An update on GAMBIT

Anders Kvellestad, Nordita

on behalf of the GAMBIT collaboration

CERN, December 13, 2016

Outline

1. Global fits
2. GAMBIT
3. ColliderBit: a GAMBIT module for collider physics
4. Preliminary GAMBIT results

1. Global fits

Comparing theories to data

Many parameters and many constraints → *perform a global fit*

Comparing theories to data

Many parameters and many constraints → perform a global fit

- Combine constraints in an overall likelihood function

$$\mathcal{L} = \mathcal{L}_{\text{Collider}}\, \mathcal{L}_{\text{Higgs}}\, \mathcal{L}_{\text{DM}}\, \mathcal{L}_{\text{EWPO}}\, \mathcal{L}_{\text{Flavor}} \ldots$$

Comparing theories to data

Many parameters and many constraints → *perform a global fit*

- Combine constraints in an overall likelihood function
- Explore likelihood across entire parameter space (smart sampling)

Comparing theories to data

Many parameters and many constraints → *perform a global fit*

- Combine constraints in an overall likelihood function
- Explore likelihood across entire parameter space (smart sampling)
- Interpretation: frequentist/Bayesian

Comparing theories to data

Many parameters and many constraints → *perform a global fit*

- Combine constraints in an overall likelihood function
- Explore likelihood across entire parameter space (smart sampling)
- Interpretation: frequentist/Bayesian
- Project down to 1 or 2 parameters (profile/marginalise)

Main challenge

[large number of observables]

$$\times$$

[long calculation time per observable per parameter point]

$$\times$$

[huge number of points required to explore parameter space]

$$\approx$$

$$\infty$$

Main challenge

[large number of observables]

- Introduce nuisance parameters
- Require many external tools

$\times$

[long calculation time per observable per parameter point]

$\times$

[huge number of points required to explore parameter space]

$\approx$

∞

Main challenge

[large number of observables]

$\times$

[long calculation time per observable per parameter point]

- E.g. simulating LHC searches

$\times$

[huge number of points required to explore parameter space]

$\approx$

∞

Main challenge

$$[\text{large number of observables}]$$

$$\times$$

$$[\text{long calculation time per observable per parameter point}]$$

$$\times$$

$$[\text{huge number of points required to explore parameter space}]$$

- Require smart sampling

$$\approx$$

$$\infty$$

Global fits require lots of work, but we shouldn't have to start from scratch every time…

Need a global fit tool with easily replaceable

- models
- theory calculators
- datasets and observables
- scanning algorithms

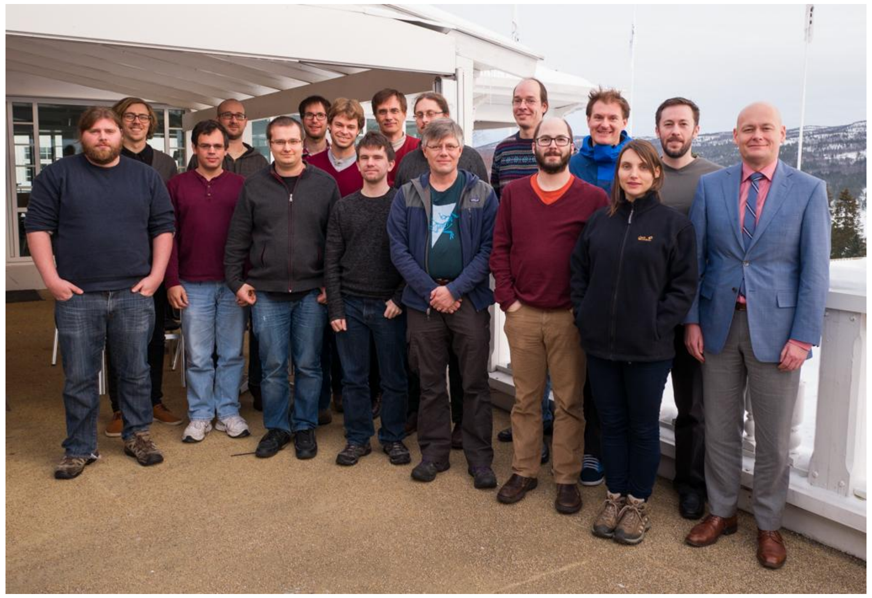
2. GAMBIT

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	A. Buckley, P. Jackson, C. Rogan, 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, J. Conrad, M. White
HESS	J. Conrad
IceCube	J. Edsjö, P. Scott
XENON/DARWIN	J. Conrad, R. Trotta
Theory	P. Athron, C. Balázs, T. Bringmann, J. Cornell, J. Edsjö, B. Farmer, T. Gonzalo, A. Fowlie, J. Harz, S. Hoof, F. Kahlhoefer, A. Krislock, A. Kvellestad, M. Pato, F.N. Mahmoudi, J. McKay, A. Raklev, R. Ruiz, P. Scott, R. Trotta, C. Weniger, M. White, S. Wild



31 Members, 9 Experiments, 4 major theory codes, 11 countries

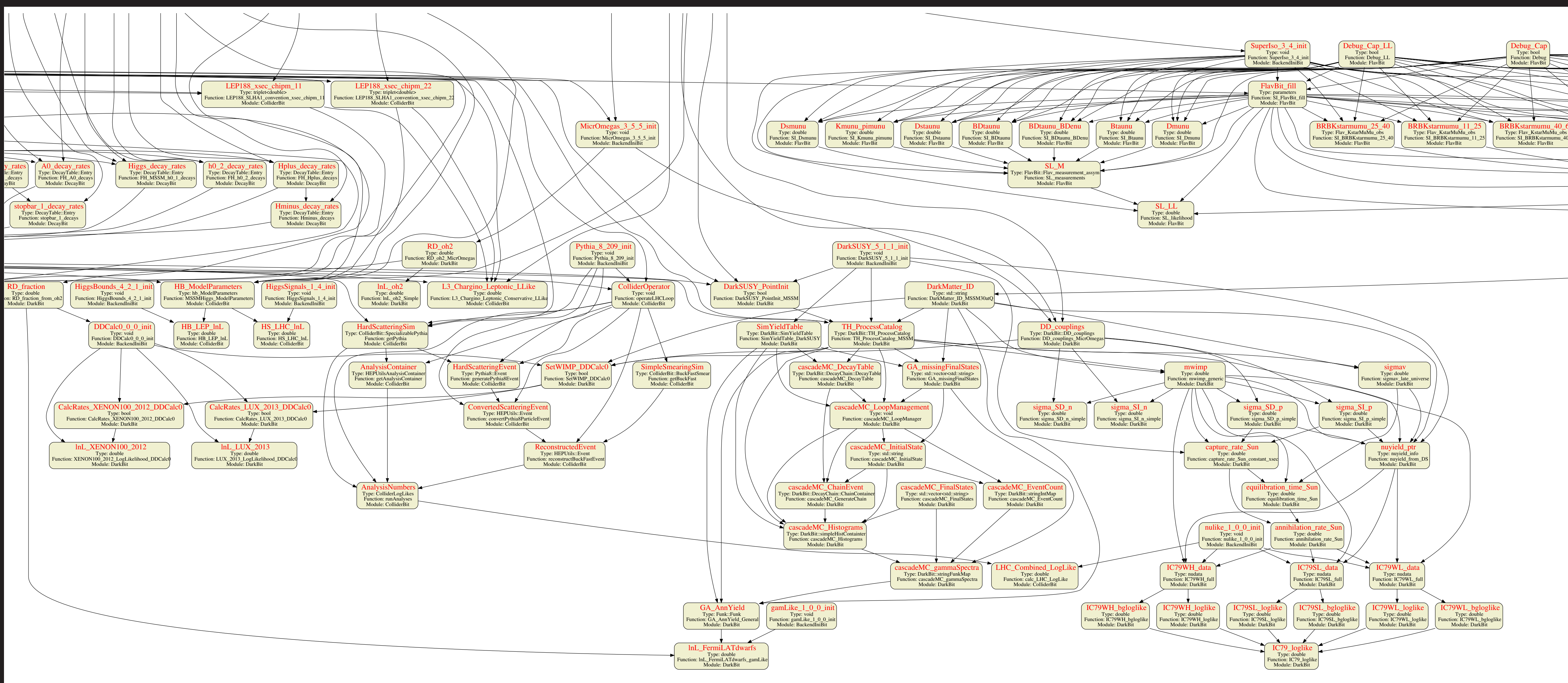
Main design principle

- Code organized in **physics modules**
- A module is a collection of **module functions**
- To ensure modularity: *No hardcoded function calls between module functions!*
- Each function presents GAMBIT with its
  - **allowed models**
  - **dependencies**
  - **capability**
- Dependency resolution at runtime: GAMBIT connects functions as required
- A theory calculation can reside in
  - a **GAMBIT module function** (C++)
  - an **external library** (Fortran, C, C++, Mathematica*, …) loaded at runtime

* Not for first release

Dependency tree

CMSSM



GAMBIT modules

- ColliderBit: Higgs data, SUSY searches at LHC & LEP
- DarkBit: Relic density, direct detection, indirect detection
- FlavBit: B, D, K decays, angular obs., theory uncertainties, correlations
  (See Nazila and Marcin's talk yesterday)
- SpecBit: Generic BSM spectrum object. RGE running, masses, mixings, etc via interchangeable interface to RGE codes.
- DecayBit: Decay widths and BRs
- PrecisionBit: SM likelihoods (nuisance par.), EW precision tests, g-2
- ScannerBit: Scanning algorithms (differential evolution, nested sampling, MCMC, t-walk, grid scan, "random sampling", …)
- [Your own module here]

