Use of event-level neutrino telescope data in global fits for theories of new physics*, *JCAP* **11** (2012) 57, [arXiv:1207.0810].
120. IceCube Collaboration: M. G. Aartsen, R. Abbasi, *et. al.*, *Search for Dark Matter Annihilations in the Sun with the 79-String IceCube Detector*, *Phys. Rev. Lett.* **110** (2013) 131302, [arXiv:1212.4097].
121. Fermi-LAT: M. Ackermann *et. al.*, *Dark matter constraints from observations of 25 Milky Way satellite galaxies with the Fermi Large Area Telescope*, *Phys. Rev. D* **89** (2014) 042001, [arXiv:1310.0828].
122. Fermi-LAT: 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** (2015) 231301, [arXiv:1503.02641].
123. HESS: 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** (2011) 161301, [arXiv:1103.3266].
124. F. Calore, I. Cholis, and C. Weniger, *Background model systematics for the Fermi GeV excess*, *JCAP* **1503** (2015) 038, [arXiv:1409.0042].
125. A. Achterberg, S. Amoroso, *et. al.*, *A description of the Galactic Center excess in the Minimal Supersymmetric Standard Model*, *JCAP* **1508** (2015) 006, [arXiv:1502.05703].
126. H. Silverwood, C. Weniger, P. Scott, and G. Bertone, *A realistic assessment of the CTA sensitivity to dark matter annihilation*, *JCAP* **1503** (2015) 055, [arXiv:1408.4131].
127. A. Chiappo, J. Cohen-Tanugi, *et. al.*, *Dwarf spheroidal J-factors without priors: A likelihood-based analysis for indirect dark matter searches*, arXiv:1608.07111.
128. D. Hooper and L. Goodenough, *Dark Matter Annihilation in The Galactic Center As Seen by the Fermi Gamma Ray Space Telescope*, *Phys. Lett. B* **697** (2011) 412–428, [arXiv:1010.2752].
129. O. Macias and C. Gordon, *Contribution of cosmic rays interacting with molecular clouds to the Galactic Center gamma-ray excess*, *Phys. Rev. D* **89** (2014) 063515, [arXiv:1312.6671].
130. K. N. Abazajian, N. Canac, S. Horiuchi, and M. Kaplinghat, *Astrophysical and Dark Matter Interpretations of Extended Gamma-Ray Emission from the Galactic Center*, *Phys. Rev. D* **90** (2014) 023526, [arXiv:1402.4090].
131. T. Daylan, D. P. Finkbeiner, *et. al.*, *The characterization of the gamma-ray signal from the central Milky Way: A case for annihilating dark matter*, *Phys. Dark Univ.* **12** (2016) 1–23,

[arXiv:1402.6703].
132. 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** (2015) 123010, [arXiv:1406.6948].
133. Fermi-LAT: M. Ajello *et. al.*, *Fermi-LAT Observations of High-Energy γ-Ray Emission Toward the Galactic Center*, *Astrophys. J.* **819** (2016) 44, [arXiv:1511.02938].
134. R. Bartels, S. Krishnamurthy, and C. Weniger, *Strong support for the millisecond pulsar origin of the Galactic center GeV excess*, *Phys. Rev. Lett.* **116** (2016) 051102, [arXiv:1506.05104].
135. S. K. Lee, M. Lisanti, B. R. Safdi, T. R. Slatyer, and W. Xue, *Evidence for Unresolved γ-Ray Point Sources in the Inner Galaxy*, *Phys. Rev. Lett.* **116** (2016) 051103, [arXiv:1506.05124].
136. F. Calore, I. Cholis, C. McCabe, and C. Weniger, *A Tale of Tails: Dark Matter Interpretations of the Fermi GeV Excess in Light of Background Model Systematics*, *Phys. Rev. D* **91** (2015) 063003, [arXiv:1411.4647].
137. H. Silverwood, "CTA morphological likelihood analysis." Private code.
138. T. Bringmann, A. J. Galea, and P. Walia, *Leading QCD Corrections for Indirect Dark Matter Searches: a Fresh Look*, *Phys. Rev. D* **93** (2016) 043529, [arXiv:1510.02473].
139. T. Bringmann and F. Calore, *Significant Enhancement of Neutralino Dark Matter Annihilation from Electroweak Bremsstrahlung*, *Phys. Rev. Lett.* **112** (2014) 071301, [arXiv:1308.1089].
140. T. Bringmann, F. Calore, A. J. Galea, and M. Garny, *Electroweak and Higgs Boson Internal Bremsstrahlung: General considerations for Majorana dark matter annihilation and application to MSSM neutralinos*, in prep. (2016).
141. T. Sjostrand, S. Ask, *et. al.*, *An Introduction to PYTHIA 8.2*, *Comp. Phys. Comm.* **191** (2015) 159–177, [arXiv:1410.3012].
