GAMBIT

Martin White

THE UNIVERSITY of ADELAIDE

SUB CRUCE LUMEN

CoEPP

ARC Centre of Excellence for Particle Physics at the Terascale

Beyond the Standard Model (BSM)

- There is no shortage of BSM physics theories

  - can either take a top-down or bottom up approach
  - we all have our favourites

- Any BSM theory could show up in lots of places

  - Higgs and supersymmetry searches at the LHC and its predecessors
  - 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 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)

This begs the question...

How do we determine which models are in, and which models are out?

Answer

- Combine results from all relevant experimental searches.

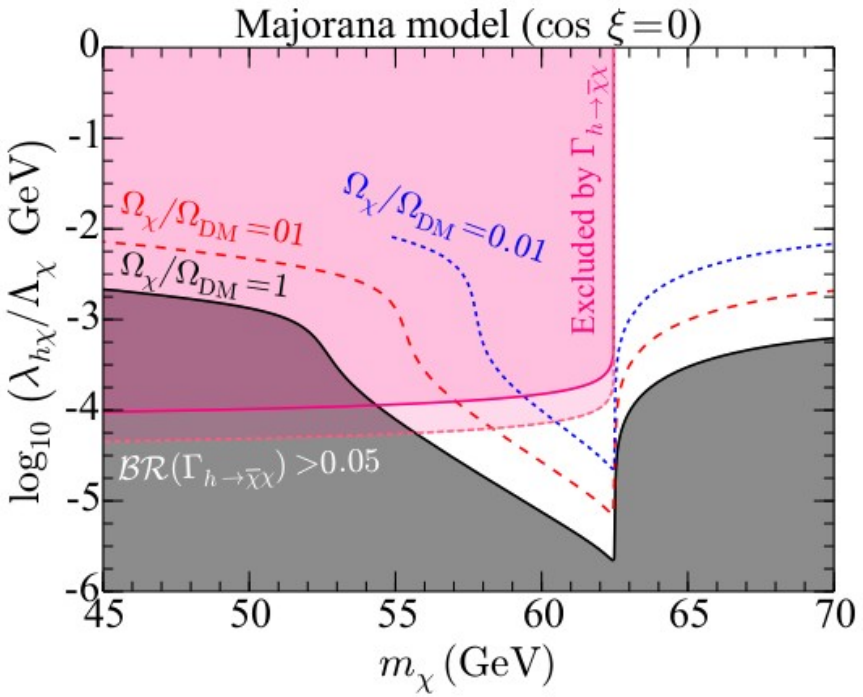
This begs the question...

- Combine results from all relevant experimental searches
- This is straightforward for models with few parameters:
- Simplest method:
  - overlay exclusion curves from different experiments
  - look for “excluded” and “non-excluded regions”

This begs the question...

- Combine results from all relevant experimental searches
- This is straightforward for models with few parameters:
- Simplest method:
  - overlay exclusion curves from different experiments
  - look for “excluded” and “non-excluded regions”

A.Beniwal, F.Rajec, C.Savage, P.Scott, MJW,A. Williams,work in progress



$$\mathcal{L}_\chi = \mathcal{L}_{\rm SM} + \frac{1}{2}\overline{\chi} i\partial\!\!\!/ \chi - \frac{1}{2} m_\chi \overline{\chi}\chi - \frac{1}{2}\frac{\lambda_{h\chi}}{\Lambda_\chi}\left[\cos\xi\,\overline{\chi}\chi + \sin\xi\,\overline{\chi} i\gamma_5\chi\right]\left(v_0 h + \frac{1}{2}h^2\right)$$

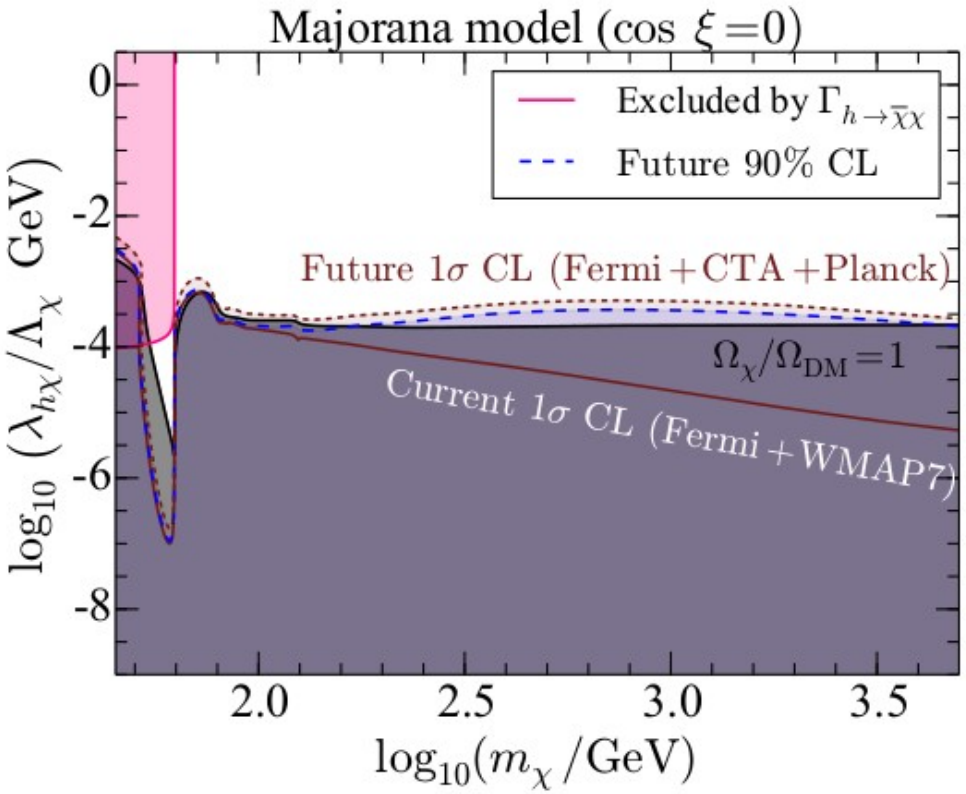
This begs the question...

