GAMBIT: What is it and where is it going?

Anders Kvellestad, University of Oslo

on behalf of the GAMBIT Collaboration

Spåtind 2018 — Nordic Conference on Particle Physics — 04/01/18

UiO : University of Oslo

GAMBIT: What is it and where is it going?

→ See Tomás Gonzalo's talk this afternoon

Anders Kvellestad, University of Oslo

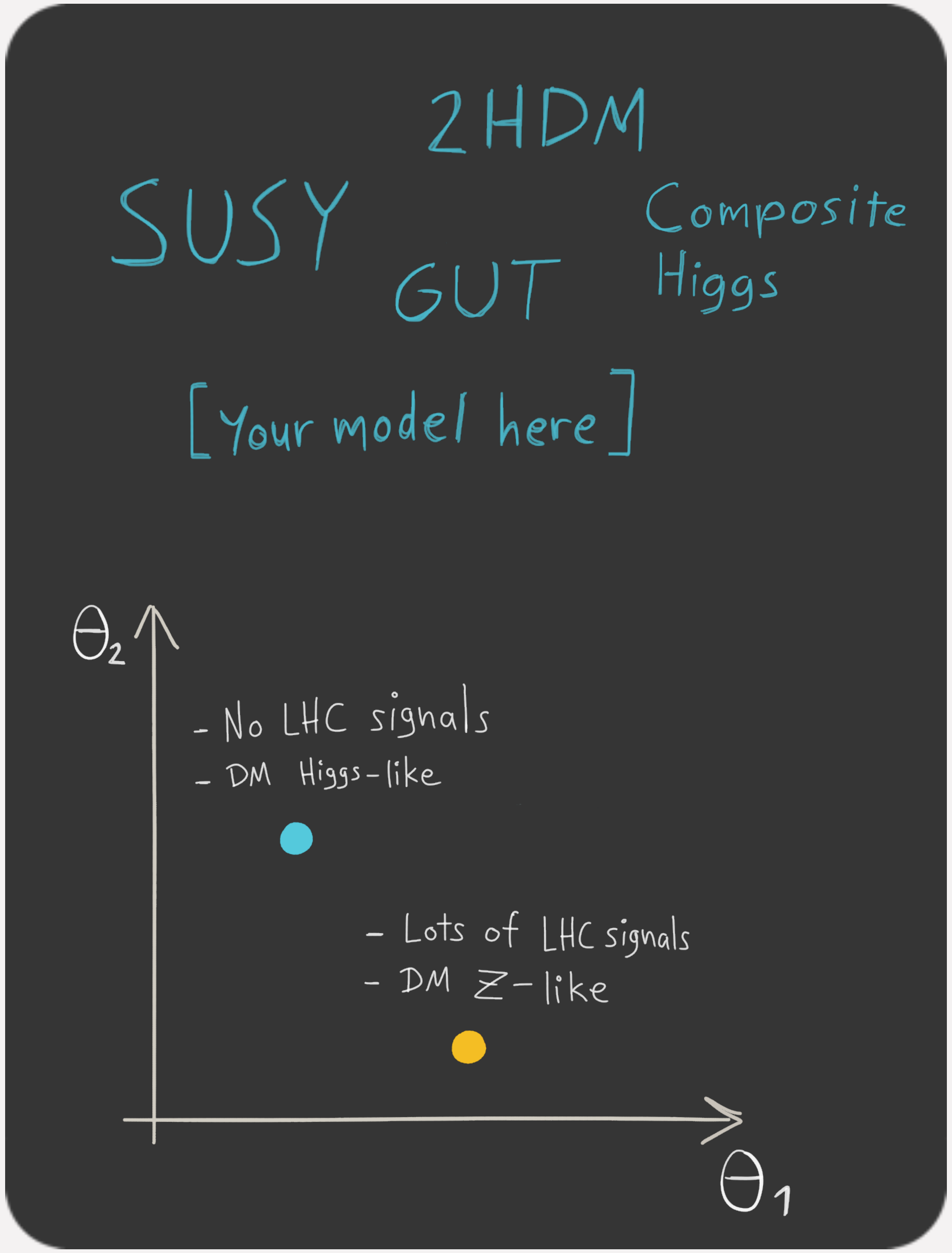
on behalf of the GAMBIT Collaboration

Spåtind 2018 — Nordic Conference on Particle Physics — 04/01/18

UiO : **University of Oslo**

Comparing BSM theories to data

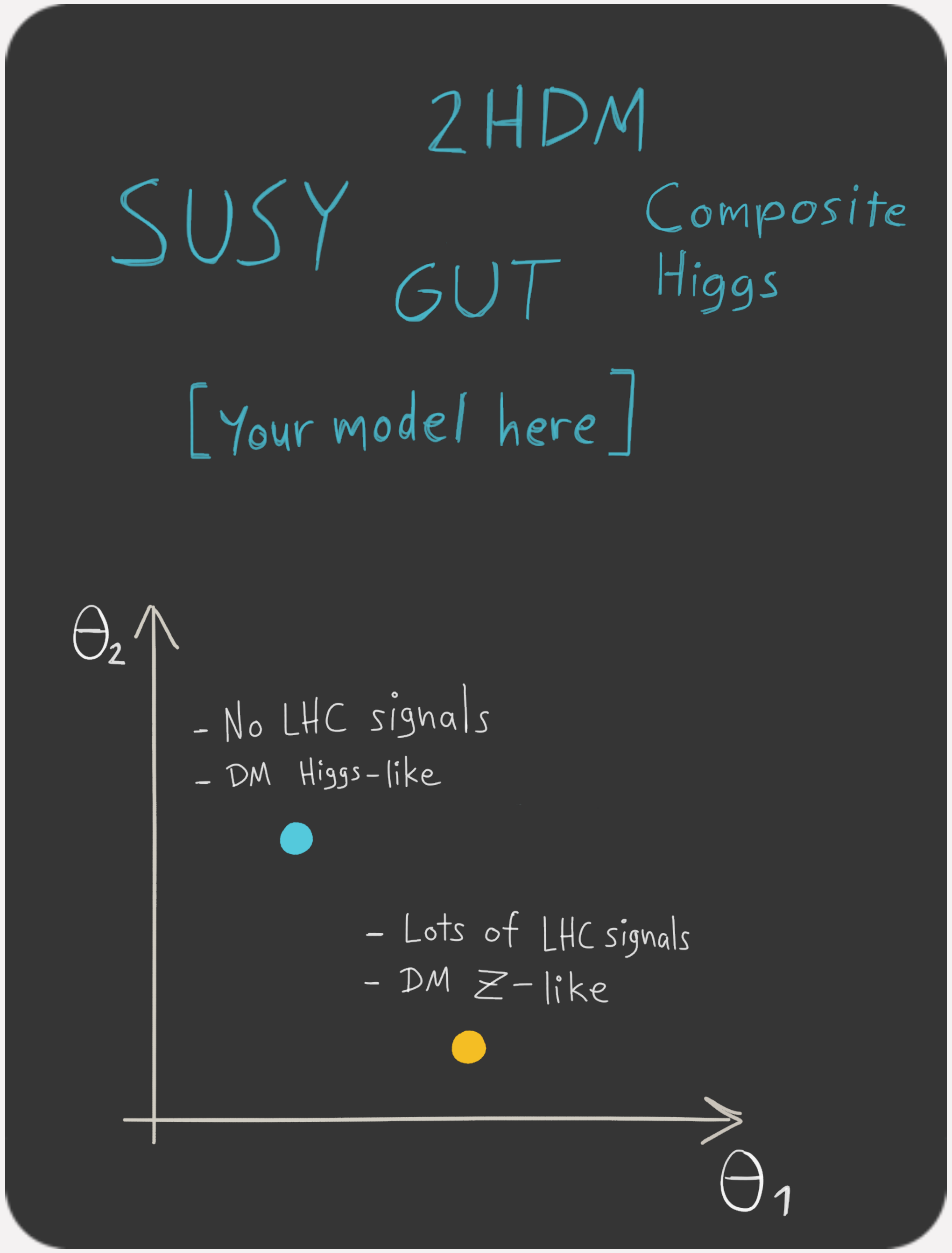
- Lots of theories for BSM physics
- For each theory, a parameter space of varying phenomenology
- Many different experiments can constrain each theory



Comparing BSM theories to data

- Lots of theories for BSM physics
- For each theory, a parameter space of varying phenomenology
- Many different experiments can constrain each theory

*Consistently compare theories against **all** available data: **global fits***



Global fits

- Calculate **combined likelihood function** including observables from collider physics, dark matter, flavor physics, +++

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

- Use **sophisticated scanning techniques** to explore likelihood function across the parameter space of the theory

- Test **parameter regions** in a statistically sensible way — not just single points *(parameter estimation)*

- Test **different theories the same way** *(model comparison)*

Need a tool designed to work with different theories, scanners, observables and theory calculators