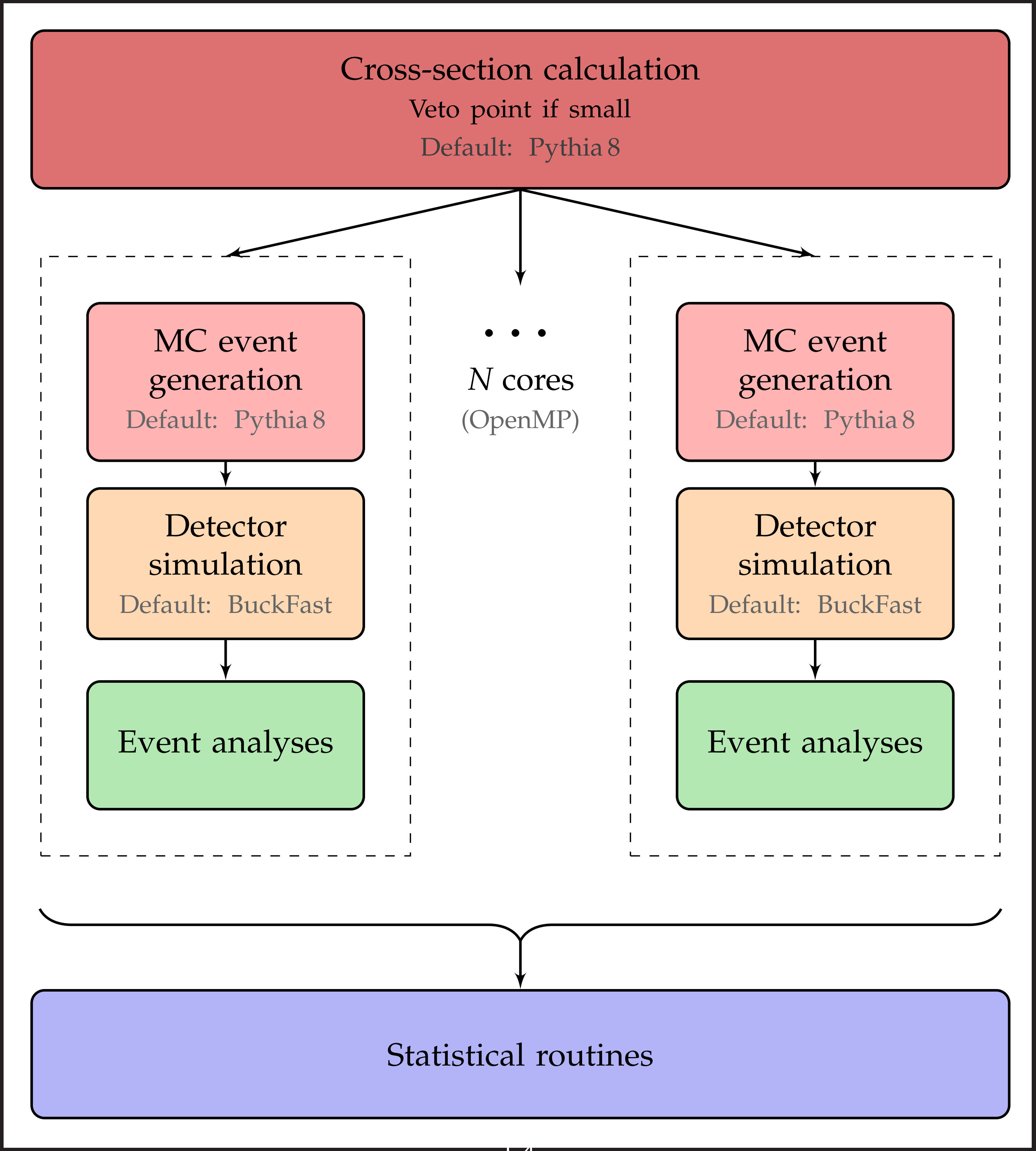
3. ColliderBit

ColliderBit: a GAMBIT module for the calculation of high-energy collider observables and likelihoods

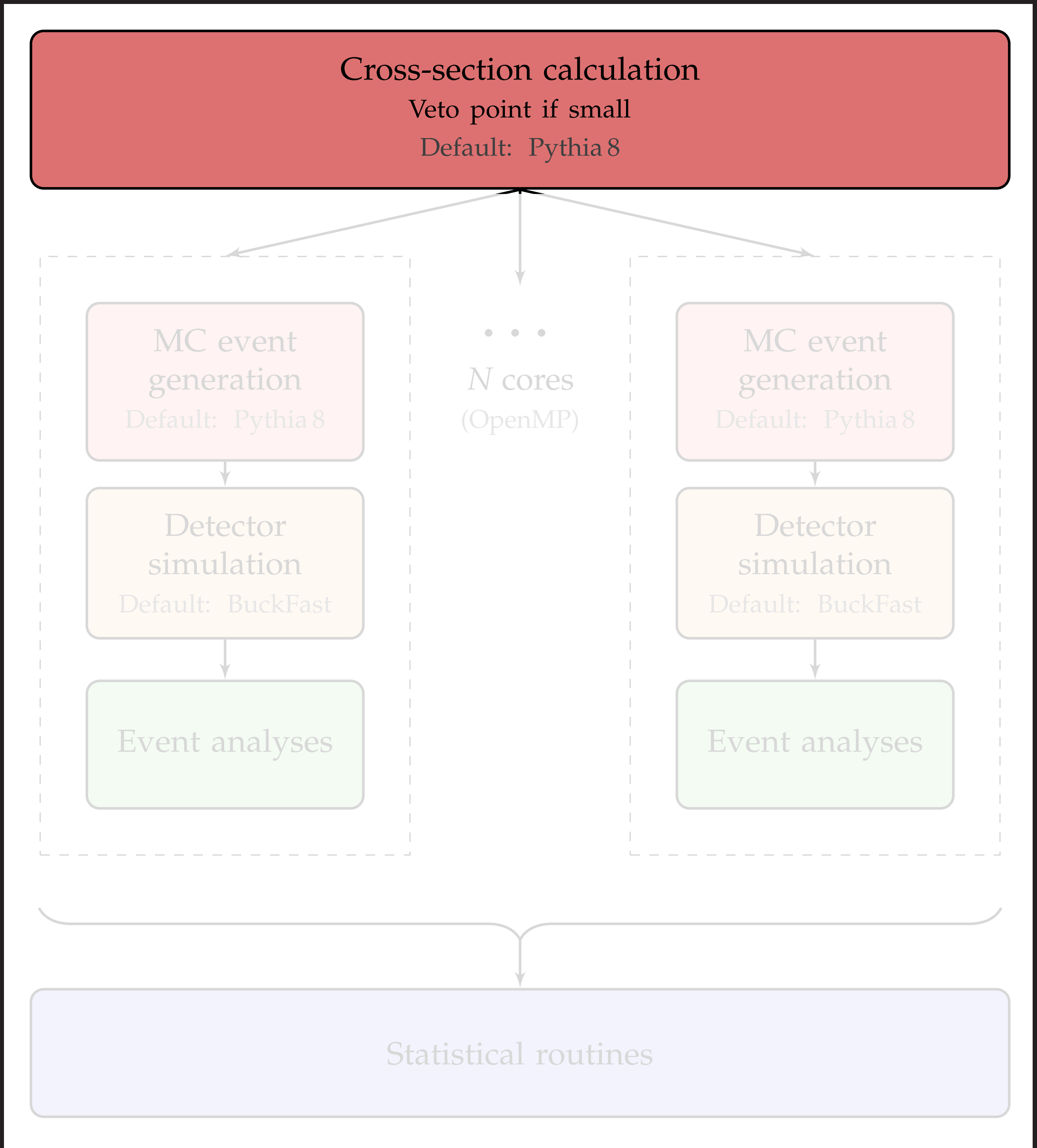
Csaba Balazs[1,2], Andy Buckley[3], Lars Dal[4], Ben Farmer[5], Paul Jackson[2,6], Abram Krislock[4], Anders Kvellestad[7], Antje Putze[8], Are Raklev[4], Christopher Rogan[9], Aldo Saavedra[2,10], Pat Scott[11], Christoph Weniger[12], Martin White[2,6]

- Higgs: Connect HiggsBounds and HiggsSignals as backends (more to come later)
- LEP limits (SUSY): Calculate $\sigma \times BR$, check against published cross section limits
- LHC particle searches: Poisson likelihood from "first principles"
    - cross section
    - MC generation
    - detector sim
    - event analysis

