GAMBIT

(Global and Modular BSM Inference Tool)

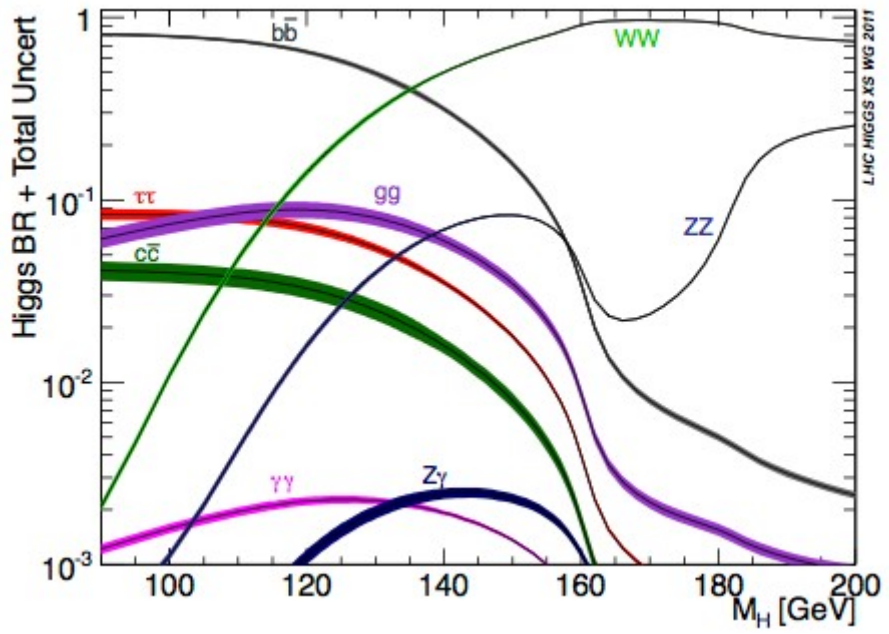
Martin White

YETI 2018

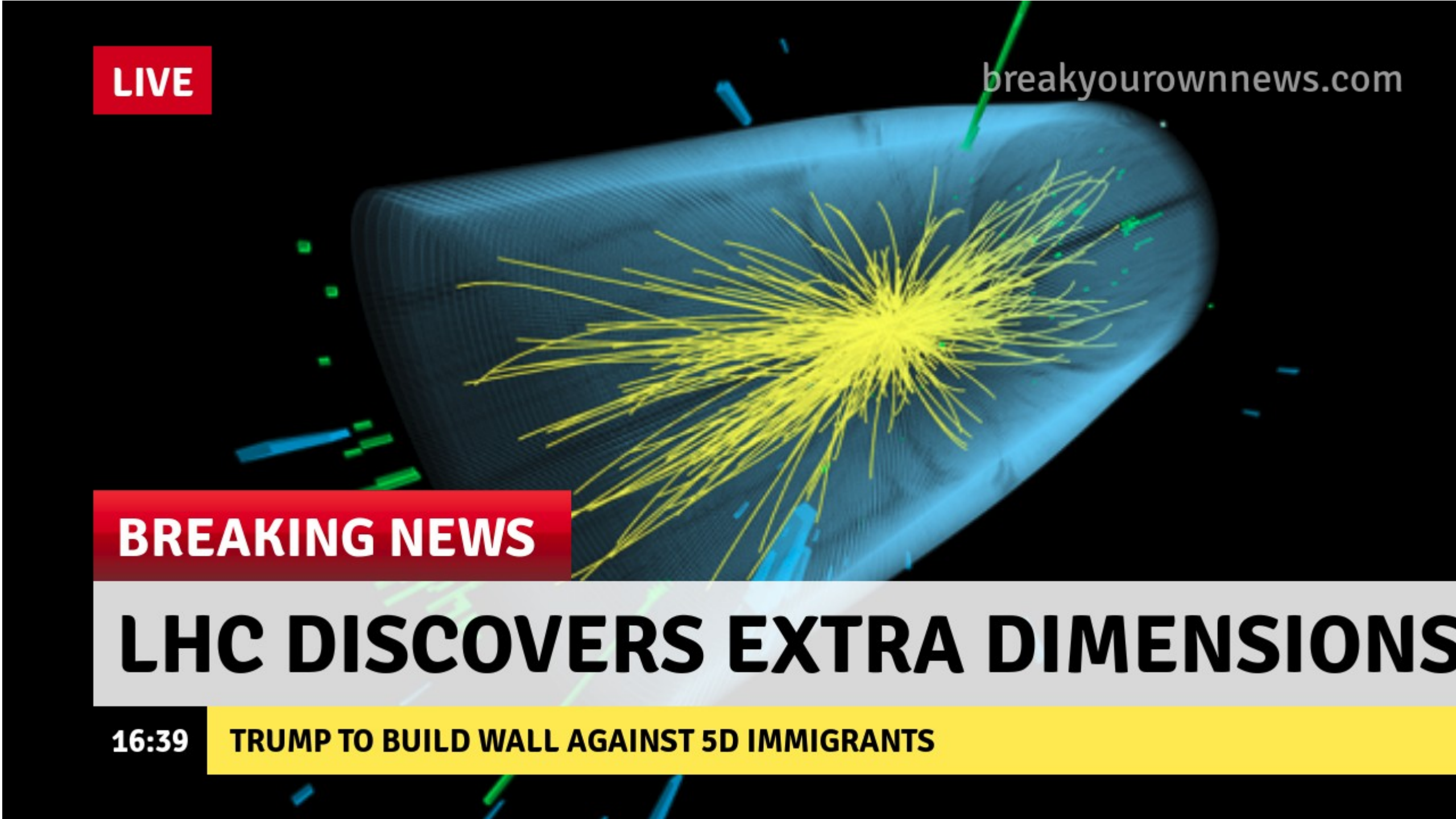
The Higgs discovery in 2012 gave us:

- The last piece of the SM
- The last occasion when we could unambiguously tell what a collider has discovered

What happens if the LHC sees a dramatic missing energy signature?




LIVE
breakyourownnews.com
BREAKING NEWS
LHC DISCOVERS SUPERSYMMETRY
16:36
GORDON KANE "PREDICTED MASS SPECTRUM IN 2003"


LIVE
breakyourownnews.com
BREAKING NEWS
LHC DISCOVERS EXTRA DIMENSIONS
16:39
TRUMP TO BUILD WALL AGAINST 5D IMMIGRANTS


LIVE
breakyourownnews.com
BREAKING NEWS
LHC DISCOVERS DARK MATTER
10:33
EXCEPT IT MIGHT NOT BE: WE ONLY KNOW IT IS STABLE ON DETECTOR TIMESCALES

Possible discoveries and assumptions

- We might discover something decaying visibly:

Default assumption: something to do with EWSB

- We might discover something decaying (semi-) invisibly

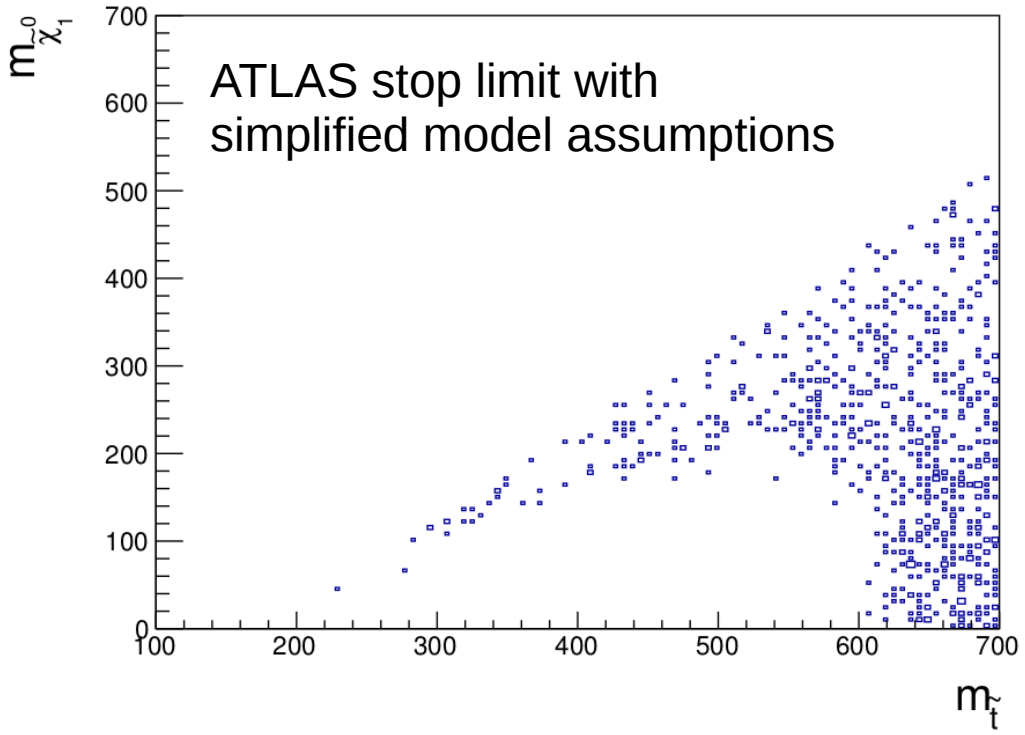
Default assumption: something to do with DM

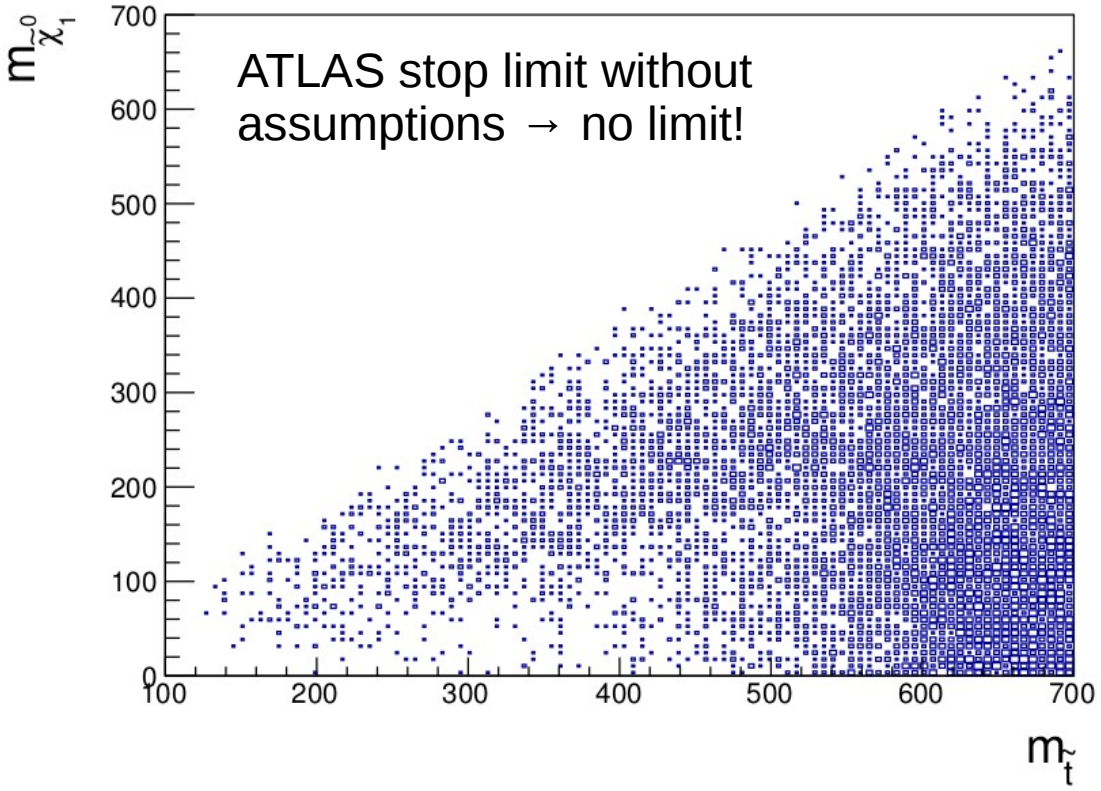
- We might discover nothing extra at all at the LHC

How do we make further progress?

What about LHC non-discoveries?

- They tell us a lot, but are infamously hard to reinterpret – how should we do that?





The answer: use as much data as possible

- Combine ATLAS+CMS null and positive results to test specific theories
- Don't forget LHCb!
- Don't forget other experiments...