GAMBIT

The Global And Modular BSM Inference Tool

- A new framework for BSM **global fits**
- Fully **open source**
- **Modular design:** easily extended with
  — new models
  — new likelihoods
  — new theory calculators
  — new scanning algorithms
- Use external codes (**backends**) as **runtime plugins**
  — Currently supported:
  C, C++, Fortran, Mathematica
  — Coming soon: Python
- **Two-level parallellization** with MPI and OpenMP
- **Hierarchical** model database
- **Flexible output streams** (ASCII, HDF5, …)
- Many **scanners** and **backends** already included

gambit.hepforge.org

GAMBIT: The Global And Modular BSM Inference Tool

gambit.hepforge.org

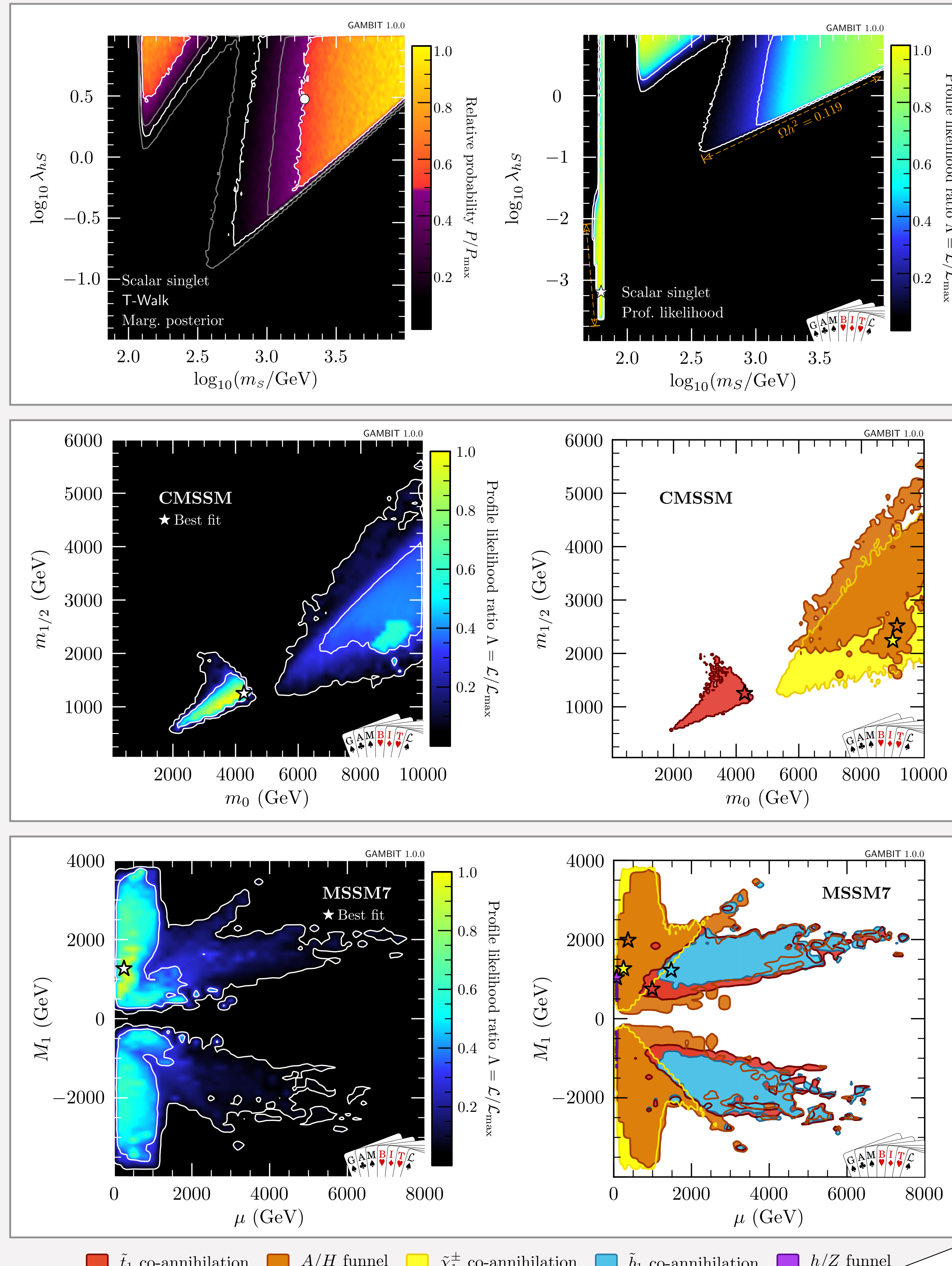
- Fast definition of new datasets and theoretical models
- Plug and play scanning, physics and likelihood packages
- Extensive model database – not just SUSY
- Extensive observable/data libraries
- Many statistical and scanning options (Bayesian & frequentist)
- *Fast* LHC likelihood calculator
- Massively 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

First results

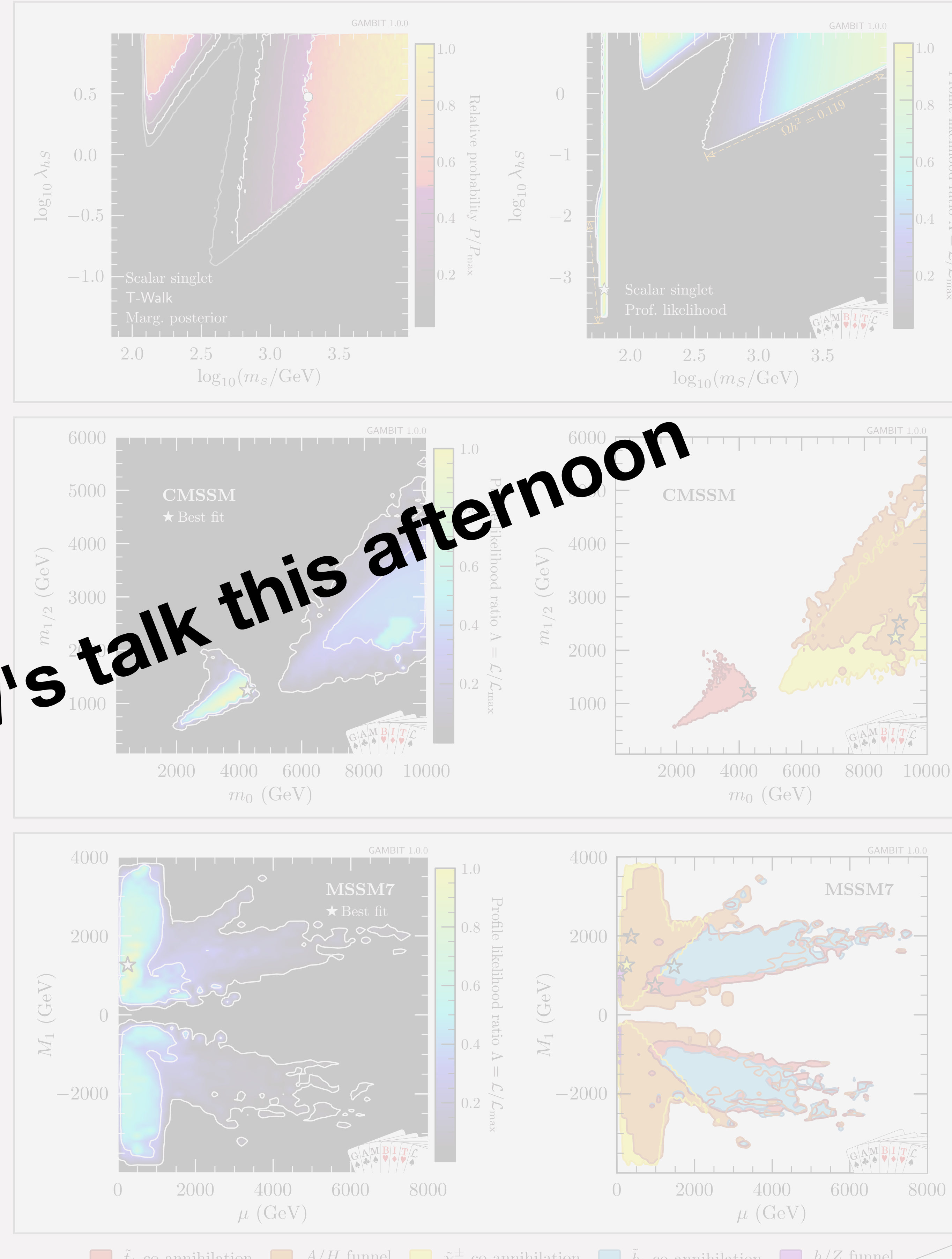
- **Scalar singlet dark matter**
  arXiv:1705.07931
- **GUT-scale MSSM**
  CMSSM, NUHM1, NUHM2
  arXiv:1705.07935
- **Weak-scale MSSM7**
  arXiv:1705.07917



First results

- **Scalar singlet dark matter**
  arXiv:1705.07931
- **GUT-scale MSSM**
  CMSSM, NUHM1, NUHM2
  arXiv:1705.07935
- **Weak-scale MSSM7**
  arXiv:1705.07917

See Are Raklev's talk this afternoon



What's in the box?