Calculating $s = \sigma \times \epsilon \times L$

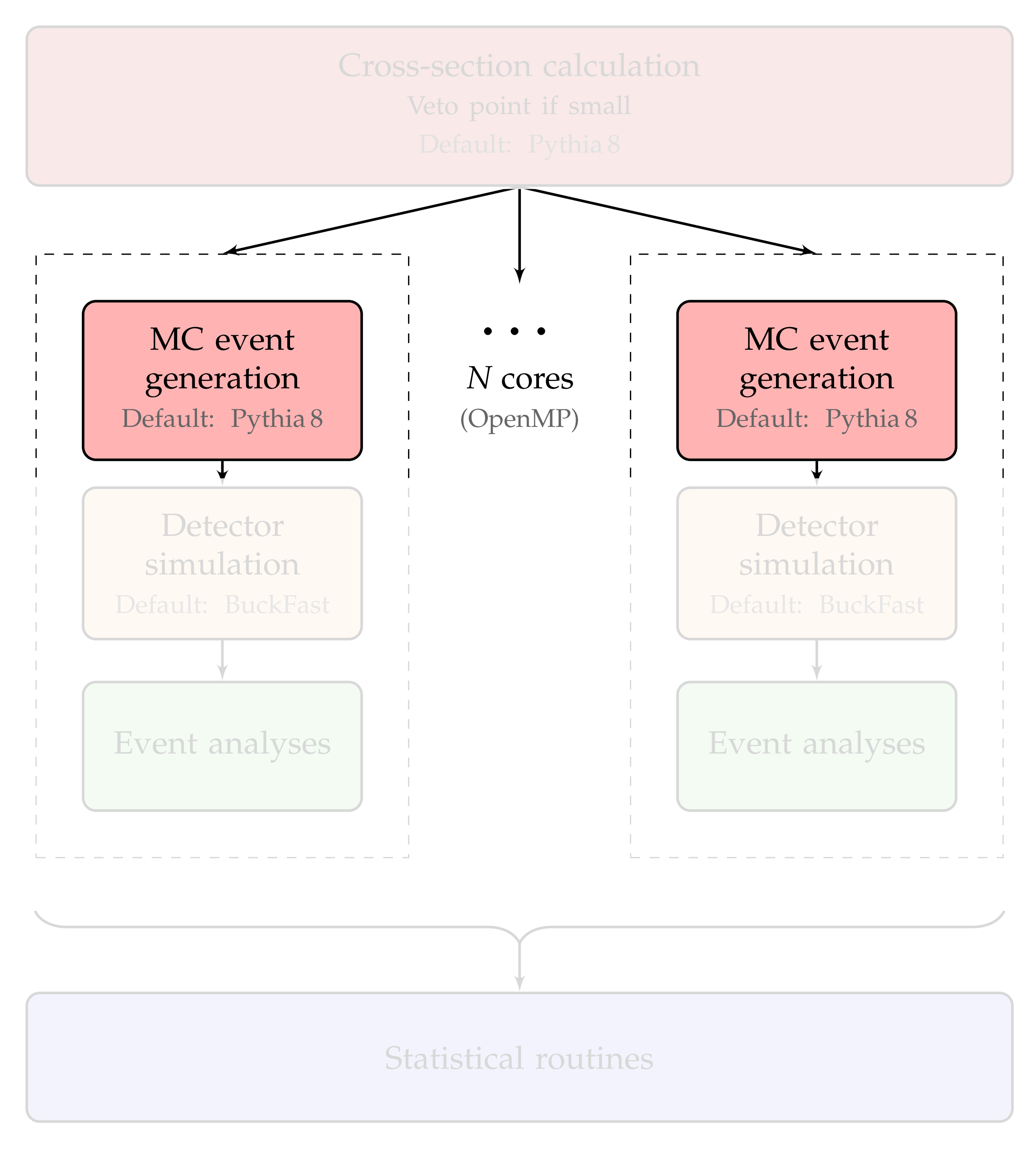


Calculating $s = \sigma \times \epsilon \times L$



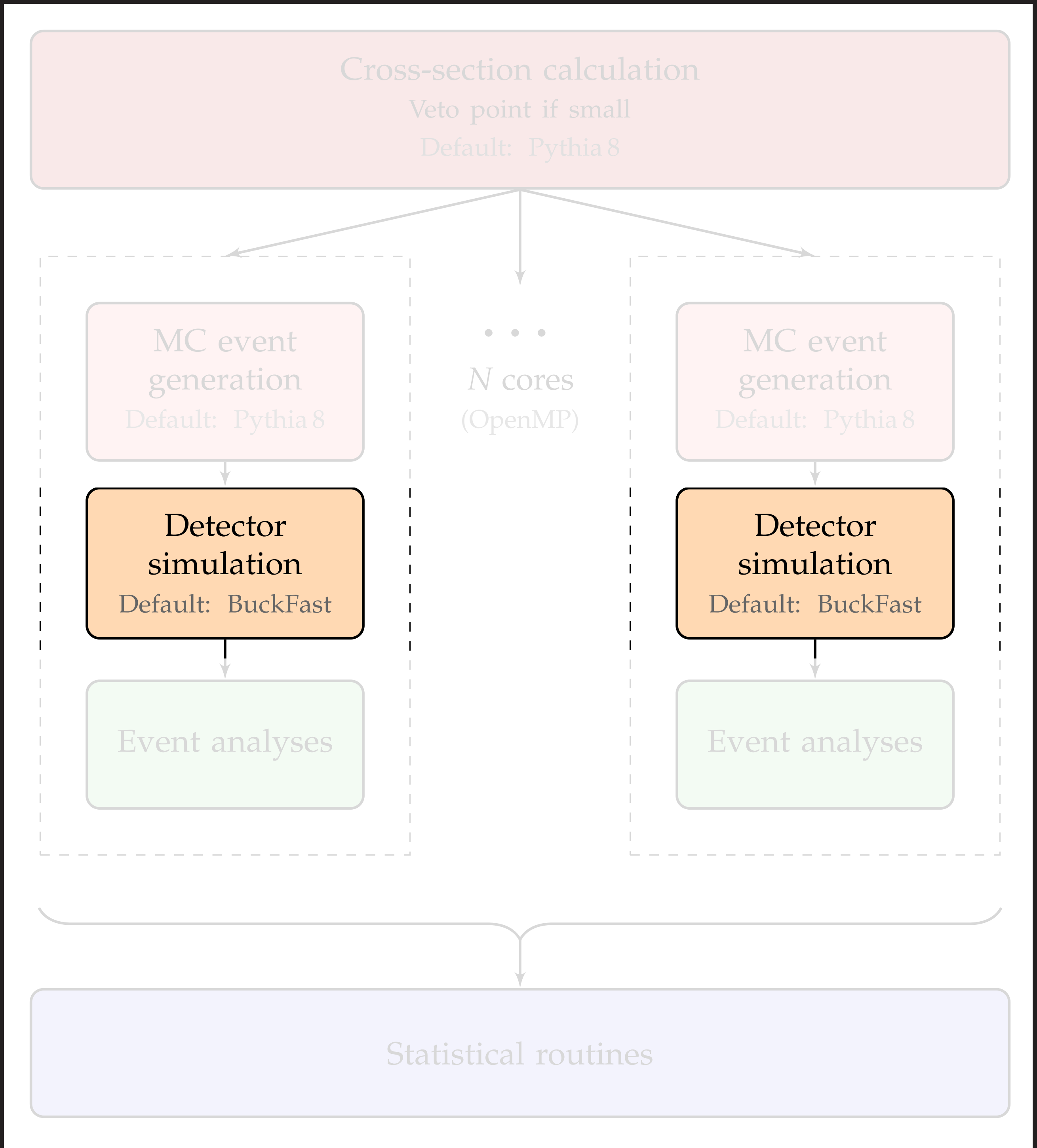
- Use LO+LL cross-sections from MC generator by default
- All that exists for many models
- Behind state-of-the-art for SUSY (but gives conservative limits)
- In global fit: skip event generation if initial max cross section estimate is very small
- *Future:* option for user-supplied cross-sections

Calculating $s = \sigma \times \epsilon \times L$



- Connects Pythia 8 as backend
- Event generation loop parallelised in ColliderBit with OpenMP
- Can generate 10^4 events in a few seconds on 8 CPUs
- Further speed-up by turning off less important Pythia options (e.g. MPI)
- Can add matrix elements for new models via MadGraph-Pythia interface

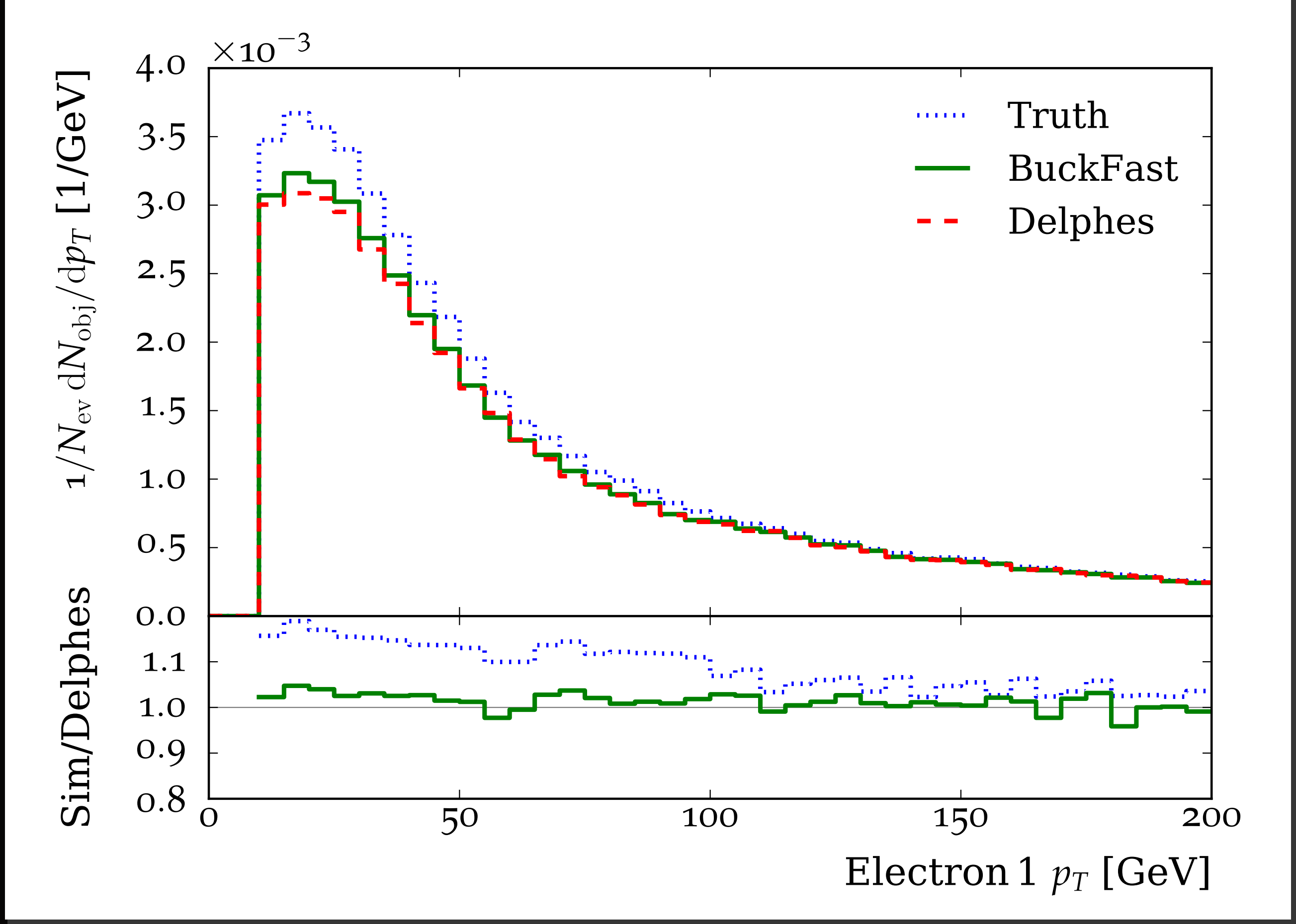
Calculating $s = \sigma \times \epsilon \times L$

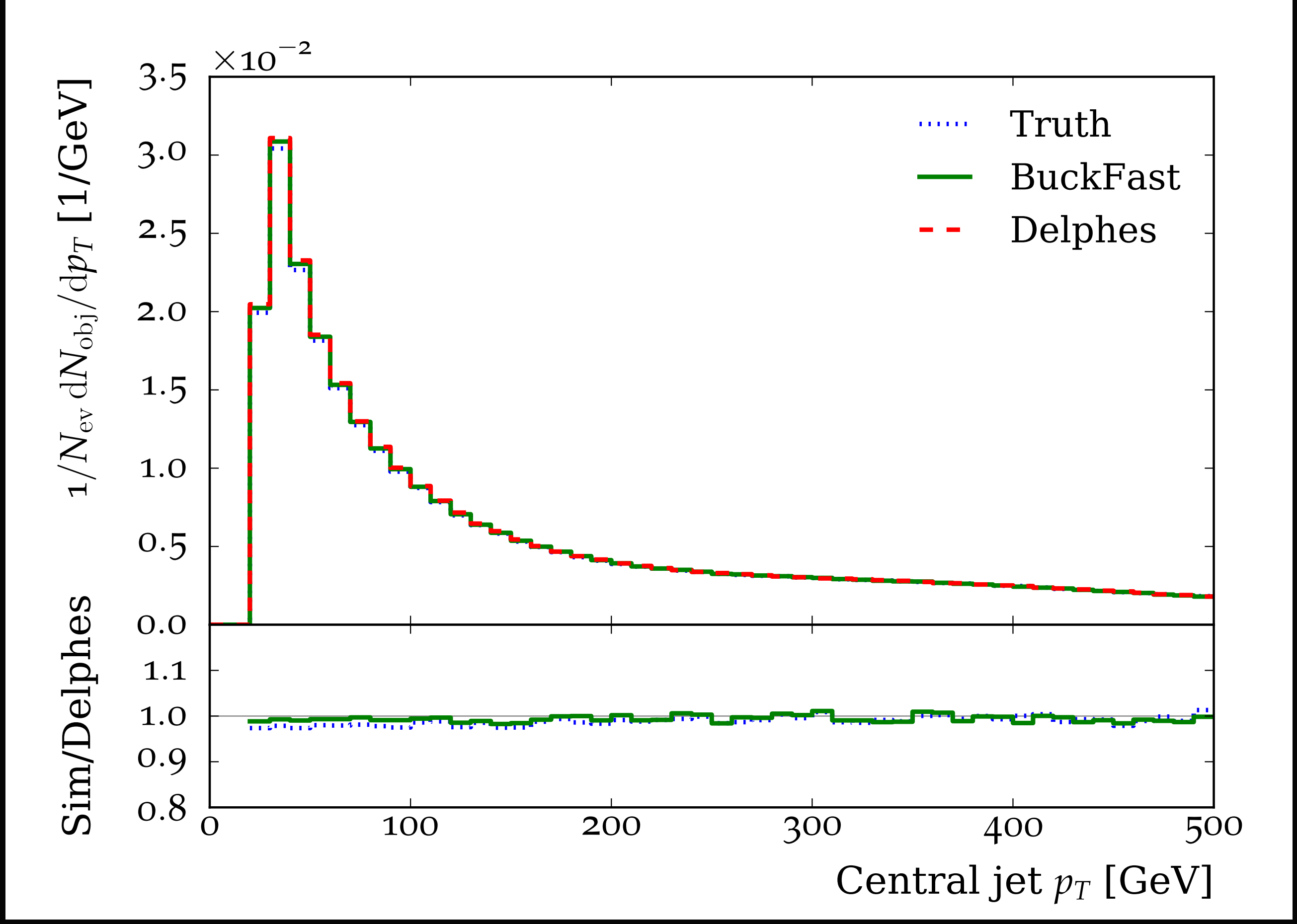


- Have interface to DELPHES…
- …but ROOT is not thread-safe
- BuckFast: Our own thread-safe detector sim based on four-vector smearing
- Run in the parallelised event loop
- Good agreement with DELPHES

