Dark matter tools

Pat Scott

Imperial College London

Slides at: www.imperial.ac.uk/people/p.scott/research.html

Imperial College London

What this talk **is definitely not**:

Imperial College London

What this talk **is definitely not**:

- a full review of all DM tools

Imperial College London

What this talk **is definitely not**:

- a full review of all DM tools
- an in depth/complete/fair representation of the physics/code/history/philosophy/politics of *your* tool

Imperial College London

What this talk **is definitely not**:

- a full review of all DM tools
- an in depth/complete/fair representation of the physics/code/history/philosophy/politics of *your* tool

What this talk **is**:

Imperial College London

What this talk **is definitely not**:

- a full review of all DM tools
- an in depth/complete/fair representation of the physics/code/history/philosophy/politics of *your* tool

What this talk **is**:

- a series of talking points == a rough personal overview of a number of open-source DM tools

Imperial College London

What this talk **is definitely not**:

- a full review of all DM tools
- an in depth/complete/fair representation of the physics/code/history/philosophy/politics of *your* tool

What this talk **is**:

- a series of talking points == a rough personal overview of a number of open-source DM tools
- an opportunity to ask "stupid" questions about what a tool does or how it does it

Imperial College London

What this talk **is definitely not**:

- a full review of all DM tools
- an in depth/complete/fair representation of the physics/code/history/philosophy/politics of *your* tool

What this talk **is**:

- a series of talking points == a rough personal overview of a number of open-source DM tools
- an opportunity to ask "stupid" questions about what a tool does or how it does it
- a chance for you to chime in to add your 2c about your tool (or any other)

Imperial College London

Outline

1. Direct detection
2. Indirect detection
3. Relic density (++)
4. Combination / Global fit codes

Imperial College London

Outline

Imperial College London

https://ddcalc.hepforge.org/

- Home
- Download
- Source Code
- Report issue
- Mailing list
- Contact

DDCalc v2

Dark matter direct detection phenomenology package

DDCalc is a software package for performing various dark matter direct detection calculations, including signal rate predictions and likelihoods for several experiments.

A full description of this package and the physics framework behind it can be found in the GAMBIT DarkBit paper:

- T Bringmann, J Conrad, JM Cornell, LA Dal, J Edsjö, B Farmer, F Kahlhoefer, A Kvellestad, A Putze, C Savage, P Scott, C Weniger, M White & S Wild 2017, EPJC 77 (2017) 831, arXiv:1705.07920

If you write a paper that uses DDCalc, please cite this paper.

Version history:

- **v2.0.0 - June 2018:** Support for full set of non-relativistic operators with general momentum and velocity dependence, new features for the definition of complex experiments with several signal regions and/or target elements, improved user interface including several new example files, new results from XENON1T (2018).

https://directdm.github.io/

DirectDM: a tool for dark matter direct detection

`DirectDM` is a program that matches Wilson coefficients of a relativistic EFT onto a Galilean-invariant non-relativistic EFT valid at the nuclear scale. The program is available as a `Mathematica` and a `python` package.

The `Mathematica` package is available from: github.com/DirectDM/directdm-mma

The `python3` package is available from: github.com/DirectDM/directdm-py

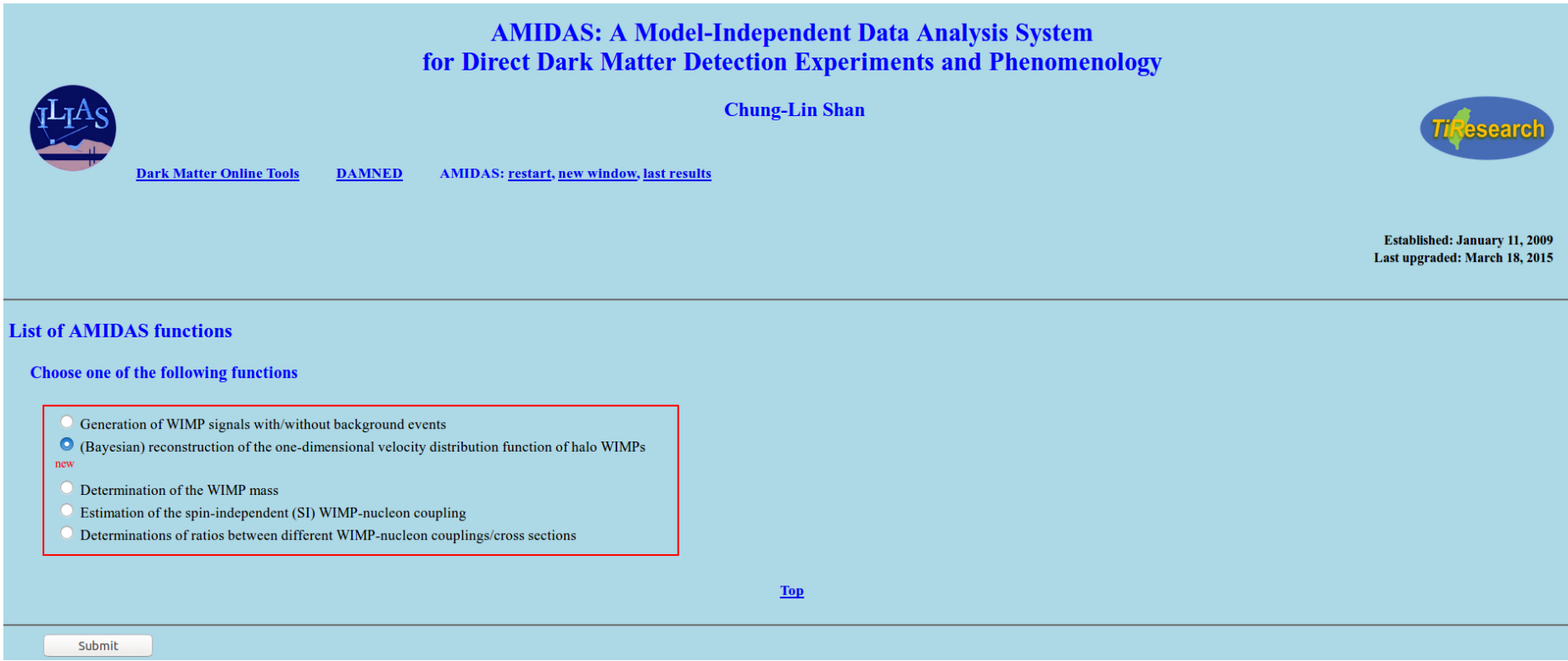
---

Citation

If you use `DirectDM` please cite

- 1611.00368: Chiral effective theory of dark matter direct detection; Bishara, Brod, Grinstein, and Zupan
- 1707.06998: From quarks to nucleons in dark matter direct detection; Bishara, Brod, Grinstein, and Zupan

http://www.tir.tw/phys/hep/dm/amidas/



Imperial College London

Outline

Imperial College London

https://gamlike.hepforge.org/

- Home
- Download
- Source Code
- Report issue
- Mailing list
- Contact

gamLike

Likelihoods for indirect dark matter searches with gamma rays

This is the development page for the gamLike project. GamLike is released in tandem with GAMBIT.

GamLike contains likelihood functions for most leading gamma-ray indirect searches for dark matter, including Fermi-LAT observations of dwarfs and the Galactic Centre (GC), HESS observations of the GC, and projected sensitivites for CTA observations of the GC.

A full description of this package and the physics framework behind it can be found in the GAMBIT DarkBit paper:

- T Bringmann, J Conrad, JM Cornell, LA Dal, J Edsjö, B Farmer, F Kahlhoefer, A Kvellestad, A Putze, C Savage, P Scott, C Weniger, M White & S Wild 2017, EPJC submitted, arXiv:1705.07920

If you write a paper that uses gamLike, please cite the above paper, as well as all relevant experimental papers.

GamLike releases can be obtained as tarballs from Hepforge. The latest and greatest version, along with a full revision history, can always be found in the bitbucket repository. Compilation and usage instructions, as well as a number of example programs, can be found in the code release.

Author: Christoph Weniger (c.weniger@uva.nl)

https://nulike.hepforge.org/

- Home
- Download
- Source Code
- Report issue
- Mailing list
- Contact

nulike

neutrino telescope likelihood tools

Nulike is software for including full event-level information in likelihood calculations for neutrino telescope searches for dark matter annihilation.

Full details can be found in the papers:

1. Scott, Savage, Edsjö & IceCube Collaboration 2012, JCAP 11:057, arXiv:1207.0810
2. IceCube Collaboration 2016, JCAP 04:022, arXiv:1601.00653

If you use nulike for preparing a paper, please cite both of these. If you use the 79-string IceCube likelihoods, don't forget to also cite the original IC79 WIMP paper:

1. IceCube Collaboration 2013, PRL 110:131302, arXiv:1212.4097

Nulike releases can be obtained as tarballs from Hepforge. The latest and greatest version, along with a full revision history, can always be found in the git repository. Compilation and usage instructions, as well as a number of example programs, can be found in the code release.

Author: Pat Scott (p.scott@imperial.ac.uk)
Additional numerical routines contributed by Chris Savage (chris@savage.name)

http://www.marcocirelli.net/PPPC4DMID.html

PPPC 4 DM ID - A Poor Particle Physicist Cookbook for Dark Matter Indirect Detection

We provide ingredients and recipes for computing signals of TeV-scale Dark Matter annihilations and decays.

Data and Results from 1012.4515 [hep-ph] (and 1009.0224 [hep-ph]), from 1312.6408 [hep-ph], 1412.5696 [astro-ph.HE], from 1505.01049 [hep-ph] and from 1511.08787 [hep-ph].

If you use the data provided on this site, please cite:
M.Cirelli, G.Corcella, A.Hektor, G.Hütsi, M.Kadastik, P.Panci, M.Raidal, F.Sala, A.Strumia,
"PPPC 4 DM ID: A Poor Particle Physicist Cookbook for Dark Matter Indirect Detection",
arXiv 1012.4515, JCAP 1103 (2011) 051.
Erratum: JCAP 1210 (2012) E01.

If you use the 'Fluxes at production', please also cite:
P. Ciafaloni, D. Comelli, A. Riotto, F. Sala, A. Strumia, A. Urbano,
"Weak corrections are relevant for dark matter indirect detection",
arXiv 1009.0224, JCAP 1103 (2011) 019,
where the energy spectra have been computed including electroweak corrections.

If you use the 'DMν : Neutrinos from the Sun', please cite:
P. Baratella, M. Cirelli, A. Hektor, J. Pata, M. Piibeleht, A. Strumia,
"PPPC 4 DM ν : A Poor Particle Physicist Cookbook for Neutrinos from DM annihilations in the Sun",
arXiv 1312.6408, JCAP 1403 (2014) 053.

If you use the 'Fluxes of charged cosmic rays at the Earth, after propagation: antiprotons', please cite:
M. Boudaud, M. Cirelli, G. Giesen, P. Salati,
"A fussy revisitation of antiprotons as a tool for Dark Matter searches",
arXiv 1412.5696, JCAP 1505 (2015) 05, 013.

If you use the 'Propagation functions...', the 'Fluxes of charged cosmic rays', the 'Fluxes of Inverse Compton gamma-rays', the 'Fluxes of bremsstrahlung gamma-rays' or the 'Fluxes of synchrotron radiation', please cite:
J. Buch, M. Cirelli, G. Giesen, M. Taoso,
"PPPC 4 DM secondary: A Poor Particle Physicist Cookbook for secondary radiation from Dark Matter",
arXiv 1505.01049, JCAP 1509 (2015) 09, 037.