142. M. Asplund, N. Grevesse, A. J. Sauval, and P. Scott, *The chemical composition of the Sun*, *ARA&A* **47** (2009) 481–522, [arXiv:0909.0948].
143. A. M. Serenelli, S. Basu, J. W. Ferguson, and M. Asplund, *New Solar Composition: The Problem with Solar Models Revisited*, *ApJ* **705** (2009) L123–L127, [arXiv:0909.2668].
144. IceCube Collaboration: R. Abbasi, Y. Abdou, *et. al.*, *Limits on a Muon Flux from Neutralino Annihilations in the Sun with the IceCube 22-String Detector*, *Phys. Rev. Lett.* **102** (2009) 201302, [arXiv:0902.2460].
145. A. L. Read, *Modified frequentist analysis of search results (the CL_s method)*, in *1st Workshop on Confidence Limits (CERN, Geneva, Switzerland)* (2000) 81–101. CERN-2000-005.
146. A. L. Read, *DURHAM IPPP WORKSHOP PAPER: Presentation of search results: the CL_s technique*, *J. Phys. G* **28** (2002) 2693–2704.
147. G. Elor, N. L. Rodd, T. R. Slatyer, and W. Xue, *Model-Independent Indirect Detection Constraints on Hidden Sector Dark Matter*, arXiv:1511.08787.
148. GAMBIT Collaboration: P. Athron, C. Balázs, *et. al.*, *Global fits of GUT-scale SUSY models with GAMBIT*, *Eur. Phys. J. C, to be submitted* (2017) [arXiv:1703.xxxxx].
149. GAMBIT Collaboration: P. Athron, C. Balázs, *et. al.*, *A global fit of the MSSM with GAMBIT*, *Eur. Phys. J. C, to be submitted* (2017) [arXiv:1703.xxxxx].
150. GAMBIT Collaboration: P. Athron, C. Balázs, *et. al.*, *Status of the scalar singlet dark matter model*, *Eur. Phys. J. C submitted* (2017) [arXiv:1703.xxxxx].
151. PandaX-II: C. Fu *et. al.*, *Spin-Dependent Weakly-Interacting-Massive-Particle–Nucleon Cross Section Limits from First Data of PandaX-II Experiment*, *Phys. Rev. Lett.* **118** (2017) 071301, [arXiv:1611.06553].
152. G. Bélanger, F. Boudjema, A. Pukhov, and A. Semenov, *micrOMEGAs4.1: Two dark matter candidates*, *Comp. Phys. Comm.* **192** (2015) 322–329, [arXiv:1407.6129].
153. F. D'Eramo and J. Thaler, *Semi-annihilation of Dark Matter*, *JHEP* **06** (2010) 109, [arXiv:1003.5912].
154. K. Petraki and R. R. Volkas, *Review of asymmetric dark matter*, *Int. J. Mod. Phys. A* **28** (2013) 1330028, [arXiv:1305.4939].
155. T. Bringmann, *Particle Models and the Small-Scale Structure of Dark Matter*, *New J. Phys.* **11** (2009) 105027, [arXiv:0903.0189].
156. J. M. Cornell, S. Profumo, and W. Shepherd, *Kinetic Decoupling and Small-Scale Structure in Effective Theories of Dark Matter*, *Phys. Rev. D* **88** (2013) 015027, [arXiv:1305.4676].
157. A. L. Fitzpatrick, W. Haxton, E. Katz, N. Lubbers, and Y. Xu, *The Effective Field Theory of Dark Matter Direct Detection*, *JCAP* **1302** (2013) 004, [arXiv:1203.3542].
158. J. Hisano, S. Matsumoto, M. M. Nojiri, and O. Saito, *Non-perturbative effect on dark matter annihilation and gamma ray signature from galactic center*, *Phys. Rev. D* **71** (2005) 063528,

[hep-ph/0412403].
159. R. Iengo, *Sommerfeld enhancement: General results from field theory diagrams*, *JHEP* **05** (2009) 024, [arXiv:0902.0688].
160. J. Bovy, *Substructure Boosts to Dark Matter Annihilation from Sommerfeld Enhancement*, *Phys. Rev. D* **79** (2009) 083539, [arXiv:0903.0413].
161. C. Arina, T. Bringmann, J. Silk, and M. Vollmann, *Enhanced Line Signals from Annihilating Kaluza-Klein Dark Matter*, *Phys. Rev. D* **90** (2014) 083506, [arXiv:1409.0007].
162. J. Choquette, J. M. Cline, and J. M. Cornell, *p-wave Annihilating Dark Matter from a Decaying Predecessor and the Galactic Center Excess*, *Phys. Rev. D* **94** (2016) 015018, [arXiv:1604.01039].