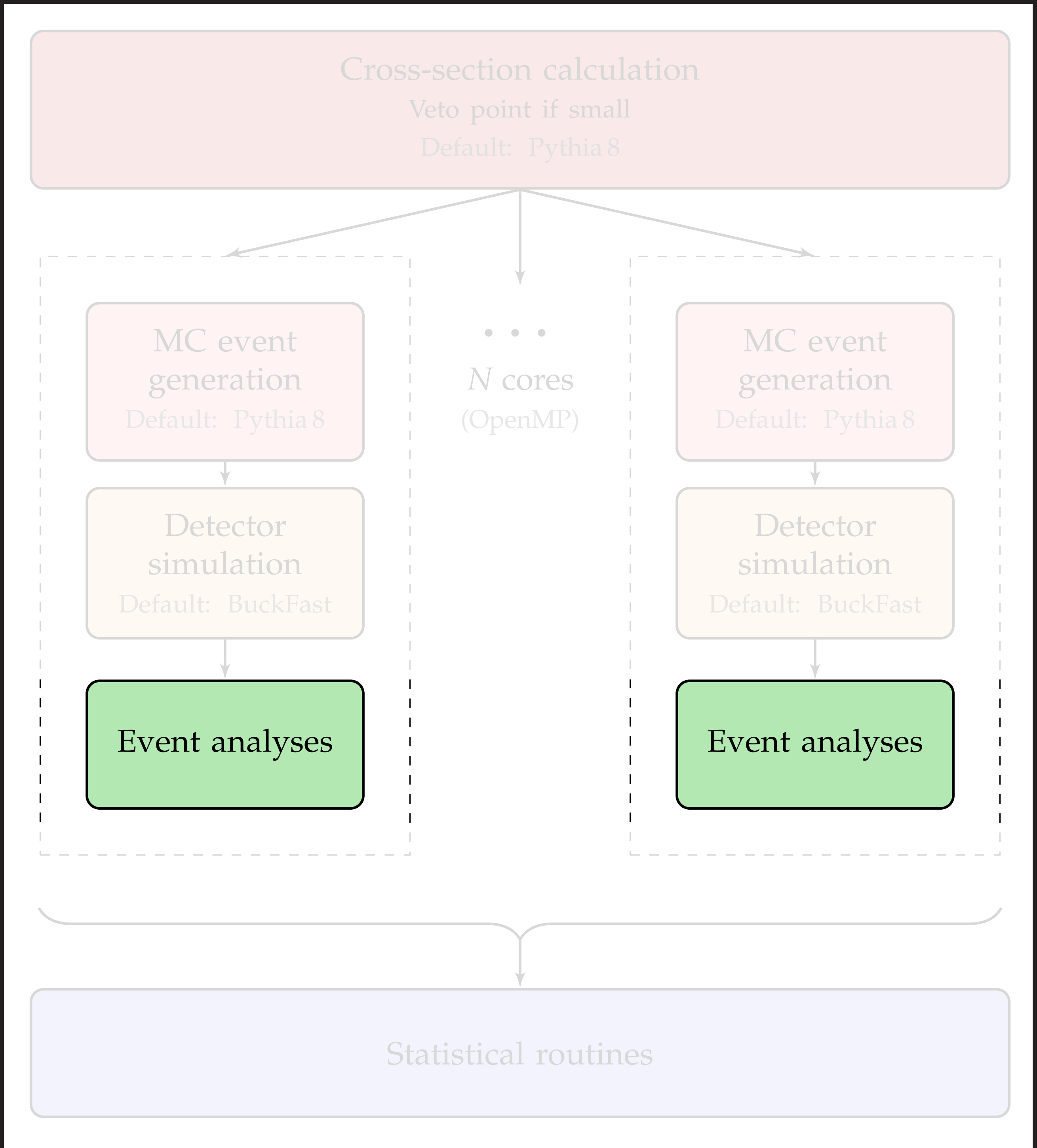
Calculating $s = \sigma \times \epsilon \times L$

- Have interface to DELPHES…



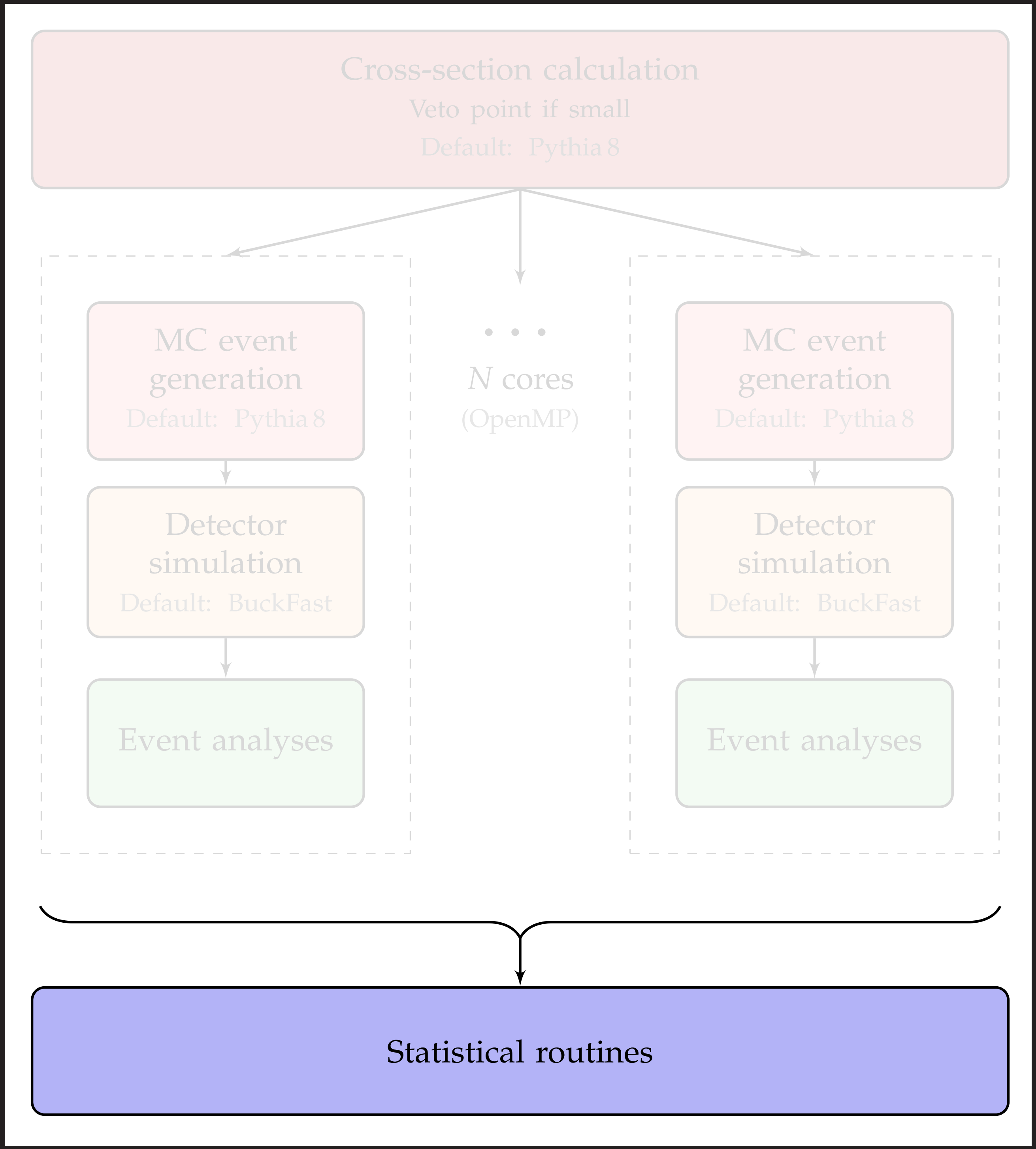


Calculating $s = \sigma \times \epsilon \times L$



- Analysis framework independent of MC generator and detector sim
- Uses public HepUtils classes
- Included analyses (8 TeV):
  - ATLAS SUSY searches:
    - 0 lep
    - 0-1-2 lep stop
    - b-jet plus MET
    - 2 lep EW
    - 3 lep EW
  - CMS multilepton SUSY search
  - CMS DM searches:
    - top pair plus MET
    - mono-b
    - mono-jet
- 13 TeV analyses in the pipeline