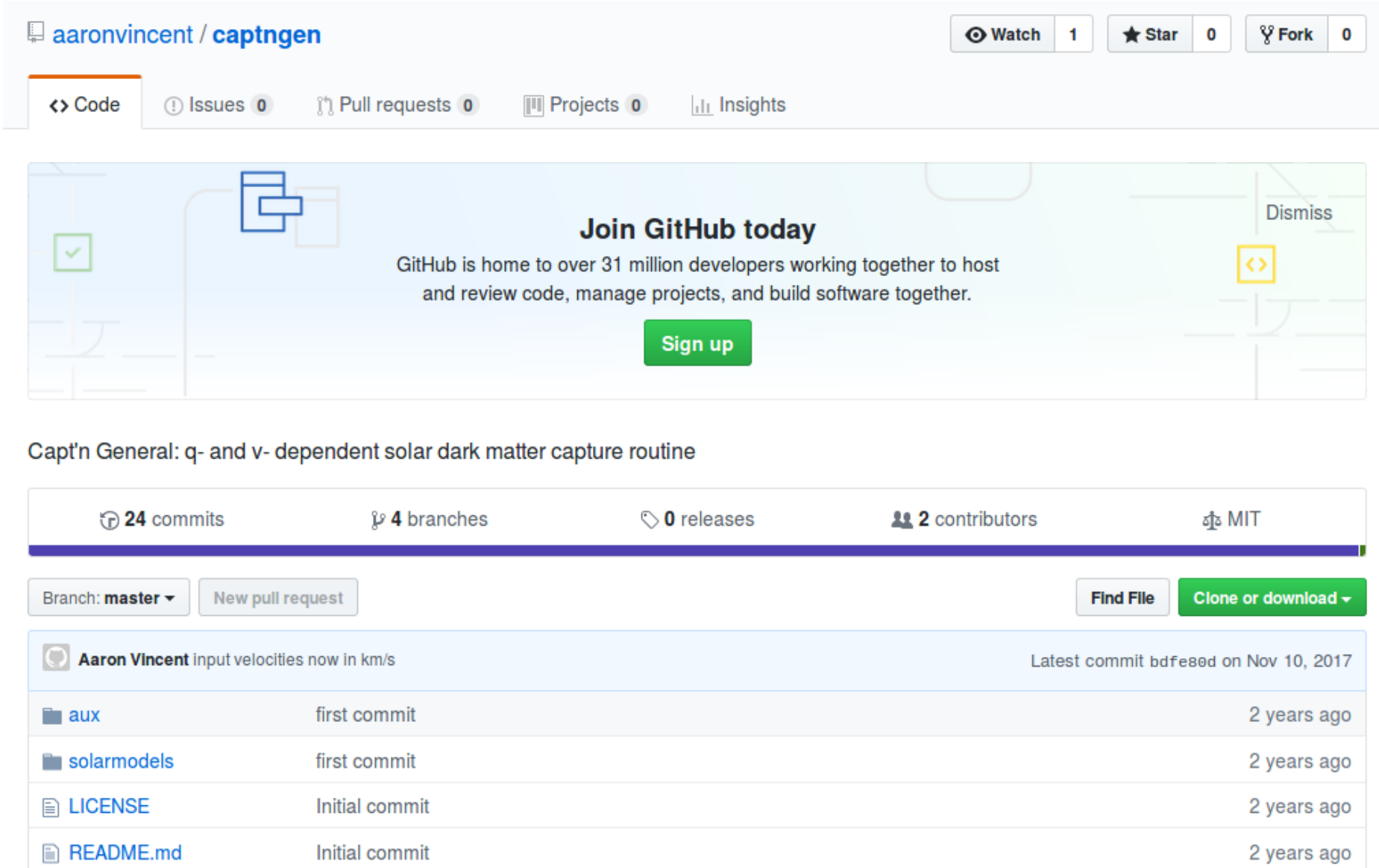
If you use the 'Fluxes at production in models with cascade decays in the hidden sector', please cite:
G. Elor, N. Rodd, T. Slatyer, W. Xue,
"Model-Independent Indirect Detection Constraints on Hidden Sector Dark Matter",
arXiv 1511.08787.

This is **Release 6.0** (Jan 2016). *Log of changes here.*

Imperial College London

Capt'n General / Capt'n Operator

https://github.com/aaronvincent/captngen



Other indirect detection

Cosmic ray propagation codes (fastest to slowest/most accurate):

- USINE: https://dmaurin.gitlab.io/USINE
- DRAGON: http://www.dragonproject.org/
- Galprop: https://galprop.stanford.edu/

Stellar evolution with dark matter:

- DarkStars: https://github.com/patscott/darkstars
- DarkStec: available on request from Aaron Vincent
- MESA: http://mesa.sourceforge.net/

Imperial College London

Outline

Imperial College London

http://www.darksusy.org/

- Home
- Download
- Documentation
- Tutorials
- Source Code
- Contact

DarkSUSY

DarkSUSY is a flexible and modular Fortran package to calculate observables for a variety of dark matter candidates. It is written by Joakim Edsjö, Torsten Bringmann, Paolo Gondolo, Piero Ullio, Lars Bergström, Mia Schelke, Ted Baltz, and Gintaras Duda. On these pages you will find information about DarkSUSY and you can also download the package.

Finally, DarkSUSY 6 has arrived with a completely new structure and the possibility to add new particle physics modules!

If you use DarkSUSY, please refer to the following publication describing DarkSUSY:

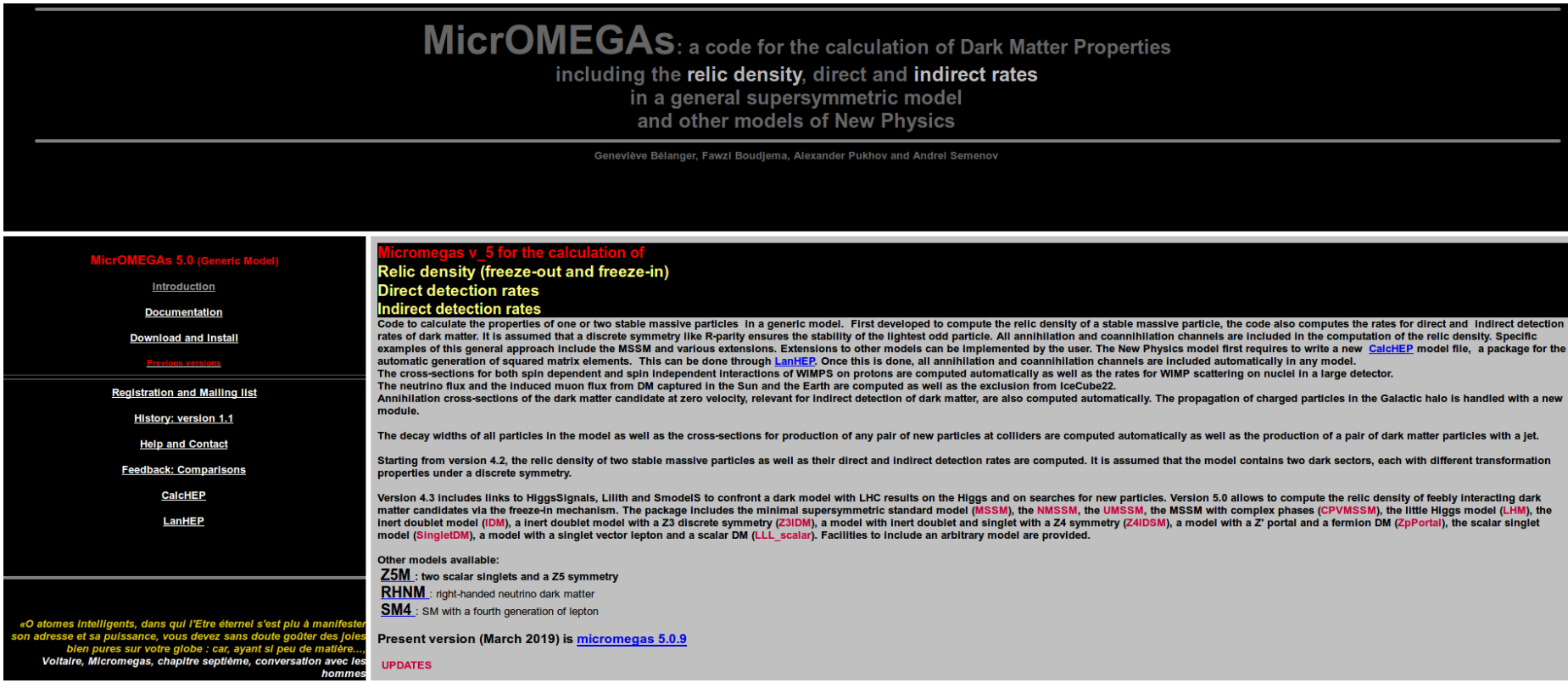
> **T. Bringmann, J. Edsjö, P. Gondolo, P. Ullio and L. Bergström, JCAP 1807 (2018) 033** [arXiv:1802.03399]

Please also cite (for code up to release version 4.1 contained in later versions)

> **P. Gondolo, J. Edsjö, P. Ullio, L. Bergström, M. Schelke and E.A. Baltz, JCAP 07 (2004) 008** [astro-ph/0406204]

Imperial College London

https://lapth.cnrs.fr/micromegas/



MicrOMEGAs: a code for the calculation of Dark Matter Properties
including the relic density, direct and indirect rates
in a general supersymmetric model
and other models of New Physics

Geneviève Bélanger, Fawzi Boudjema, Alexander Pukhov and Andrei Semenov

MicrOMEGAs 5.0 (Generic Model)

Introduction

Documentation

Download and Install

Previous versions

Registration and Mailing list

History: version 1.1

Help and Contact

Feedback: Comparisons

CalcHEP

LanHEP

«O atomes intelligents, dans qui l'Etre éternel s'est plu à manifester son adresse et sa puissance, vous devez sans doute goûter des joies bien pures sur votre globe : car, ayant si peu de matière..., Voltaire, Micromegas, chapitre septième, conversation avec les hommes

Micromegas v_5 for the calculation of
Relic density (freeze-out and freeze-in)
Direct detection rates
Indirect detection rates

Code to calculate the properties of one or two stable massive particles in a generic model. First developed to compute the relic density of a stable massive particle, the code also computes the rates for direct and indirect detection rates of dark matter. It is assumed that a discrete symmetry like R-parity ensures the stability of the lightest odd particle. All annihilation and coannihilation channels are included in the computation of the relic density. Specific examples of this general approach include the MSSM and various extensions. Extensions to other models can be implemented by the user. The New Physics model first requires to write a new CalcHEP model file, a package for the automatic generation of squared matrix elements. This can be done through LanHEP. Once this is done, all annihilation and coannihilation channels are included automatically in any model.
The cross-sections for both spin dependent and spin independent interactions of WIMPS on protons are computed automatically as well as the rates for WIMP scattering on nuclei in a large detector.
The neutrino flux and the induced muon flux from DM captured in the Sun and the Earth are computed as well as the exclusion from IceCube22.
Annihilation cross-sections of the dark matter candidate at zero velocity, relevant for indirect detection of dark matter, are also computed automatically. The propagation of charged particles in the Galactic halo is handled with a new module.

The decay widths of all particles in the model as well as the cross-sections for production of any pair of new particles at colliders are computed automatically as well as the production of a pair of dark matter particles with a jet.

Starting from version 4.2, the relic density of two stable massive particles as well as their direct and indirect detection rates are computed. It is assumed that the model contains two dark sectors, each with different transformation properties under a discrete symmetry.

Version 4.3 includes links to HiggsSignals, Lilith and SmodelS to confront a dark model with LHC results on the Higgs and on searches for new particles. Version 5.0 allows to compute the relic density of feebly interacting dark matter candidates via the freeze-in mechanism. The package includes the minimal supersymmetric standard model (MSSM), the NMSSM, the UMSSM, the MSSM with complex phases (CPVMSSM), the little Higgs model (LHM), the inert doublet model (IDM), a inert doublet model with a Z3 discrete symmetry (Z3IDM), a model with inert doublet and singlet with a Z4 symmetry (Z4IDSM), a model with a Z' portal and a fermion DM (ZpPortal), the scalar singlet model (SingletDM), a model with a singlet vector lepton and a scalar DM (LLL_scalar). Facilities to include an arbitrary model are provided.

Other models available:
Z5M : two scalar singlets and a Z5 symmetry
RHNM : right-handed neutrino dark matter
SM4 : SM with a fourth generation of lepton

Present version (March 2019) is micromegas 5.0.9

UPDATES

Imperial College London

http://superiso.in2p3.fr/relic/

SuperIso Relic

By Alexandre Arbey, Farvah Nazila Mahmoudi & Glenn Robbins

SuperIso

- Description
- Manual

SuperIso Relic

- Description
- Manual

Download

- SuperIso
- SuperIso Relic

AlterBBN

Links

Calculation of flavour physics and dark matter observables

SuperIso Relic is a mixed C - Fortran code which computes the dark matter observables in the MSSM and NMSSM. SuperIso Relic is an extension of SuperIso and therefore gives access to many flavour observables at the same time.

The computation of the relic density requires the calculation of thousands of annihilation and coannihilation Feynman diagrams. In SuperIso Relic, all these diagrams have been analytically computed at tree level using FeynArts/FormCalc. They are then calculated numerically during execution. The widths of the Higgs bosons are also computed using FeynHiggs or Hdecay. The necessary libraries are included in the SuperIso Relic packages.