Other experiments

- low-energy accelerators
- measurements of the magnetic moment of the muon
- beam dump/fixed target
- electroweak precision tests
- dark matter direct detection experiments
- searches for antimatter in cosmic rays
- nuclear cosmic ray ratios
- radio astronomy data
- effects of dark matter on reionisation, recombination and helioseismology
- the observed dark matter cosmological abundance
- neutrino masses and mixings
- gamma ray searches (e.g. FERMI-LAT, HESS, CTA, etc)

How to combine data

- Correct answer is to use a global statistical fit
- Frequentist or Bayesian methods available
- Calculate a **combined likelihood**:

$$\mathcal{L} = \mathcal{L}_{\text{collider}} \mathcal{L}_{\text{DM}} \mathcal{L}_{\text{flavor}} \mathcal{L}_{\text{EWPO}} \cdots$$

Parameter estimation

Given a particular model, which set of parameters best fits the available data

(Rigorous exclusion limits and parameter measurements)

Model comparison

Given a set of models, which is the best description of the data, and how much better is it?

(Model *X* is now worse than model *Y*)

The dream

Global fit results

- Recent years have seen an explosion of tools that make study of user-defined Lagrangians easier

  - e.g. Feynrules → Madgraph, CalcHEP → Micromegas, MadDM, NLOCT + much, much more

- Even so, a general global fit tool requires some very tricky innovations:

  - calculations are not allowed to know about Lagrangian parameters – how do you do that?
  - how do you make an easy interface for tying existing code together?
  - how do you store parameters in a scale independent way, but reintroduce scales in calculations?
  - how do you make LHC constraints model independent?
  - how do you make astrophysical constraints model independent?
  - ***how do we do all of this fast enough to get convergence within the age of the universe?***

GAMBIT: The Global And Modular BSM Inference Tool

gambit.hepforge.org

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

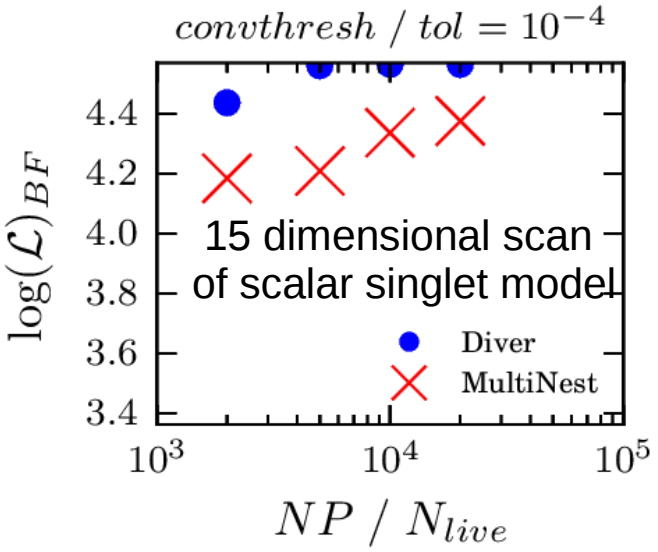
ATLAS	F. Bernlochner, A. Buckley, P. Jackson, M. White
LHCb	M. Chrząszcz, N. Serra
Belle-II	F. Bernlochner, P. Jackson
Fermi-LAT	J. Conrad, J. Edsjö, G. Martinez, P. Scott
CTA	C. Balázs, T. Bringmann, M. White
CMS	C. Rogan
IceCube	J. Edsjö, P. Scott
XENON/DARWIN	B. Farmer, R. Trotta
Theory	P. Athron, C. Balázs, S. Bloor, T. Bringmann, J. Cornell, J. Edsjö, B. Farmer, A. Fowlie, T. Gonzalo, J. Harz, S. Hoof, F. Kahlhoefer, S. Krishnamurthy, A. Kvellestad, F.N. Mahmoudi, J. McKay, A. Raklev, R. Ruiz, P. Scott, R. Trotta, A. Vincent, C. Weniger, M. White, S. Wild

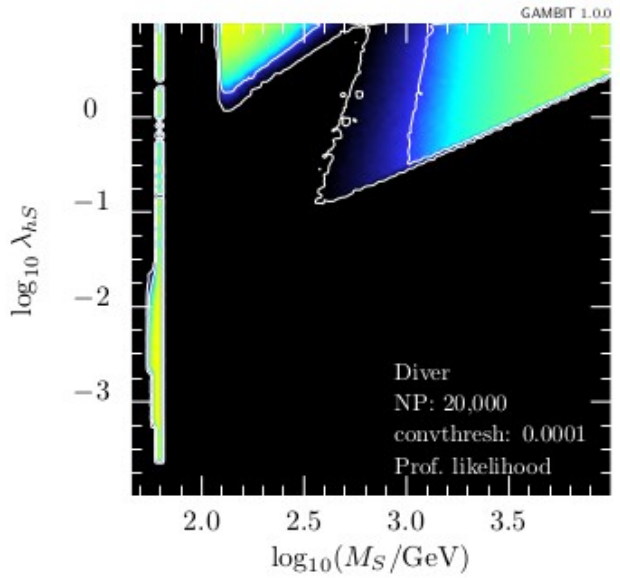
31 Members in 9 Experiments, 12 major theory codes, 11 countries

Global

- Complete global statistical fit framework
- Can be Bayesian, Frequentist or other (random, grid, etc)
- Interfaced to the best + fastest scanners available:

  Multinest, MCMC, Diver (new differential evolution scanner)

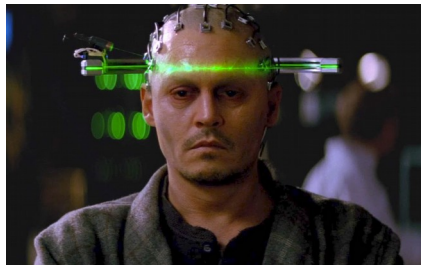




Publication ready plots available using *pippi* plotting code on the GAMBIT HDF5 output

Global and Modular

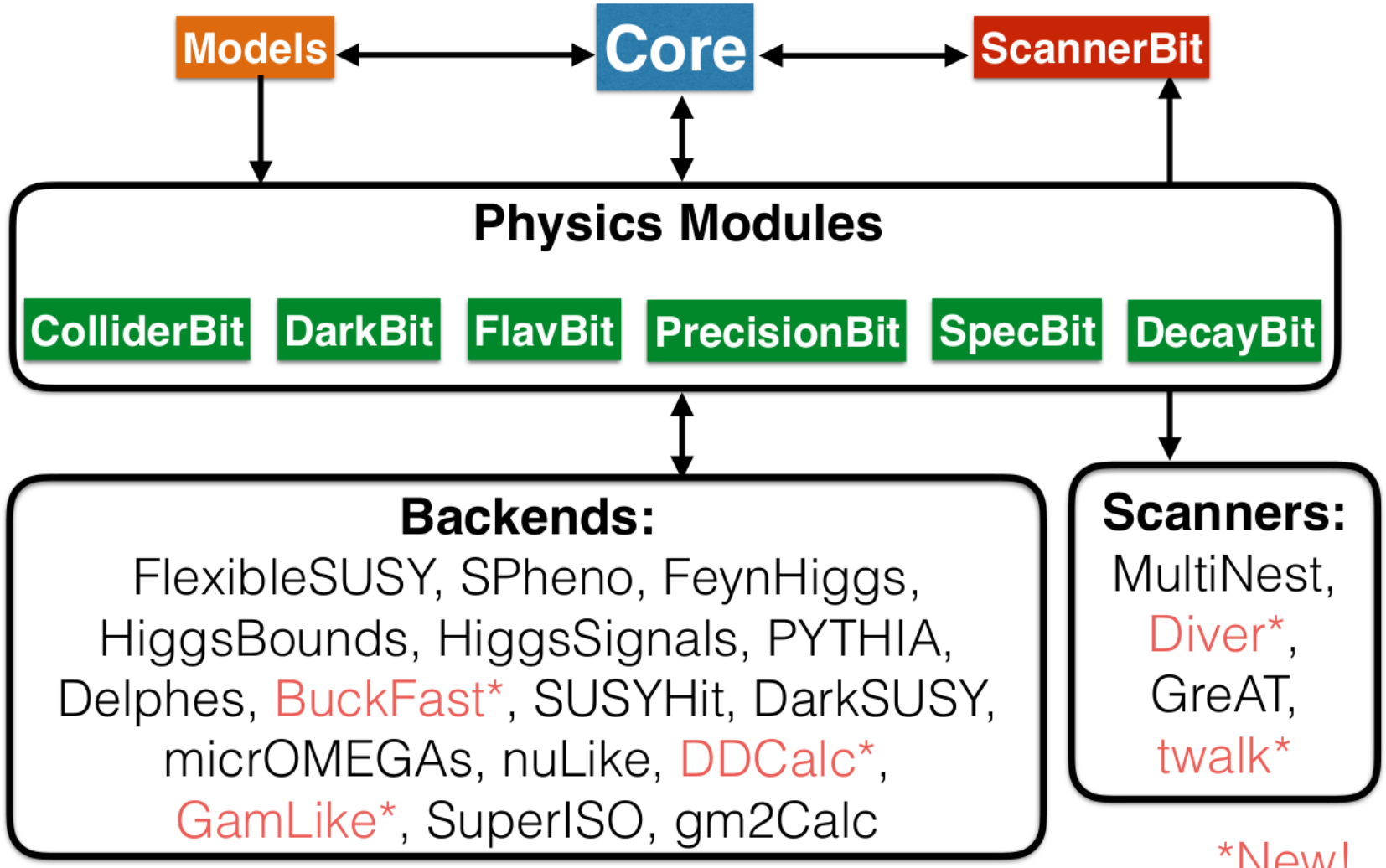
- **ColliderBit:** collider observables including Higgs + SUSY Searches from ATLAS, CMS, LEP
- **DarkBit:** dark matter observables (relic density, direct & indirect detection)
- **FlavBit:** including $g-2$, $b \to s\gamma$, B decays (new channels), angular obs., theory unc., LHCb likelihoods
- **SpecBit:** generic BSM spectrum object, providing RGE running, masses, mixings
- **DecayBit:** decay widths for all relevant SM and BSM particles
- **PrecisionBit:** precision EW tests (mostly via interface to FeynHiggs or SUSY-POPE)
- **ScannerBit:** manages stats, sampling and optimisation



What's in a module?

- Module functions (actual bits of GAMBIT C++ code)
- These can depend on other module functions
- Or can they can depend on *backends*(external codes)
- Adding new things is ***easy*** (detailed manual)
- Hooking up new backends or swapping them is ***easy***
- Module functions are **tagged** according to what they can calculate → plug and play!

GAMBIT code structure

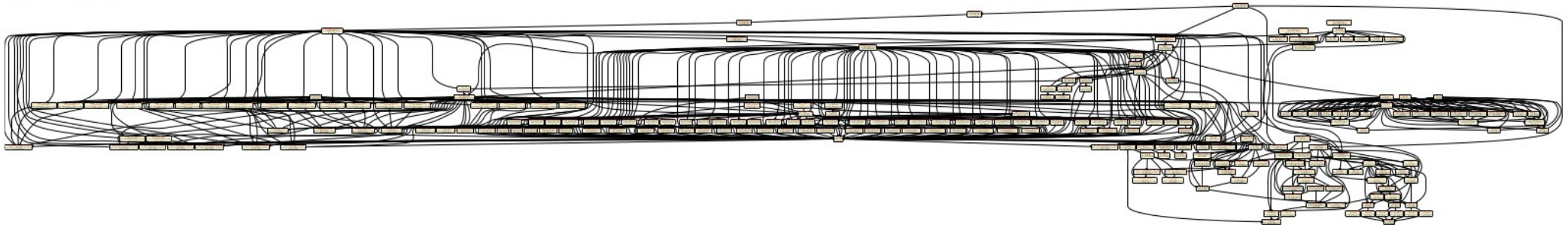


How does GAMBIT work?