Calculating $s = \sigma \times \epsilon \times L$



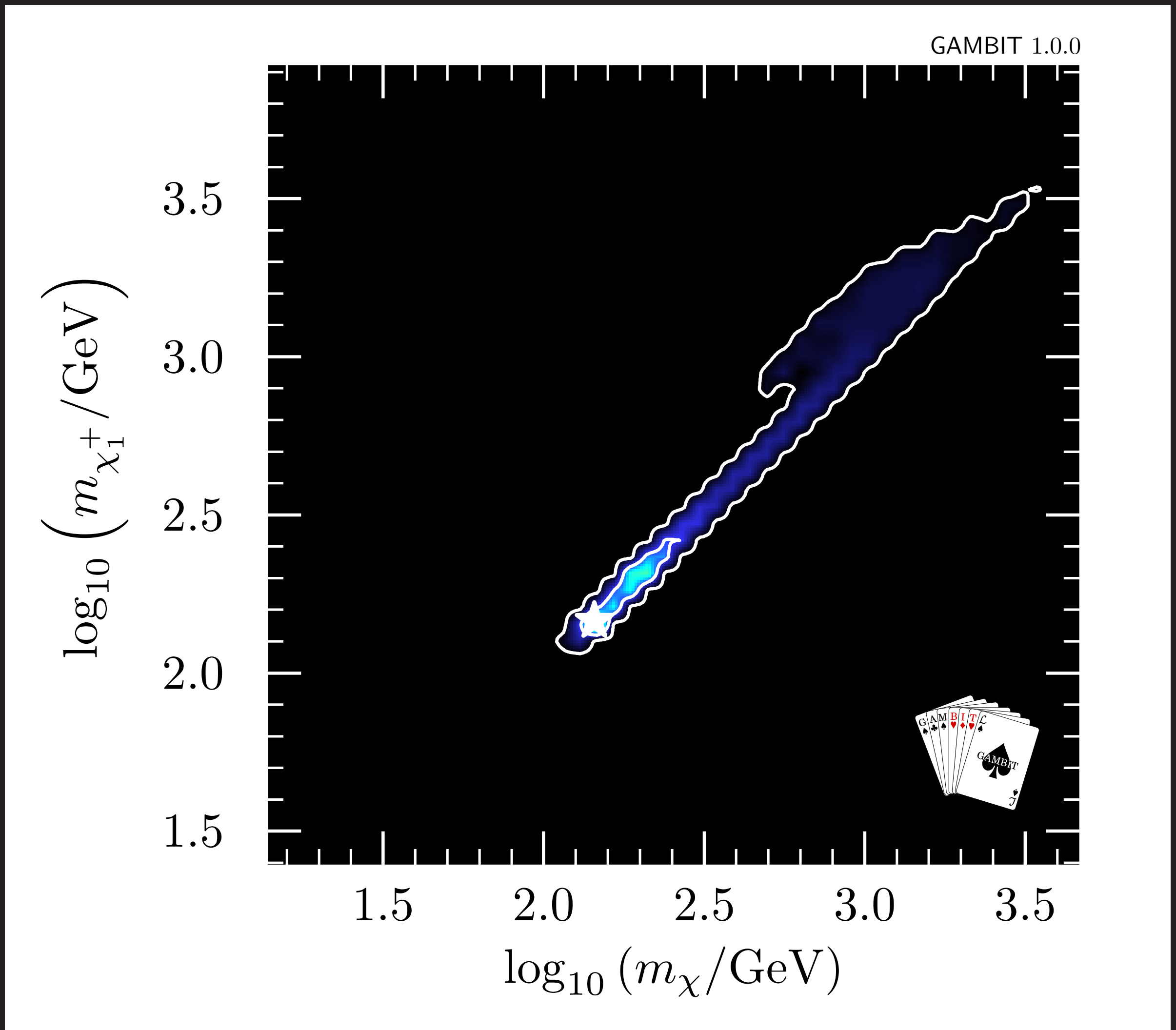
- Poisson distribution, marginalised over systematic uncertainty (via nulike)

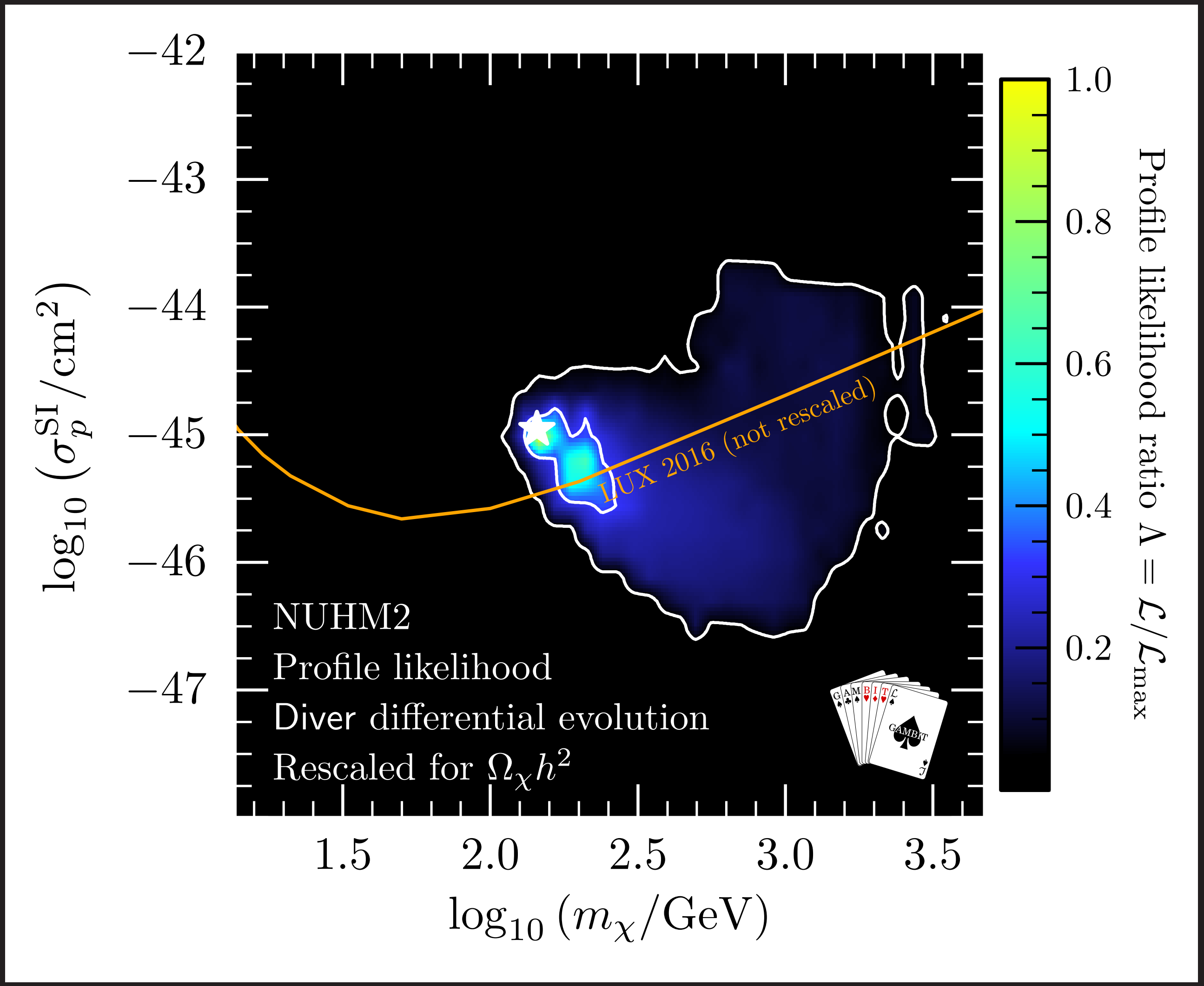
$$\mathcal{L}(n|s,b) = \int_0^\infty \frac{[\xi(s+b)]^n \, e^{-\xi(b+s)}}{n!} P(\xi)\mathrm{d}\xi$$

- For each analysis: Use likelihood from the signal region with the best *expected* sensitivity
- Combined likelihood on assumption that the different analyses are orthogonal (user's responsibility!)

4. Some preliminary GAMBIT results

GUT-scale MSSM: NUHM2 (preliminary)

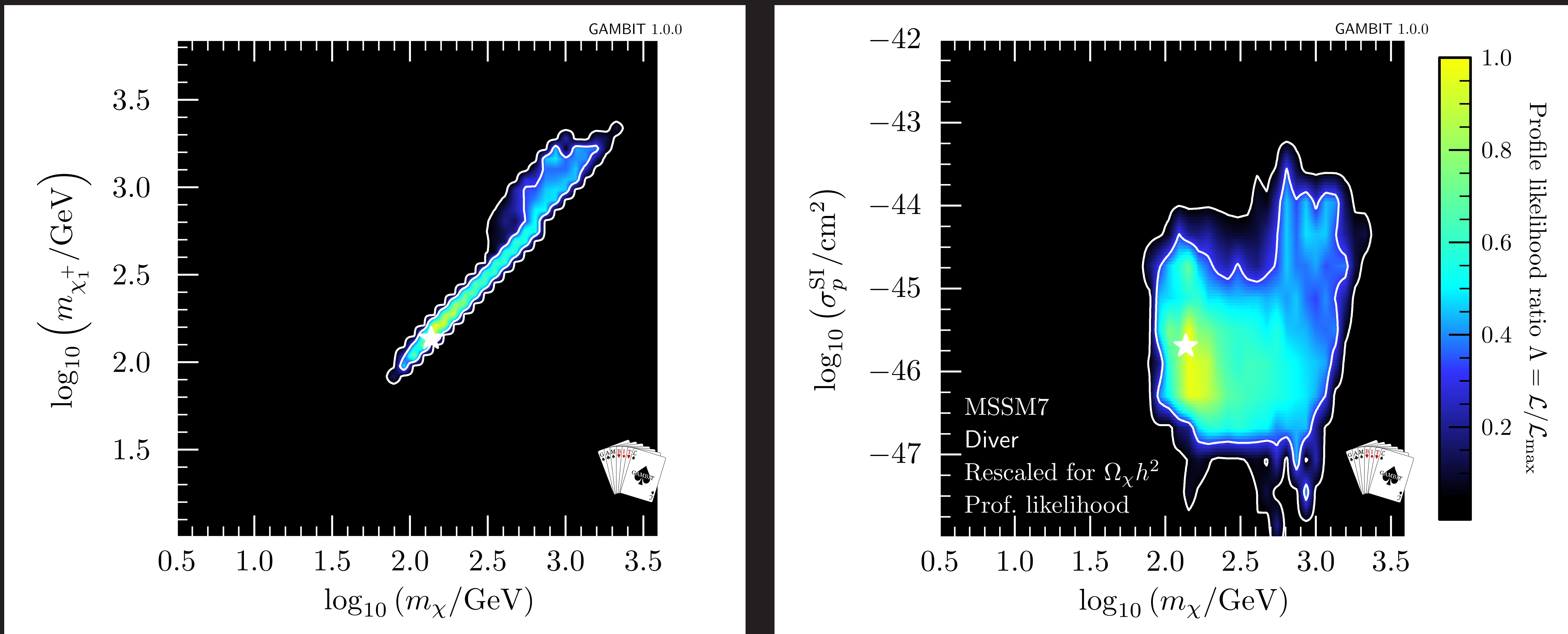




$$m_0, m_{\frac{1}{2}}, A_0, m_{H_u}, m_{H_d}, \tan\beta \quad (+5 \text{ nuisance par.})$$