Core

- Models — arXiv:1705.07908

Physics modules

- ColliderBit: *fast* LHC sim, Higgs searches, LEP SUSY limits — arXiv:1705.07919
- DarkBit: relic density, gamma ray signal yields, ID/DD likelihoods — arXiv:1705.07920
- FlavBit: wide range of flavour observables & likelihoods — arXiv:1705.07933
- SpecBit: spectrum objects, RGE running
- DecayBit: decay widths
- PrecisionBit: precision BSM tests

(SpecBit, DecayBit, PrecisionBit) — arXiv:1705.07936

Statistics and sampling

- ScannerBit: stats & sampling (Diver, MultiNest, T-Walk, ++) — arXiv:1705.07959

Backends (external tools)

[arXiv:1705.07959]

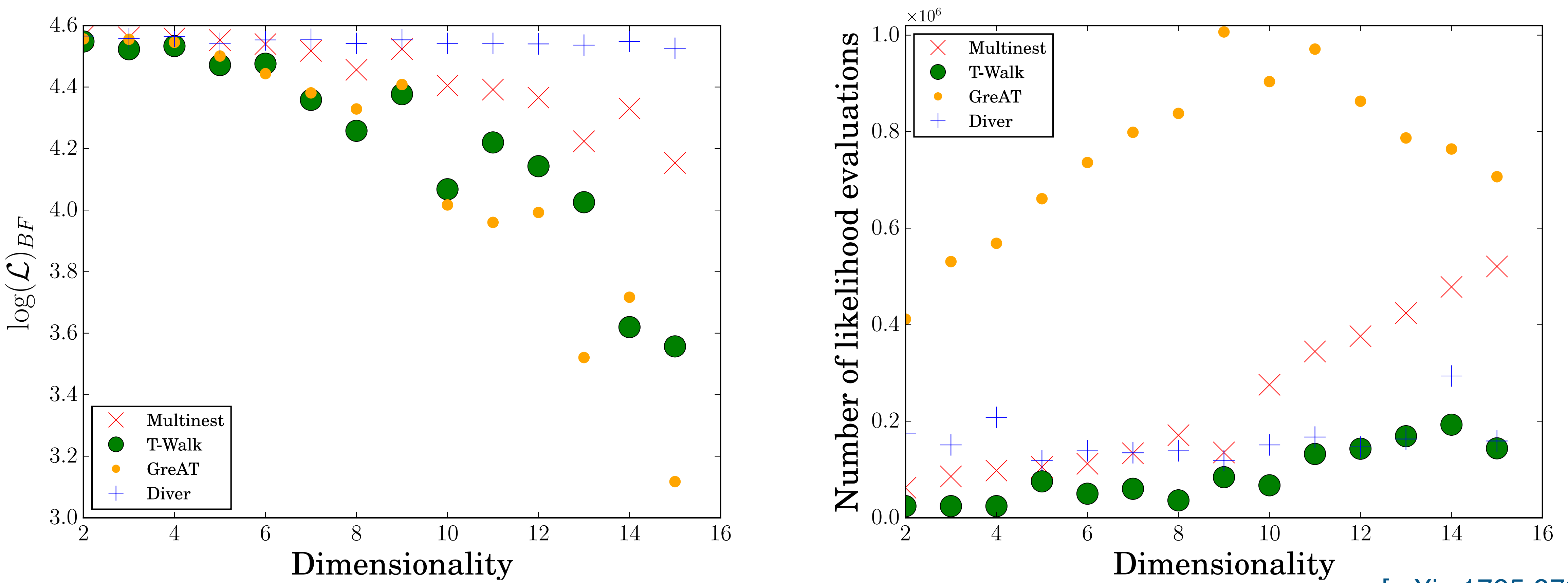
What's in the box?

Core

- Models arXiv:1705.07908

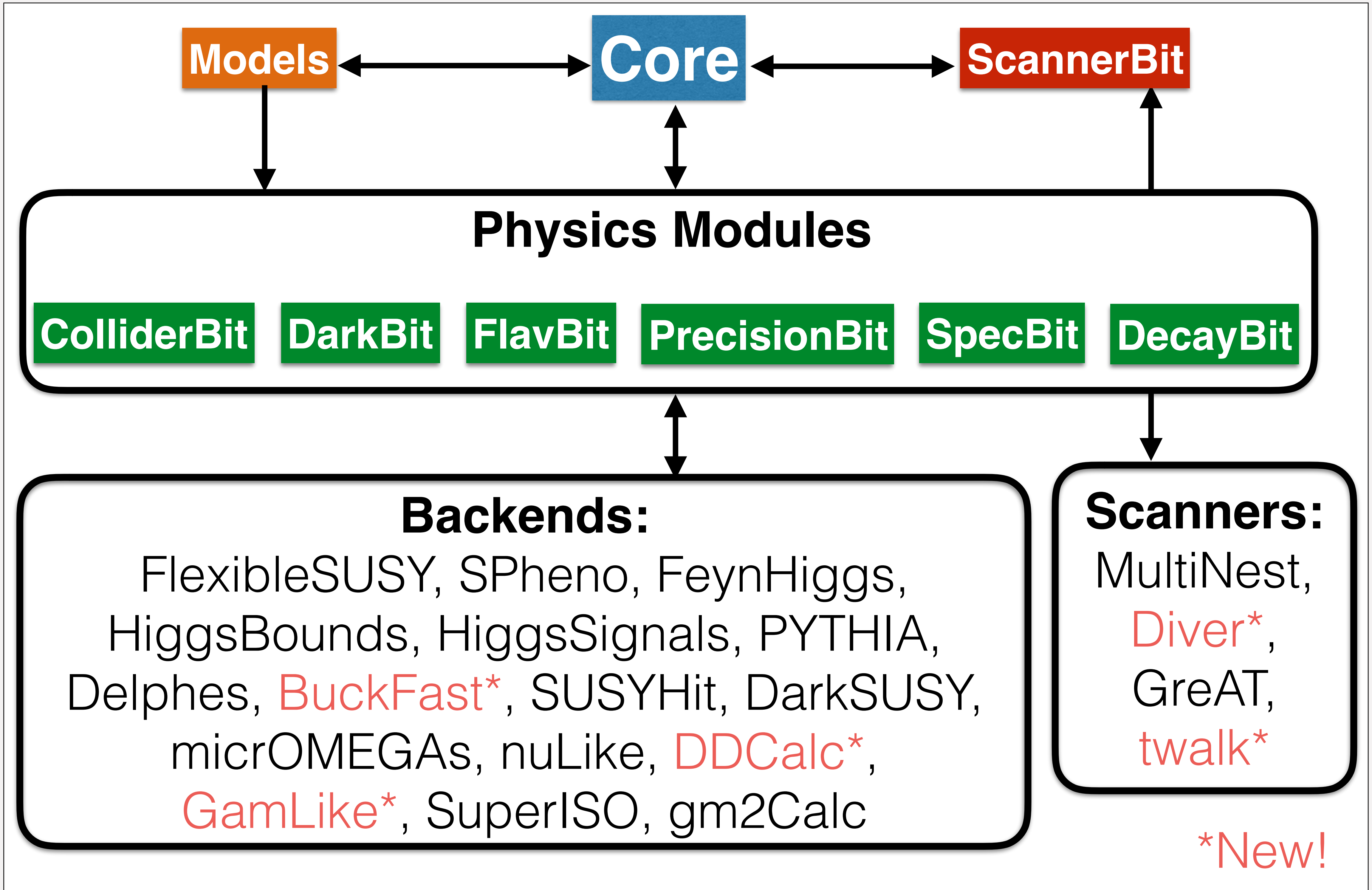
Physics modules

- ColliderBit: *fast* LHC sim, Higgs searches, LEP SUSY limits arXiv:1705.07919

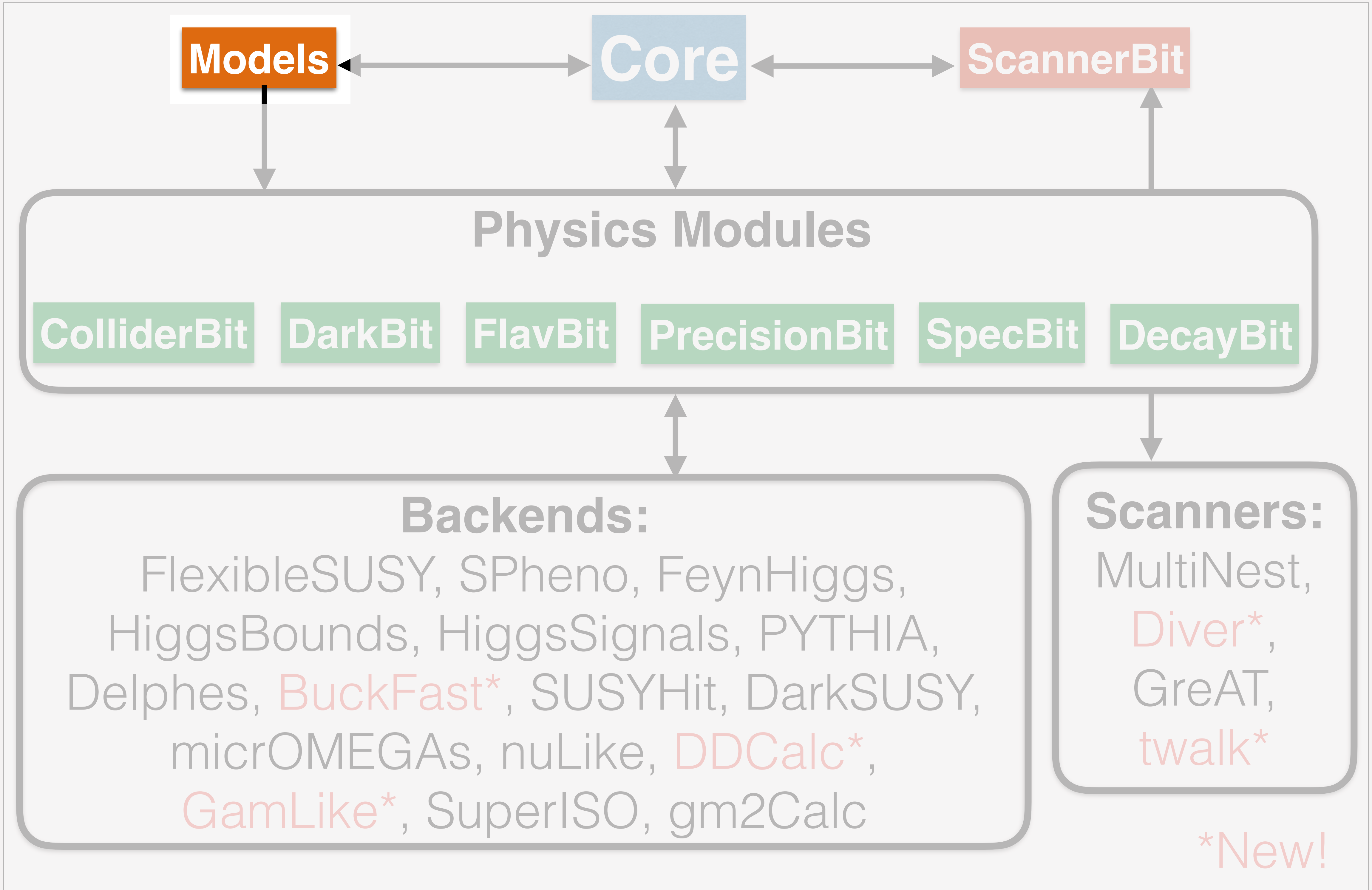


[arXiv:1705.07959]