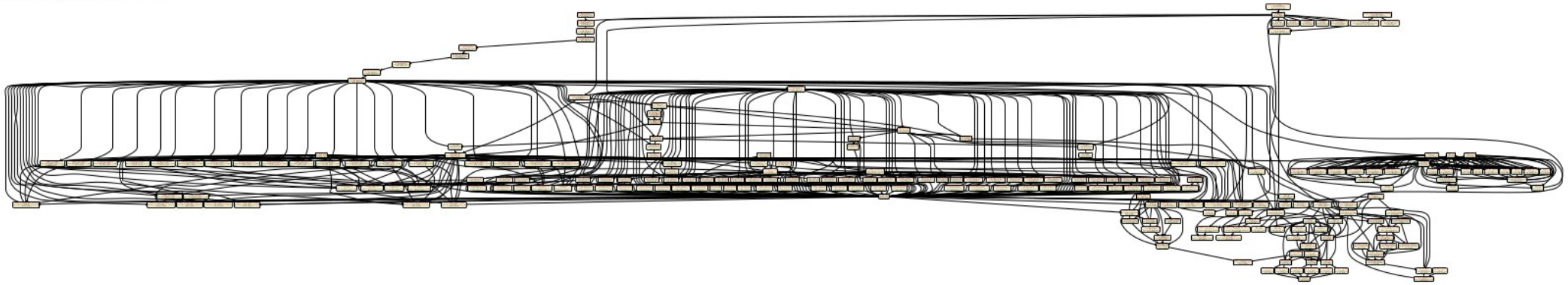
- You specify what to calculate and how (yaml input file)
- GAMBIT checks to see which functions can do it
- A dependency resolver stitches things together in the right order, and calculations are also ordered by speed
- GAMBIT performs the scan and writes output
- Pippi makes the plots
- You(r student) write(s) the paper

Dependency resolution in action

CMSSM:



MSSM7:



A peek inside a yaml file (more later...)

specify parameters, ranges, priors

select the scanner

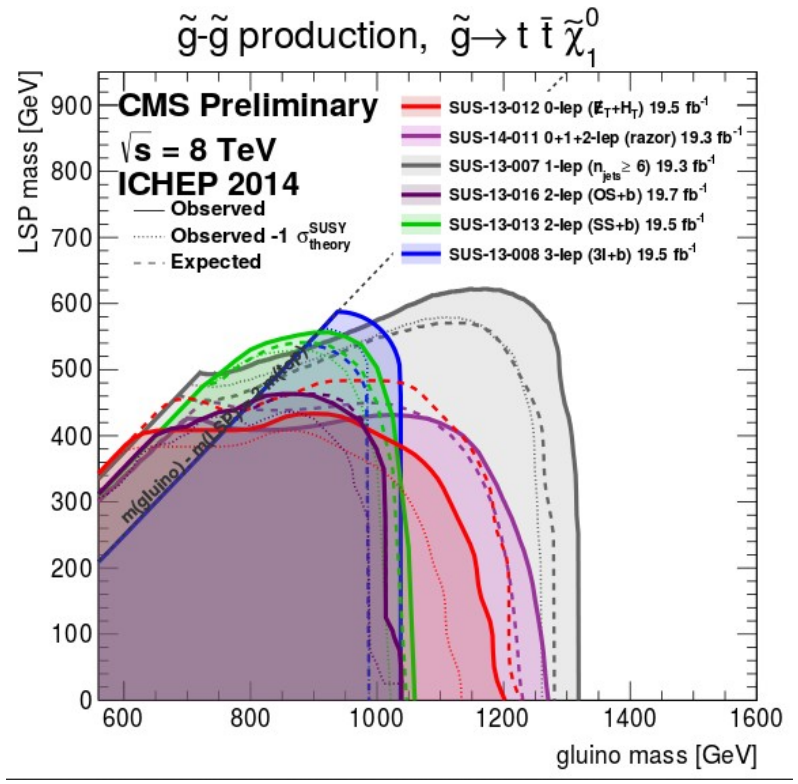
select likelihood components and other observables to calculate

define generic rules for how to fill dependencies

define generic rules for options to be passed to module functions

```
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
```

LHC limits: the problem



ColliderBit

- Handles LHC and LEP limits
- LEP: complete recast of sparticle xsec limits
- SUSY & Exotic LHC search limits from real-time MC simulation
- LHC resonance search limits from HiggsBounds+HiggsSignals
- Future: new resonance limits (beyond NW?)
- Future: interface to Rivet for LHC analyses

Model independent LHC limits

- Custom parallelised Pythia MC + custom detector sim
- Can generate 20,000 events on 12 cores in < 5 s
- Then apply Poisson likelihood with nuisance parameters for systematics
- Combine analyses using best expected exclusion
- The best you can do without extra public info from the experiments. CMS are getting better at this:

  https://cds.cern.ch/record/2242860/files/NOTE2017_001.pdf

ColliderBit likelihood

- We use a Poissonian likelihood marginalized over a rescaling parameter ξ to account for systematic uncertainties:

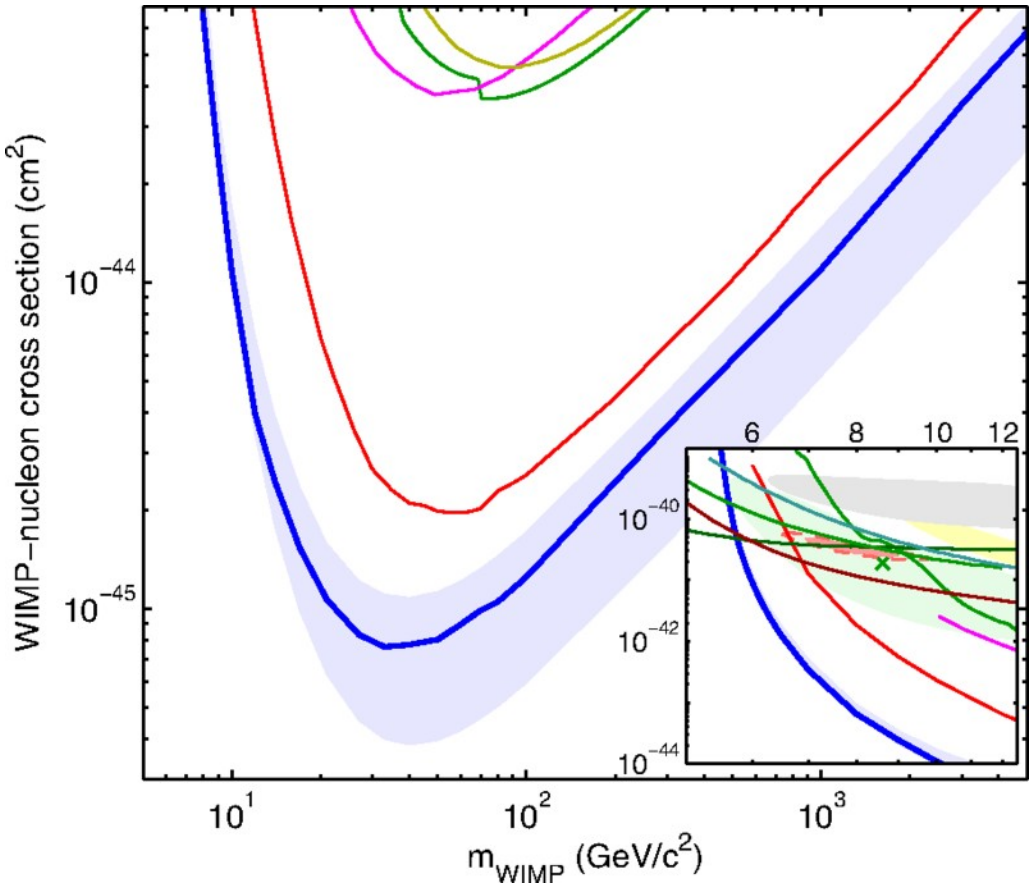
$$\mathcal{L}(n|s,b) = \int_0^\infty \frac{[\xi(s+b)]^n\, e^{-\xi(b+s)}}{n!} P(\xi)\mathrm{d}\xi$$

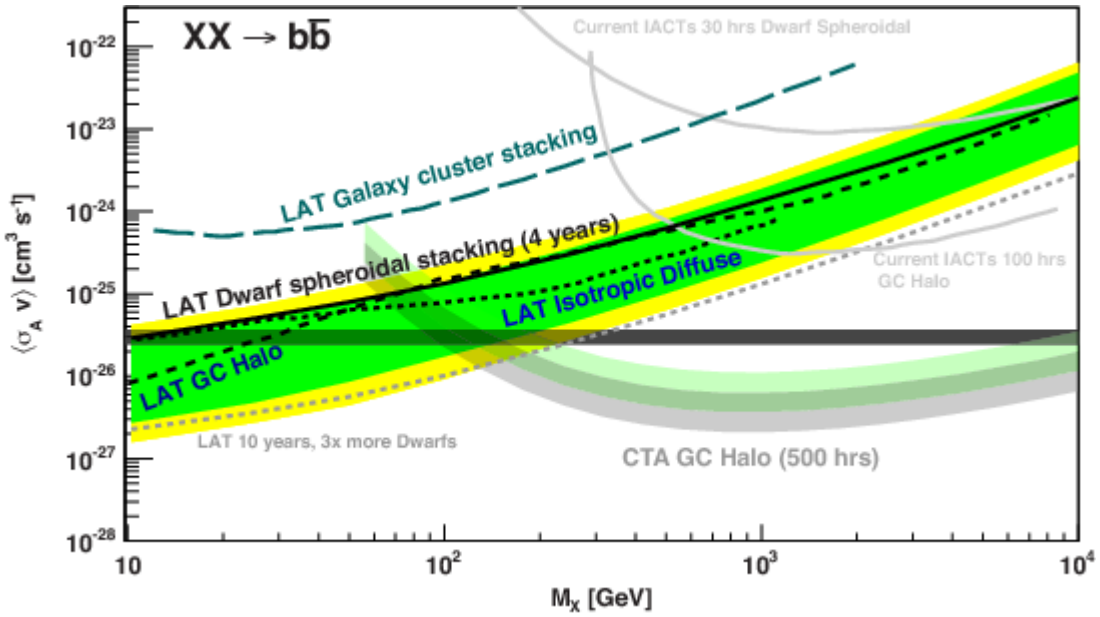
$$P(\xi|\sigma_\xi) \approx \frac{1}{\sqrt{2\pi}\sigma_\xi}\frac{1}{\xi}\exp\left[-\frac{1}{2}\left(\frac{\ln\xi}{\sigma_\xi}\right)^2\right] \qquad \text{where} \quad \sigma_\xi^2 = \sigma_s^2 + \sigma_b^2$$

- *n*, *s* and *b* are for signal region *expected* to give the strongest limit
- Currently available analyses (all 8 TeV):
  - ATLAS SUSY searches (0 lepton*, 0-1-2 lepton stop, b jets + MET, 2 lepton EW, 3 lepton EW)
  - CMS DM searches (top pair + MET, mono-b, mono-jet)
  - CMS multilepton SUSY search

*13 TeV also available

Astro limits: the problem

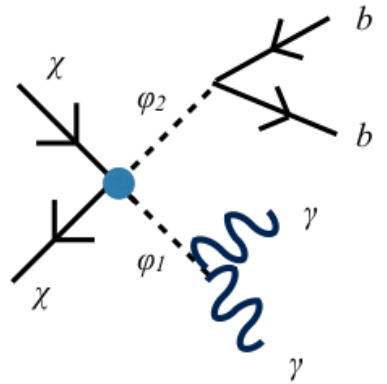


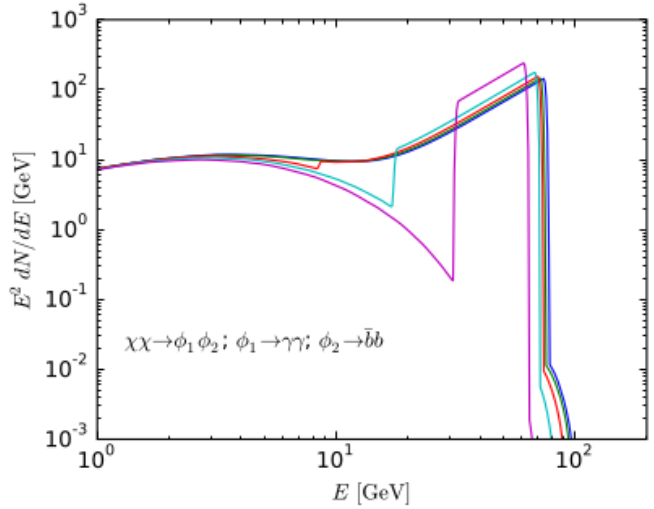


DarkBit: indirect detection

Gamma rays:

- Theoretical spectra calculated using branching fractions and tabulated gamma-ray yields
- Non-SM final state particles and Higgs are decayed on the fly with cascade Monte Carlo
- gamLike (gamlike.hepforge.org): New standalone code with likelihoods for DM searches from Fermi-LAT (dwarf spheroidals, galactic centre) and H.E.S.S. (galactic halo)