- Relic density as upper bound
- Mostly chargino co-annihilation and A/H funnel

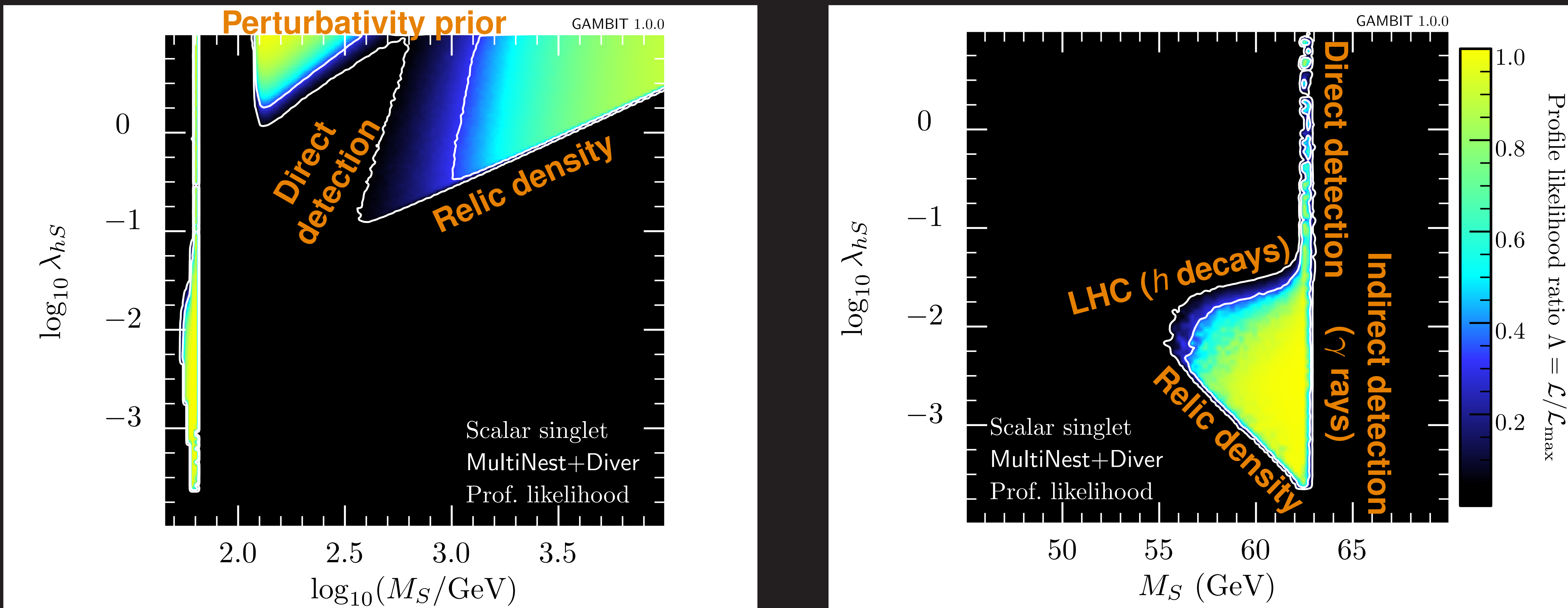
Weak-scale MSSM: MSSM-7 (preliminary)



$$m_{\tilde{f}}, M_2, A_u, A_d, m_{H_u}, m_{H_d}, \tan\beta \quad (+5 \text{ nuisance par.})$$

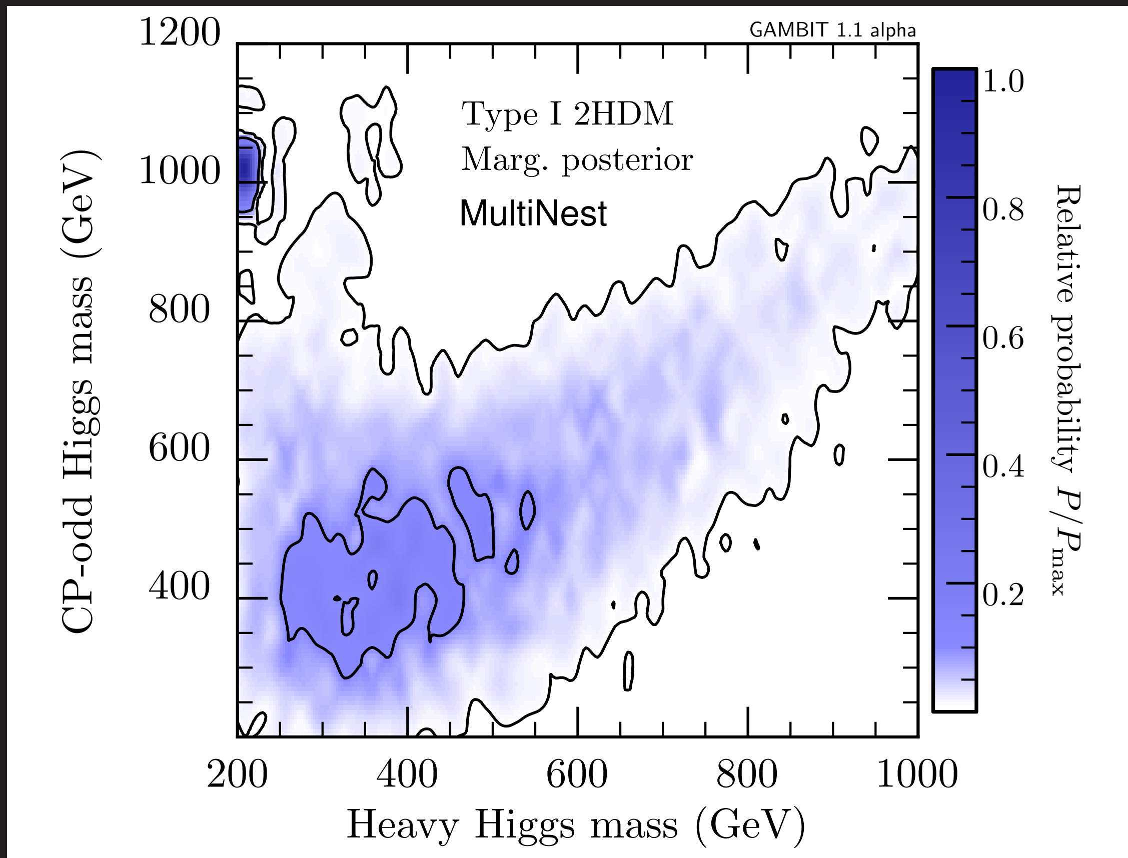
- Relic density as upper bound
- Mostly chargino co-annihilation and A/H funnel, some stau co-ann.

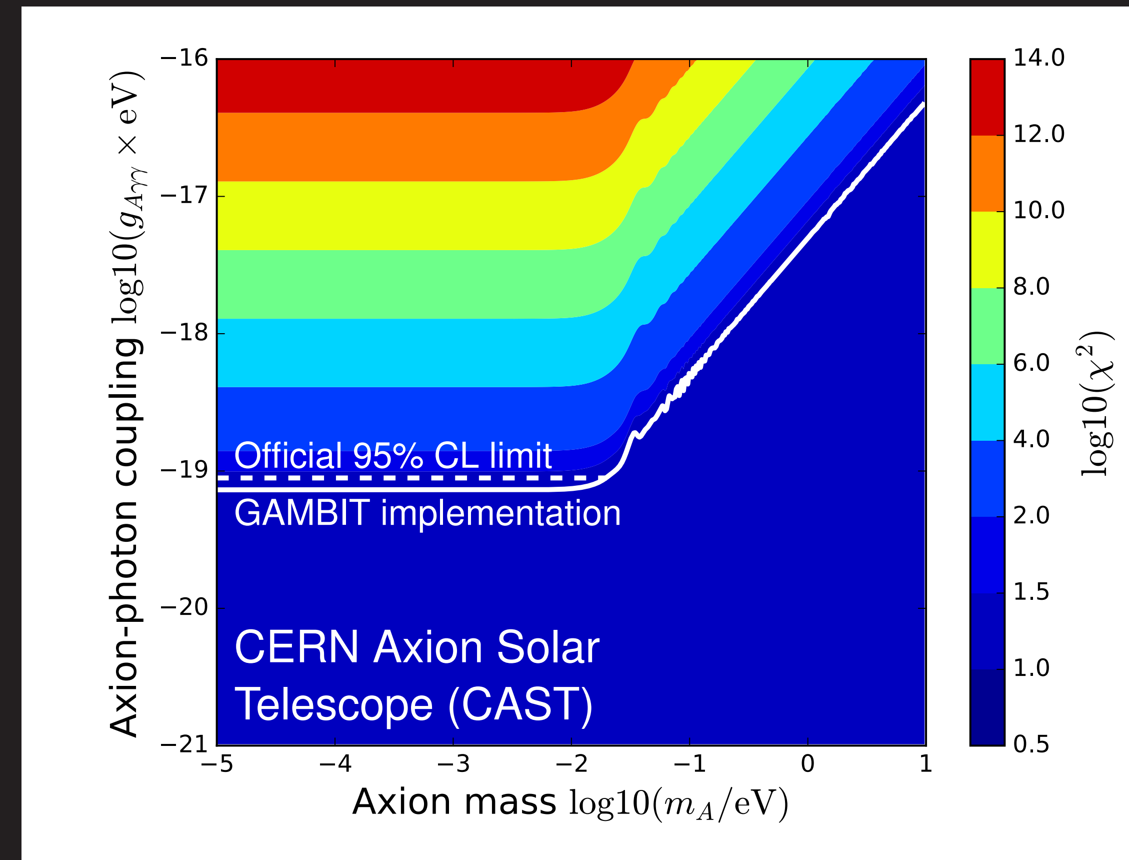
Scalar singlet DM (preliminary)

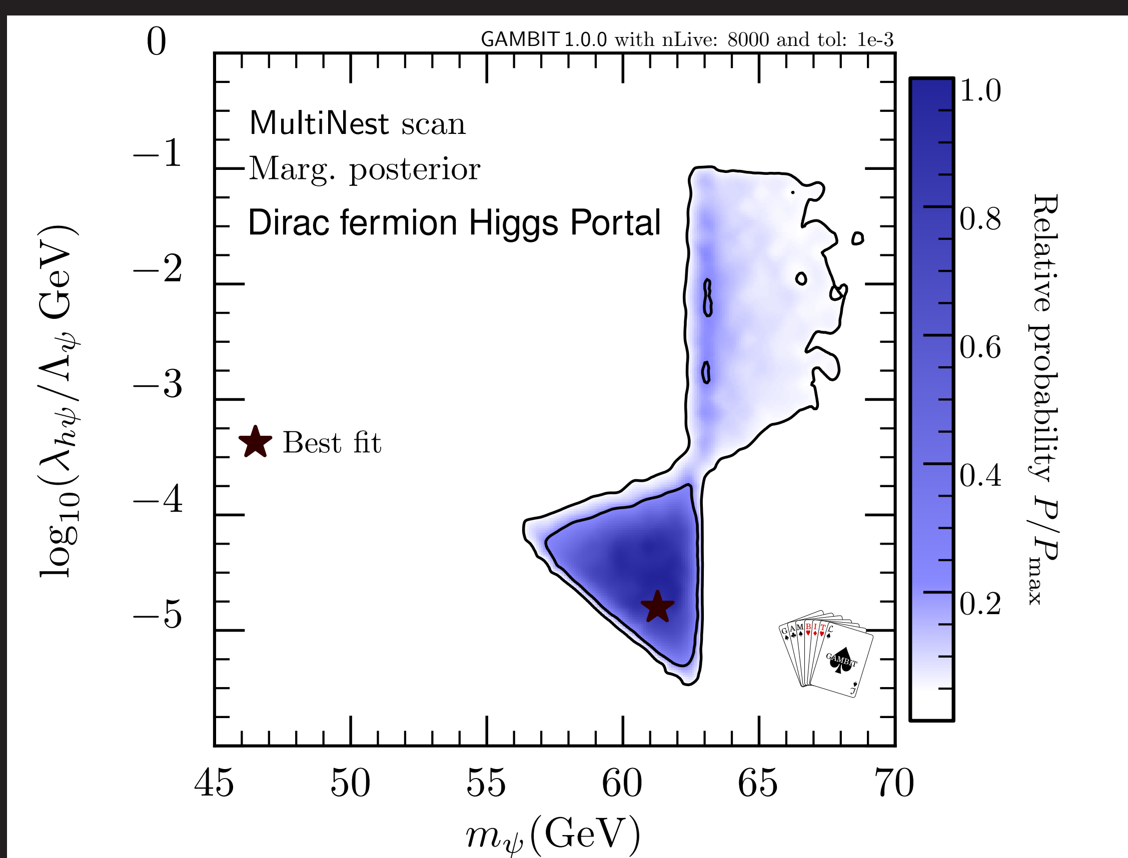


$$m_S, \lambda_{hS} \quad (+13 \text{ nuisance par.})$$

…and some ongoing work on other models…







[Your model here]