Code structure

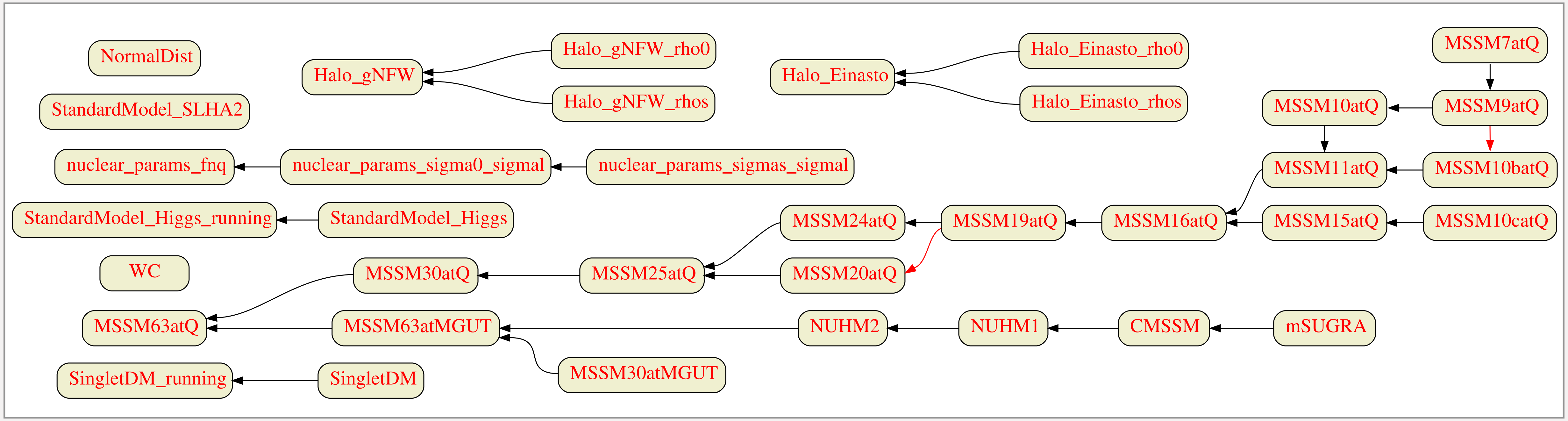


Code structure



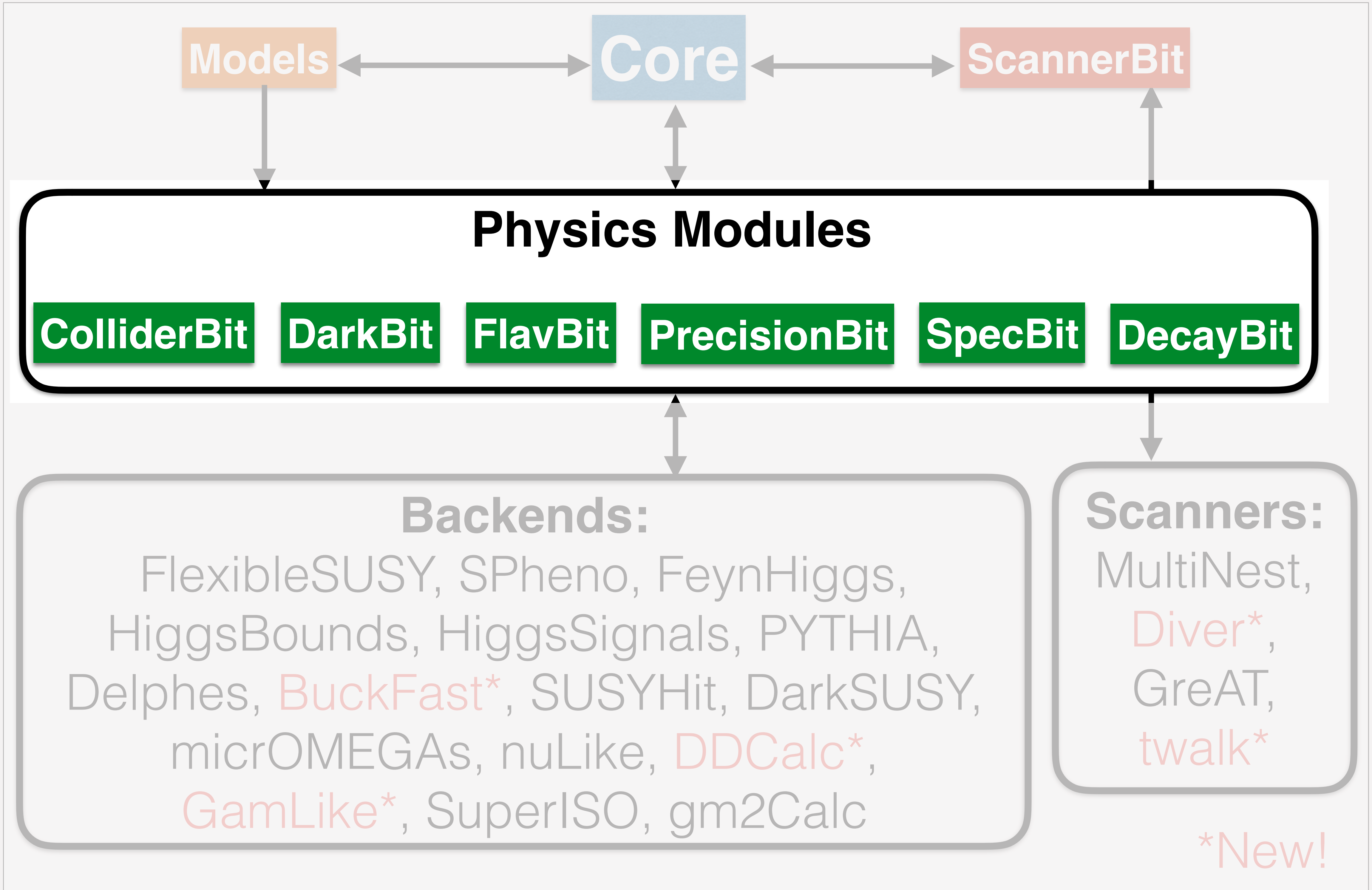
Hierarchical model database

- A **model** is a collection of named parameters
- Models can be **related** (e.g. MSSM9 is a parent of MSSM7)
- Points in child model **automatically translated** to ancestor models
- Ensures **maximum reuse** of calculations and minimizes risk of mistakes and inconsistencies



GAMBIT

Code structure



Physics modules

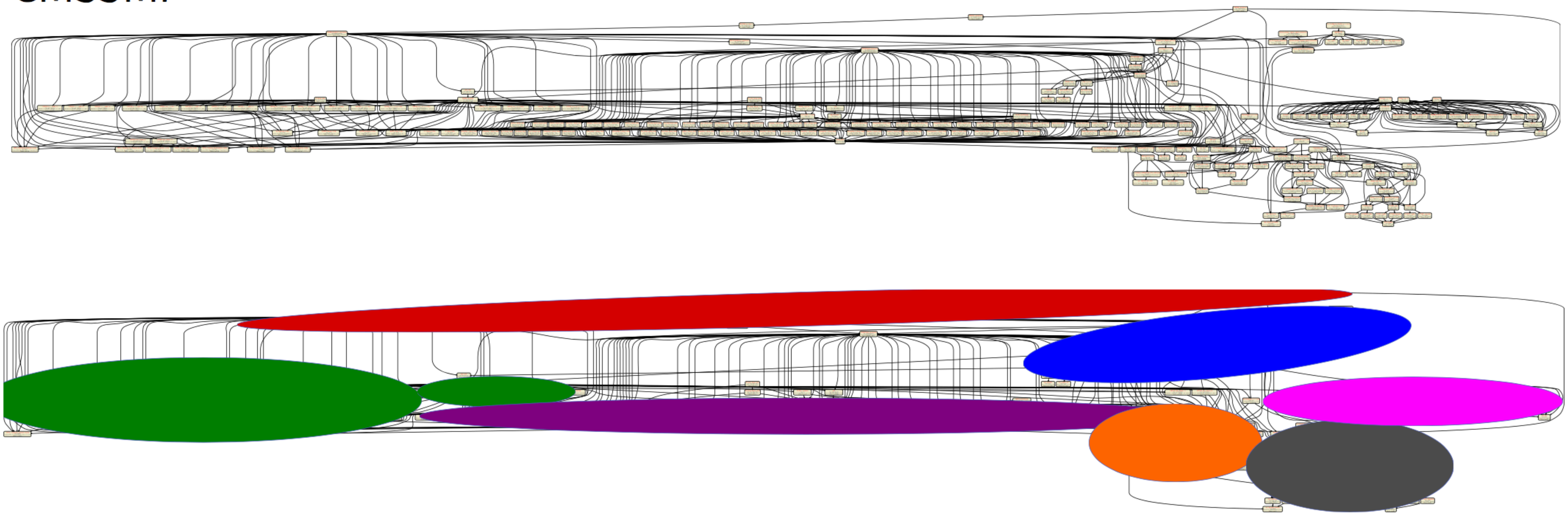
- Basic building blocks: **module functions**
- A physics module: **a collection of module functions** related to the same physics topic
- Each module function has a single **capability** (what it calculates)
- A module function can have **dependencies** on the results of other module functions
- A module function can declare which **models** it can work with
- GAMBIT determines which module functions should be run in which order for a given scan (**dependency resolution**)

```
void function_name(double &result)
{
  ...
  result = ... // something useful
}
```

```
// Observable: BR(B -> tau nu)
#define CAPABILITY Btaunu
START_CAPABILITY
  #define FUNCTION SI_Btaunu
  START_FUNCTION(double)
  DEPENDENCY(SuperIso_modelinfo, parameters)
  BACKEND_REQ(Btaunu, (libsuperiso), double, (const parameters*))
  BACKEND_OPTION( (SuperIso, 3.6), (libsuperiso) )
  #undef FUNCTION
#undef CAPABILITY
```