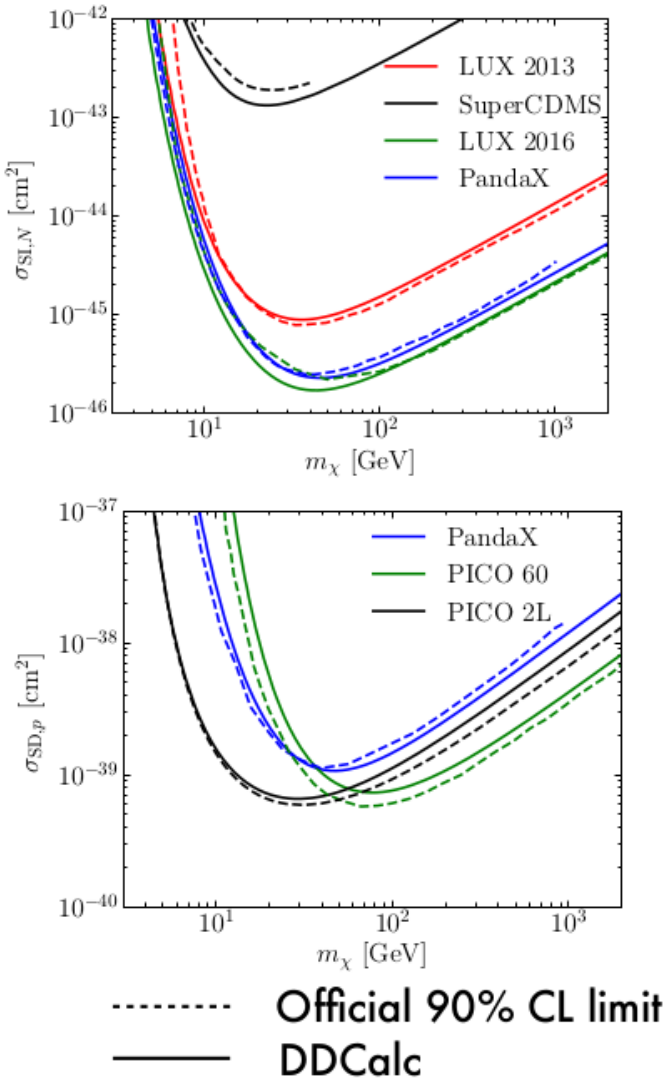


Solar neutrinos:

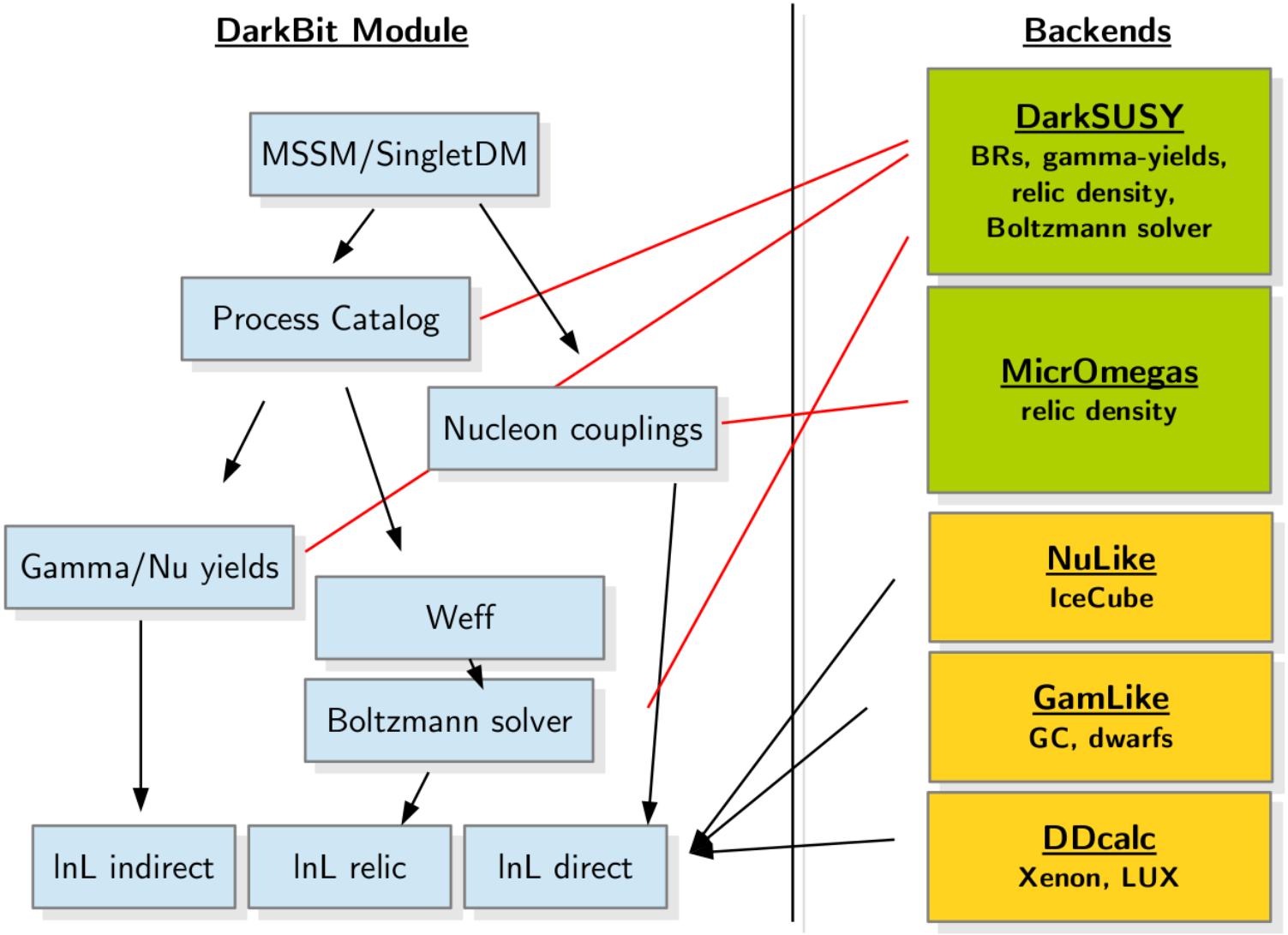
- Yields from DM annihilation in sun calculated by DarkSUSY. IceCube likelihoods contained in nulike (nulike.hepforge.org) standalone code.

DarkBit: direct detection

- In parallel with GAMBIT, we introduce *DDCalc* (ddcalc.hepforge.org), a tool to calculate event rates and complete likelihood functions for direct detection experiments taking into account:
  - A mix of both spin-independent and dependent contributions to the scattering rate.
  - Halo parameters (local density, DM velocity dispersion, etc.) chosen by the user.
- We currently have implemented likelihoods for Xenon(1T, 100), LUX, PandaX, SuperCDMS, PICO(60, 2L), and SIMPLE



DarkBit



- **Event level neutrino telescope and gamma ray likelihoods!**
- **First principles treatment of direct search limits → easily extendable to non-trivial operators**
- **Very large range of experiments included (includes future, e.g. CTA)**

FlavBit

- Models a series of experimental flavour anomalies
- Theoretical predictions currently based on SuperIso
- Theoretical and experimental uncertainties are carefully considered for each observable (including correlations)

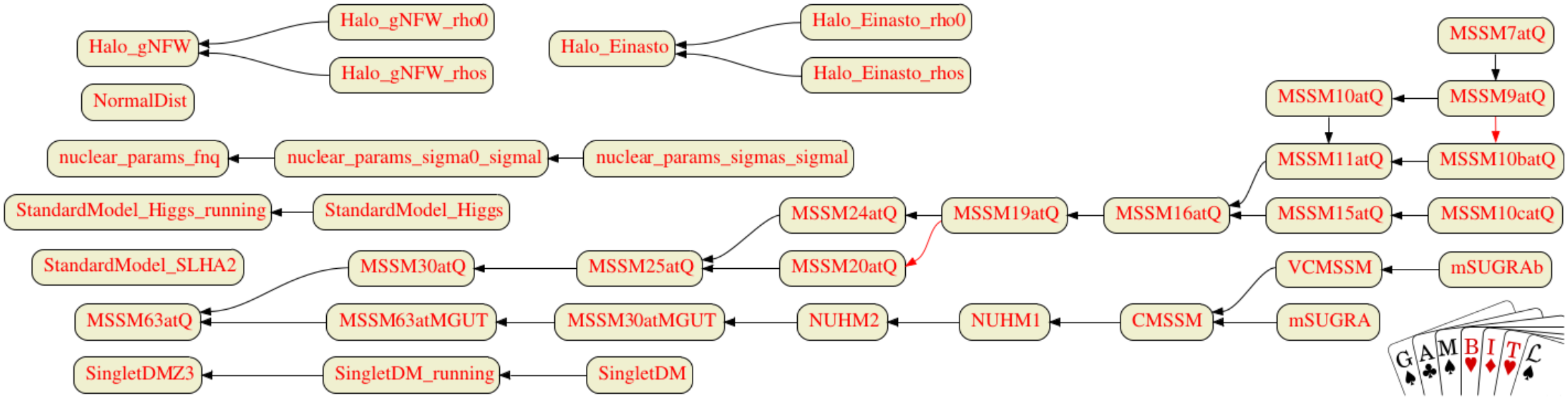
FlavBit likelihoods (more to come...)

- $B \to D^{(*)}\mu\nu$ (PDG, take correlated theoretical uncertainties from SuperIso study)
- $R_{D^{(*)}} \equiv \mathcal{B}(B \to D^{(*)}\tau\nu_\tau)/\mathcal{B}(B \to D^{(*)}\ell\nu_\ell)$ (HFAG, take correlated experimental uncertainties)
- $B^\pm \to \ell\nu_\ell$ (PDG value)
- $D^\pm \to \mu\nu_\mu$, $D_s^\pm \to \tau\nu_\tau$, $D_s^\pm \to \mu\nu_\mu$ (PDG values, theoretical correlations considered)
- Angular observables of $B^0 \to K^{*0}\mu^+\mu^-$ in q^2 bins (LHCb) (experimental correlations within each bin + theory correlations)
- $B_s^0 \to \mu^+\mu^-$ (latest LHCb result)
- $B^0 \to \mu^+\mu^-$ (combined LHCb and CMS)
- $B \to X_s\gamma$ for $E_\gamma > 1.6$ GeV (BaBar and Belle average by Misiak et al)
- ΔM_s: (average of CDF and LHCb, theory error is an order of magnitude greater than experimental uncertainty)

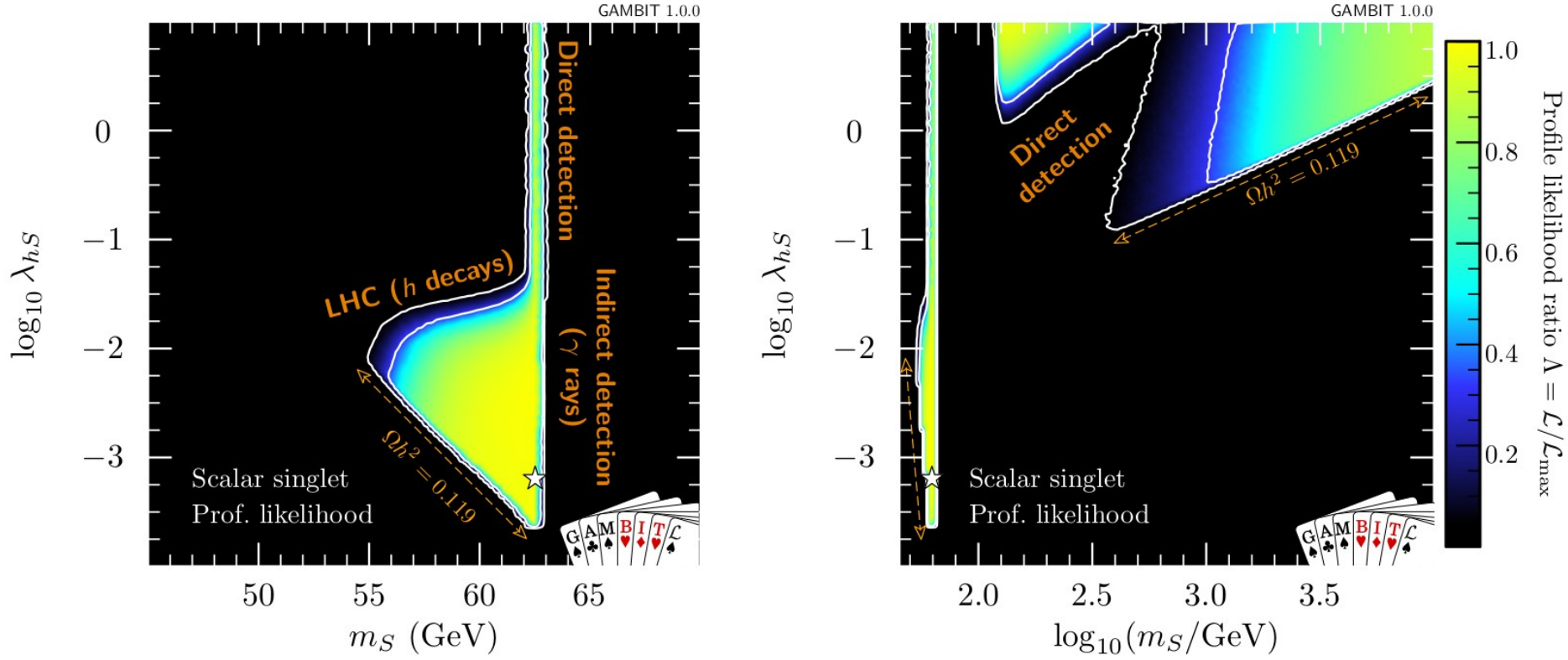
$$\begin{aligned}\frac{1}{\Gamma}\frac{\mathrm{d}^3(\Gamma+\bar{\Gamma})}{\mathrm{d}\cos\theta_\ell\,\mathrm{d}\cos\theta_K\,\mathrm{d}\phi} = \frac{9}{32\pi}\Big[&\frac{3}{4}(1-F_\mathrm{L})\sin^2\theta_K \\ &+ F_\mathrm{L}\cos^2\theta_K + \frac{1}{4}(1-F_\mathrm{L})\sin^2\theta_K\cos 2\theta_\ell \\ &- F_\mathrm{L}\cos^2\theta_K\cos 2\theta_\ell + S_3\sin^2\theta_K\sin^2\theta_\ell\cos 2\phi \\ &+ S_4\sin 2\theta_K\sin 2\theta_\ell\cos\phi + S_5\sin 2\theta_K\sin\theta_\ell\cos\phi \\ &+ \frac{4}{3}A_\mathrm{FB}\sin^2\theta_K\cos\theta_\ell + S_7\sin 2\theta_K\sin\theta_\ell\sin\phi \\ &+ S_8\sin 2\theta_K\sin 2\theta_\ell\sin\phi + S_9\sin^2\theta_K\sin^2\theta_\ell\sin 2\phi\Big]\end{aligned}$$