Summary and outlook

- GAMBIT is a general framework for BSM global fits
- Public code release Jan/Feb 2017
- 9 papers in preparation:

  - 3 physics studies with GAMBIT
  - the main GAMBIT paper
  - 5 module papers (incl. ColliderBit)
- ColliderBit is a fast and model-independent tool for LHC recasting
- *We would like to backend your code!*

Backup slides

Screenshot of config file for SUSY scans

```
# LHC likelihoods
- purpose:    LogLike
  capability: LHC_Combined_LogLike

- purpose:    LogLike
  capability: LHC_Higgs_LogLike

# LEP likelihoods
- purpose:    LogLike
  capability: LEP_Higgs_LogLike

- purpose:    LogLike
  capability: ALEPH_Selectron_LLike

- purpose:    LogLike
  capability: ALEPH_Smuon_LLike

- purpose:    LogLike
  capability: ALEPH_Stau_LLike

- purpose:    LogLike
  capability: L3_Selectron_LLike

- purpose:    LogLike
  capability: L3_Smuon_LLike

- purpose:    LogLike
  capability: L3_Stau_LLike

- purpose:    LogLike
  capability: L3_Neutralino_Leptonic_LLike

- purpose:    LogLike
  capability: L3_Chargino_Leptonic_LLike

- purpose:    LogLike
  capability: OPAL_Chargino_Hadronic_LLike

- purpose:    LogLike
  capability: OPAL_Chargino_SemiLeptonic_LLike

- purpose:    LogLike
  capability: OPAL_Chargino_Leptonic_LLike

- purpose:    LogLike
  capability: OPAL_Neutralino_Hadronic_LLike
```

```
# Dark matter likelihoods
- capability: lnL_oh2
  purpose:    LogLike

- capability: lnL_FermiLATdwarfs
  purpose:    LogLike

- capability: XENON100_2012_LogLikelihood
  purpose:    LogLike

- capability: LUX_2015_LogLikelihood
  purpose:    LogLike

- capability: LUX_2016_prelim_LogLikelihood
  purpose:    LogLike

- capability: PandaX_2016_LogLikelihood
  purpose:    LogLike

- capability: PICO_2L_LogLikelihood
  purpose:    LogLike

- capability: PICO_60_F_LogLikelihood
  purpose:    LogLike

- capability: SuperCDMS_2014_LogLikelihood
  purpose:    LogLike

- capability: SIMPLE_2014_LogLikelihood
  purpose:    LogLike

- capability: IC79_loglike
  purpose:    LogLike
```

```
# Flavour physics likelihoods
- purpose:    LogLike
  capability: b2ll_LL

- purpose:    LogLike
  capability: SL_LL

- purpose:    LogLike
  capability: b2sll_LL

- purpose:    LogLike
  capability: b2sgamma_LL

# Precision physics likelihoods
- capability: lnL_W_mass
  purpose:    LogLike

- capability:  lnL_gm2
  purpose:     LogLike
```

```
# SI nuclear nuisance parameter likelihood
- capability: lnL_SI_nuclear_parameters
  purpose:    LogLike

# DM Local Halo likelihoods
- capability: lnL_rho0
  purpose: LogLike

# SM nuisance parameter likelihoods
- capability: lnL_t_mass
  purpose:    LogLike

- capability: lnL_mbmb
  purpose:    LogLike

- capability: lnL_alpha_s
  purpose:    LogLike
```

Anders Kvellestad

Diagnostics on connected backends

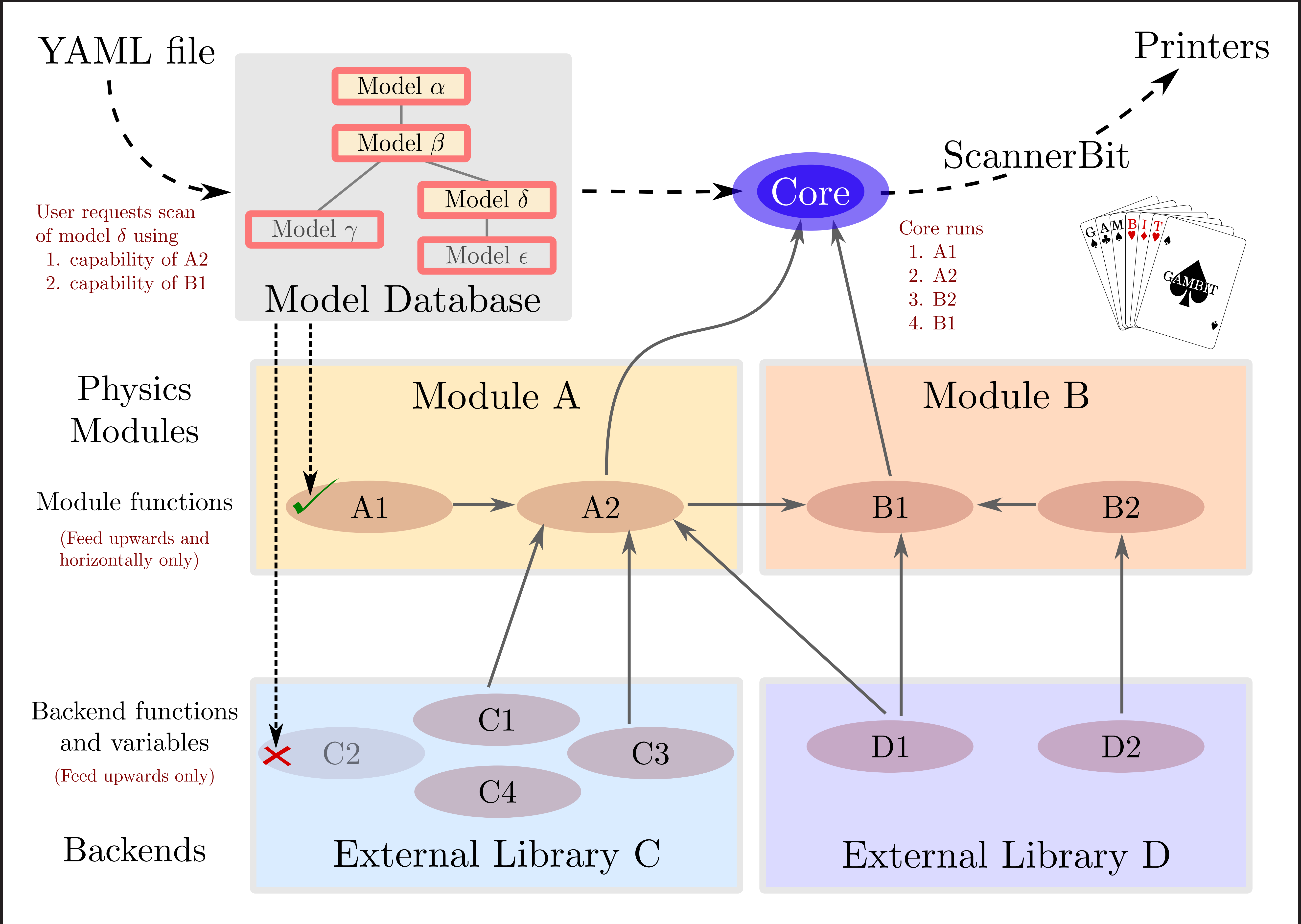
BACKENDS	VERSION	PATH TO LIB	STATUS	#FUNC	#TYPES	#CTORS
DDCalc0	0.0	Backends/installed/DDCalc/0.0/libDDCalc0.so	OK	62	0	0
DarkSUSY	5.1.1	Backends/installed/DarkSUSY/5.1.1/lib/libdarksusy.so	OK	68	0	0
FastSim	1.0	Backends/installed/fastsim/1.0/libfastsim.so	absent/broken	1	0	0
FeynHiggs	2.11	Backends/installed/FeynHiggs/2.11.2/lib/libFH.so	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)

Other features

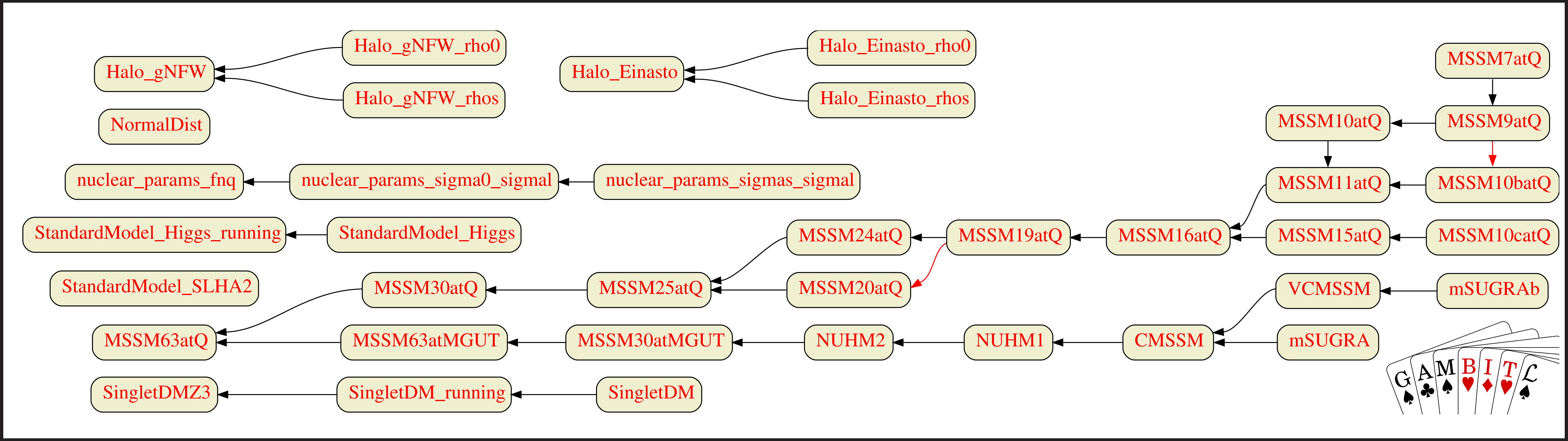
- Easy to add new observables and likelihoods (backup slides)
- **User interface**: yaml file (backup slides)
- **Scanners**: Nested sampling, differential evolution, MCMC, genetic algorithm, t-walk. . .
- Mixed-mode **MPI + openMP** parallelisation, mostly automated → scales to 10k+ cores
- diskless 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, . . .)
- more more more. . .