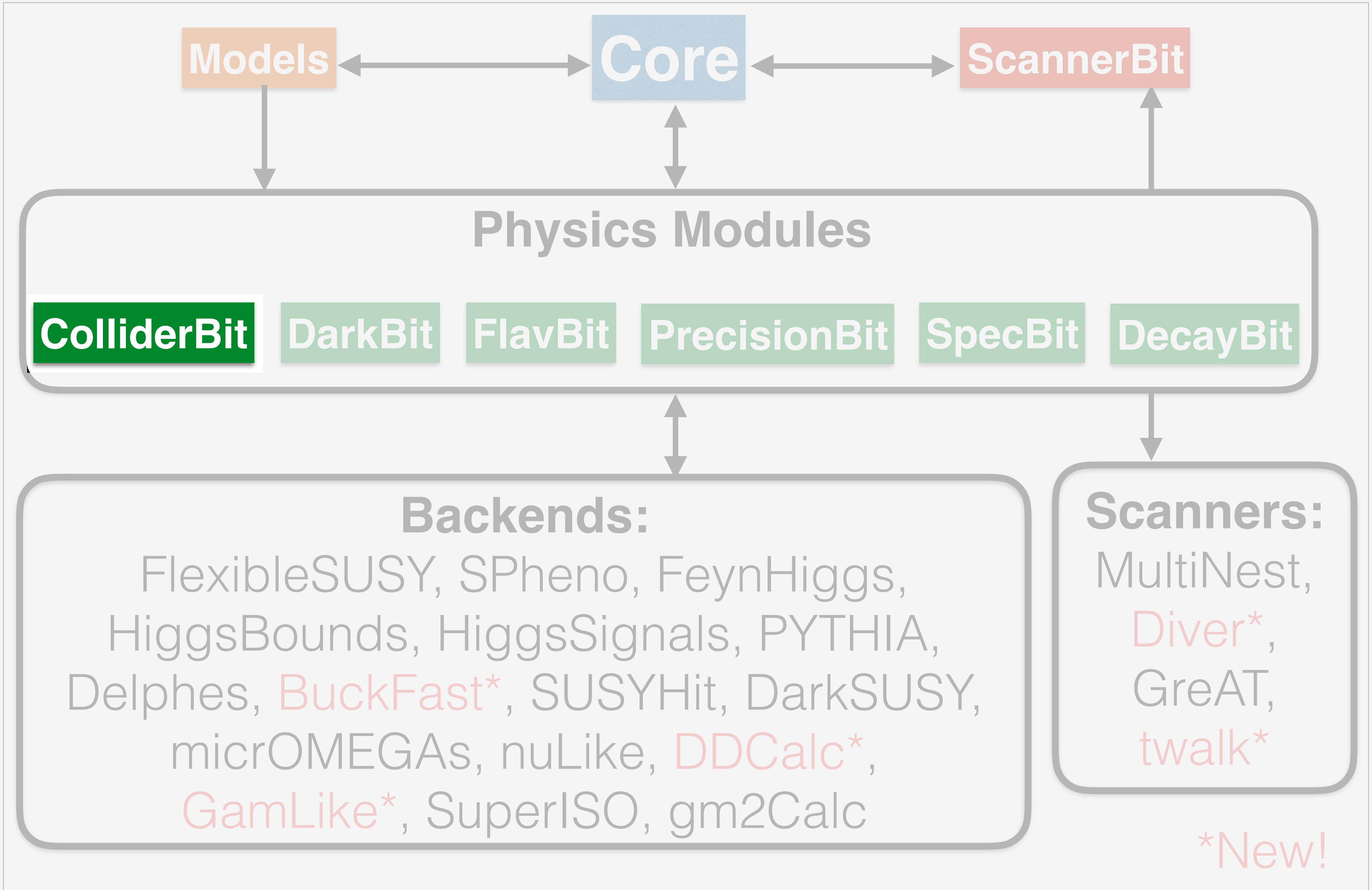
```
/// Br B->tau nu_tau decays
void SI_Btaunu(double &result)
{
  using namespace Pipes::SI_Btaunu;

  parameters const& param = *Dep::SuperIso_modelinfo;
  result = BEreq::Btaunu(&param);
}
```

CMSSM:



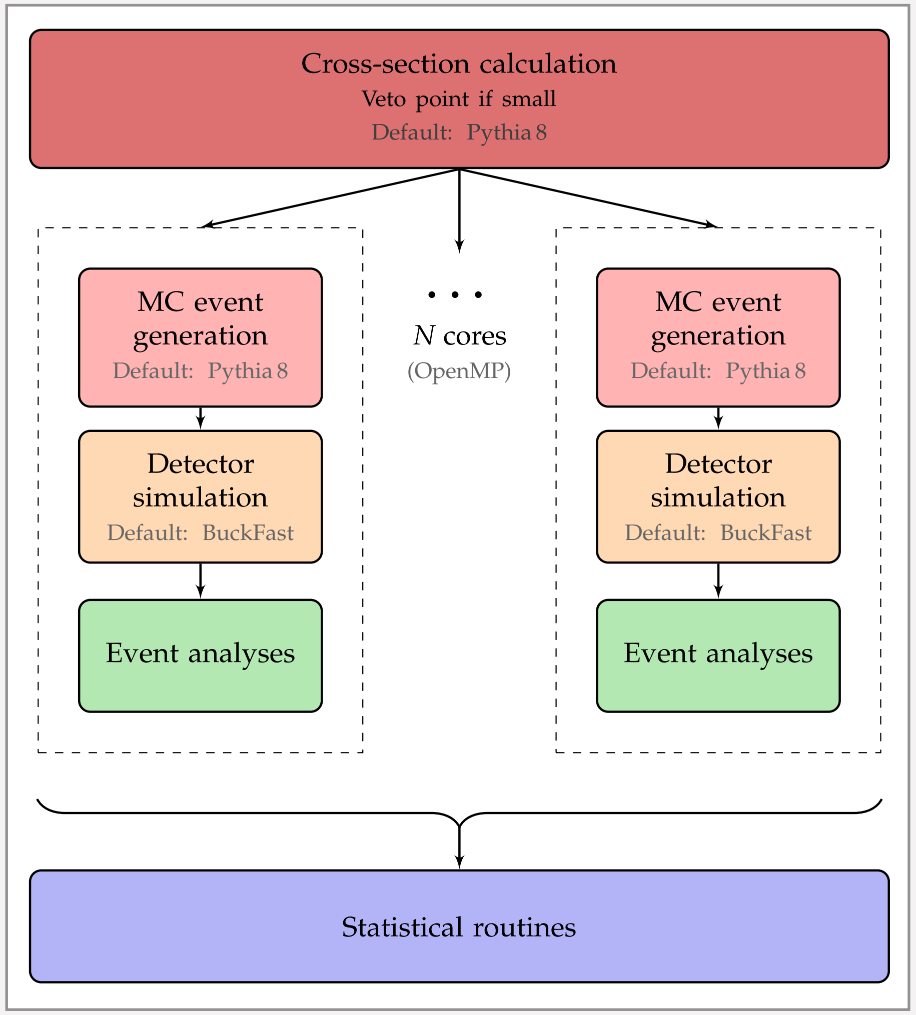
Red: Model parameter translations
Blue: Precision calculations
Green: LEP rates+likelihoods
Purple: Decays
Orange: LHC observables and likelihoods
Grey: DM direct, indirect and relic density
Pink: Flavour physics

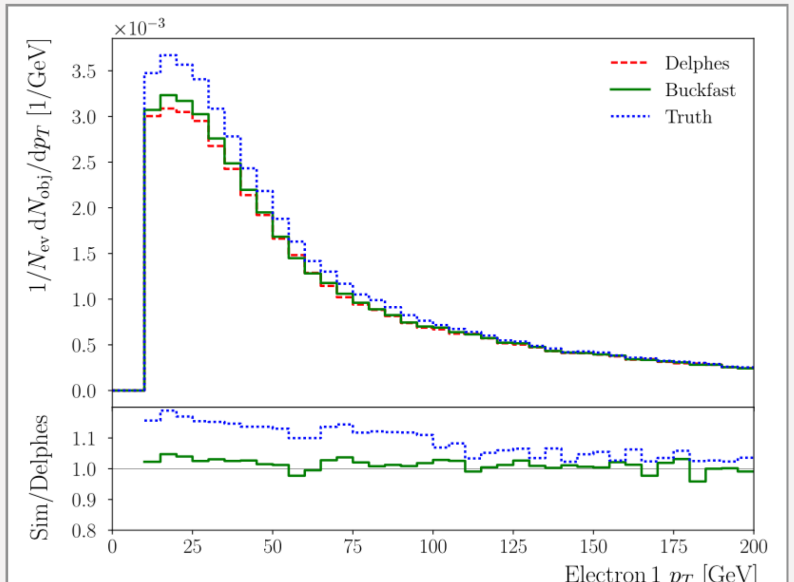
Code structure



ColliderBit

- **Higgs:** Connect HiggsBounds and HiggsSignals as backends (more to come)
- **LEP limits (SUSY):** Calculate $\sigma \times BR$ and check against published limits
- **LHC particle searches:** Full Poisson likelihood from fast MC simulation of LHC searches
  - Parallellized MC event generation and analysis loop inside ColliderBit
  - Event generation with Pythia 8
  - Fast detector simulator: BuckFast (4-vector smearing)