Global and Modular BSM

- Models are defined by their parameters and relations to each other
- Models can inherit from parent models, easy translation between relations
- We have so far scanned SUSY + Higgs portal + axion + two Higgs doublet models



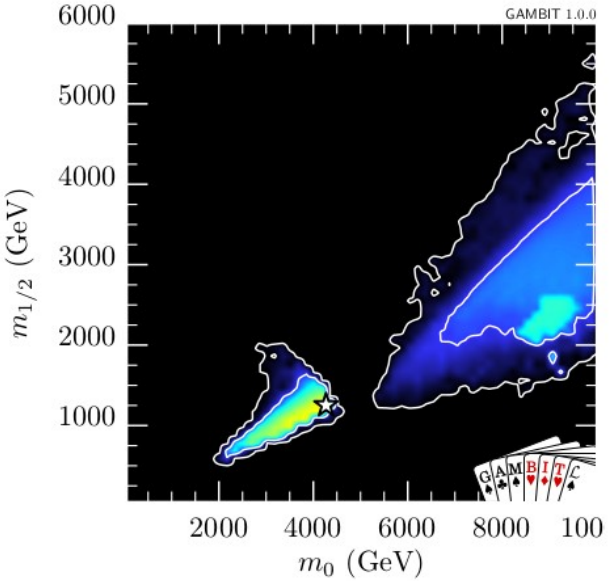
Global and Modular BSM Inference: Scalar singlet DM

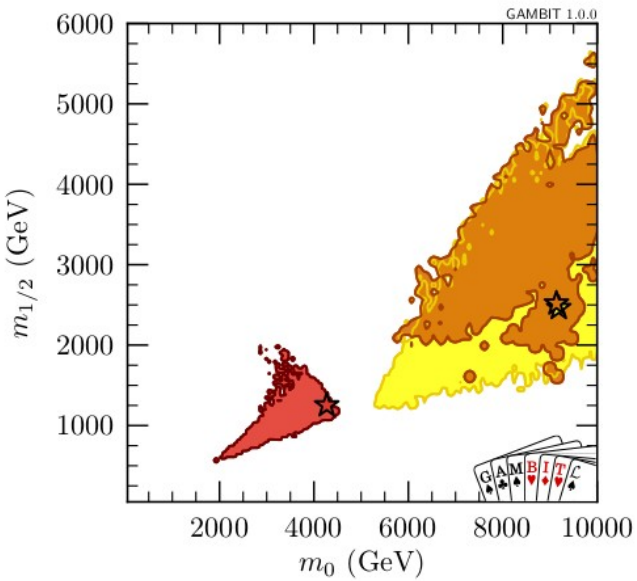


$$\mathcal{L} = \frac{1}{2}\mu_S^2 S^2 + \frac{1}{2}\lambda_{hS} S^2 |H|^2 + \frac{1}{4}\lambda_S S^4 + \frac{1}{2}\partial_\mu S \partial^\mu S$$

(m_S, λ_{hS} + 13 nuisances)

Global and Modular BSM Inference: CMSSM

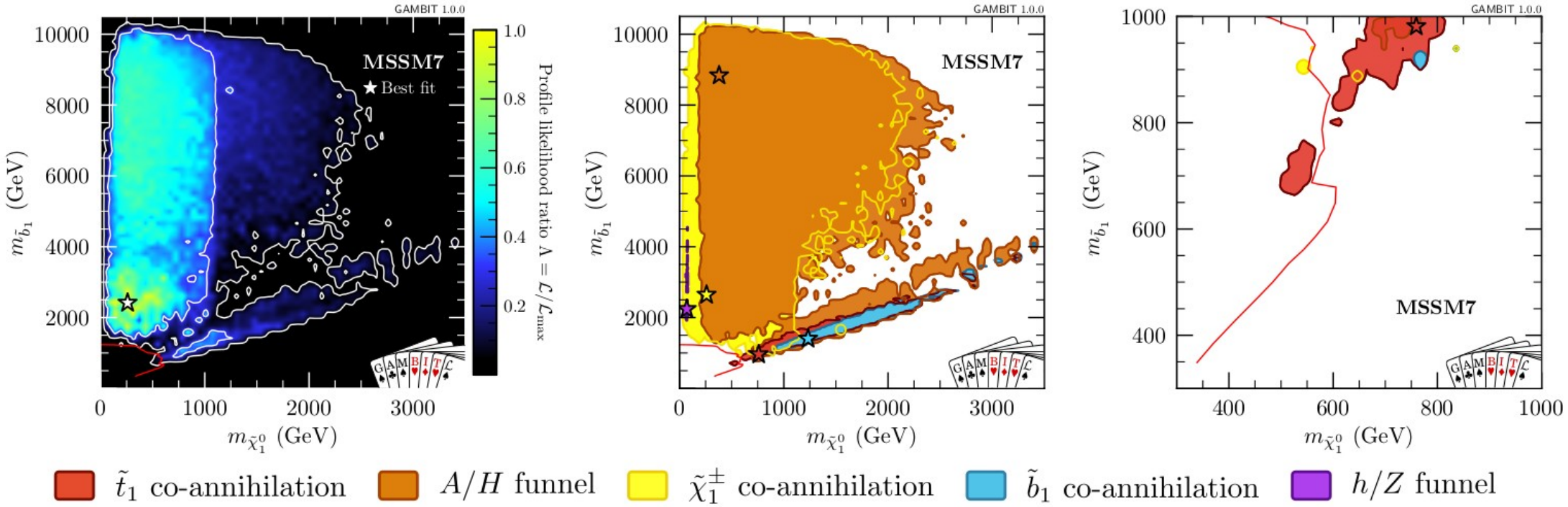




- m_0, $m_{\frac{1}{2}}$, A_0, $\tan\beta$ + 5 nuisances
- H/A^0 funnel, $\chi^{\pm}$ co-annihilation, $\tilde{t}$ co-annihilation
- $\tilde{\tau}$ **co-annihilation now ruled out**
- Includes LUX 2016, Panda-X + direct simulation of LHC Run 1 & Run 2 limits.

(also have NUHM1 and NUHM2 results)

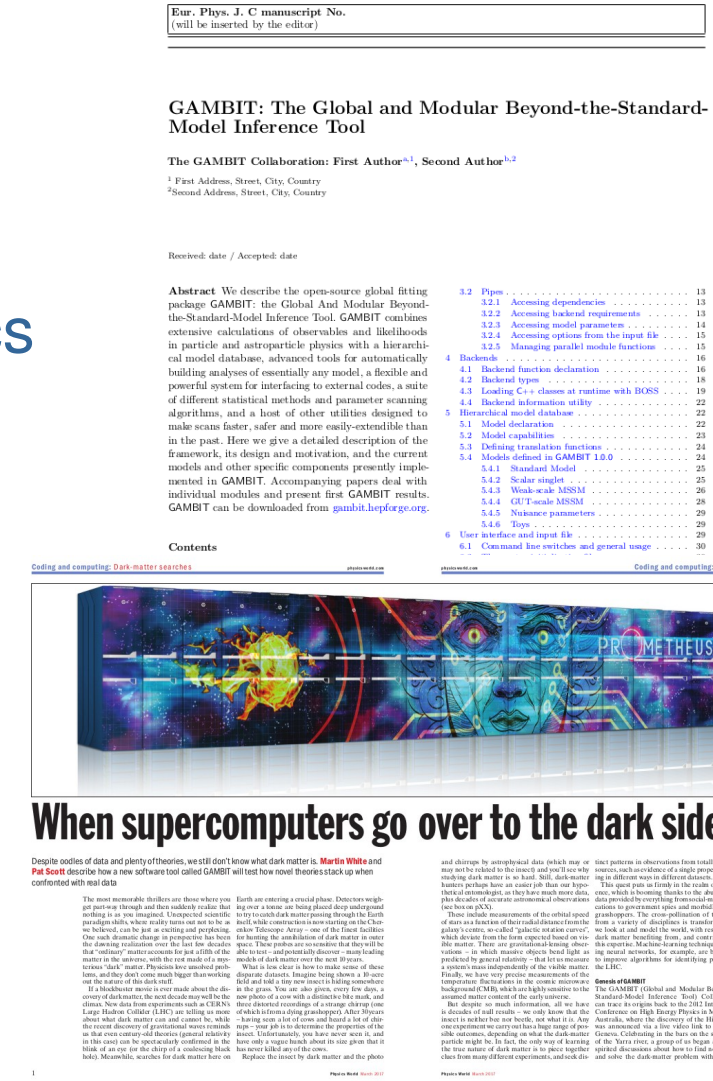
Global and Modular BSM Inference: MSSM7



- $m_{\tilde{f}}$, M_2, A_u, A_d, m_{Hu}, m_{Hd}, $\tan\beta$ + 5 nuisances
- H/A^0 funnel, h/Z funnel, $\chi^\pm$ co-annihilation, $\tilde{t}/\tilde{b}$ co-annihilation
- Includes LUX 2016, Panda-X + direct simulation of LHC Run 1 & Run 2 limits.

Global and Modular BSM Inference Tool

- GAMBIT has just been released as an ***open source public tool***
- 9 papers published in EPJC (design, manual + first physics results)
- Feature article in *Physics World* March 2017 issue if you want a gentler introduction
- See **gambit.hepforge.org** for more info



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

GAMBIT: The Global and Modular Beyond-the-Standard-Model Inference Tool

The GAMBIT Collaboration: First Author[a,1], Second Author[b,2]

[1] First Address, Street, City, Country
[2] Second Address, Street, City, Country

Received: date / Accepted: date

Abstract We describe the open-source global fitting package GAMBIT: the Global And Modular Beyond-the-Standard-Model Inference Tool. GAMBIT combines extensive calculations of observables and likelihoods in particle and astroparticle physics with a hierarchical model database, advanced tools for automatically building analyses of essentially any model, a flexible and powerful system for interfacing to external codes, a suite of different statistical methods and parameter scanning algorithms, and a host of other utilities designed to make scans faster, safer and more easily-extendible than in the past. Here we give a detailed description of the framework, its design and motivation, and the current models and other specific components presently implemented in GAMBIT. Accompanying papers deal with individual modules and present first GAMBIT results. GAMBIT can be downloaded from gambit.hepforge.org.

Contents

Coding and computing: Dark-matter searches

physicsworld.com

When supercomputers go over to the dark side

Despite oodles of data and plenty of theories, we still don't know what dark matter is. Martin White and Pat Scott describe how a new software tool called GAMBIT will test how novel theories stack up when confronted with real data

The most memorable thrillers are those where you get part-way through and then suddenly realize that nothing is as you imagined. Unexpected scientific paradigm shifts, where reality turns out not to be as we believed, can be just as exciting and perplexing. One such dramatic change in perspective has been the dawning realization over the last few decades that "ordinary" matter accounts for just a fifth of the matter in the universe, with the rest made of a mysterious "dark" matter. Physicists love unsolved problems, and they don't come much bigger than working out the nature of this dark stuff.