What if there are many constraints?

What if there are many constraints?

- Need to combine them properly into a *joint likelihood*

- For two parameter models, can then continue as before



A.Beniwal, F.Rajec, C.Savage, P.Scott,
MJW,A. Williams,work in progress

$$\mathcal{L}_\chi = \mathcal{L}_{\rm SM} + \frac{1}{2}\overline{\chi} i\partial\!\!\!/ \chi - \frac{1}{2} m_\chi \overline{\chi}\chi - \frac{1}{2}\frac{\lambda_{h\chi}}{\Lambda_\chi}\left[\cos\xi\,\overline{\chi}\chi + \sin\xi\,\overline{\chi} i\gamma_5\chi\right]\left(v_0 h + \frac{1}{2}h^2\right)$$

Another question

*What if there are many **parameters**?*

What if there are many parameters?

- Much harder in principle

- Need to:

  - scan the space (need very smart methods for a large number of parameters)
  - interpret the results (Bayesian/frequentist)
  - project down to parameters of interest (marginalise/profile)

i.e. need a global statistical fit

The final question

*What if there are many **models**?*

What if there are many models?

- Of course, there *are* many models
- Can rinse and repeat the above procedure
- Notice that we have two distinct problems

Parameter estimation

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

Model comparison

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

Existing "global fit" solutions

MasterCode	SuperBayeS	Fittino	Rizzo et al.
Monolithic Frequentist Nested sampling, MCMC, grad. descent	Monolithic Freq./Bayes. Nested sampling, MCMC	(∼)Monolithic Frequentist MCMC	Monolithic None None (random)
CMSSM$\pm\epsilon$	(p)MSSM-15, CMSSM$\pm\epsilon$, mUED	CMSSM$\pm\epsilon$	(p)MSSM-19
Basic: $\Omega_{\rm DM}$, LUX, XENON	Basic: $\Omega_{\rm DM}$, Fermi, IceCube, XENON	Basic: $\Omega_{\rm DM}$, Fermi, HESS, XENON	Event-level: Fermi. Basic: $\Omega_{\rm DM}$, IceCube, CTA
ATLAS resim, HiggsSignals, basic flavour.	ATLAS direct sim, Higgs mass only, basic flavour.	ATLAS resim, HiggsSignals, basic flavour.	ATLAS+CMS +Tevatron direct sim, basic flavour.
m_t, m_Z, $\alpha_{\rm EM}$, hadronic matrix elements	m_t, m_b, α_s, $\alpha_{\rm EM}$, DM halo, hadronic matrix elems.	m_t	None

Issues with current global fit codes

- Strongly wedded to a few theories (e.g. constrained MSSM / mSUGRA)
- Strongly wedded to a few theory calculators
- All datasets and observables basically hardcoded
- Rough or non-existent treatment of most experiments (astroparticle + collider especially)
- Sub-optimal statistical methods / search algorithms
- Not all codes are publicly available

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 interface, CalcHEP interface to Micromegas, MadDM, automated NLO calculations through Madgraph/NLOCT + much, much more

- The global fit world has not kept up with this

  - most people hard-code their own solution for each particular study

- Several innovations are need to rectify this:
  - how do we store model parameters in a sufficiently abstract way?
  - how do we tie disparate codes together?
  - how do we make LHC and other constraints model independent?

Introducing GAMBIT

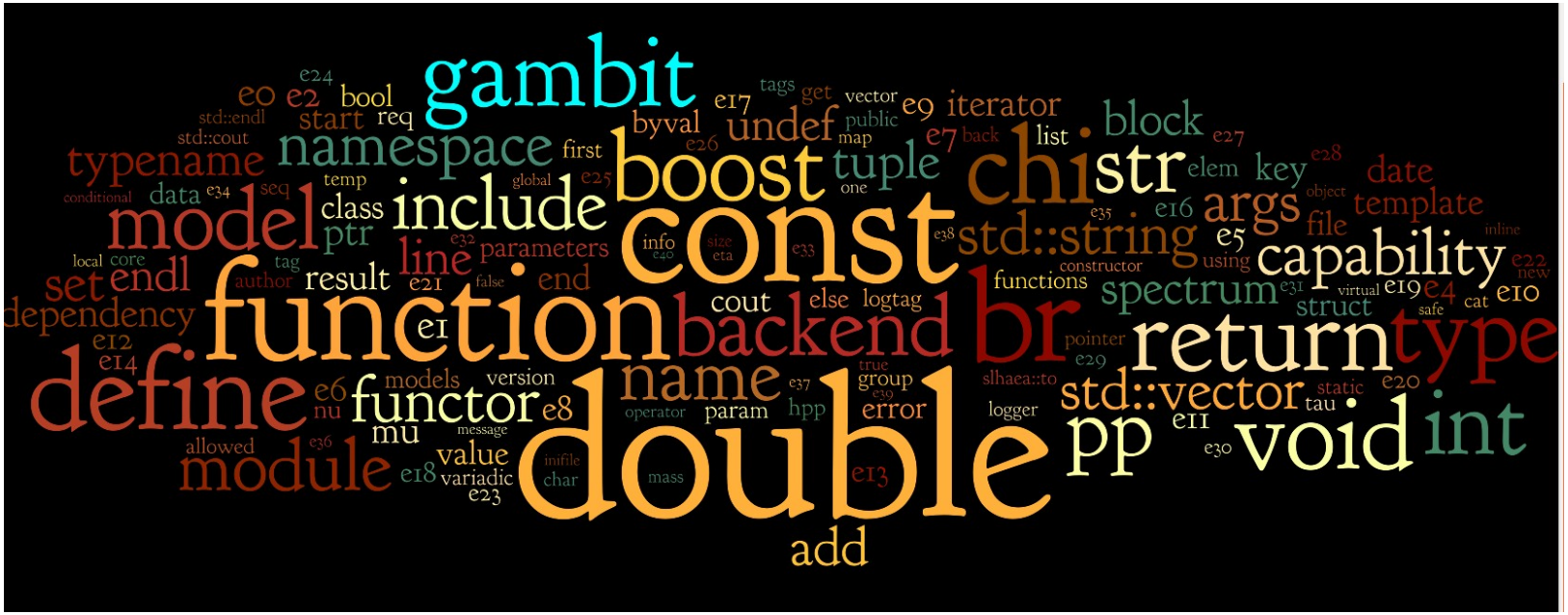
- GAMBIT: The Global And Modular BSM Inference Tool is:

  1) A collaboration of ~30 theorists and experimentalists

  2) A new *public* global fitting code

  3) A programme of physics analyses that (1) performs with the code

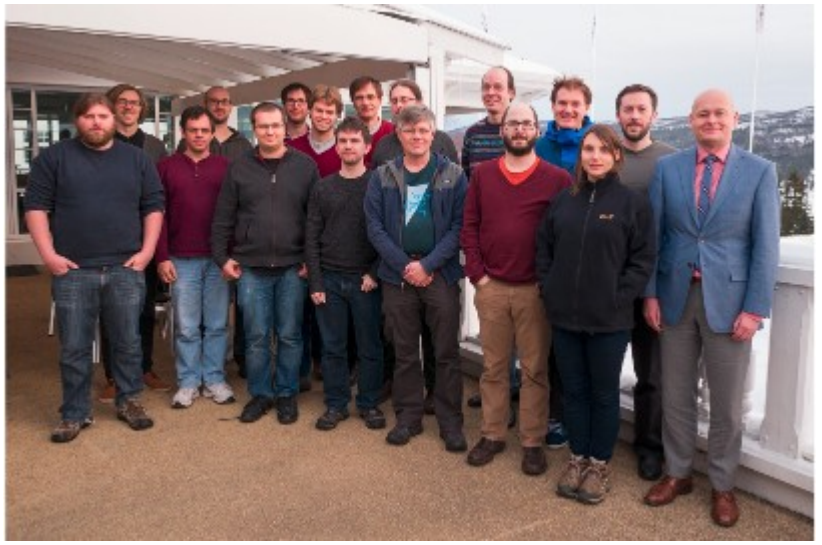
First results and code release: later this year



Introducing GAMBIT

- 26 members, 15 institutions, 9 countries, 8 experiments, 4 major theory codes

Fermi-LAT	J. Conrad, J. Edsjö, G. Martinez P. Scott
ATLAS	A. Buckley, P. Jackson, C. Rogan, A. Saavedra, M. White
CTA	C. Balázs, T. Bringmann, J. Conrad, M. White
HESS	J. Conrad
LHCb	M. Chrząszcz, N. Serra
IceCube	J. Edsjö, C. Savage, P. Scott
AMS-02	A. Putze
CDMS, DM-ICE	L. Hsu
XENON/DARWIN	J. Conrad
Theory	P. Athron, C. Balázs, T. Bringmann, J. Cornell, L. Dal, J. Edsjö, B. Farmer, A. Krislock, A. Kvellestad, M. Pato, F. Mahmoudi, A. Raklev, C. Savage, P. Scott, C. Weniger, M. White



Design principles of GAMBIT

- We set out to create a package that would:
  - be fully general: easy to add new datasets or theoretical models
  - have plug and play scanning, physics and likelihood components
  - have an extensive built in model database (not just SUSY…)
  - have a comprehensive library of observables/data
  - allow general statistical treatments (random/Bayesian/frequentist)
  - be *fast* (smart scanning algorithms, massive parallelisation)
  - perform more detailed modelling of physical observables than previous codes

GAMBIT modules

- **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 BSMspectrum object,providing RGE running, masses, mixings via interchangable interfaces to different RGE codes
- **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?

- GAMBIT modules consist of a number of standalone *module functions*

  - Module functions can depend on each other, or on “backends”

  - backends are external code libraries (DarkSUSY, FeynHiggs, etc) that include different functions, some or all of which might be needed

  - GAMBIT automates and abstracts the interfaces to backends (backend functions are tagged according to what they calculate)

  - with appropriate module design, different backends and their functions can be used interchangeably

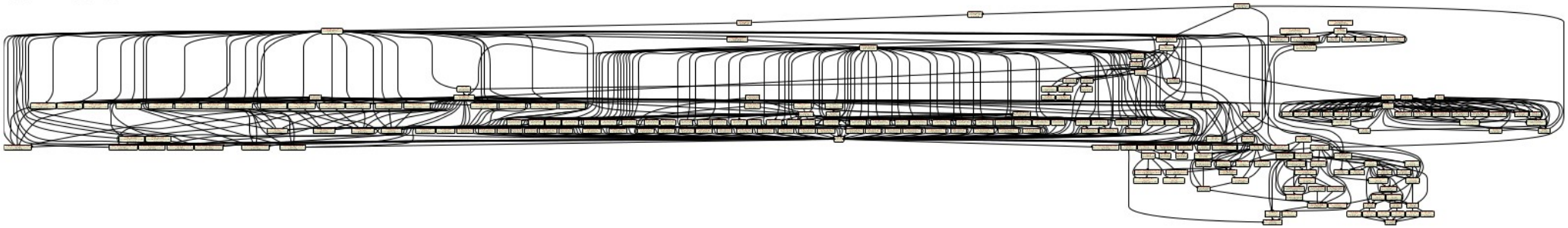
  - GAMBIT dynamically adapts to use whichever backends are locally present

How GAMBIT operates (very roughly)