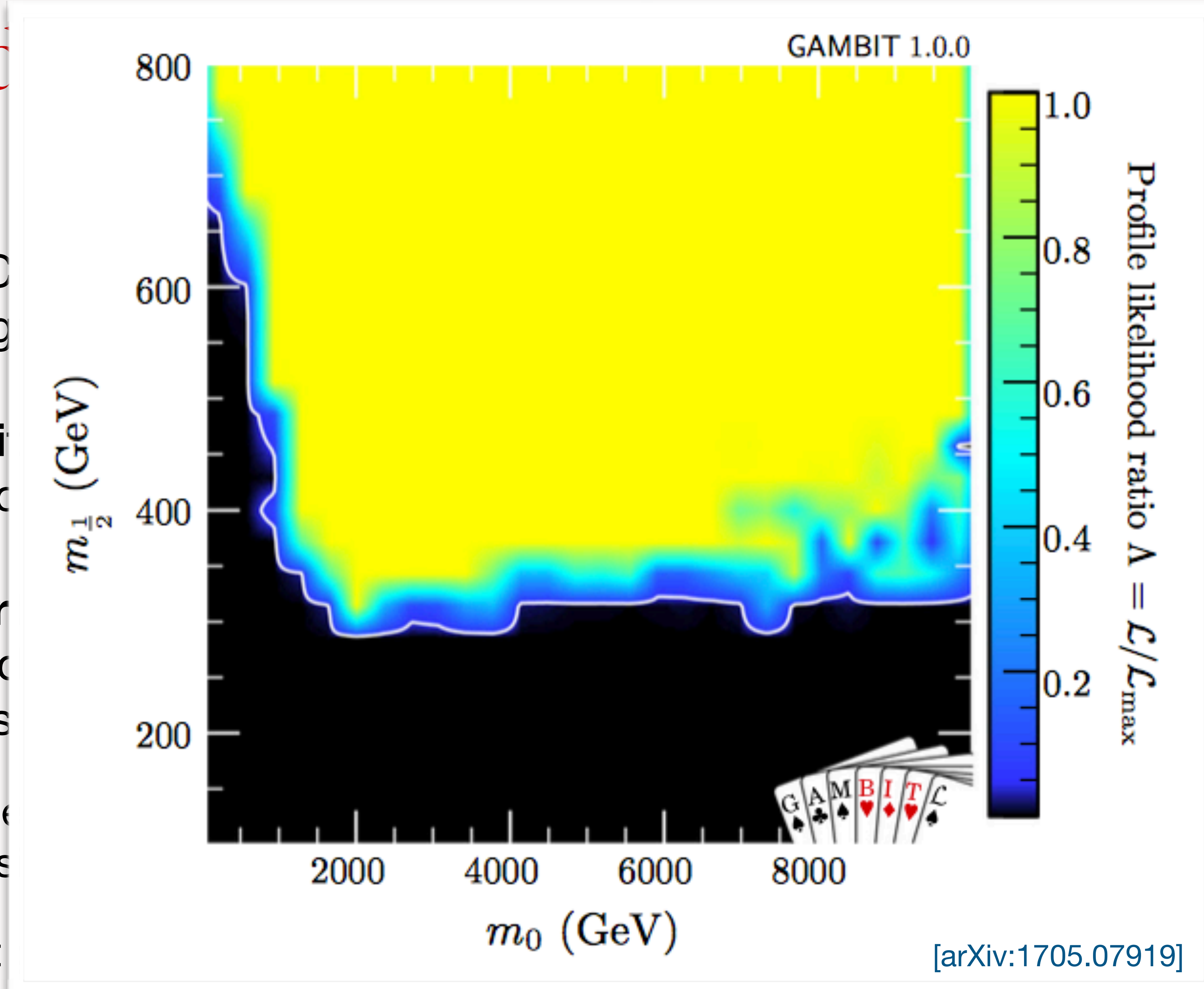




p_T distribution of leading electrons.

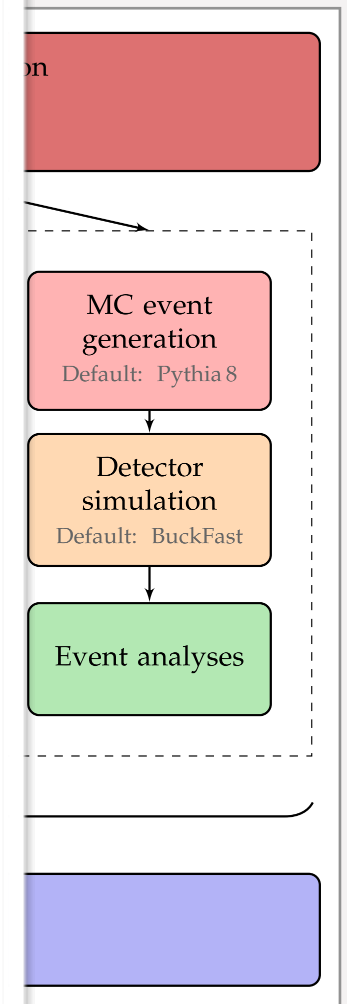
[arXiv:1705.07919]

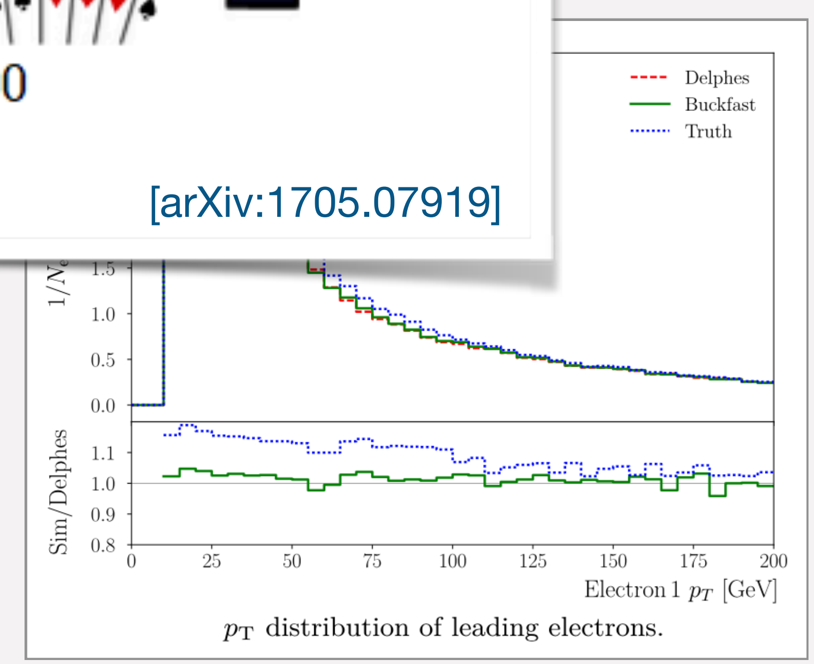
Collic



[arXiv:1705.07919]

- **Higgs:** C
  HiggsSig
- **LEP limi**
  and chec
- **LHC par**
  likelihood
  searches
  - Paralle
    analys
  - Event
  - Fast detector simulator: BuckFast (4-vector smearing)