If a blockbuster movie is ever made about the discovery of dark matter, the next decade may well be the climax. New data from experiments such as CERN's Large Hadron Collider (LHC) are telling us more about what dark matter can and cannot be, while the recent discovery of gravitational waves reminds us that even century-old theories (general relativity in this case) can be spectacularly confirmed in the blink of an eye (or the chirp of a coalescing black hole). Meanwhile, searches for dark matter here on Earth are entering a crucial phase. Detectors weighing over a tonne are being placed deep underground to try to catch dark matter passing through the Earth itself, while construction is now starting on the Cherenkov Telescope Array – one of the finest facilities for hunting the annihilation of dark matter in outer space. These probes are so sensitive that they will be able to test – and potentially discover – many leading models of dark matter over the next 10 years.

What is less clear is how to make sense of these disparate datasets. Imagine being shown a 10-acre field and told a tiny new insect is hiding somewhere in the grass. You are also given, every few days, a new photo of a cow with a distinctive bite mark, and three distorted recordings of a strange chirrup (one of which is from a dying grasshopper). After 30 years – having seen a lot of cows and heard a lot of chirrups – your job is to determine the properties of the insect. Unfortunately, you have never seen it, and have only a vague hunch about its size given that it has never killed any of the cows.

Replace the insect by dark matter and the photo and chirrups by astrophysical data (which may or may not be related to the insect) and you'll see why studying dark matter is so hard. Still, dark-matter hunters perhaps have an easier job than our hypothetical entomologist, as they have much more data, plus decades of accurate astronomical observations (see box on p[XX]).

These include measurements of the orbital speed of stars as a function of their radial distance from the galaxy's centre, so-called "galactic rotation curves", which deviate from the form expected based on visible matter. There are gravitational-lensing observations – in which massive objects bend light as predicted by general relativity – that let us measure a system's mass independently of the visible matter. Finally, we have very precise measurements of the temperature fluctuations in the cosmic microwave background (CMB), which are highly sensitive to the assumed matter content of the early universe.

But despite so much information, all we have is decades of null results – we only know that the insect is neither bee nor beetle, not what it is. Any one experiment we carry out has a huge range of possible outcomes, depending on what the dark-matter particle might be. In fact, the only way of learning the true nature of dark matter is to piece together clues from many different experiments, and seek distinct patterns in observations from totally different sources, such as evidence of a single property emerging in different ways in different datasets.

This quest puts us firmly in the realm of data science, which is booming thanks to the abundance of data provided by everything from social-media applications to government spies and mobbily operatic grasshoppers. The cross-pollination of techniques from a variety of disciplines is transforming how we look at and model the world, with research into dark matter benefiting from, and contributing to, this expertise. Machine-learning techniques, including neural networks, for example, are being used to improve algorithms for identifying particles at the LHC.

Genesis of GAMBIT

The GAMBIT (Global and Modular Beyond-the-Standard-Model Inference Tool) Collaboration can trace its origins back to the 2012 International Conference on High Energy Physics in Melbourne, Australia, where the discovery of the Higgs boson was announced via a live video link to CERN in Geneva. Celebrating in the bars on the south bank of the Yarra river, a group of us began a series of spirited discussions about how to find new physics and solve the dark-matter problem with software.

Martin White is a particle astrophysicist and Australian Research Council Future Fellow at the University of Adelaide, e-mail martin.white@adelaide.edu.au, and Pat Scott is an STFC Rutherford Fellow and lecturer in fundamental physics at Imperial College London, e-mail p.scott@imperial.ac.uk. They are writing on behalf of the GAMBIT Collaboration, which Scott leads and White co-leads (gambit.hepforge.org).

What's next for GAMBIT?

- More models (2HDM, axion models, more Higgs portal models, RH neutrinos)
- Better LHC Higgs likelihoods
- Direct link to CalcHEP and MadGraph
- Implementation of more complex LHC likelihoods (e.g. CMS simplified likelihood analyses for monojet and 0 lepton searches
- A new cosmology module

Today

- Have four hands-on tutorials that introduce you to the basic features of GAMBIT

1) Wilson coefficient fit with 2 parameters (runs quickly, allow you to produce and scrutinise output within the next few hours)

2) CMSSM example: see how to run a single points or scan with lots of likelihoods included

3) Example of adding new model to ColliderBit for single point tests. Note: setting up a scan in this case is a very advanced topic, so we will use the ColliderBit standalone code.

4) Adding new model for use with astrophysical likelihoods

Installing GAMBIT

- Follow the instructions in: Installation_before_tutorial.txt
- If you have not yet done this, ask me for a USB stick with GAMBIT on it

Tutorial 1: Wilson Coefficients

$$\mathcal{H}_{\text{eff}} = -\frac{4G_F}{\sqrt{2}} V_{tb}V_{ts}^* \sum_{i=1}^{10} \Big(C_i(\mu)\mathcal{O}_i(\mu) + C_i'(\mu)\mathcal{O}_i'(\mu) \Big)$$

Wilson Coefficients

$$\mathcal{O}_1 = (\bar{s}\gamma_\mu T^a P_L c)(\bar{c}\gamma^\mu T^a P_L b)$$

$$\mathcal{O}_2 = (\bar{s}\gamma_\mu P_L c)(\bar{c}\gamma^\mu P_L b)$$

$$\mathcal{O}_3 = (\bar{s}\gamma_\mu P_L b)\sum_q (\bar{q}\gamma^\mu q)$$

$$\mathcal{O}_4 = (\bar{s}\gamma_\mu T^a P_L b)\sum_q (\bar{q}\gamma^\mu T^a q)$$

$$\mathcal{O}_5 = (\bar{s}\gamma_{\mu_1}\gamma_{\mu_2}\gamma_{\mu_3} P_L b)\sum_q (\bar{q}\gamma^{\mu_1}\gamma^{\mu_2}\gamma^{\mu_3} q)$$

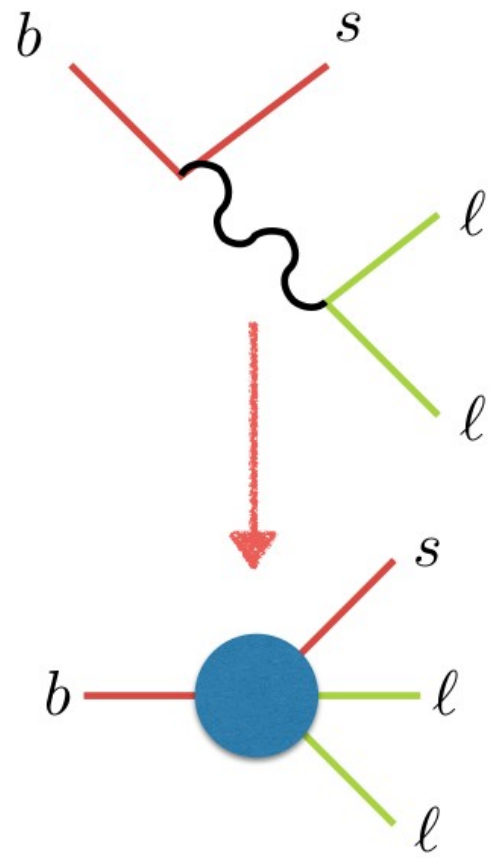
$$\mathcal{O}_6 = (\bar{s}\gamma_{\mu_1}\gamma_{\mu_2}\gamma_{\mu_3} T^a P_L b)\sum_q (\bar{q}\gamma^{\mu_1}\gamma^{\mu_2}\gamma^{\mu_3} T^a q)$$

$$\mathcal{O}_7 = \frac{e}{(4\pi)^2} m_b (\bar{s}\sigma^{\mu\nu} P_R b) F_{\mu\nu}$$

$$\mathcal{O}_8 = \frac{g}{(4\pi)^2} m_b (\bar{s}\sigma^{\mu\nu} T^a P_R b) G^a_{\mu\nu}$$

$$\mathcal{O}_9 = \frac{e^2}{(4\pi)^2} (\bar{s}\gamma^\mu P_L b)(\bar{\ell}\gamma_\mu \ell)$$

$$\mathcal{O}_{10} = \frac{e^2}{(4\pi)^2} (\bar{s}\gamma^\mu P_L b)(\bar{\ell}\gamma_\mu \gamma_5 \ell)$$



See arxiv.org/pdf/1705.07933.pdf for more detail

Tutorial 1: Wilson Coefficients

- 2D Wilson coefficient fit

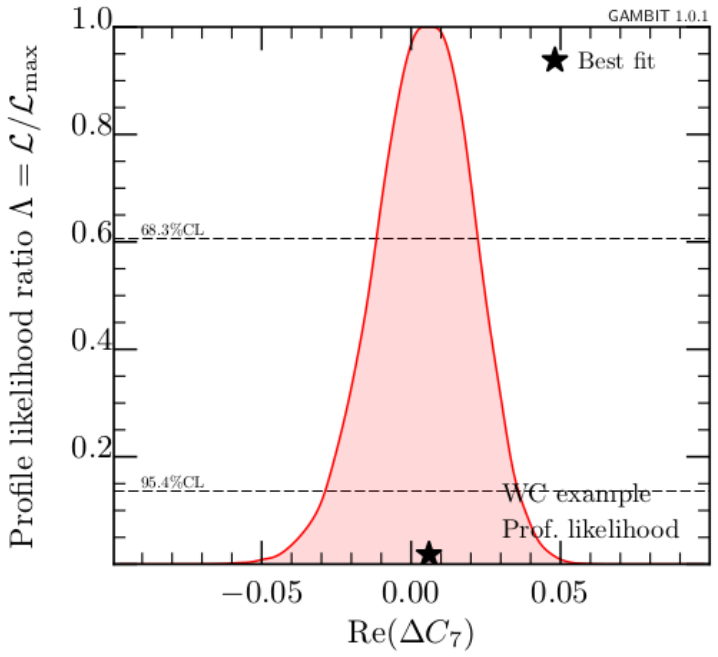
$$\Delta C_x \equiv C_{x,BSM} - C_{x,SM}$$

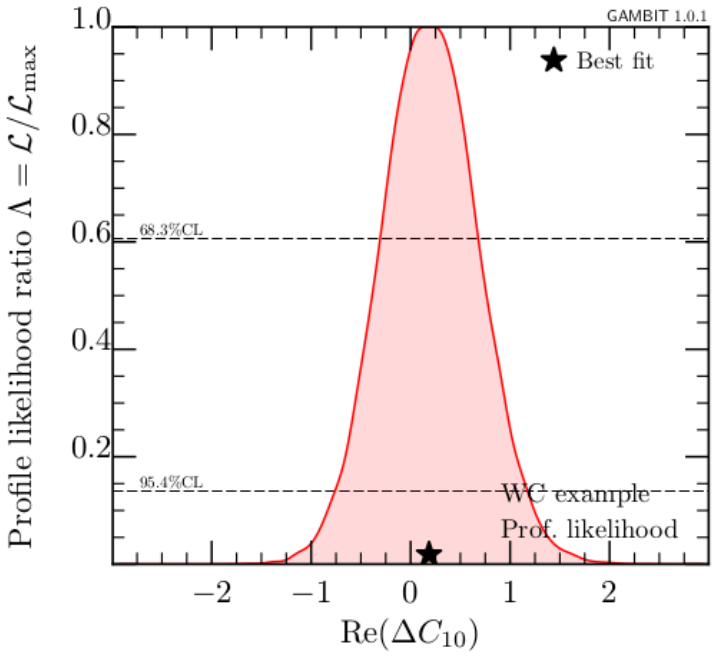
- Free parameters: ΔC_7 `Re_DeltaC7`
  ΔC_{10} `Re_DeltaC10`