From the cosmological point of view, SuperIso Relic performs the relic density calculation in the cosmological standard model, but also offers the possibility to alter the equation of state of radiation or modify the density and entropy content of the pre-BBN Universe. BBN constraints to check the validity of the altered model are automatically computed using the AlterBBN code included in the package.

Since its version 4, SuperIso Relic also includes routines to compute observables related to dark matter direct and indirect detections, and incorporates in particular experimental constraints from FERMI-LAT, AMS-02, XENON1T, PANDA-X and PICO60.

For any comment, question or bug report please contact Alexandre Arbey, Nazila Mahmoudi or Glenn Robbins.

https://launchpad.net/maddm

MadDM

Overview | Code | Bugs | Blueprints | Translations | Answers

Registered 2014-03-27 by Mihailo Backovic

ATTENTION: MadDM is now a plugin for MadGraph 5. In order to install it and run it, start madgraph and type

install maddm

in the command line. Then exit and start maddm with ./maddm.py.

M5G_aMC@NLO v.2.6.2 is required to be able to run MadDM v.3.0.

MadDM v.3.0 is a numerical tool to compute dark matter relic abundance, dark matter nucleus scattering rates and dark matter indirect detection predictions in a generic model. The code is based on the existing MadGraph 5 architecture and as such is easily integrable into any MadGraph collider study. A simple Python interface offers a level of user-friendliness characteristic of MadGraph 5 without sacrificing functionality.

MadDM is able to calculate the dark matter relic abundance in models which include a multi-component dark sector, resonance annihilation channels and co-annihilations.

The direct detection module of the MadDM code calculates spin independent / spin dependent dark matter-nucleon cross sections and differential recoil rates as a function of recoil energy

Outline

Imperial College London

Existing codes

- GAMBIT
- MasterCode (not public)
- HEPFit (no DM)
- SuperBayeS (retired)
- Fittino
- SFitter

Imperial College London

GAMBIT: The Global And Modular BSM Inference Tool

gambit.hepforge.org EPJC **77** (2017) 784 arXiv:1705.07908

- Extensive model database – not just SUSY
- Extensive observable/data libraries
- Many statistical and scanning options (Bayesian & frequentist)
- *Fast* LHC likelihood calculator
- Massively parallel
- Fully open-source

- Fast definition of new datasets and theories
- Plug and play scanning, physics and likelihood packages

Members of:
ATLAS, Belle-II, CLiC, CMS, CTA, *Fermi*-LAT, DARWIN, IceCube, LHCb, SHiP, XENON

Authors of:
DarkSUSY, DDCalc, Diver, FlexibleSUSY, gamlike, GM2Calc, IsaTols, nulike, PolyChord, Rivet, SoftSUSY, SuperISO, SUSY-AI, WIMPSim

Recent collaborators:
Peter Athron, Csaba Balázs, Ankit Beniwal, Sanjay Bloor, Torsten Bringmann, Andy Buckley, José Eliel Camargo-Molina, Marcin Chrząszcz, Jonathan Cornell, Matthias Danninger, Joakim Edsjö, Ben Farmer, Andrew Fowlie, Tomás E. Gonzalo, Will Handley, Sebastian Hoof, Selim Hotinli, Felix Kahlhoefer, Anders Kvellestad, Julia Harz, Paul Jackson, Farvah Mahmoudi, Greg Martinez, Are Raklev, Janina Renk, Chris Rogan, Roberto Ruiz de Austri, Pat Scott, Patrick Stöcker, Aaron Vincent, Christoph Weniger, Martin White, Yang Zhang

40+ participants in 11 experiments and 14 major theory codes

Imperial College London

Modules

Physics modules

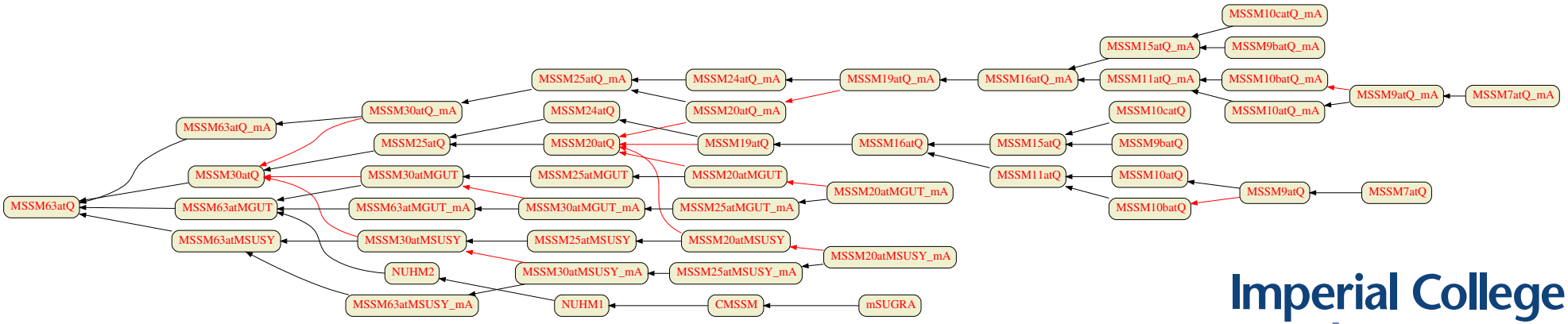
- DarkBit – dark matter observables (relic density, direct + indirect detection) (EPJC, arXiv:1705.07920)
- ColliderBit – collider observables inc. Higgs + SUSY searches from ATLAS, CMS + LEP (EPJC, arXiv:1705.07919)
- FlavBit – flavour physics inc. $g-2$, $b \to s\gamma$, B decays (new channels, angular obs., theory uncerts, LHCb likelihoods) (EPJC, arXiv:1705.07933)
- SpecBit – generic BSM spectrum object, providing RGE running, masses, mixings, etc via interchangeable interfaces to different RGE codes (EPJC, arXiv:1705.07936)
- DecayBit – decay widths for all relevant SM & BSM particles (EPJC, arXiv:1705.07936)
- PrecisionBit – SM likelihoods, precision BSM tests (W mass, $\Delta\rho$ etc) (EPJC, arXiv:1705.07936)

Each consists of a number of **module functions** with **dependencies** on each other

Imperial College London

- Models are defined by their parameters and relations to each other
- Models can inherit from (be subspaces of) **parent models**
- Points in child models can be **automatically translated** to ancestor models
- **Friend models** also allowed (cross-family translation)
- Model dependence of every function/observable is tracked

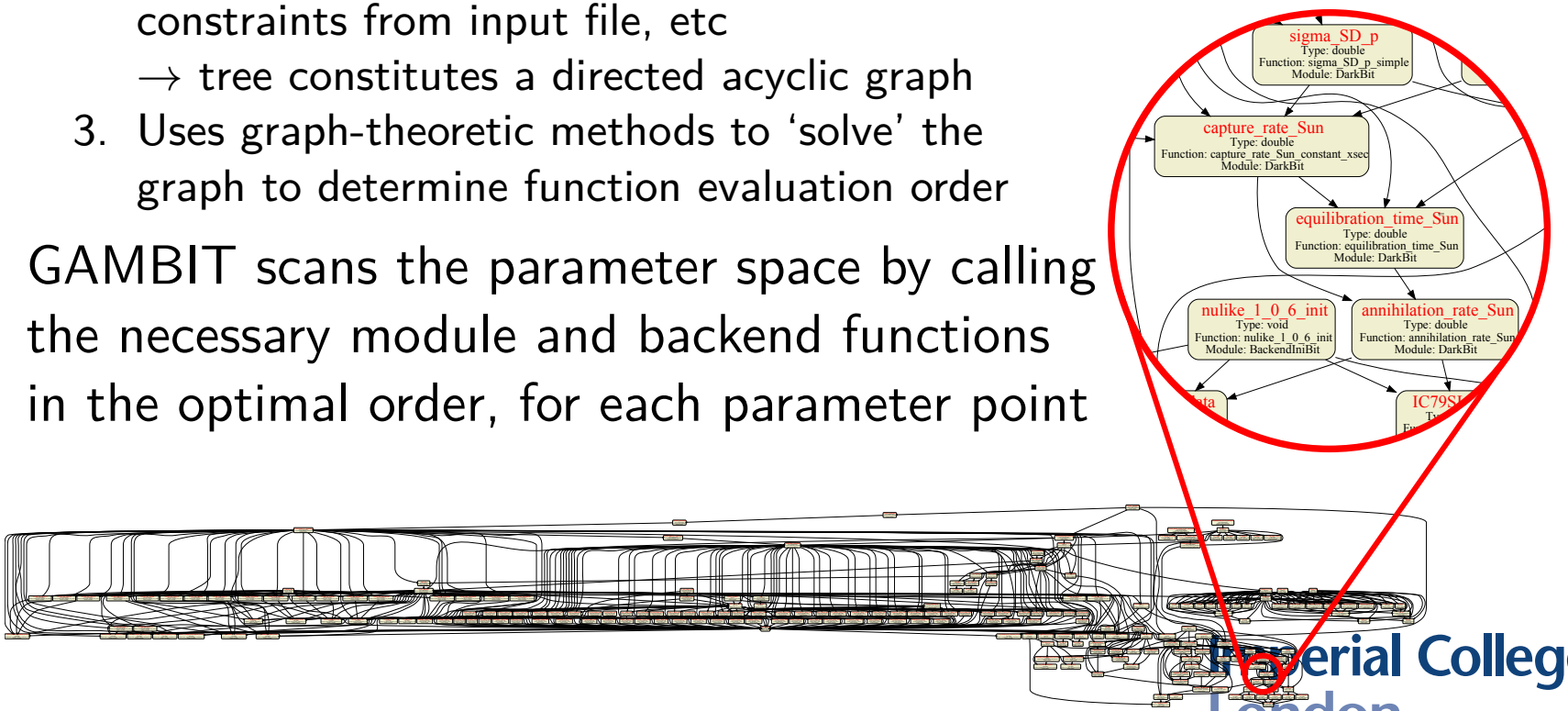
  $\Longrightarrow$ maximum safety, maximum reuse



Imperial College London

Anatomy of a GAMBIT run (EPJC, arXiv:1705.07908)

- User chooses a model to scan, which observables to include, and the scanning method
- GAMBIT constructs a **dependency tree**
  1. Identifies which functions and inputs are needed to compute the requested observables
  2. Obeys **rules** at each step: allowed models, allowed backends, constraints from input file, etc
     $\rightarrow$ tree constitutes a directed acyclic graph
  3. Uses graph-theoretic methods to 'solve' the graph to determine function evaluation order
- GAMBIT scans the parameter space by calling the necessary module and backend functions in the optimal order, for each parameter point



Backends: mix and match (EPJC, arXiv:1705.07908)

- Module functions can require specific functions from **backends**
- Backends are external code libraries (DarkSUSY, FeynHiggs, etc) that include different functions
- GAMBIT automates and abstracts the interfaces to backends $\rightarrow$ backend functions are tagged according to what they calculate
- $\rightarrow$ with appropriate module design, different backends and their functions can be used interchangeably
- GAMBIT dynamically adapts to use whichever backends are actually present on a user's system (+ provides details of what it decided to do of course)

Imperial College London

Backends: mix and match (EPJC, arXiv:1705.07908)

pat@xpspedition: ~/gambit 163x45

All relative paths are given with reference to /home/pat/gambit.

BACKENDS	VERSION	PATH TO LIB	STATUS	#FUNC	#TYPES	#CTORS
DDCalc0	0.0	Backends/installed/DDCalc/0.0/libDDCalc0.so	OK	62	0	0
DarkSUSY	5.1.1	Backends/installed/DarkSUSY/5.1.1/lib/libdarksusy.so	OK	68	0	0
FastSim	1.0	Backends/installed/fastsim/1.0/libfastsim.so	absent/broken	1	0	0
FeynHiggs	2.11	Backends/installed/FeynHiggs/2.11.2/lib/libFH.so	OK	14	0	0
HiggsBounds	4.2.1	Backends/installed/HiggsBounds/4.2.1/lib/libhiggsbounds.so	OK	10	0	0
HiggsSignals	1.4	Backends/installed/HiggsSignals/1.4.0/lib/libhiggssignals.so	OK	11	0	0
LibFarrayTest	1.0	Backends/examples/libFarrayTest.so	OK	9	0	0
LibFirst	1.0	Backends/examples/libfirst.so	OK	8	0	0
	1.1	Backends/examples/libfirst.so	OK	15	0	0
LibFortran	1.0	Backends/examples/libfortran.so	OK	6	0	0
MicrOmegas	3.5.5	Backends/installed/micromegas/3.5.5/MSSM/MSSM/libmicromegas.so	OK	15	0	0
MicrOmegasSingletDM	3.5.5	Backends/installed/micromegas/3.5.5/SingletDM/SingletDM/libmicromegas.so	OK	13	0	0
Pythia	8.186	Backends/installed/Pythia/8.186/lib/libpythia8.so	absent/broken	0	27	105
	8.209	Backends/installed/Pythia/8.209/lib/libpythia8.so	OK	0	28	107
SUSYPOPE	0.2	no path in config/backend_locations.yaml	absent/broken	3	0	0
SUSY_HIT	1.5	Backends/installed/SUSY-HIT/1.5/libsusyhit.so	OK	55	0	0
SuperIso	3.4	Backends/installed/SuperIso/3.4/libsuperiso.so	OK	32	0	0
gamLike	1.0.0	Backends/installed/gamLike/1.0.0/lib/gamLike.so	OK	3	0	0
nulike	1.0.0	Backends/installed/nulike/1.0.0/lib/libnulike.so	OK	4	0	0

Gambit diagnostic backend line 1 (press h for help or q to quit)

Backends: mix and match

(EPJC, arXiv:1705.07908)

pat@xpspedition: ~/gambit 163x45

All relative paths are given with reference to /home/pat/gambit.