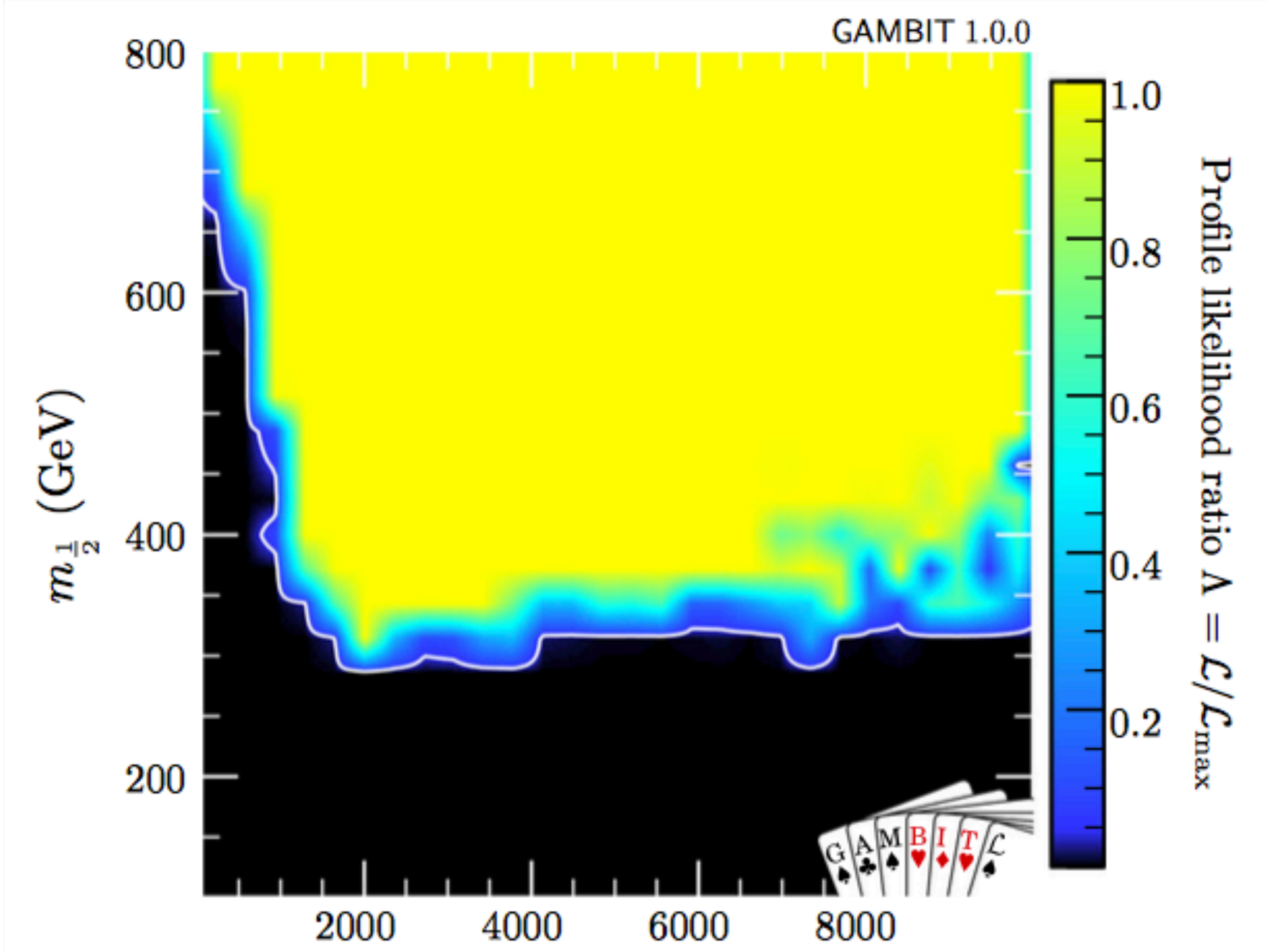


p_T distribution of leading electrons.

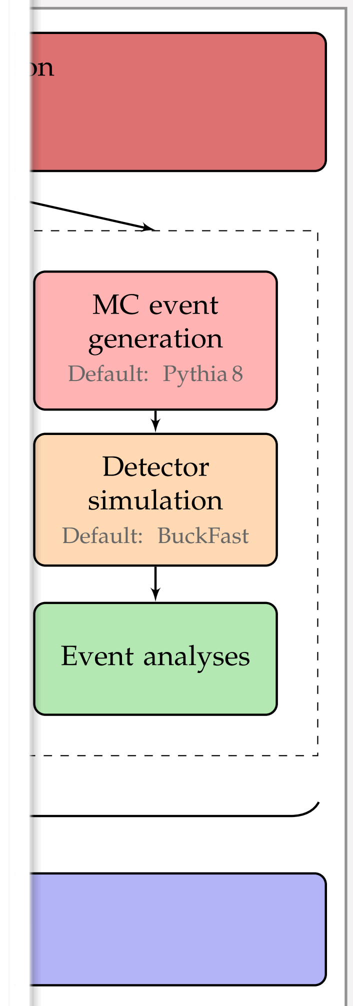
[arXiv:1705.07919]

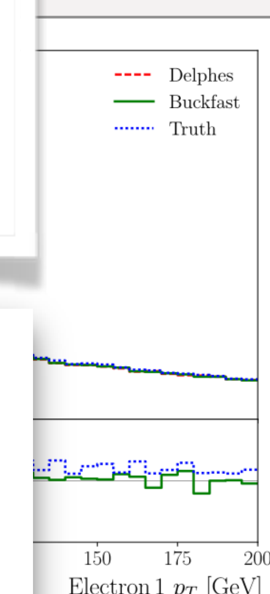
Collid

- **Higgs:** C
  HiggsSig
- **LEP limi**
  and chec
- **LHC par**
  likelihood
  searches
  - Paralle
    analys
  - Event
  - Fast d
    (4-vec



[arXiv:1705.07919]





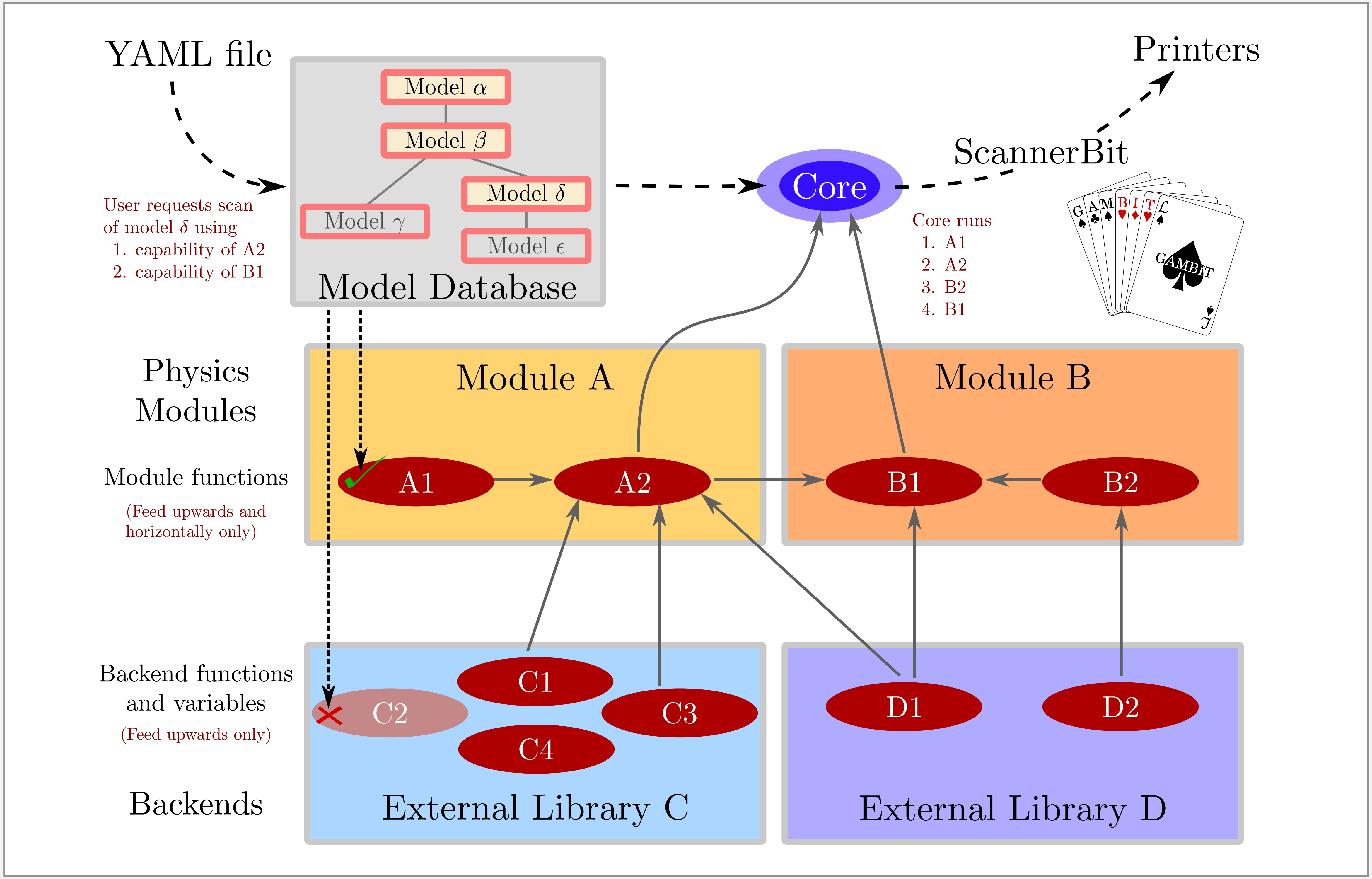
electrons.

Possible important improvement:

- **Fast** and **general** NLO cross section calculation
- SUSY: see **Jon Vegard Sparre's** talk this afternoon!

[arXiv:1705.07919]

GAMBIT scan overview



[arXiv:1705.07908]

Summary

- **GAMBIT is a new open-source framework for BSM global fits**
  - Includes several stand-alone physics modules
  - 6 code papers and 3 physics papers published in EPJC
- **First physics results**
  - Singlet DM
  - GUT-scale SUSY
  - Weak-scale MSSM7

→ **See Are Raklev's talk**

- **More results coming soon**
  - Sterile neutrinos, axions, 2HDMs, MSSM9, Higgs portals, ++

→ **See Tomás Gonzalo's talk**

- **Future plans**
  - More models! More likelihoods!
  - GAMBIT 2.0: Interface with Lagrangian-level tools for automatic code generation

Summary

- **GAMBIT is a new open-source framework for BSM global fits**
  - Includes several stand-alone physics modules
  - 6 code papers and 3 physics papers published in EPJC

- **First physics results**
  - Singlet DM
  - GUT-scale SUSY
  - Weak-scale MSSM7

→ **See Are Raklev's talk**

- **More results coming soon**
  - Sterile neutrinos, axions, 2HDMs, MSSM9, Higgs portals, ++

→ **See Tomás Gonzalo's talk**

- **Future plans**
  - More models! More likelihoods!
  - GAMBIT 2.0: Interface with Lagrangian-level tools for automatic code generation

Thank you!