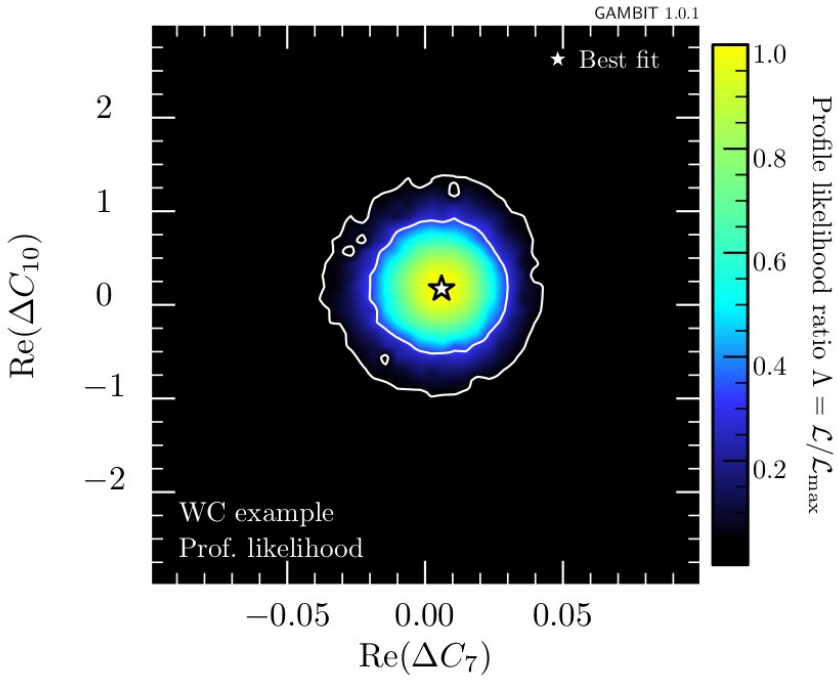
- Observables: $BR(B \to X_s\gamma)$ `b2sgamma`
  $BR(B_d \to \mu^+\mu^-)$ `b2ll`
  $BR(B_s \to \mu^+\mu^-)$

- Follow the steps in: WC_tutorial_commands.txt

Tutorial 1: Results should look like this







- Feel free to ask for assistance with plotting and interpreting the *.pip file

Tutorial 2: CMSSM file

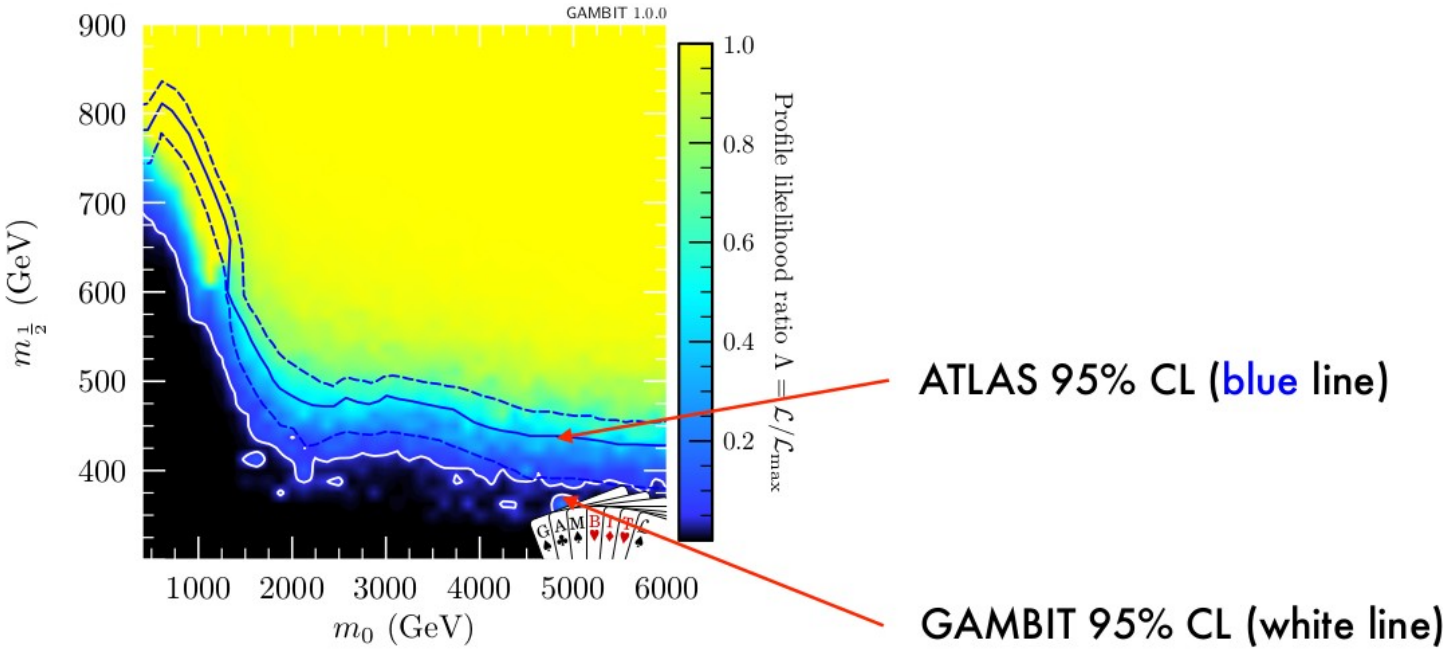
- Put the file ColliderBit_CMSSM_tutorial.yaml in your GAMBIT directory
- Run it using:

./gambit -f ColliderBit_CMSSM_tutorial.yaml

- After a few minutes, GAMBIT will spit out a likelihood value the ATLAS 8 TeV 0 lepton analysis, compare with:

95% CL exclusion →

$$\Delta \ln \mathcal{L} = -3.0$$



Tutorial 2: Questions

- How can you change the number of generated events?
- How can you add more LHC analyses?
- How can you run both ATLAS and CMS analyses?
- How would you scan the CMSSM rather than run one point?
- How would you go about scanning the MSSM rather than the CMSSM?
- How would you add astrophysical likelihoods?

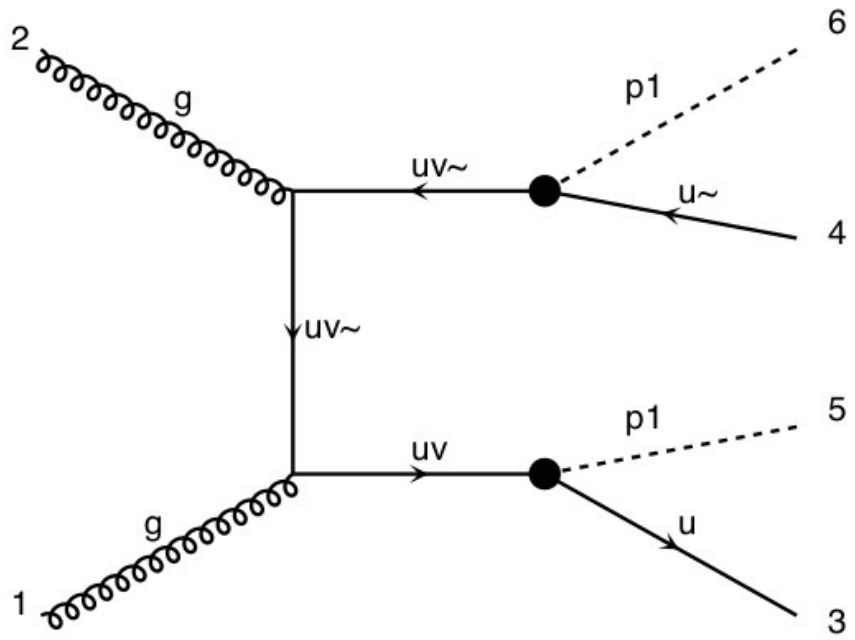
See the yaml_files/ directory for hints and examples

Tutorial 3: Add new model to ColliderBit

An example with two new particles:

- ϕ_1 , **real singlet scalar, mass** m_1
- U, **colored fermion, SU(2) singlet, mass** M_U

$$\mathcal{L}_{\rm int} = \lambda_1 \phi_1 \bar{U} u_R + h.c.$$



Tutorial 3: Add new model to ColliderBit

To put this model into ColliderBit, we have implemented it into FeynRules, passed the UFO file to MadGraph, and then MadGraph generated Pythia code for the process of interest:

$$pp \to \bar{U}U$$

The MadGraph output is in `Backends/patches/pythia8.212.EM/ExternalModel` along with a modified version of `ProcessContainer.cc` from Pythia:

```
22  #include "Sigma_GambitDemo_UFO_gg_uvuvx.h"
23  #include "Sigma_GambitDemo_UFO_uux_uvuvx.h"
24  #include "Sigma_GambitDemo_UFO_ccx_uvuvx.h"
25  #include "Sigma_GambitDemo_UFO_ddx_uvuvx.h"
26  #include "Sigma_GambitDemo_UFO_ssx_uvuvx.h"
27
```

```
2478
2479  if (settings.flag("UserModel:all")) {
2480    sigmaPtr = new Sigma_GambitDemo_UFO_ccx_uvuvx();
2481    containerPtrs.push_back( new ProcessContainer(sigmaPtr) );
2482    sigmaPtr = new Sigma_GambitDemo_UFO_ddx_uvuvx();
2483    containerPtrs.push_back( new ProcessContainer(sigmaPtr) );
2484    sigmaPtr = new Sigma_GambitDemo_UFO_gg_uvuvx();
2485    containerPtrs.push_back( new ProcessContainer(sigmaPtr) );
2486    sigmaPtr = new Sigma_GambitDemo_UFO_ssx_uvuvx();
2487    containerPtrs.push_back( new ProcessContainer(sigmaPtr) );
2488    sigmaPtr = new Sigma_GambitDemo_UFO_uux_uvuvx();
2489    containerPtrs.push_back( new ProcessContainer(sigmaPtr) );
2490  }
```

These changes were automatically applied to Pythia when you ran `make pythia_8.212.EM`.

Tutorial 3: Add new model to ColliderBit

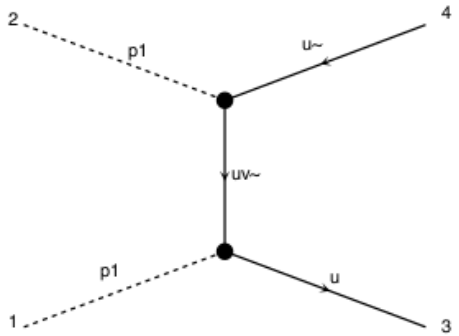
- To quickly check the collider likelihoods for the new model, we will use a ColliderBit standalone program.
- To do a scan with GAMBIT, we would need to properly implement the model and a corresponding spectrum object and decay table into GAMBIT. Then we would make ColliderBit aware of the new model (see GAMBIT core, SpecBit, DecayBit, and ColliderBit papers for more details). This takes a little bit of work, so we will leave it for another tutorial.
- To compile the standalone, type `make ColliderBit_standalone` (in the `./build` directory)

Tutorial 4: Add new model to DarkBit

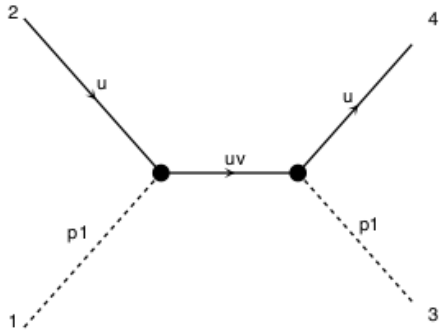
An example with two new particles: $\mathcal{L}_{\rm int} = \lambda_1 \phi_1 \bar{U} u_R + h.c.$

- ϕ_1, real singlet scalar, mass m_1, DM candidate
- U, colored fermion, SU(2) singlet, mass M_U

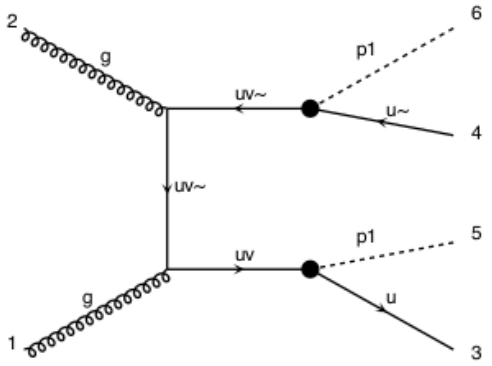
DM annihilation: relic density, indirect detection



DM-quark scattering: direct detection



U production at LHC



In the remaining part of this tutorial, we will complete a simple implementation of the model into GAMBIT, and find the best fit regions of parameter space in light of the these constraints.