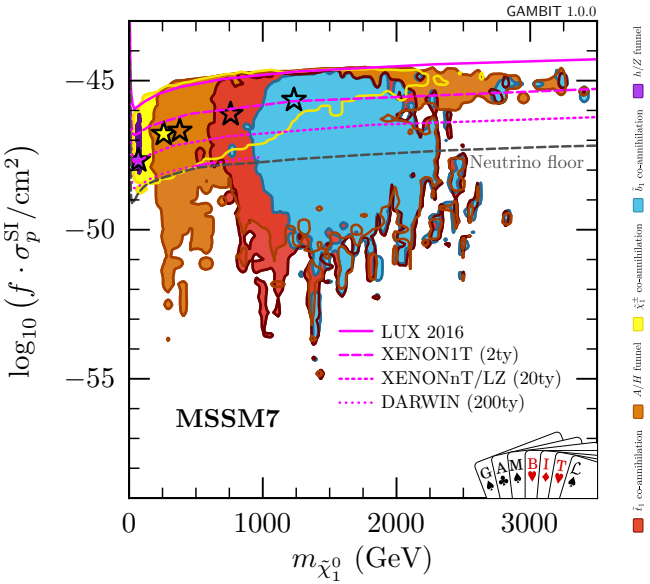
BACKENDS	VERSION	PATH TO LIB	STATUS	#FUNC	#TYPES	#CTORS
DDCalc0	0.0	Backends/installed/DDCalc/0.0/libDDCalc0.so	OK	62	0	0
DarkSUSY	5.1.1	Backends/installed/DarkSUSY/5.1.1/lib/libdarksusy.so	OK	68	0	0
FastSim	1.0	Backends/installed/fastsim/1.0/libfastsim.so	absent/broken	1	0	0
FeynHiggs	2.11	Backends/installed/FeynHiggs/2.11.2/lib/libFH.so	OK	14	0	0
HiggsBounds	4.2.1	Backends/installed/HiggsBounds/4.2.1/lib/libhiggsbounds.so	OK	10	0	0
HiggsSignals	1.4	Backends/installed/HiggsSignals/1.4.0/lib/libhiggssignals.so	OK	11	0	0
LibFarrayTest	1.0	Backends/examples/libFarrayTest.so	OK	9	0	0
LibFirst	1.0	Backends/examples/libfirst.so	OK	8	0	0
	1.1	Backends/examples/libfirst.so	OK	15	0	0
LibFortran	1.0	Backends/examples/libfortran.so	OK	6	0	0
MicrOmegas	3.5.5	Backends/installed/micromegas/3.5.5/MSSM/MSSM/libmicromegas.so	OK	15	0	0
MicrOmegasSingletDM	3.5.5	Backends/installed/micromegas/3.5.5/SingletDM/SingletDM/libmicromegas.so	OK	13	0	0
Pythia	8.186	Backends/installed/Pythia/8.186/lib/libpythia8.so	absent/broken	0	27	105
	8.209	Backends/installed/Pythia/8.209/lib/libpythia8.so	OK	0	28	107
SUSYPOPE	0.2	no path in config/backend_locations.yaml	absent/broken	3	0	0
SUSY_HIT	1.5	Backends/installed/SUSY-HIT/1.5/libsusyhit.so	OK	55	0	0
SuperIso	3.4	Backends/installed/SuperIso/3.4/libsuperiso.so	OK	32	0	0
gamLike	1.0.0	Backends/installed/gamLike/1.0.0/lib/gamLike.so	OK	3	0	0
nulike	1.0.0	Backends/installed/nulike/1.0.0/lib/libnulike.so	OK	4	0	0

Gambit diagnostic backend line 1 (press h for help or q to quit)

13 GAMBIT papers so far (since 2017)

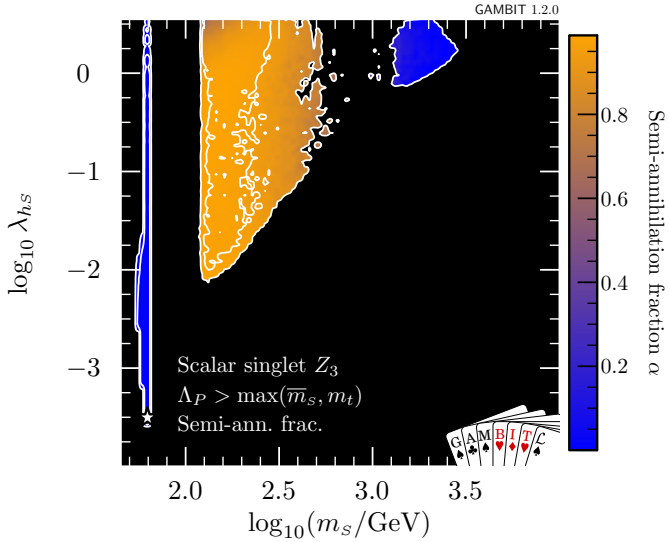
Supersymmetry

EPJC arXiv:1705.07917, 1705.07935, 1809.02097



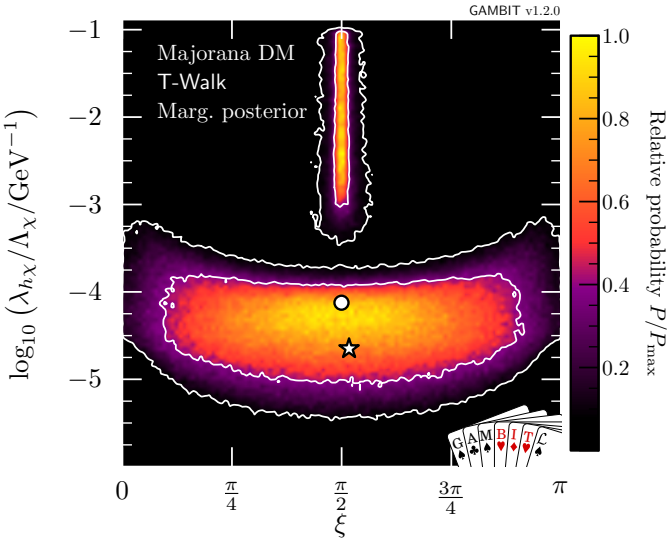
Scalar singlet dark matter

EPJC arXiv:1705.07931, 1806.11281



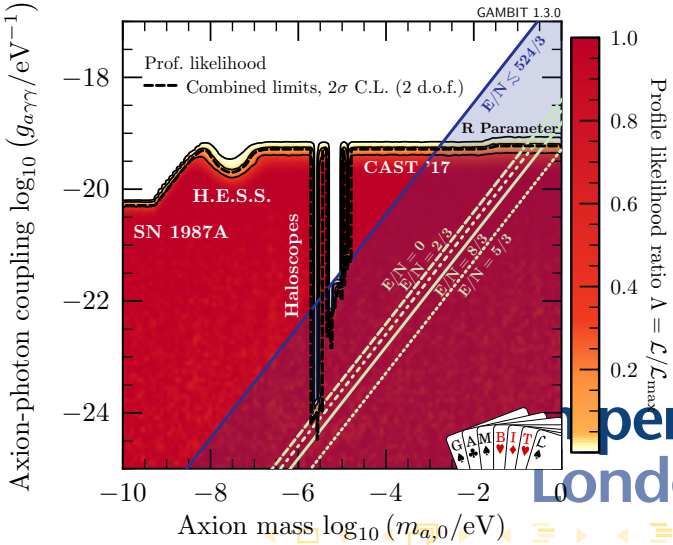
Vector & fermion Higgs-portal dark matter

EPJC arXiv:1808.10465



Axions & axion-like particles

JHEP arXiv:1810.07192



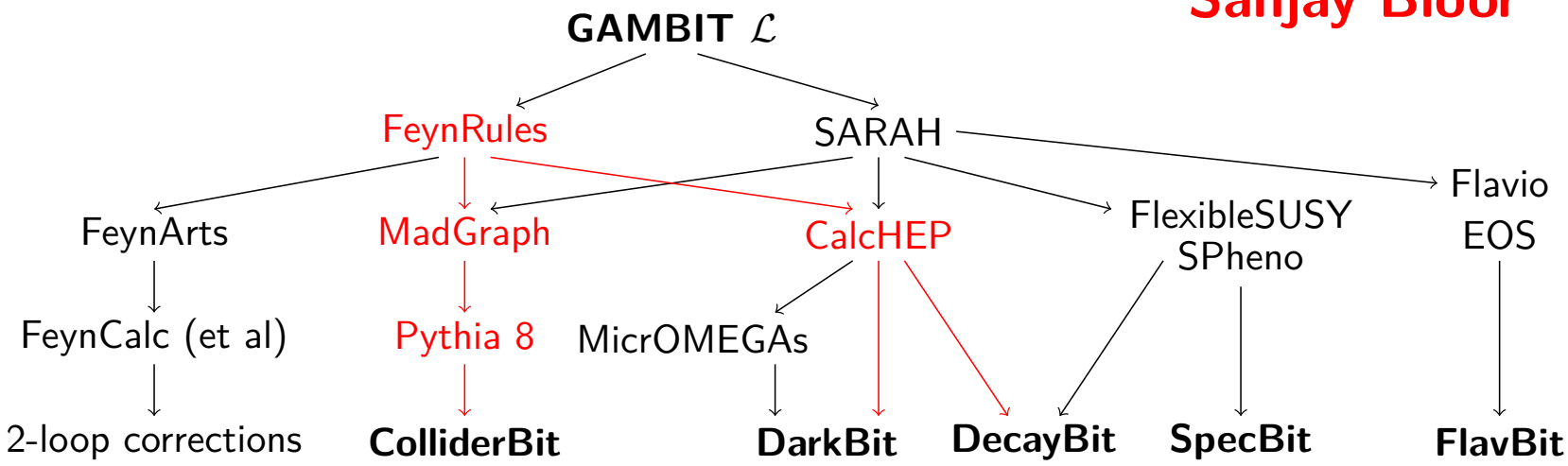
Imperial College London

GAMBIT 2.0 (under development)