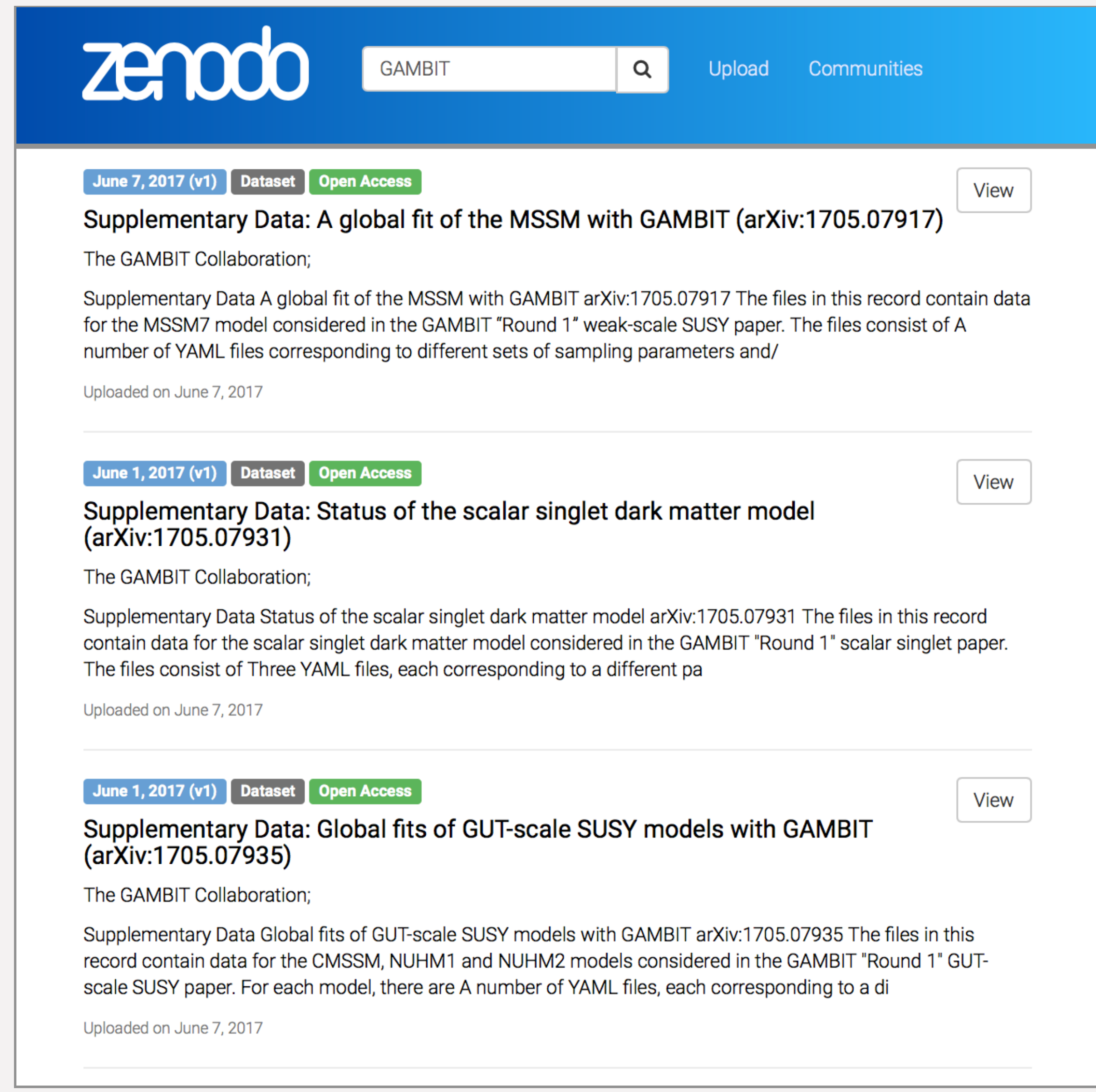
Backup slides

All GAMBIT results are publicly available

Results available on zenodo.cern.ch

- Parameter point samples (hdf5 files)
- GAMBIT input files for all scans
- Example plotting routines

Links at gambit.hepforge.org/pubs



Get started with GAMBIT

Clone git repository from GitHub

- github.com/patscott/gambit_1.1

Download tarballs

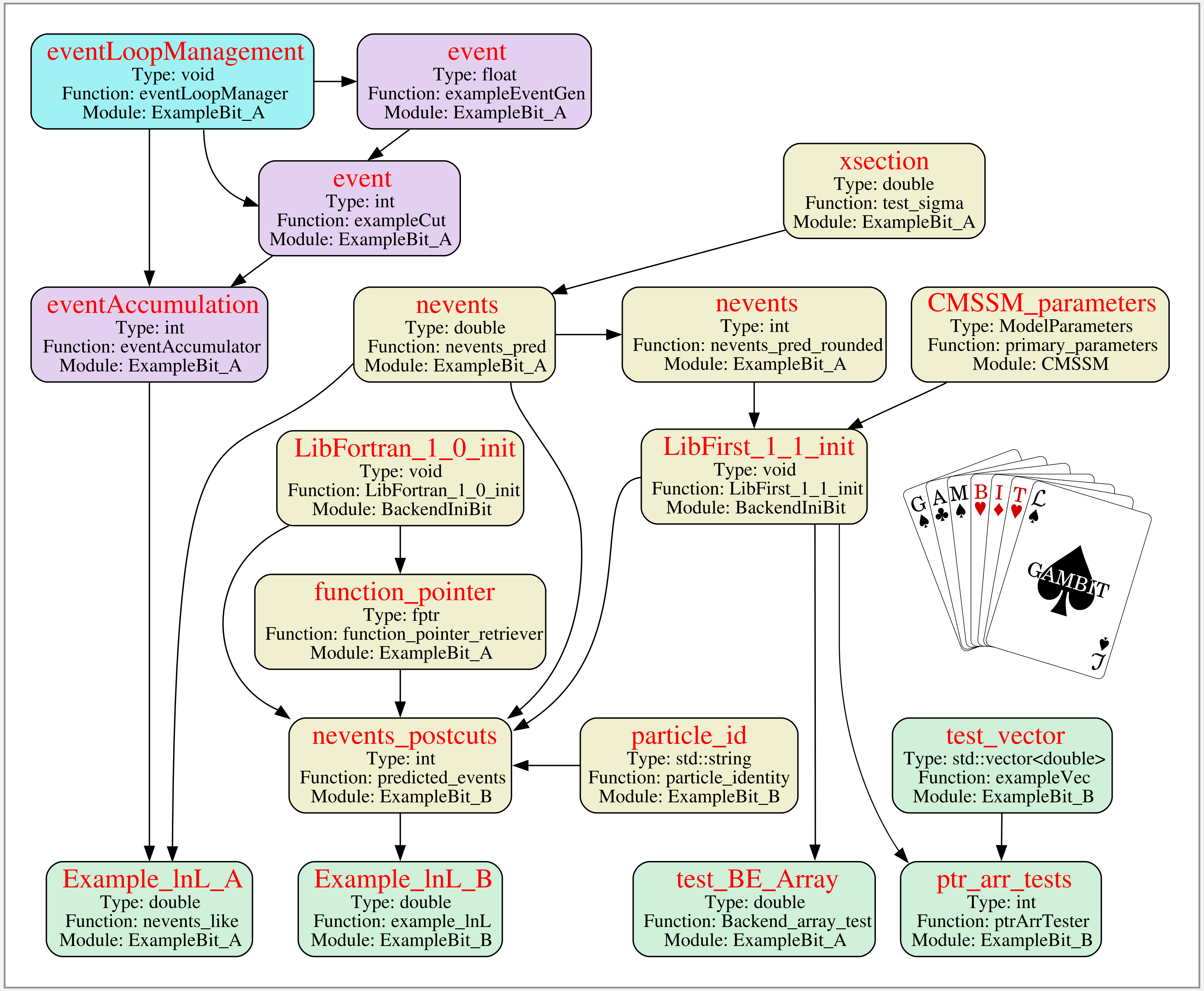
- hepforge.org/downloads/gambit

Pre-compilied version with Docker [Sebastian Liem]

- `docker run -it sliem/gambit`

See quick start guide in arXiv:1705.07908

Dependency resolution



Adding a new model to GAMBIT

(From Pat Scott)

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.

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)

ColliderBit speed

Table 1 Time taken for the ColliderBit LHC likelihood calculation as a function of the number of cores, for 100,000 SUSY events at the SPS1a parameter point [83,84], including all sub-processes. The processes were run on a single computer node, with ISR, FSR, and full hadronisation enabled, but multiple parton interactions and tau decay spin correlations disabled. GAMBIT was compiled with full optimisation settings (cf. Sect. 11 of Ref. [1])

Num. cores	t (10^5 events) (s)	Speed-up
1	421	1
4	128	3.3
8	67	6.3
16	38	11.1
20	33	12.8

Table 2 Single-thread CPU effects of sequentially disabling event simulation components, for 100,000 SUSY events at the SPS1a parameter point [83,84], including all sub-processes. The disabled components have a major effect on CPU, and a minor (sometimes even positive) effect on physics performance. The third row corresponds to the first row in Table 1. Note that the few percent difference is typical of the variation with local CPU load on the cluster on which this was tested

Configuration	t (10^5 events) (s)	Speed-up
All	1529	1
$\hookrightarrow$ −MPI	516	3.0
$\hookrightarrow$ −τ correlations	434	3.5
$\hookrightarrow$ −FSR	195	7.8
$\hookrightarrow$ −hadrons	102	15.0

[arXiv:1705.07919]