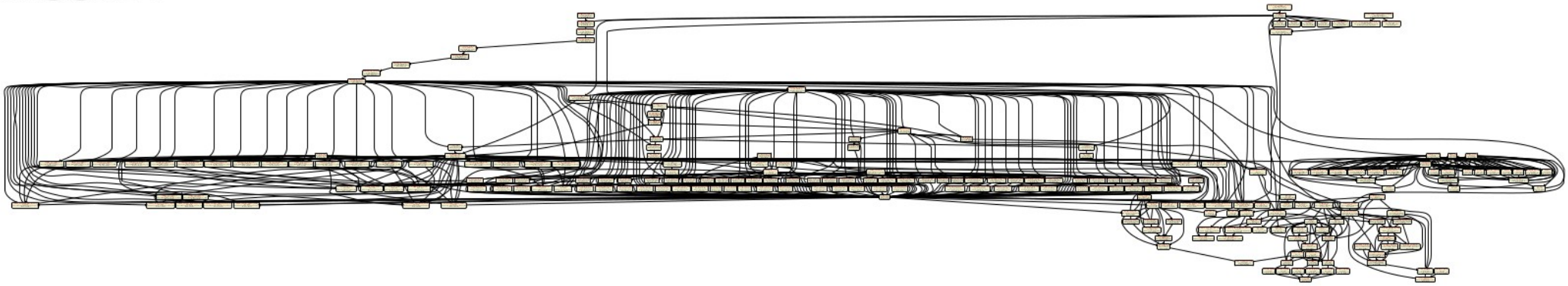
- The user states what they want to calculate
- GAMBIT works out which module functions and backends can calculate those things
  - user must specify a choice in the case of things that can be calculated by multiple codes
- We then have a dependency problem:
  - some things can only be calculated after others
- GAMBIT thus uses a complex dependency resolver (based on graph theoretic techniques):
  - Module functions and backend functions get arranged into a dependency tree
  - Starting with requested observables and likelihoods, fills each dependency and backend requirement
  - Obeys rules at each step: allowed models, allowed backends, constraints from input file, etc
- GAMBIT calculates observables plus likelihoods, outputs results in user-specified form

Dependency resolution

CMSSM:

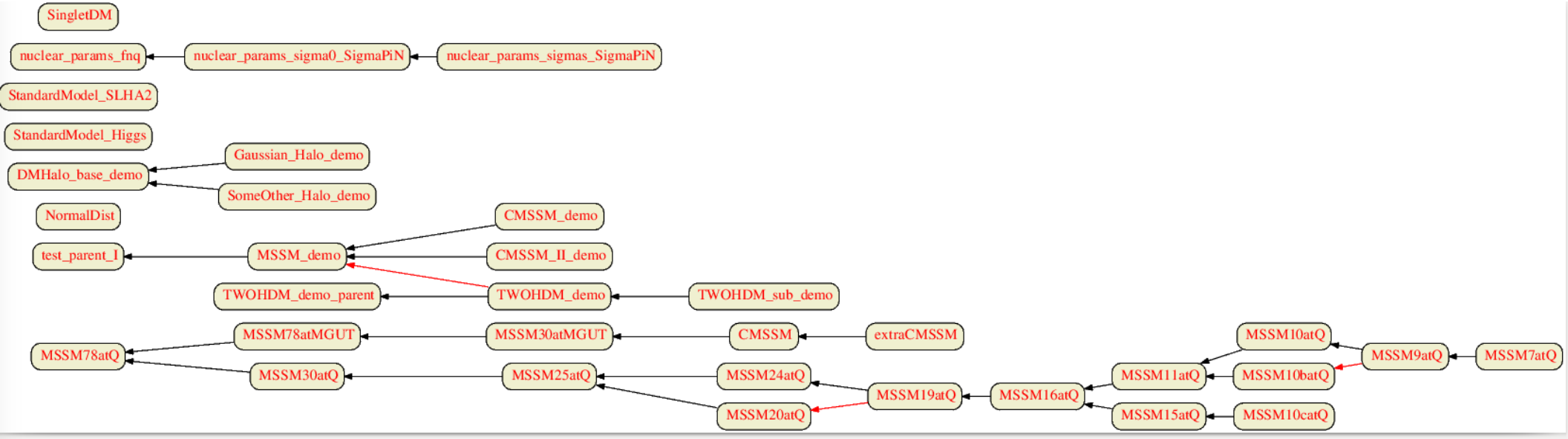


MSSM7:



Hierarchical model database

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



User input

- Basic interface is a yaml file

  specify parameters, ranges, priors

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

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
```

User additions

- Adding new module functions is straightforward

  - detailed instructions will be supplied with the code
  - user adds C++ code and hooks it up using a standard recipe

- Adding new models is also straightforward

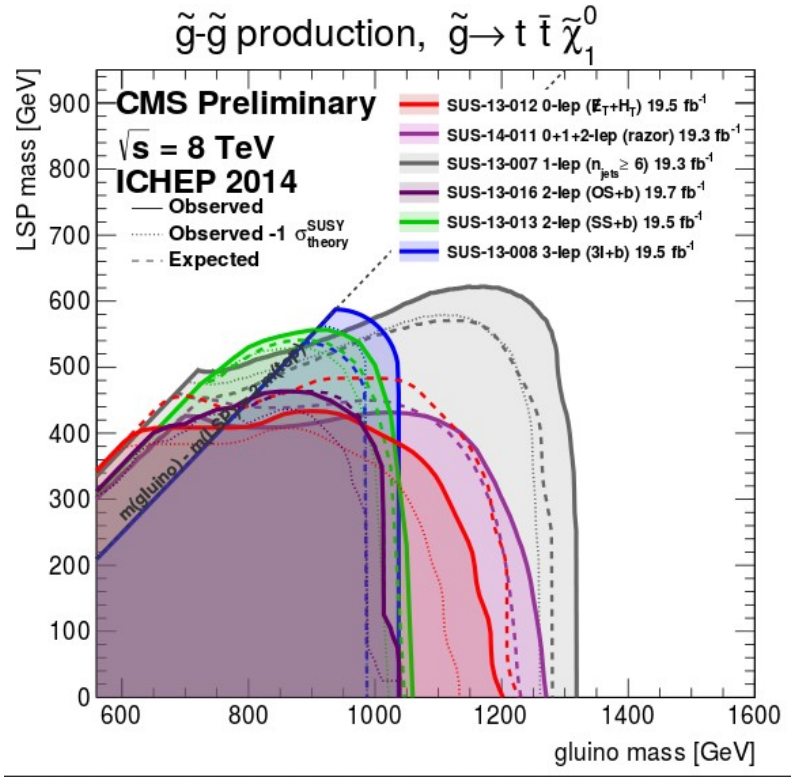
  - a detailed worked example will be supplied with the first release
  - can exploit existing interfaces to calculate observables in Micromegas/Pythia
  - more generally, the code is flexible enough to cope with new backends

ColliderBit basics

- Handles LEP and LHC constraints on BSM models (Tevatron information to come)
- Higgs constraints handled via interface to HiggsBounds/HiggsSignals plus invisible width data

- LEP results:
  - complete model-independent recast of direct sparticle searches (new code)

- ATLAS and CMS results are more complicated to reinterpret, typically shown like this:



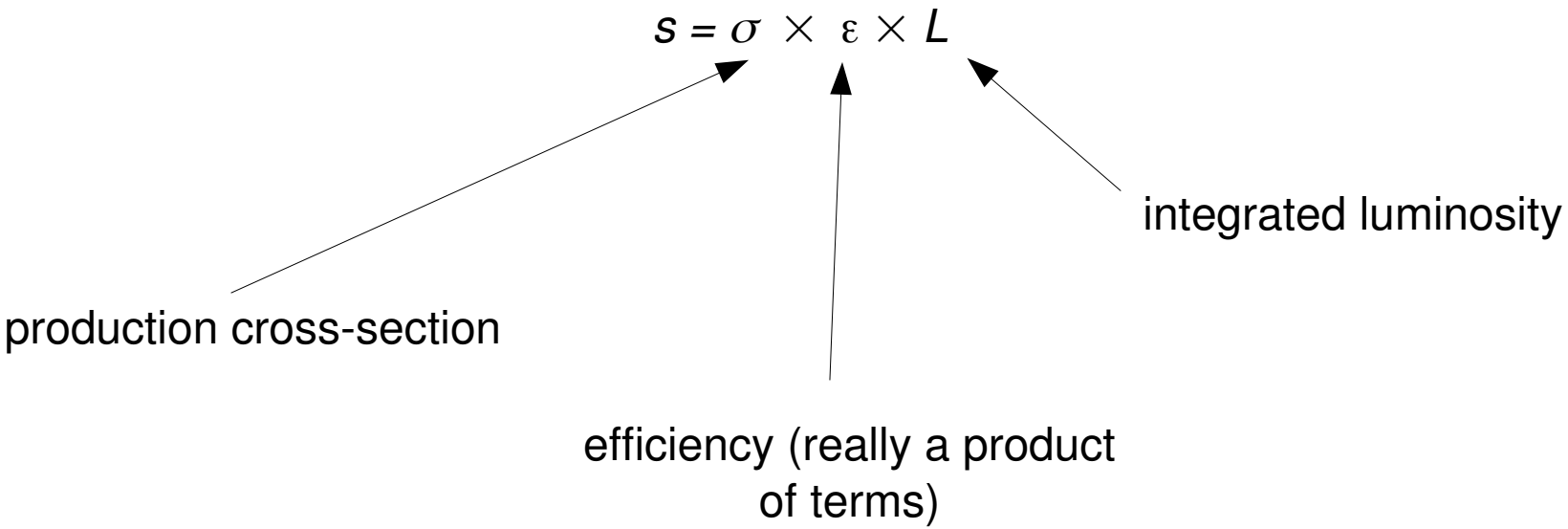
How can we apply this limit to other models?

• Given *s* expected signal events, *b* background events and *o* observed events, we have a Poisson likelihood for the likelihood of a model after an LHC search:

$$L = e^{-(s+b)}(s+b)^{o} \div o!$$

• Need to include the effects of systematics, and combine likelihoods from different searches (not that hard, modulo published information on correlations)

• The only rigorous approach to reinterpreting collider limits is to accurately calculate *s*



Calculating *s*

$$s = \sigma \times \varepsilon \times L$$

CALCULATE PRODUCTION CROSS-SECTION(S)

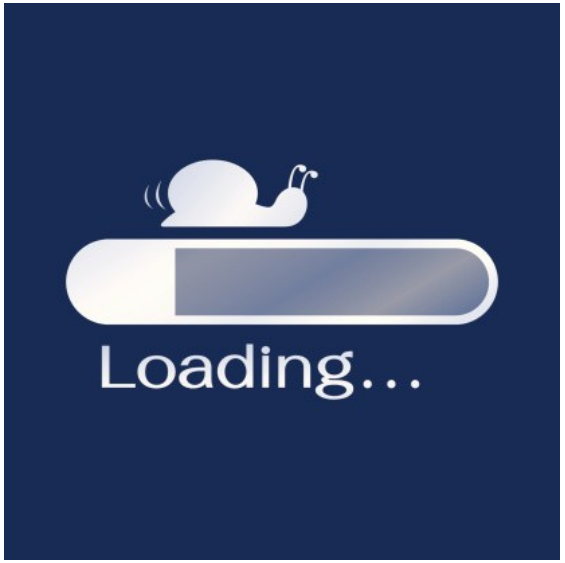
Generate MC events

Simulate LHC detector effects

Apply LHC analysis cuts

- Each of these steps may take minutes or hours

  - this has traditionally limited the statistical rigour of attempts to present LHC results



Current tools for applying LHC constraints

• Several tools for the application of collider limits have appeared in recent years:

CHECKMATE: Contains a custom version of the DELPHES event simulation plus analysis code for lots of LHC searches. Does not do any MC event generation or cross-section calculation.

SMODELS: Looks at LHC "simplified model" limits and checks to see if a given model is excluded by those limits (by scaling each topology by the appropriate cross-section times branching ratio). Gives very conservative limits and is only applicable to SUSY.

FASTLIM: Uses simplified model limits plus efficiency tables to approximate the efficiency for general SUSY models. Only applicable to SUSY.