$\mathcal{L}_{\rm BSM} \rightarrow$ global fit

Red = complete

Sanjay Bloor



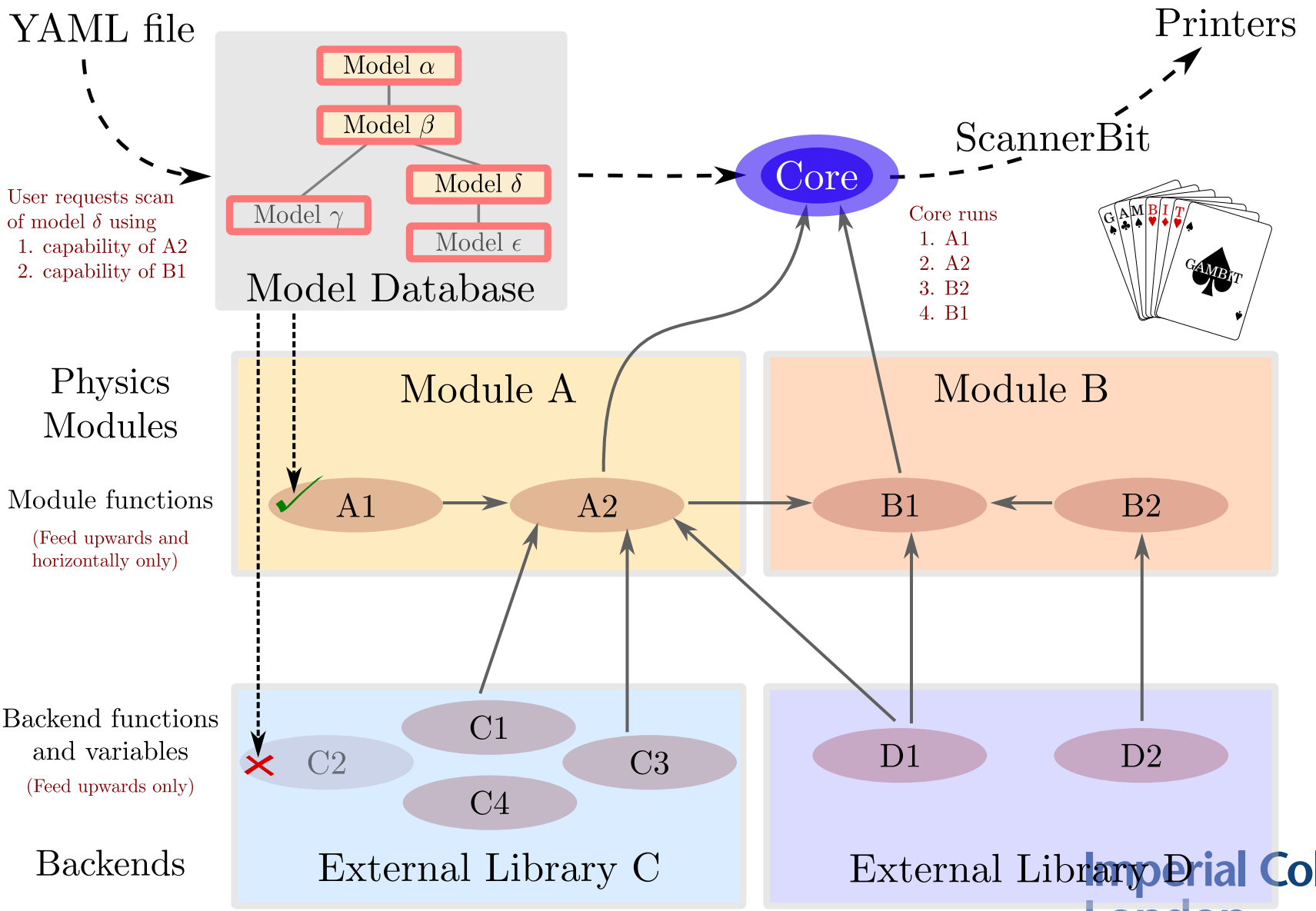
Imperial College London

Backup slides

Imperial College London

Functional overview of GAMBIT

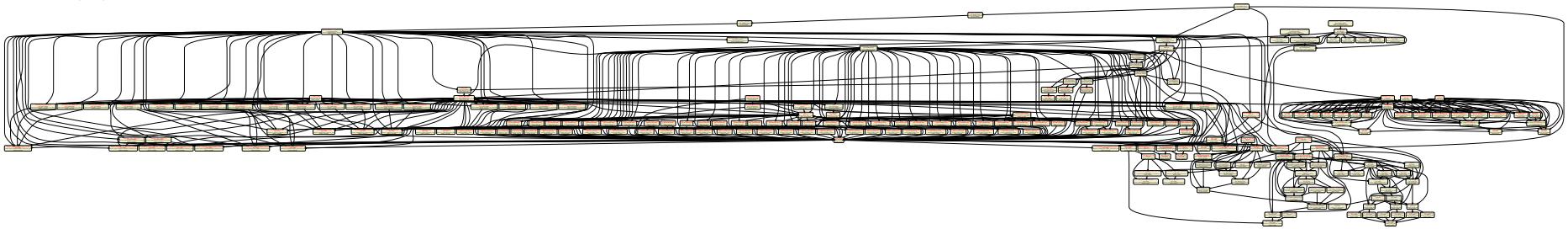
(EPJC, arXiv:1705.07908)



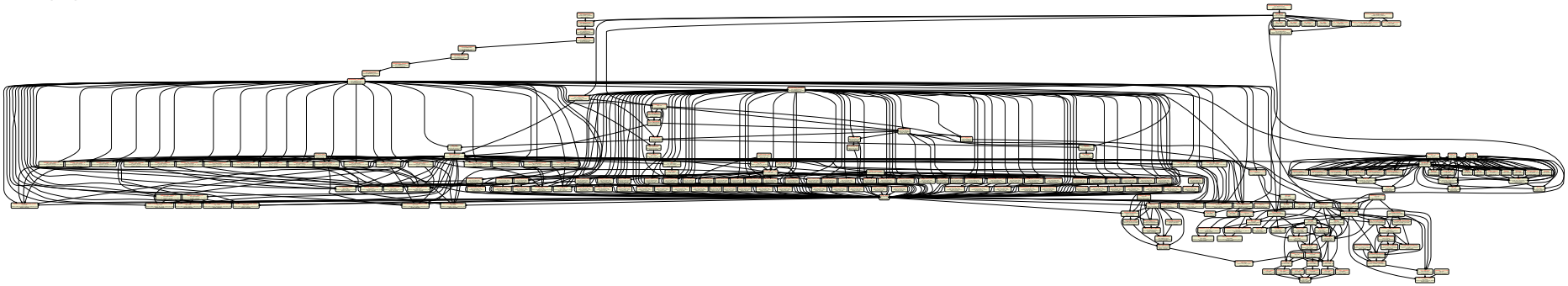
Imperial College London

Dependency Resolution

CMSSM:



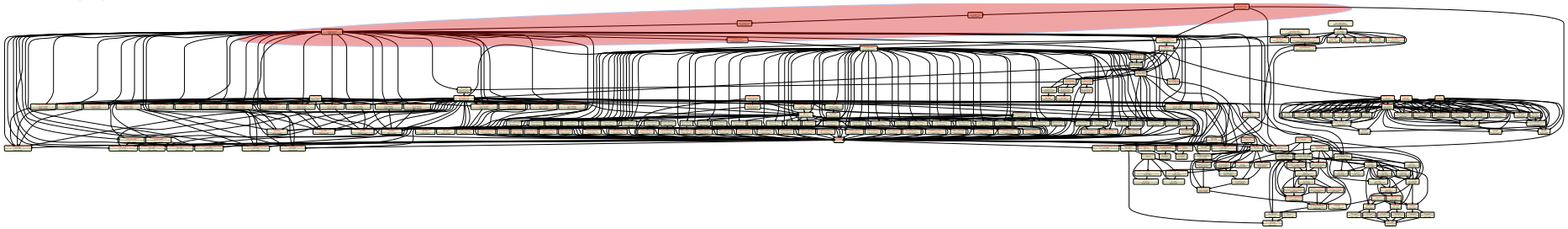
MSSM7:



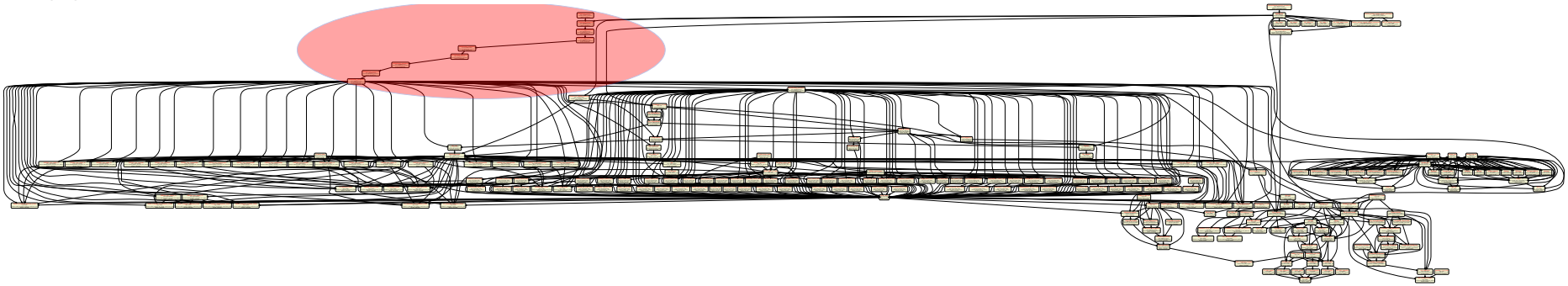
Imperial College London

Dependency Resolution

CMSSM:



MSSM7:

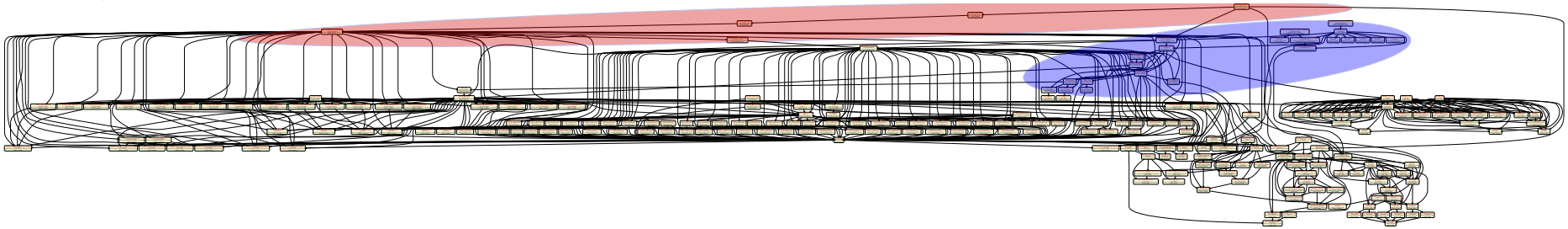


Red: Model parameter translations

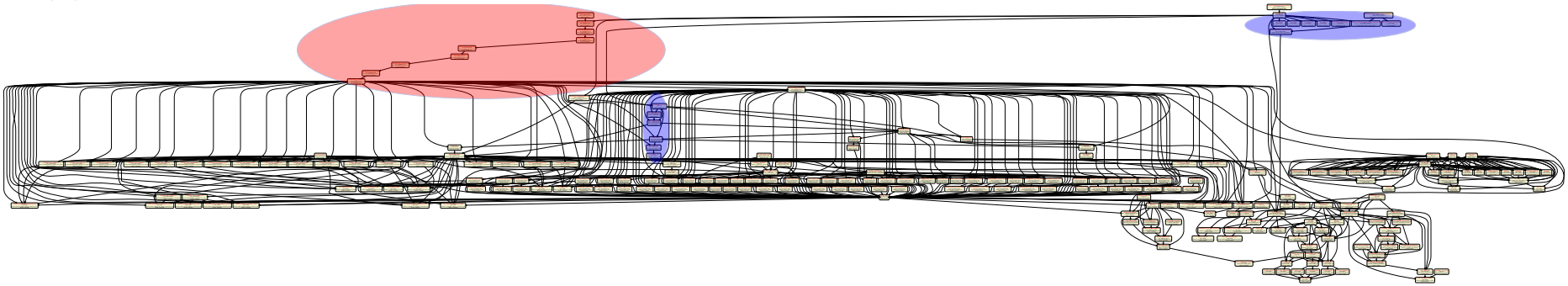
Imperial College London

Dependency Resolution

CMSSM:



MSSM7:



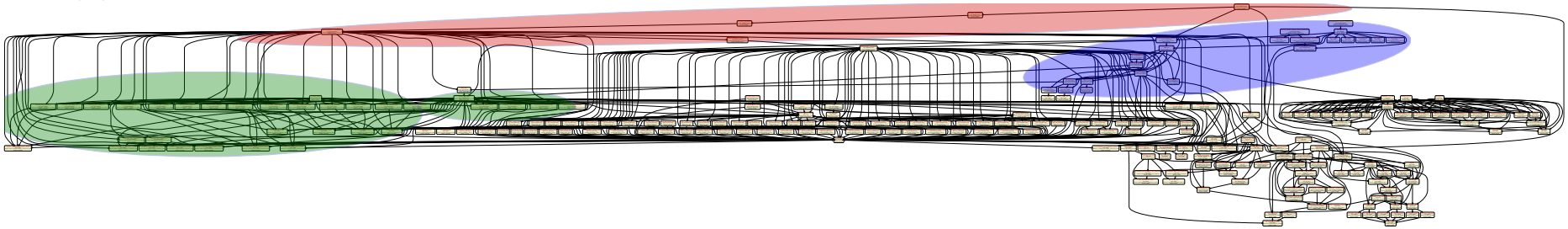
Red: Model parameter translations

Blue: Precision calculations

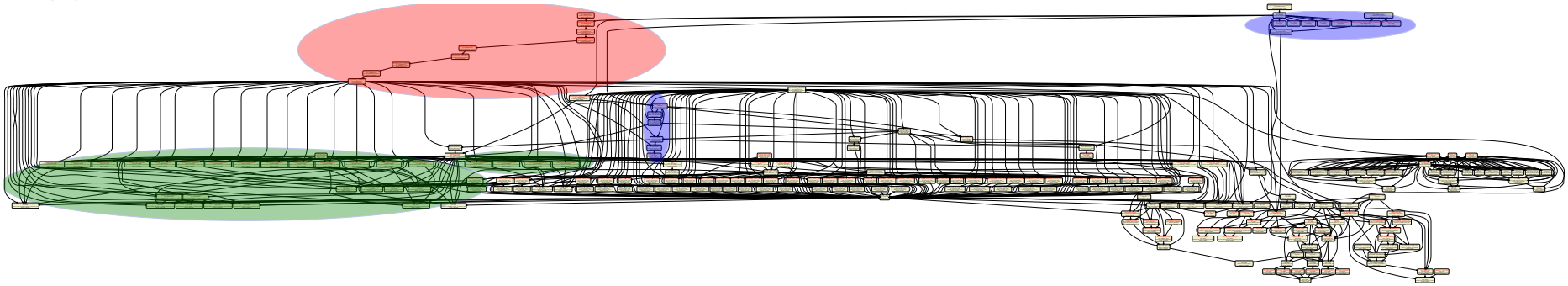
Imperial College London

Dependency Resolution

CMSSM:



MSSM7:

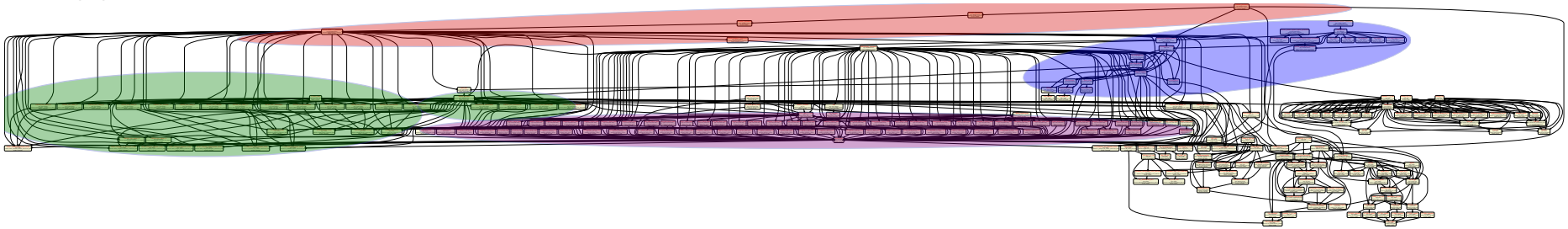


Red: Model parameter translations
Blue: Precision calculations
Green: LEP rates+likelihoods

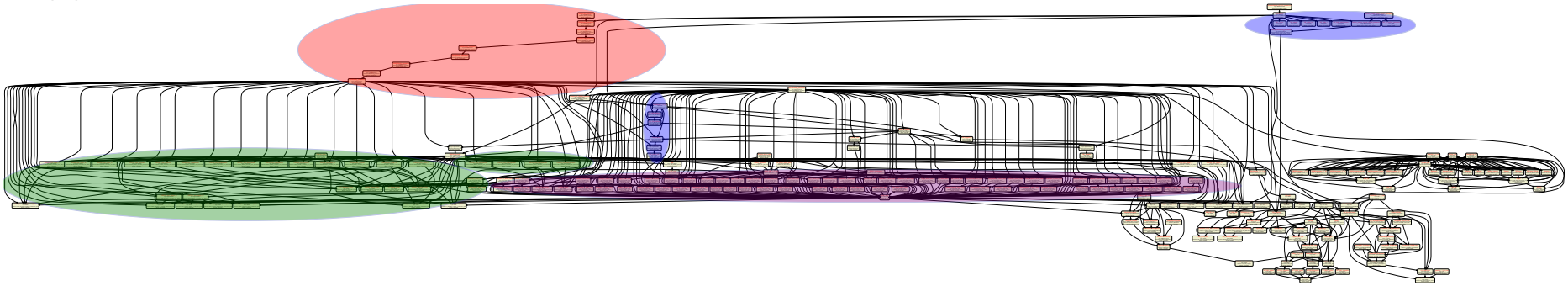
Imperial College London

Dependency Resolution

CMSSM:



MSSM7:

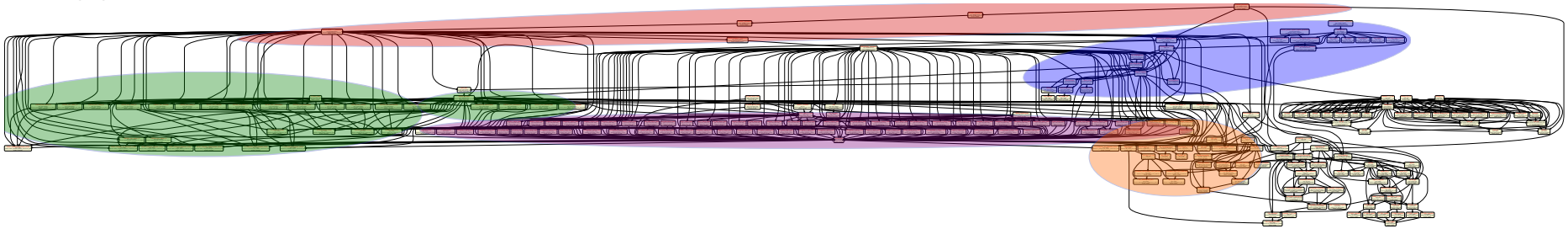


Red: Model parameter translations
Blue: Precision calculations
Green: LEP rates+likelihoods
Purple: Decays

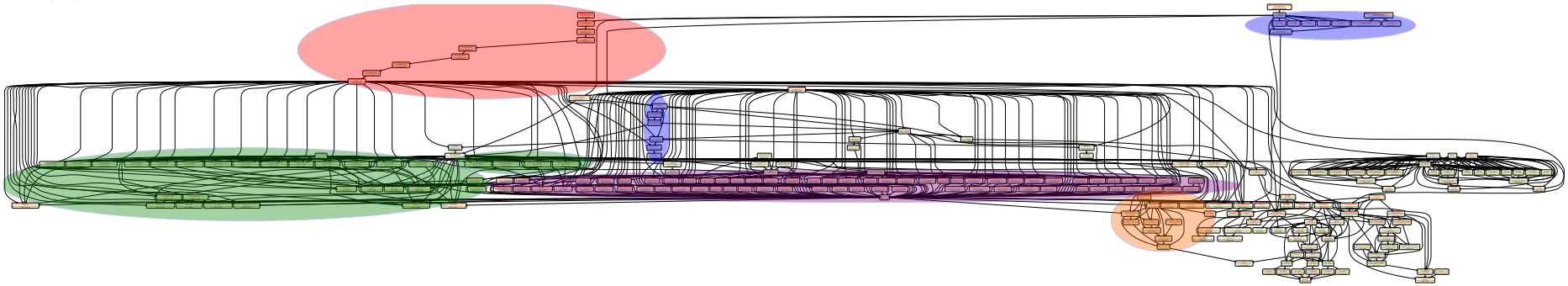
Imperial College London

Dependency Resolution

CMSSM:



MSSM7:

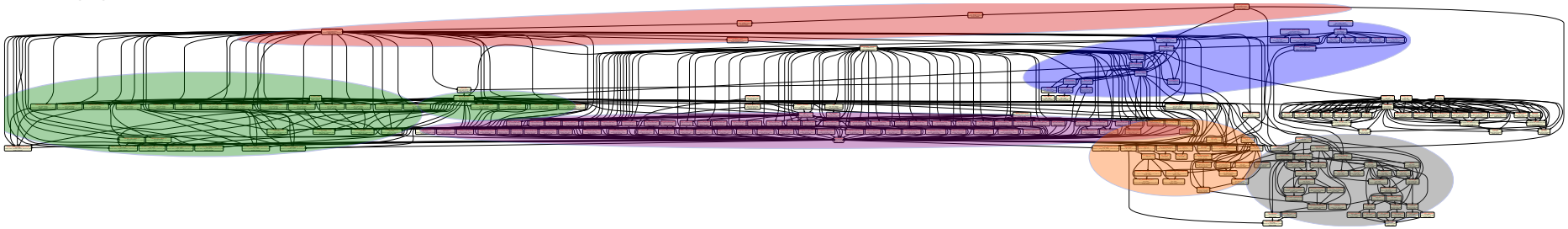


Red: Model parameter translations
Blue: Precision calculations
Green: LEP rates+likelihoods
Purple: Decays
Orange: LHC observables and likelihoods

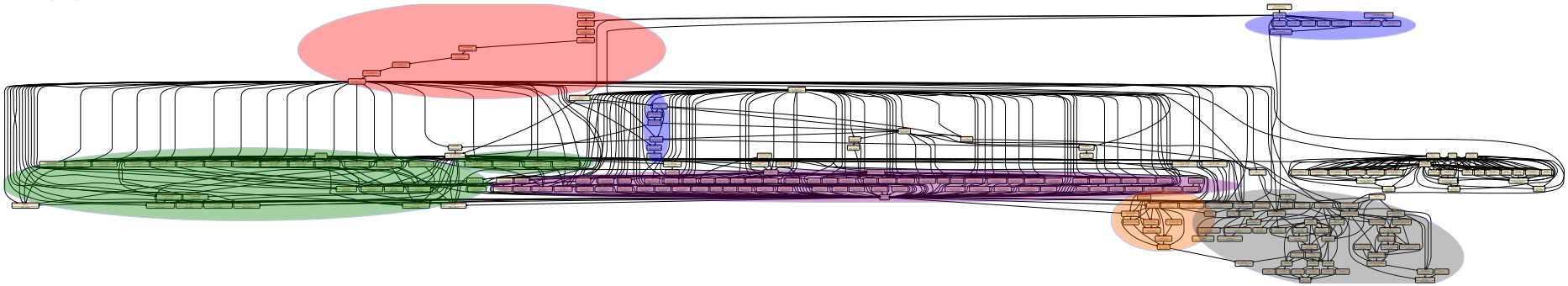
Imperial College London

Dependency Resolution

CMSSM:



MSSM7:

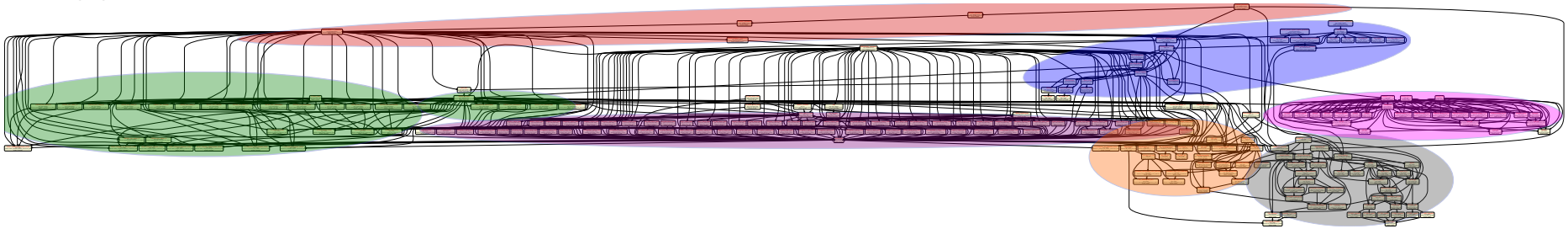


Red: Model parameter translations
Blue: Precision calculations
Green: LEP rates+likelihoods
Purple: Decays
Orange: LHC observables and likelihoods
Grey: DM direct, indirect and relic density