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



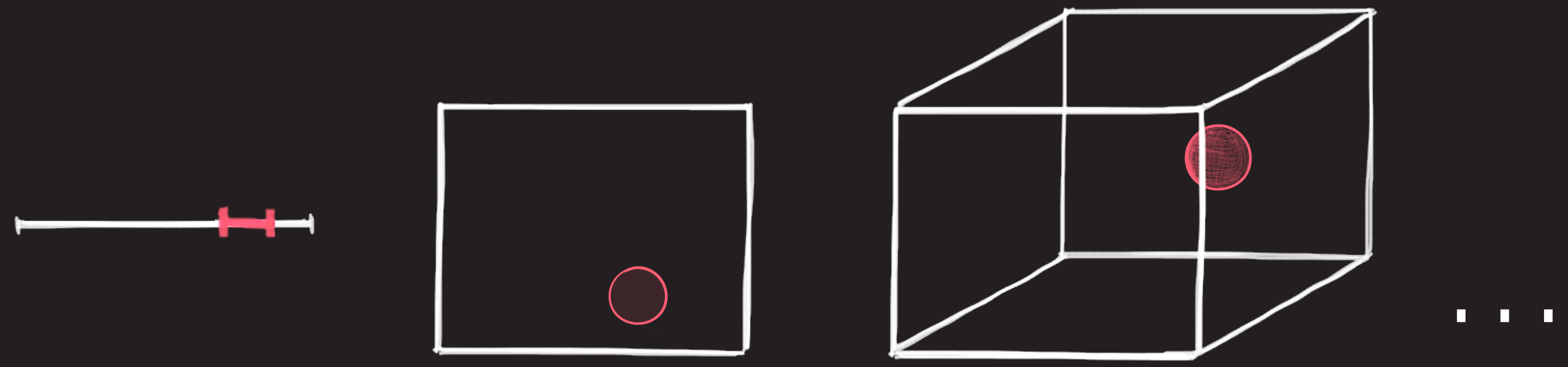
Anders Kvellestad

Hierarchical model database



Finding interesting parameter regions gets harder with increasing number of dimensions…

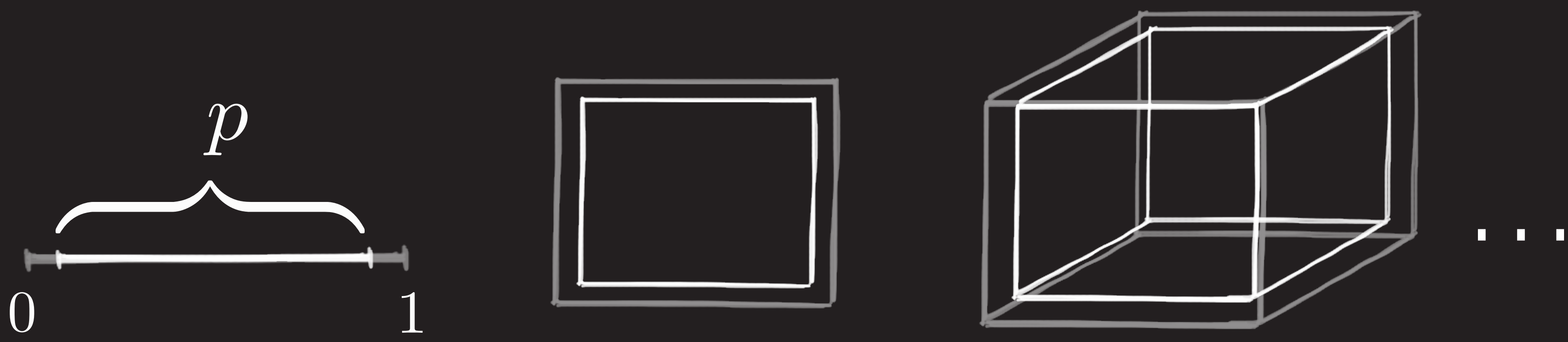
$$\lim_{D\to\infty} \frac{V_{\text{interesting}}}{V_{\text{total}}} = 0$$



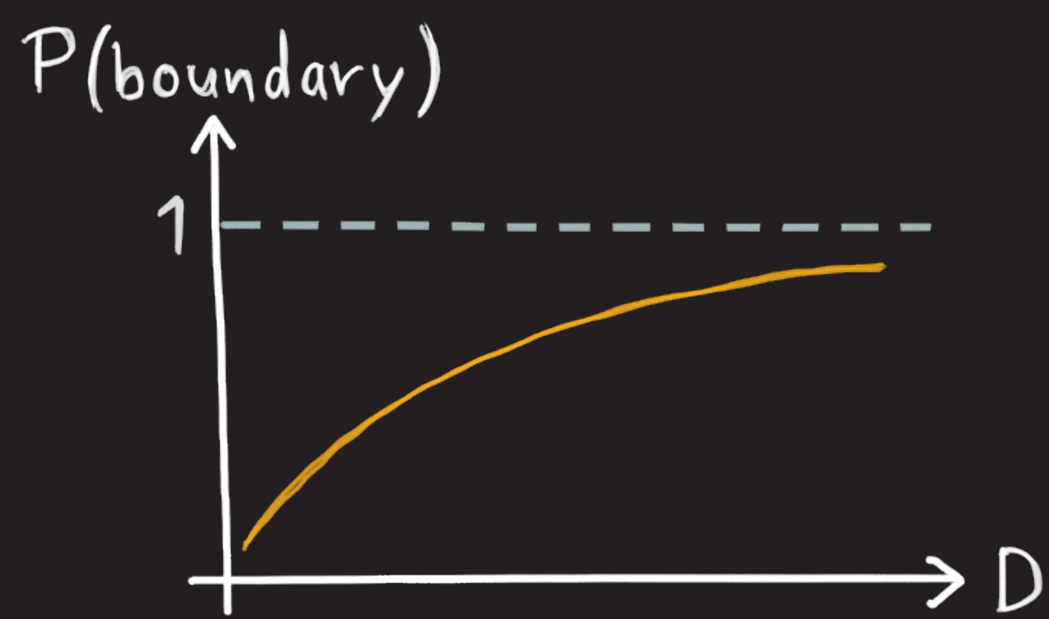
…so simply picking points «at random» will be highly inefficient…

…and it will mainly explore the boundary of the parameter space!

$$\vec{x} = (x_1, x_2, \ldots, x_D) \qquad x_i \sim U(0,1)$$



$$P(\text{boundary}) = 1 - P(\text{not boundary}) = 1 - p^D$$



Marginalization vs profiling (maximising)

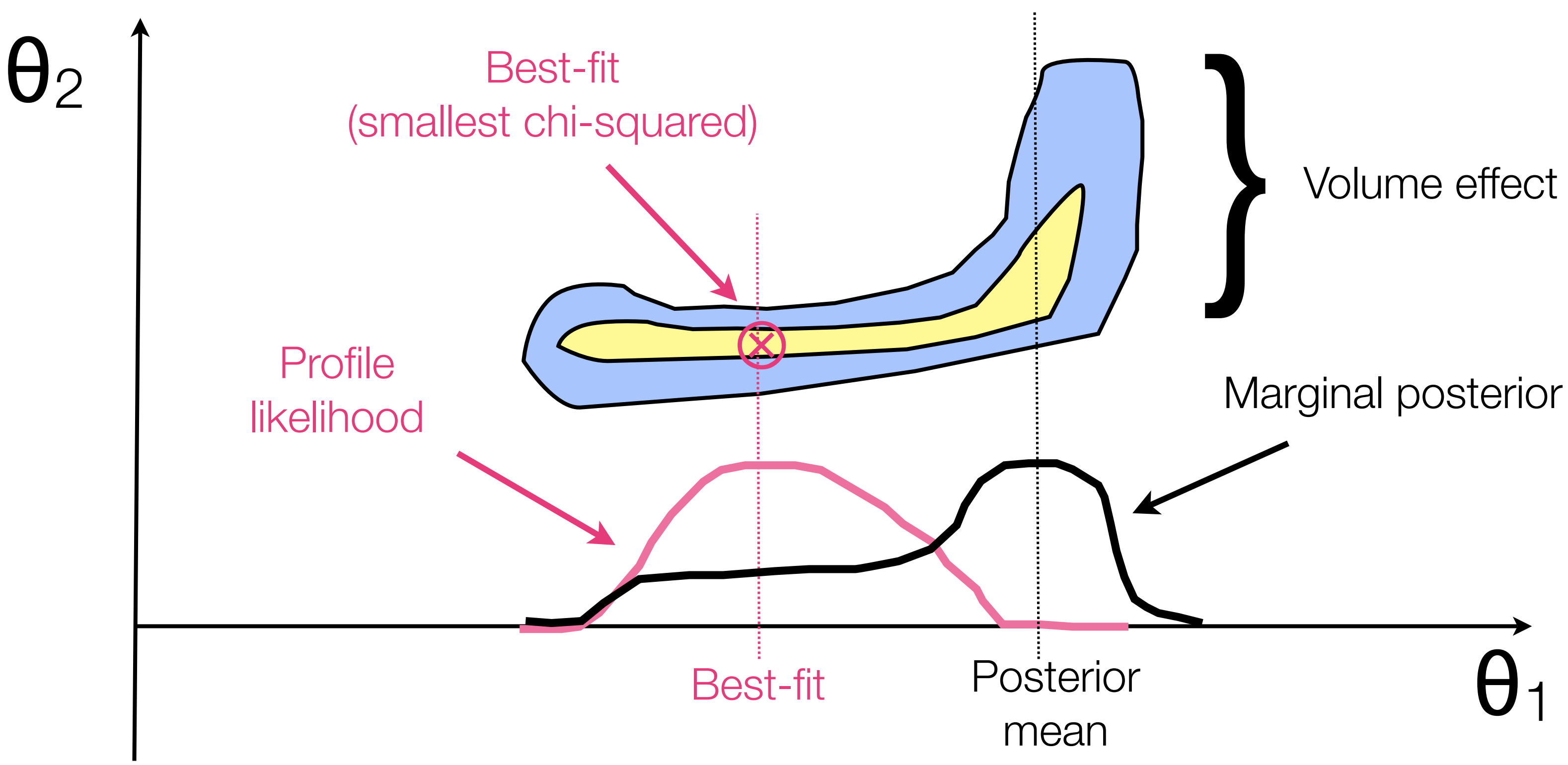
Imperial College London

Marginal posterior:

$$P(\theta_1|D) = \int L(\theta_1,\theta_2)p(\theta_1,\theta_2)d\theta_2$$

Profile likelihood:

$$L(\theta_1) = max_{\theta_2} L(\theta_1,\theta_2)$$



(2D plot depicts likelihood contours - prior assumed flat over wide range)

from Roberto Trotta