Tutorial 4: Files

- Download Materials.zip from the indico page and unzip it

Tutorial 4 Step 1: Add a new model file

Add the file `ExternalModel.hpp` to the `Models/include/gambit/Models/models` directory.

```
1  //  GAMBIT: Global and Modular BSM Inference Tool
2  //  *********************************************
3  //
4  //  External Model for tutorial
5  //
6  //  *********************************************
7  //
8  //  Authors
9  //  =======
10 //
11 //  Your Name Here
12 //
13 //
14 //  *********************************************
15
16 #ifndef __ExternalModel_hpp__
17 #define __ExternalModel_hpp__
18
19 #define MODEL ExternalModel
20   START_MODEL
21   DEFINEPARS(Mp1, Muv, lam1)
22 #undef MODEL
23
24 #endif
```

You also need to add a description of the model to the GAMBIT diagnostic system, otherwise GAMBIT will judge you and refuse to run. Add something like this to `config/models.dat`:

```
383  ExternalModel: |
384
385    The theory of everything.
```

Tutorial 4 Step 2: micrOMEGAs

- Go to the directory `Backends/installed/micromegas/3.6.9.2/`
- Run `./newProject ExternalModel` to make a micrOMEGAs directory for the new model.
- Enter the `./ExternalModel` directory. Copy all of the provided CalcHEP files for this model into the `./work/models` directory.
- Copy the provided `Makefile` into the `ExternalModel` directory.
- Run `make sharedlib main=main.c`. This should create a library `libmicromegas.so`, which GAMBIT will use to run micrOMEGAs functions.

Tutorial 4 Step 3: micrOMEGAs frontend

- Open the provided files `MicrOmegas_ExternalModel_3_6_9_2.*`. These files make GAMBIT aware of various functions and variables in micrOMEGAs, as well as providing an initialization function for the backend.
- Copy `MicrOmegas_ExternalModel_3_6_9_2.hpp` to `Backends/include/gambit/Backends/frontends` and `MicrOmegas_ExternalModel_3_6_9_2.cpp` to `Backends/src/frontends`

Tutorial 4 Step 4: Tell GAMBIT where to find the new micrOMEGAs library

Copy `config/backend_locations.yaml.default` to `config/backend_locations.yaml`. Add the following lines to the file:

```
63  MicrOmegas_ExternalModel:
64    3.6.9.2:      ./Backends/installed/micromegas/3.6.9.2/ExternalModel/libmicromegas.so
```

Tutorial 4 Step 5: make existing DarkBit function aware of the new model

- Our implementation of the this model into micrOMEGAs allows us to calculate relic density and direct detection likelihoods for it using DarkBit. To make the existing DarkBit functions aware of this new model and backend, make the following changes to `DarkBit/include/gambit/DarkBit/DarkBit_rollcall.hpp`:

```
529  #define FUNCTION DD_couplings_MicrOmegas
530    START_FUNCTION(DM_nucleon_couplings)
531    BACKEND_REQ(nucleonAmplitudes, (gimmemicro), int, (double(*)(double,double,double,double), double*, double*, double*, double*)
532    BACKEND_REQ(FeScLoop, (gimmemicro), double, (double, double, double, double))
533    BACKEND_REQ(MOcommon, (gimmemicro), MicrOmegas::MOcommonSTR)
534    ALLOW_MODEL_DEPENDENCE(nuclear_params_fnq, MSSM63atQ, SingletDM, ExternalModel)
535    MODEL_GROUP(group1, (nuclear_params_fnq))
536    MODEL_GROUP(group2, (MSSM63atQ, SingletDM, ExternalModel))
537    ALLOW_MODEL_COMBINATION(group1, group2)
538    BACKEND_OPTION((MicrOmegas_MSSM),(gimmemicro))
539
540    BACKEND_OPTION((MicrOmegas_ExternalModel),(gimmemicro))
541
542  #undef FUNCTION
```

```
190    // Routine for cross checking RD density results
191    #define FUNCTION RD_oh2_MicrOmegas
192      START_FUNCTION(double)
193      BACKEND_REQ(oh2, (MicrOmegas_MSSM,MicrOmegas_SingletDM,MicrOmegas_ExternalModel), double, (double*,int,double)
194      ALLOW_MODELS(MSSM63atQ,SingletDM,ExternalModel)
195    #undef FUNCTION
196  #undef CAPABILITY
```

Tutorial 4 Step 6: add a function to satisfy the *mwimp* capability

The direct detection functions have a dependency on the *mwimp* capability. Add a new function to satisfy this dependency for the new model:

- First register the function in `DarkBit/include/gambit/DarkBit/DarkBit_rollcall.hpp`:

```
495  // Simple WIMP property extractors ======================================
496
497  #define CAPABILITY mwimp
498    START_CAPABILITY
499    #define FUNCTION mwimp_ExternalModel
500      START_FUNCTION(double)
501      ALLOW_MODELS(ExternalModel)
502    #undef FUNCTION
503  #undef CAPABILITY
504
505  // Retrieve the DM mass in GeV for generic models
506  QUICK_FUNCTION(DarkBit, mwimp, OLD_CAPABILITY, mwimp_generic, double, (),
507       (TH_ProcessCatalog, DarkBit::TH_ProcessCatalog), (DarkMatter_ID, std::string))
```

The new function.

Be sure to change this from what is there now!

- Then add a new function to `DarkBit/src/DarkBit.cpp`:

```
67    void mwimp_ExternalModel(double & result)
68    {
69        using namespace Pipes::mwimp_ExternalModel;
70        result = *Param["Mp1"];
71    }
```

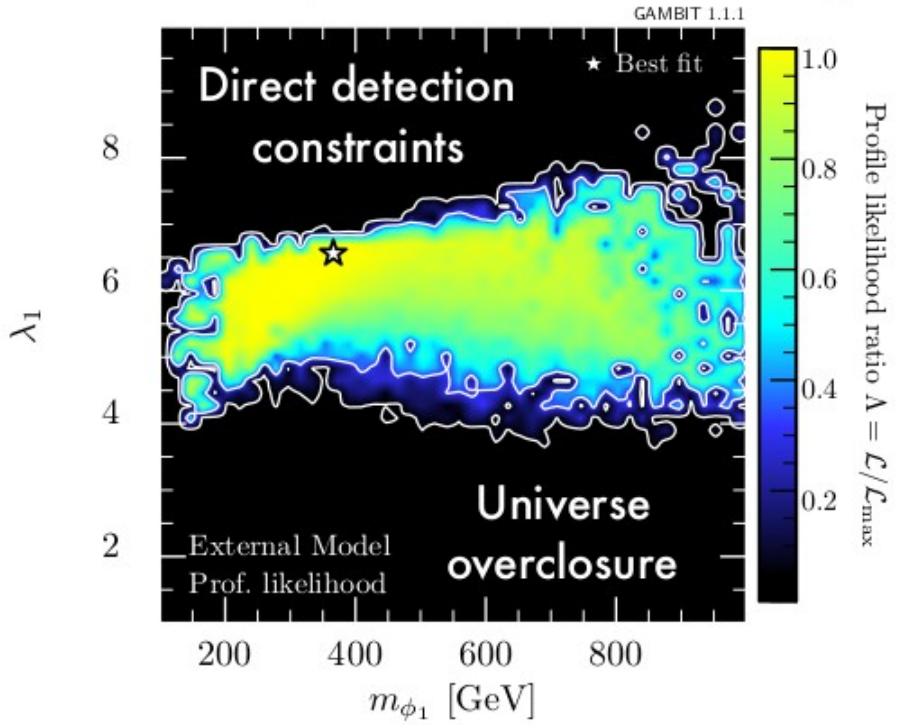
This function just returns the dark matter mass from the model parameters.

Tutorial 4 Step 7: Recompile and scan

- Run `make clean` and then `make -jn gambit` in the build directory to incorporate all of your changes. (This takes a long time, sorry :( )
- Run `./gambit backends` and `./gambit models`. You should see the backend and model you added there. Run `./gambit ExternalModel` to learn more about our new model.
- Use the `ExternalModel_DM.yaml` file to run a scan of the new model! You can stop GAMBIT from complaining about missing descriptions of new capabilities with the `--developer` flag.

Tutorial 4 : Scan results

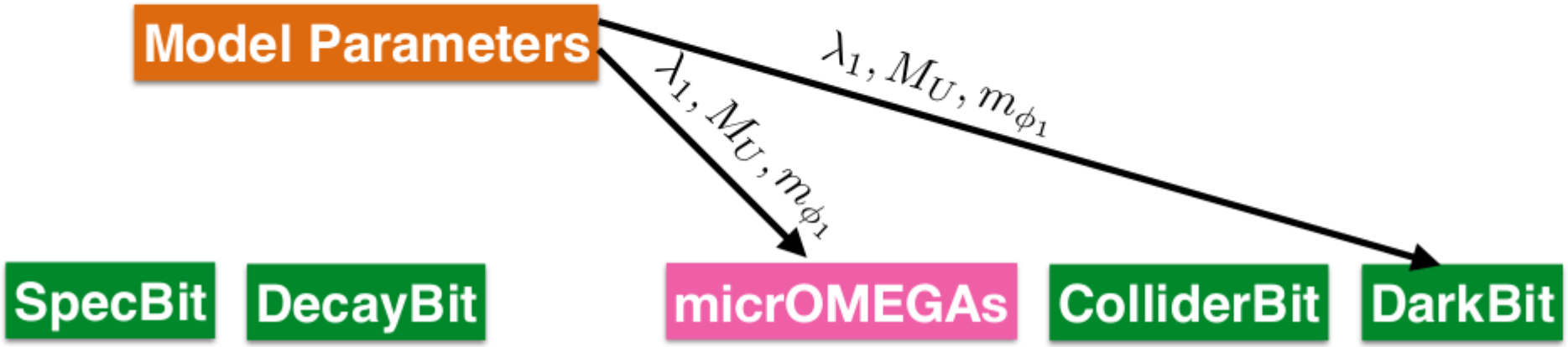
Allowing M_U to be greater than 400 GeV, I find this best fit region of the parameter space when scanning over M_U, m_1, and λ_1



Can you reproduce this? How does the plot change if M_U is restricted to higher masses?

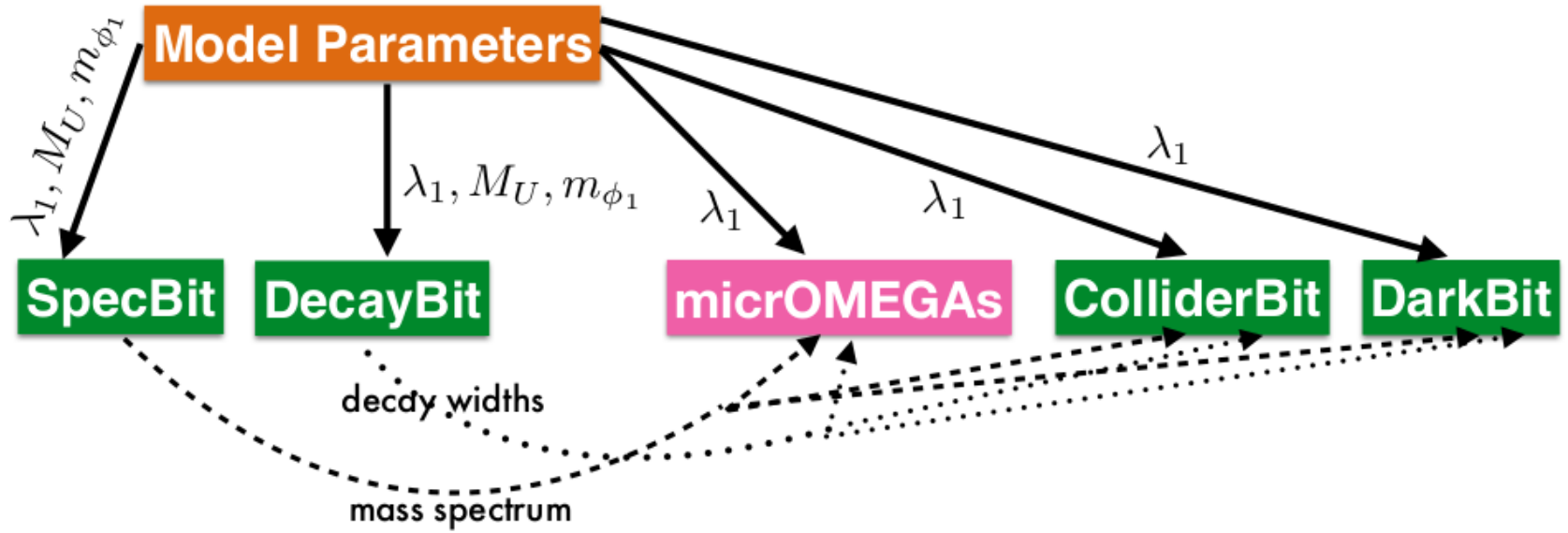
Tutorial 4 : SpecBit and DecayBit

In this quick and dirty example, we bypassed SpecBit and DecayBit and passed model parameters directly to our backend:



Tutorial 4 : SpecBit and DecayBit

Many of the observable calculations in GAMBIT require spectrum objects from SpecBit and decay tables from DecayBit, so the more canonical implementation would take this form:



For more details, see the SpecBit paper.