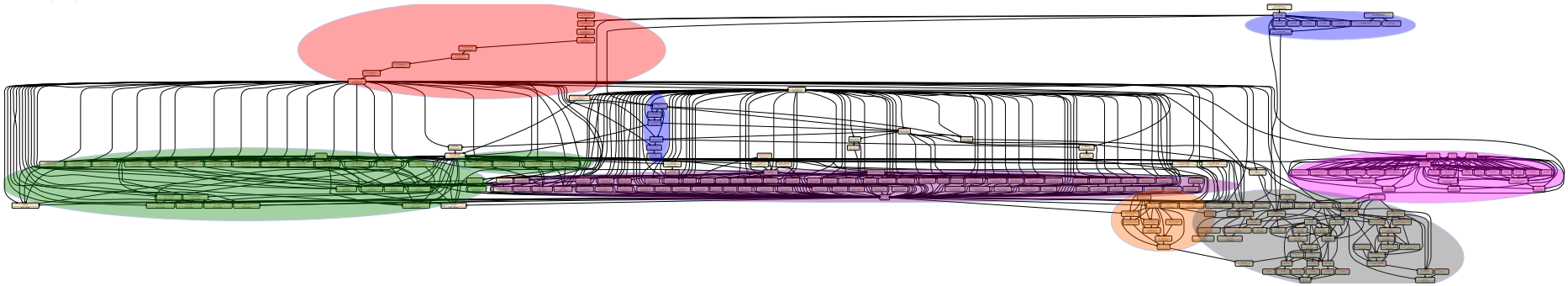
Imperial College London

Dependency Resolution

CMSSM:



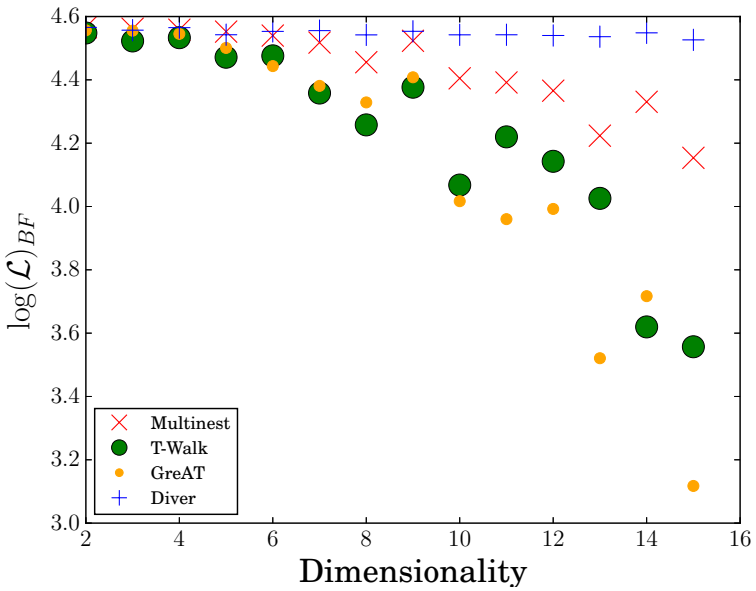
MSSM7:

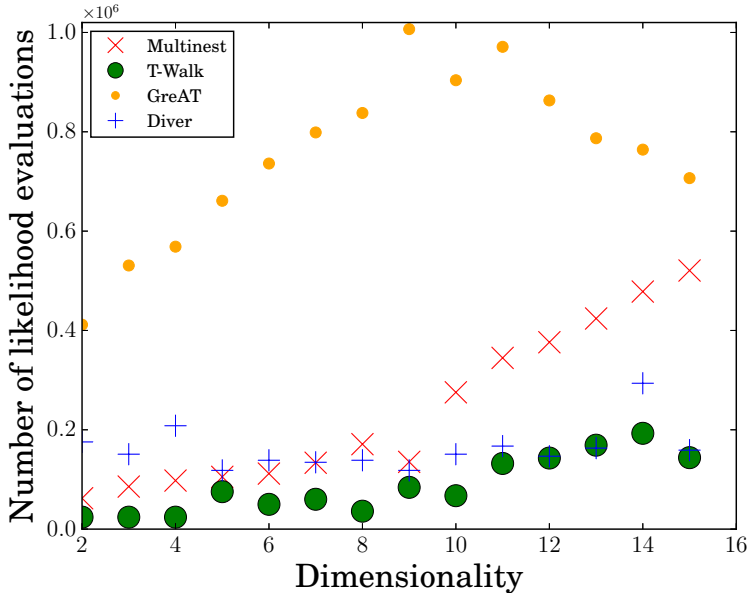


Red: Model parameter translations
Blue: Precision calculations
Green: LEP rates+likelihoods
Purple: Decays
Orange: LHC observables and likelihoods
Grey: DM direct, indirect and relic density
Pink: Flavour physics

Imperial College London

Extensive scanner tests on scalar singlet model with different numbers of nuisance paramters





Diver scales far better with dimensionality than MultiNest or other scanners

Imperial College London

Expansion: adding new observables and likelihoods

Adding a new module function is easy:

1. Declare the function to GAMBIT in a module's **rollcall header**
   - Choose a capability
   - Declare any **backend requirements**
   - Declare any **dependencies**
   - Declare any specific **allowed models**
   - other more advanced declarations also available

```
#define MODULE FlavBit                                  // A tasty GAMBIT module.
START_MODULE

  #define CAPABILITY Rmu                                // Observable: BR(K->mu nu)/BR(pi->mu nu)
  START_CAPABILITY
    #define FUNCTION SI_Rmu                             // Name of a function that can compute Rmu
    START_FUNCTION(double)                              // Function computes a double precision result
    BACKEND_REQ(Kmunu_pimunu, (my_tag), double, (const parameters*)) // Needs function from a backend
    BACKEND_OPTION( (SuperIso, 3.6), (my_tag) )                      // Backend must be SuperIso 3.6
    DEPENDENCY(SuperIso_modelinfo, parameters)          // Needs another function to calculate SuperIso info
    ALLOW_MODELS(MSSM63atQ, MSSM63atMGUT)               // Works with weak/GUT-scale MSSM and descendents
    #undef FUNCTION
  #undef CAPABILITY
```

2. Write the function as a standard C++ function (one argument: the result)

Imperial College London

Expansion: adding new models

1. Add the model to the **model hierarchy**:
   - Choose a model name, and declare any **parent model**
   - Declare the model's parameters
   - Declare any **translation function** to the parent model

```
#define MODEL NUHM1
#define PARENT NUHM2
  START_MODEL
  DEFINEPARS(M0,M12,mH,A0,TanBeta,SignMu)
  INTERPRET_AS_PARENT_FUNCTION(NUHM1_to_NUHM2)
#undef PARENT
#undef MODEL
```

2. Write the translation function as a standard C++ function:

```
void MODEL_NAMESPACE::NUHM1_to_NUHM2 (const ModelParameters &myP, ModelParameters &targetP)
{
   // Set M0, M12, A0, TanBeta and SignMu in the NUHM2 to the same values as in the NUHM1
   targetP.setValues(myP,false);
   // Set the values of mHu and mHd in the NUHM2 to the value of mH in the NUHM1
   targetP.setValue("mHu", myP["mH"]);
   targetP.setValue("mHd", myP["mH"]);
}
```

3. If needed, declare that existing module functions work with the new model, or add new functions that do.

Imperial College London

Interface: yaml file

Basic interface for a scan is a YAML initialisation file

- specify parameters, ranges, priors
- select likelihood components
- select other observables to calculate
- define generic rules for how to fill dependencies
- define generic rules for options to be passed to module functions
- set global options (scanner, errors/warnings, logging behaviour, etc)

```yaml
Parameters:
  StandardModel_SLHA2: !import StandardModel_SLHA2_default
  MSSM25atQ: !import LesHouches.in.MSSM_1.yaml

Priors:
    # none: all parameters fixed in this example.

Scanner:

  use_scanner: toy_mcmc

  scanners:
    toy_mcmc:
      plugin: toy_mcmc
      point_number: 2000
      output_file:  output
      like:  Likelihood

ObsLikes:

  # Test DecayBit
  - purpose:      Test
    capability:   decay_rates
    type:         DecayTable

  # 79-string IceCube likelihood
  - capability: IceCube_likelihood
    purpose: Likelihood
    function: IC79_loglike

Rules:

  - capability: MSSM_spectrum
    function: get_MSSMatQ_spectrum
    options:
      invalid_point_fatal: true
```

Interface: yaml file

Basic interface for a scan is a YAML initialisation file

- specify parameters, ranges, priors
- select likelihood components
- select other observables to calculate
- define generic rules for how to fill dependencies
- define generic rules for options to be passed to module functions
- set global options (scanner, errors/warnings, logging behaviour, etc)

```
Parameters:
  StandardModel_SLHA2: !import StandardModel_SLHA2_default
  MSSM25atQ: !import LesHouches.in.MSSM_1.yaml

Priors:
    # none: all parameters fixed in this example.

Scanner:

  use_scanner: toy_mcmc

  scanners:
    toy_mcmc:
      plugin: toy_mcmc
      point_number: 2000
      output_file:  output
      like:  Likelihood

ObsLikes:

  # Test DecayBit
  - purpose:      Test
    capability:   decay_rates
    type:         DecayTable

  # 79-string IceCube likelihood
  - capability: IceCube_likelihood
    purpose: Likelihood
    function: IC79_loglike

Rules:

  - capability: MSSM_spectrum
    function: get_MSSMatQ_spectrum
    options:
      invalid_point_fatal: true
```

Interface: yaml file

Basic interface for a scan is a YAML initialisation file

- specify parameters, ranges, priors
- select likelihood components
- select other observables to calculate
- define generic rules for how to fill dependencies
- define generic rules for options to be passed to module functions
- set global options (scanner, errors/warnings, logging behaviour, etc)

```yaml
Parameters:
  StandardModel_SLHA2: !import StandardModel_SLHA2_default
  MSSM25atQ: !import LesHouches.in.MSSM_1.yaml

Priors:
    # none: all parameters fixed in this example.

Scanner:

  use_scanner: toy_mcmc

  scanners:
    toy_mcmc:
      plugin: toy_mcmc
      point_number: 2000
      output_file:  output
      like:  Likelihood

ObsLikes:

  # Test DecayBit
  - purpose:      Test
    capability:   decay_rates
    type:         DecayTable

  # 79-string IceCube likelihood
  - capability: IceCube_likelihood
    purpose: Likelihood
    function: IC79_loglike

Rules:

  - capability: MSSM_spectrum
    function: get_MSSMatQ_spectrum
    options:
      invalid_point_fatal: true
```

Interface: yaml file

Basic interface for a scan is a YAML initialisation file

- specify parameters, ranges, priors
- select likelihood components
- select other observables to calculate
- define generic rules for how to fill dependencies
- define generic rules for options to be passed to module functions
- set global options (scanner, errors/warnings, logging behaviour, etc)

```yaml
Parameters:
  StandardModel_SLHA2: !import StandardModel_SLHA2_default
  MSSM25atQ: !import LesHouches.in.MSSM_1.yaml

Priors:
    # none: all parameters fixed in this example.

Scanner:

  use_scanner: toy_mcmc

  scanners:
    toy_mcmc:
      plugin: toy_mcmc
      point_number: 2000
      output_file:  output
      like:  Likelihood

ObsLikes:

  # Test DecayBit
  - purpose:      Test
    capability:   decay_rates
    type:         DecayTable

  # 79-string IceCube likelihood
  - capability: IceCube_likelihood
    purpose: Likelihood
    function: IC79_loglike

Rules:

  - capability: MSSM_spectrum
    function: get_MSSMatQ_spectrum
    options:
      invalid_point_fatal: true
```

London

Interface: yaml file

Basic interface for a scan is a YAML initialisation file

- specify parameters, ranges, priors
- select likelihood components
- select other observables to calculate
- define generic rules for how to fill dependencies
- define generic rules for options to be passed to module functions
- set global options (scanner, errors/warnings, logging behaviour, etc)

```yaml
Parameters:
  StandardModel_SLHA2: !import StandardModel_SLHA2_default
  MSSM25atQ: !import LesHouches.in.MSSM_1.yaml

Priors:
    # none: all parameters fixed in this example.

Scanner:

  use_scanner: toy_mcmc

  scanners:
    toy_mcmc:
      plugin: toy_mcmc
      point_number: 2000
      output_file:  output
      like:  Likelihood

ObsLikes:

  # Test DecayBit
  - purpose:      Test
    capability:   decay_rates
    type:         DecayTable

  # 79-string IceCube likelihood
  - capability: IceCube_likelihood
    purpose: Likelihood
    function: IC79_loglike

Rules:

  - capability: MSSM_spectrum
    function: get_MSSMatQ_spectrum
    options:
      invalid_point_fatal: true
```

Other nice technical features

- **Scanners**: Nested sampling, differential evolution, MCMC, t-walk...
- Mixed-mode **MPI + openMP** parallelisation, mostly automated → scales to 10k+ cores
- diskless generalisation of various Les Houches Accords
- **BOSS**: dynamic loading of C++ classes from backends (!)
- **all-in or module standalone** modes – easily implemented from single cmake script
- **automatic getters** for obtaining, configuring + compiling backends[1]
- **flexible output streams** (ASCII, databases, HDF5, ...)
- available as docker plugin or vagrant virtual machine
- more more more...