• Although some of the approaches taken above may generalise, there is no fully general package capable of quickly applying LHC limits to new models out of the box

- in ColliderBit we perform the full simulation chain, with a variety of speed improvements

The ColliderBit approach: Cross-sections

- Use LO+LL cross-sections supplied by MC generator as a default

  - this is all that exists for most WIMP models
  - lags behind state of the art for others (SUSY), but will give conservative limits

- A look up table of NLO cross-sections has been in development for some time for SUSY

  - tricky to do generally in large dimensional spaces
  - the only successful approaches use significant approximations that reduce generality (e.g. NLLFAST)
  - not ready for first release

The ColliderBit approach: MC event generation

- We have made a custom OpenMP parallelised version of the Pythia 8 generator

  - includes other hacks to speed up Pythia execution (e.g. remove redundant flag checks)
  - can generate tens of thousands of events on 8 CPUs in seconds
  - Pythia 8 is nice and general for running arbitrary WIMP models

- First release will be capable of everything Pythia is out of the box plus:

  - will add matrix elements for some key models (using Madgraph-Pythia interface)

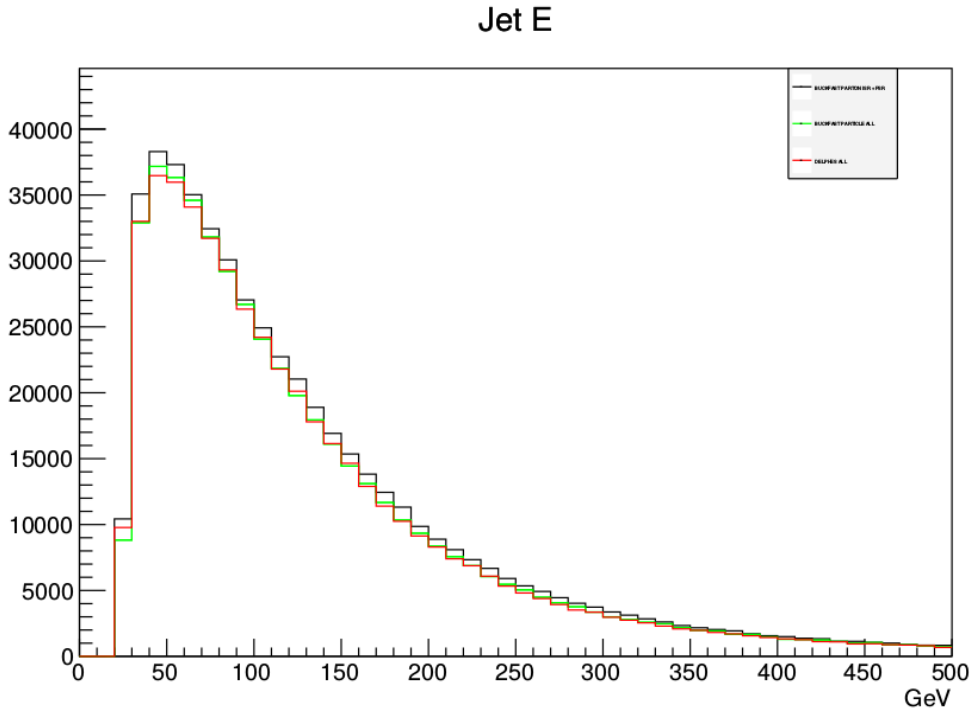
- A more general solution for future versions:

  - interface ColliderBit to general matrix element calculators (e.g. Madgraph)

The ColliderBit approach: Detector simulation

- Have an interface to DELPHES (a standard LHC detector simulation)
- Have also written a faster detector simulation based on four-vector smearing

  - results agree very closely with DELPHES



- A custom “cluster-based” smearing simulation is also in development
- The simulation is run in the main event loop, so that it is implicitly parallelised

The ColliderBit approach: Collider analysis cuts

- Have built a custom event analysis framework

  - this is “simulation blind”: can be used on event info from any source
  - uses public HepUtils event, jet and particle classes

- First release will contain implementations of (8 TeV):

  - ATLAS SUSY searches (0 lep, 0-1-2 lep stop, b jets plus MET, 2 lep EW, 3 lep EW)
  - CMS DM searches (top pair plus MET, mono-b, mono-jet)
  - CMS multilepton SUSY search
  - good coverage for SUSY and DM effective field theory

- We use the GAMBIT statistical routines for likelihood calculations

  - includes effect of systematics on Poisson likelihoods
  - we will approximate LHC exclusion by picking the most constraining signal region

Astrophysics data

- Most astrophysics limits are published using simplified models

  - e.g. effective WIMP, one annihilation channel,etc

- Need to able to cope with:

  - arbitrary direct detection couplings (i.e. potential q and v dependence of cross-section)
  - arbitrary indirect detection decay/annihilation fractions

- Impacts of new unstable particles are hard

  - need to simulate decays “on the fly”

- Calculating relic density for general models is also challenging

  - we need to somehow abstract the calculation rather than have a presumed model
  - i.e. feed in partial annihilation rates, co-annihilations, resonances, etc

DarkBit details

- DarkBit is designed to:

  - make the full suite of indirect and direct searches available for global fits

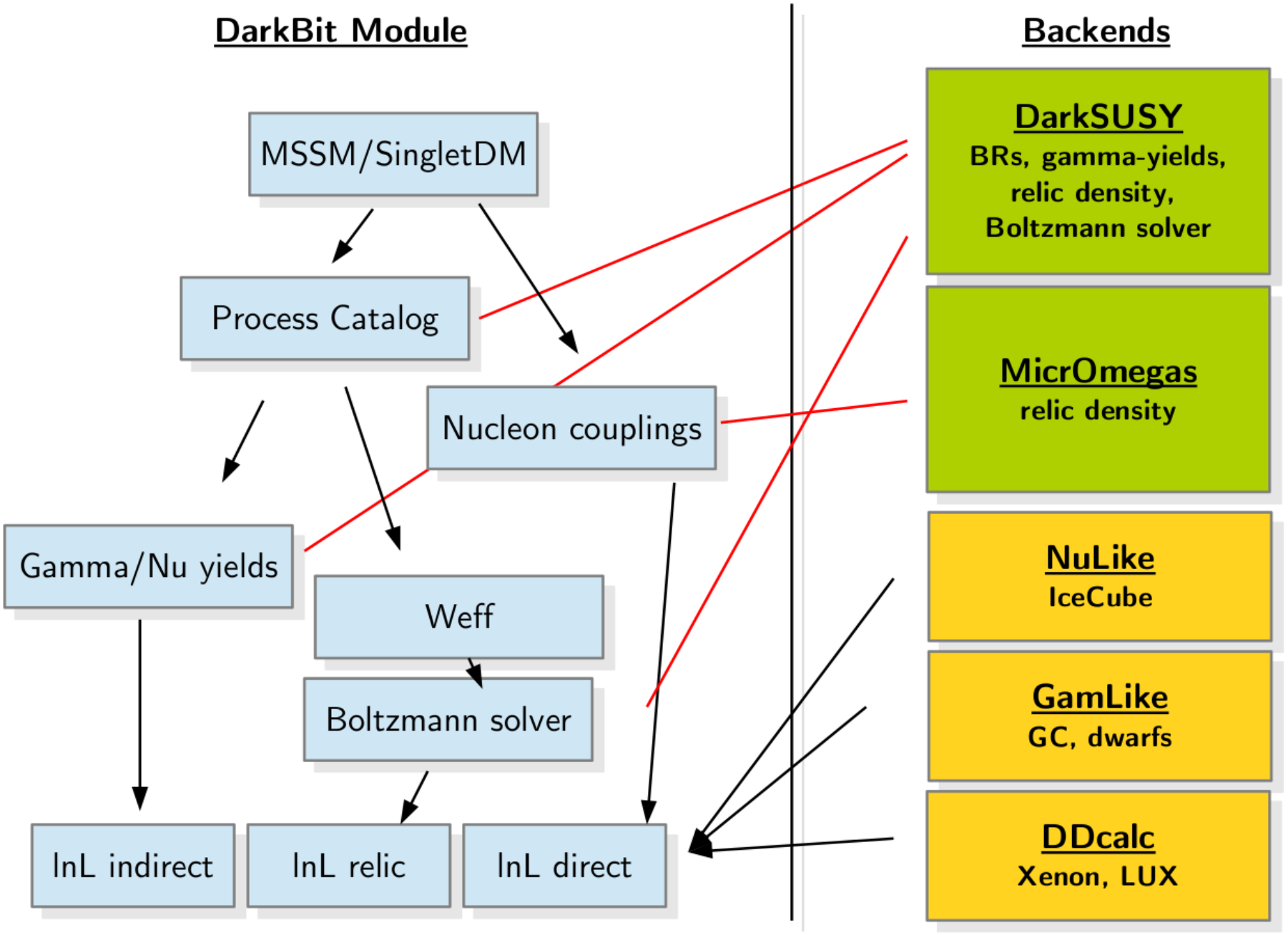
  - have the most accurate likelihoods available (event-level where possible!)

  - include experimental, astrophysical and nuclear systematics consistently in one fit

  - allow more control of relic density calculations

  - do it all in a modular and easily extendable way (i.e. DM might be something else...)

DarkBit overview



DarkBit: Gamma ray likelihoods (gamLike)

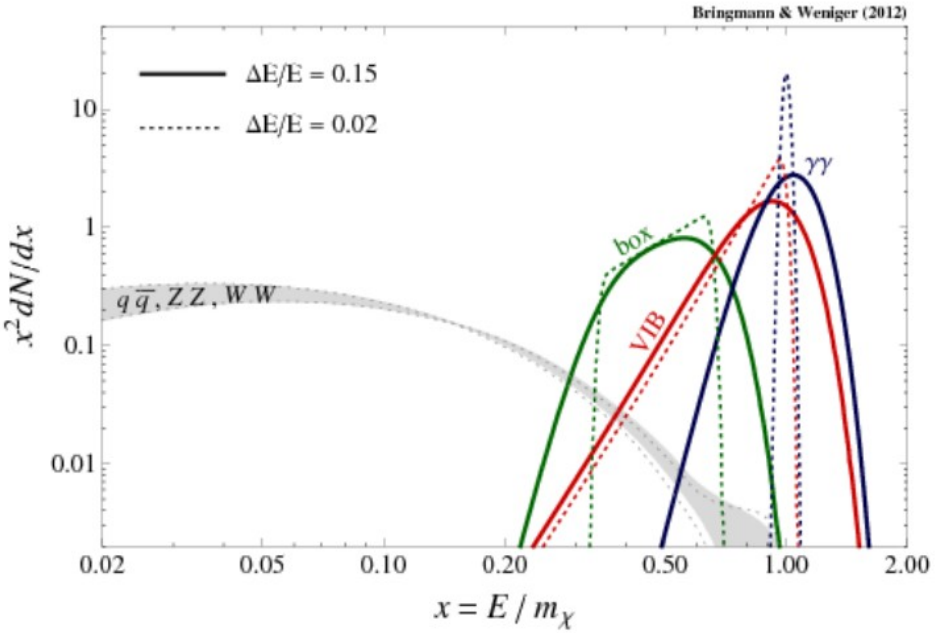
Process Catalog
Annihilation & decay channels

$$\left.\frac{d(\sigma v_{\rm rel})}{dE_1 \ldots dE_n}\right|_v$$

Cascade decay
Monte Carlo

DarkSUSY
PPPC
MicrOmegas

Tabulated yields



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

- Uncertainties in the DM distribution are internally marginalised over
- Correct treatment of energy dispersion and spectral singularities (lines, virtual internal Bremsstrahlung, boxes).

Other nice GAMBIT features

- **Scanners:** MultiNest, Diver (diff. evolution), PIKAIA (genetic algorithms), GreAT (MCMC)
- **Statistics:** Bayesian, Profile Likelihood, later full Neyman
- **Parallelisation:** Mixed-mode MPI + openMP, mostly automated
- **Data handling:** d*iskless* generalisation of various Les Houches Accords
- **Backends:** dynamic loading of C++ classes from backends (BOSS)
- **Compilation:**

  - all-in or module standalone modes – easily implemented from single cmake script
  - automatic getters for obtaining, configuring + compiling backends

- **Output:** flexible output streams (ASCII, databases, HDF5, . . .)
- more more more. . .

Comparison with existing tools

Table 1: Features of BSM scanning codes. $\pm\epsilon \equiv$ small variants thereof.

Aspect	GAMBIT	MasterCode	SuperBayeS	Fittino	Rizzo et al.
Design	Modular, Adaptive	Monolithic	Monolithic	(∼)Monolithic	Monolithic
Statistics	Frequentist, Bayesian	Frequentist	Freq., Bayes.	Frequentist	None
Scanners	Differential evolution, genetic algorithms, random forests, t-walk, t-nest, particle swarm, nested sampling, MCMC, gradient descent	Nested sampling, MCMC, grad. descent	Nested sampling, MCMC	MCMC	None (random)