[1] if a backend won't compile/crashes/kills your cat, blame the authors (not us...except where we **are** the authors...)

Imperial College London

ColliderBit details

LEP likelihoods

- complete model-independent recast of direct sparticle searches

Higgs likelihoods:

- for now: HiggSignals + HiggsBounds + constraints from invisible fits (Bernon, Dumont, Kraml et al)
- future: full simulation and ATLAS+CMS combination, more correlations, CP info, no SM-like coupling assumptions

Fast LHC likelihoods

- no simplified models, just faster direct simulation

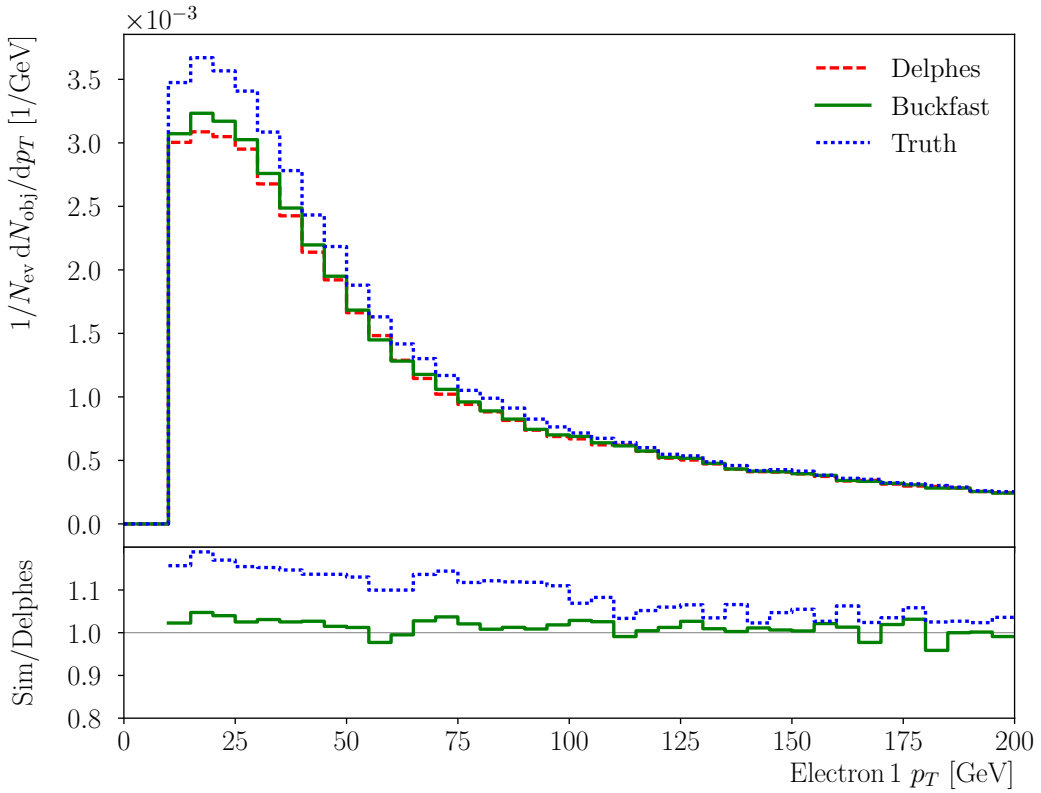
Imperial College London

ColliderBit details

LHC likelihoods:

- MC generation: Pythia8 parallelised with OpenMP + other speed tweaks
- Detector simulation: fast simulation based on 4-vector smearing
  $\rightarrow$ matches DELPHES results very closely (but much faster!)

Leading electron p_{T} distribution (CMSSM example):
red: detector-level simulation with DELPHES
green: 4-vector smearing with GAMBIT
blue: truth-level distribution



ColliderBit details

LHC likelihoods:

- MC generation: Pythia8 parallelised with OpenMP + other speed tweaks
- Detector simulation: fast simulation based on 4-vector smearing
  $\rightarrow$ matches DELPHES results very closely (but much faster!)
- Cross-sections: LO + LL from MC generator by default (fast NLO in works for SUSY)
- Analysis framework: custom event-level, independent of experiment or simulation
- Likelihood: inline systematic error marginalisation (via `nulike`)
- v1.0 shipped with:
  - ATLAS SUSY searches (0ℓ, $0/1/2\ell$ $\tilde{t}$, b jets + MET, $2/3\ell$ EW)
  - CMS multi-ℓ SUSY
  - CMS DM (t pair + MET, mono-b, monojet)
  - ATLAS + CMS Run II 0ℓ
- v1.2 now released with a bucketload of additional Run II analyses, 80 fb^{-1} analyses coming soon too

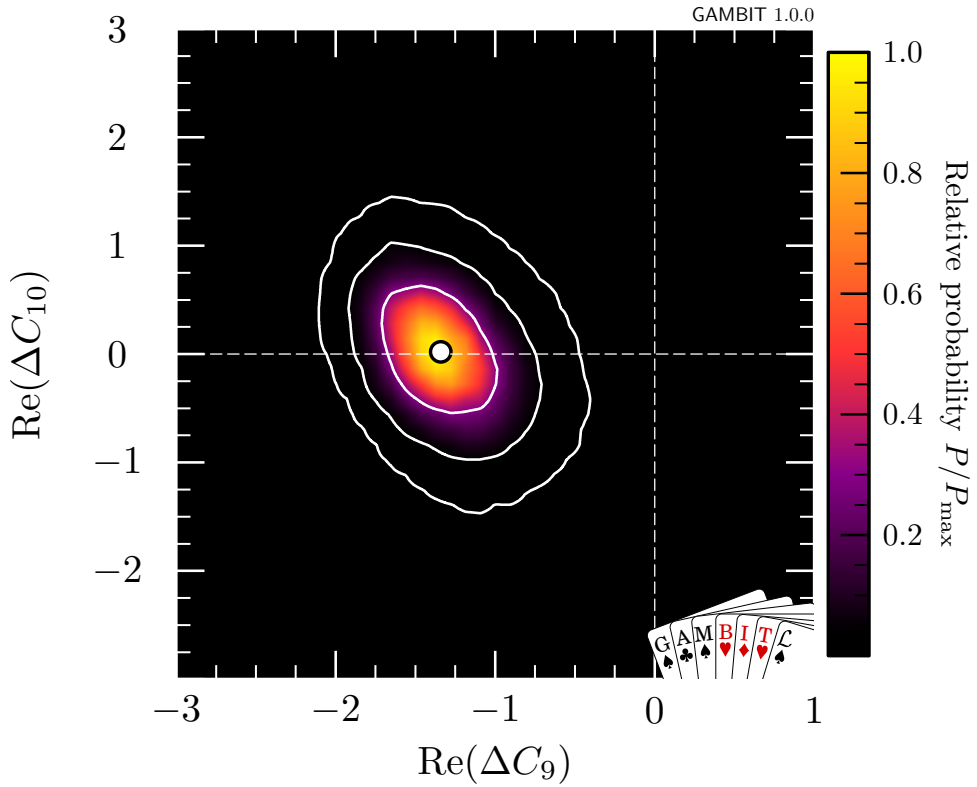
Imperial College London

Flavour physics global fits

LHCb sees possible hints of lepton flavour non-universality in neutral currents
→ GAMBIT flavour EFT global fit (Wilson coefficients as model parameters)

Flavour likelihoods in GAMBIT:

- $(g-2)_\mu$
- $B \to X_s\gamma$
- $B \to \mu\mu$
- $B_s \to \mu\mu$
- $B \to K^*\mu\mu$ + angular observables
- $B \to \tau\nu$
- $B \to D\mu\nu$
- $B \to D\tau\nu$
- $B \to D^*\mu\nu$
- $B \to D^*\tau\nu$
- $D \to \mu\nu$
- $D_s \to \mu\nu$
- $D_s \to \tau\nu$
- $\dfrac{\mathcal{B}(K \to \mu\nu)}{\mathcal{B}(\pi \to \mu\nu)}$

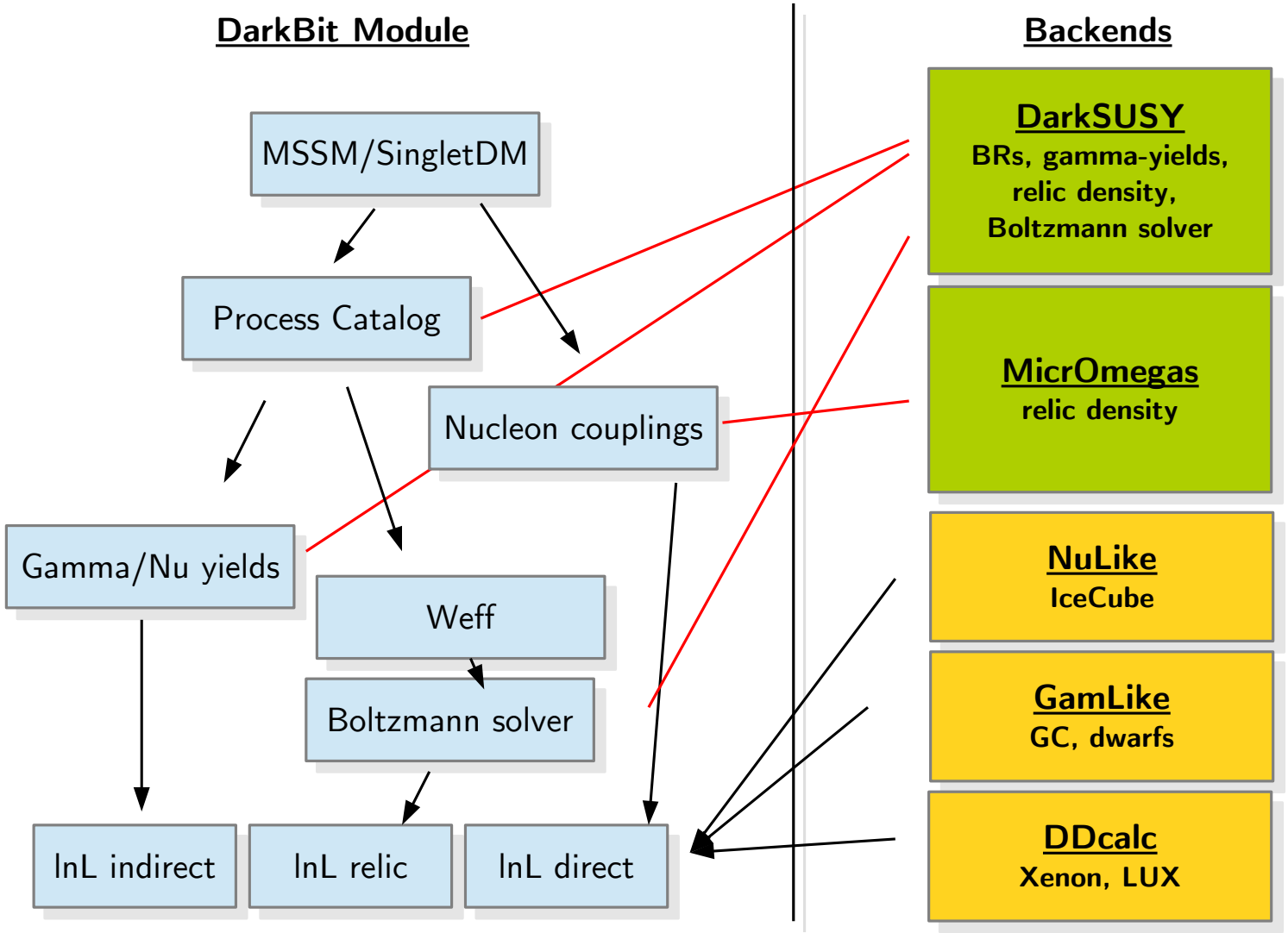


(EPJC, arXiv:1705.07933)

Fit to $\mathcal{O}_7$ (photons), $\mathcal{O}_9$ (leptons, vector), $\mathcal{O}_{10}$ (leptons, axial-vector)

Imperial College London

DarkBit overview



Backends – gamLike

C++ library with simple interface to most relevant likelihood functions from Fermi LAT and IACTs

Particle physics input:

$$\frac{1}{m_\chi^2}\frac{d\sigma v}{dE}(v, E)$$

Output: lnL

Uncertainties in the DM distribution (or astrophysical foregrounds) are internally marginalized over.

Correct treatment of energy dispersion and spectral singularities (lines, virtual internal Bremsstrahlung, boxes).