Theories	(p)MSSM-25, CMSSM$\pm\epsilon$, GMSB, AMSB, gaugino mediation, E6MSSM, NMSSM, BMSSM, PQMSSM, effective operators, iDM, XDM, ADM, UED, Higgs-portal DM, extended Higgs sectors	CMSSM$\pm\epsilon$	(p)MSSM-15, CMSSM$\pm\epsilon$, mUED	CMSSM$\pm\epsilon$	(p)MSSM-19
Astro-particle	Event-level: IceCube, Fermi, LUX, XENON, CDMS, DM-ICE. Basic: $\Omega_{\rm DM}$, AMS-02, COUPP, KIMS, CRESST, CoGeNT, SIMPLE, PAMELA, Planck, HESS. Predictions: CTA, DARWIN, GAPS	Basic: $\Omega_{\rm DM}$, LUX, XENON	Basic: $\Omega_{\rm DM}$, Fermi, IceCube, XENON	Basic: $\Omega_{\rm DM}$, Fermi, HESS, XENON	Event-level: Fermi. Basic: $\Omega_{\rm DM}$, IceCube, CTA
LHC	ATLAS+CMS multi-analysis with neural net and fast detector simulation. Higgs multi-channel with correlations and no SM assumptions. Full flavour inc. complete $B \to X_s ll$ and $B \to K^* ll$ angular set.	ATLAS resim, HiggsSignals, basic flavour.	ATLAS direct sim, Higgs mass only, basic flavour.	ATLAS resim, HiggsSignals, basic flavour.	ATLAS+CMS+ Tevatron direct sim, basic flavour.
SM, theory and related uncerts.	m_t, m_b, $\alpha_{\rm s}$, $\alpha_{\rm EM}$, DM halo, hadronic matrix elements, detector responses, QCD and EW corrections (LHC & DM signal & BG), astro BGs, cosmic ray hadronisation, coalescence and propagation.	m_t, m_Z, $\alpha_{\rm EM}$, hadronic matrix elements	m_t, m_b, $\alpha_{\rm s}$, $\alpha_{\rm EM}$, DM halo, hadronic matrix elems.	m_t	None

Summary

- Proper comparison of BSM physics theories to data requires global fits
- Existing tools are increasingly unable to cope with the rapid accumulation of relevant data
- GAMBIT is to be released later this year

  - will allow global fits to much more general models
  - will allow much more data to be used in simple “2 parameter” scans
  - open source, public software
  - faster, more complete exploration of theoretical BSM options
  - easy way to generate benchmark points for experimental optimisations

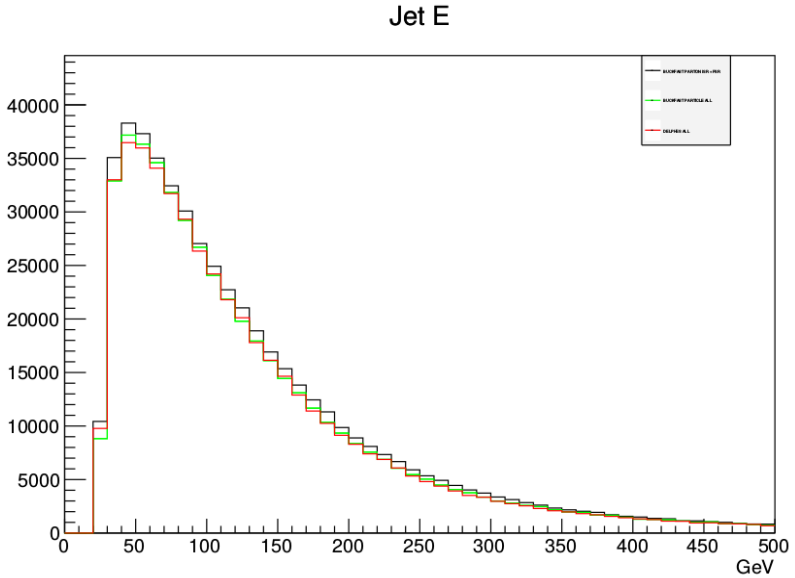
I'm very happy to speak to anyone who might have a use case

ColliderBit calculation details

- **Cross-sections:** Use LO+LL cross-sections supplied by MC generator as a default (Pythia 8) (fast NLO lookup able for SUSY is in progress)

- **MC generation:** custom OpenMP parallelised version of the Pythia 8 generator, plus other speed hacks (tens of thousands of events on 8 CPUs in seconds). NLO generation to come later.

- **Detector sim:** Interface to DELPHES, plus a new faster sim based on four vector smearing



- **Analysis framework:** custom framework, independent of event level (e.g. det sim, parton level, hadron level, etc)

- **Likelihoods:** inline systematic error marginalisation (via nulike)

- **Initial analyses:**

  - ATLAS SUSY searches (0*l*, 0/1/2*l* stop, *b* jets plus MET, 2/3*l* EW
  - CMS multi-*l* SUSY
  - CMS DM (t pair + MET, mono-*b*, monojet)

How GAMBIT operates (more detail)