GAMBIT Tutorial

Tomás Gonzalo

Institute for Theoretical Particle Physics and Cosmology
RWTH Aachen

Overview

1. What is GAMBIT?
2. How to write code for GAMBIT?
3. How to use GAMBIT?

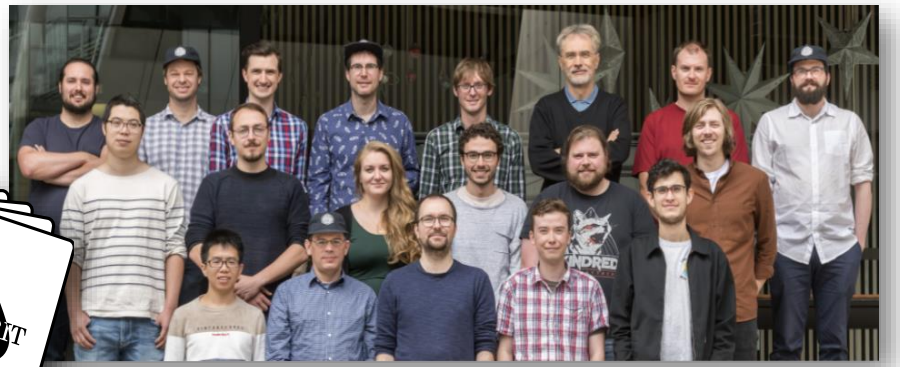
What is GAMBIT?

Institute for Theoretical Particle Physics and Cosmology | RWTH AACHEN UNIVERSITY

GAMBIT: The Global And Modular BSM Inference Tool

gambit.hepforge.org github.com/GambitBSM EPJC 77 (2017) 784 arXiv:1705.07908

- Extensive model database, beyond SUSY
- Fast definition of new datasets, theories
- Extensive observable/data libraries
- Plug&play scanning/physics/likelihood packages
- Various statistical options (frequentist /Bayesian)
- Fast LHC likelihood calculator
- Massively parallel
- Fully open-source



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

Authors of: BubbleProfiler, Capt'n General, Contur, DarkAges, DarkSUSY, DDCalc, DirectDM, Diver, EasyScanHEP, ExoCLASS, FlexibleSUSY, gamLike, GM2Calc, HEPLike, IsaTools, MARTY, nuLike, PhaseTracer, PolyChord, Rivet, SOFTSUSY, SuperIso, SUSY-AI, xsec, Vevacious, WIMPSim

Recent collaborators: P Athron, C Balázs, A Beniwal, S Bloor, T Bringmann, A Buckley, J-E Camargo-Molina, C Chang, M Chrzaszcz, J Conrad, J Cornell, M Danninger, J Edsjö, T Emken, A Fowlie, T Gonzalo, W Handley, J Harz, S Hoof, F Kahlhoefer, A Kvellestad, P Jackson, D Jacob, C Lin, N Mahmoudi, G Martinez, MT Prim, A Raklev, C Rogan, R Ruiz, N Serra, P Scott, P Stöcker, A Vincent, C Weniger, M White, Y Zhang, ++

70+ participants in many experiments and numerous major theory codes

Modules (Bits)

- Physics Modules
  - → **ColliderBit**: collider searches [Eur.Phys.J. C77 (2017) no.11, 795]
  - → **DarkBit**: relic density, dd,. . . [Eur.Phys.J. C77 (2017) no.12, 831]
  - → **FlavBit**: flavour observables [Eur.Phys.J. C77 (2017) no.11, 786]
  - → **SpecBit**: spectra, RGE running [Eur.Phys.J. C78 (2018) no.1, 22]
  - → **DecayBit**: decay widths [Eur.Phys.J. C78 (2018) no.1, 22]
  - → **PrecisionBit**: precision tests [Eur.Phys.J. C78 (2018) no.1, 22]
  - → **NeutrinoBit**: neutrino likelihoods [Eur.Phys.J.C 80 (2020) no.6, 569]
  - → **CosmoBit**: cosmological constraints [JCAP 02 (2021) 022]
- **ScannerBit** : stats and sampling [Eur.Phys.J. C77 (2017) no.11, 761]
  - → Diver, GreAT, Multinest, Polychord, . . .
- **Models**: hierarchical model database
- **Core** : dependency resolution [Eur.Phys.J. C78 (2018) no.2, 98]
- **Backends** : External tools to calculate observables
- **GUM**: Autogeneration of code [Eur.Phys.J. C81 (2021) no 12, 1103]

How to write code for GAMBIT?

Module functions - what?

- **Module functions** are the building blocks of GAMBIT
- Module functions provide a **capability**
- They have **dependencies** on other capabilities
- They have **backend requirements**
- Can be allowed for specific **models**
- GAMBIT resolves the dependeny graph at runtime

```
#define CAPABILITY RD_oh2
START_CAPABILITY

  /// General Boltzmann solver from DarkSUSY, using arbitrary Weff
  #define FUNCTION RD_oh2_DS_general
    START_FUNCTION(double)
    DEPENDENCY(RD_spectrum_ordered, RD_spectrum_type)
    DEPENDENCY(RD_eff_annrate, fptr_dd)
    BACKEND_REQ(rdpars, (ds6), DS_RDPARS)
    BACKEND_REQ(rdtime, (ds6), DS_RDTIME)
    BACKEND_REQ(dsrdcom, (ds6), void, ())
    BACKEND_REQ(dsrdstart,(ds6),void,(int&, double(&)[1000], double(&)[100
    BACKEND_REQ(dsrdens, (ds6), void, (double(*)(double&), double&, double
    BACKEND_OPTION((DarkSUSY_MSSM),(ds6))
    BACKEND_OPTION((DarkSUSY_generic_wimp),(ds6))
    FORCE_SAME_BACKEND(ds6)
  #undef FUNCTION

  /// Routine for cross checking relic density results, using MicrOmegas
  #define FUNCTION RD_oh2_MicrOmegas
    START_FUNCTION(double)
    BACKEND_REQ(oh2, (gimmemicro), double, (double*,int,double))
    BACKEND_OPTION((MicrOmegas_MSSM), (gimmemicro))
    BACKEND_OPTION((MicrOmegas_ScalarSingletDM_Z2), (gimmemicro))
    BACKEND_OPTION((MicrOmegas_ScalarSingletDM_Z3), (gimmemicro))
    BACKEND_OPTION((MicrOmegas_VectorSingletDM_Z2), (gimmemicro))
    BACKEND_OPTION((MicrOmegas_MajoranaSingletDM_Z2), (gimmemicro))
    BACKEND_OPTION((MicrOmegas_DiracSingletDM_Z2),(gimmemicro))
    ALLOW_MODELS(MSSM63atQ, MSSM63atMGUT,
                 ScalarSingletDM_Z2, ScalarSingletDM_Z2_running,
                 ScalarSingletDM_Z3, ScalarSingletDM_Z3_running,
                 DiracSingletDM_Z2, MajoranaSingletDM_Z2, VectorSingletDM_
  #undef FUNCTION

  /// Routine for computing axion energy density today from vacuum misalig
  #define FUNCTION RD_oh2_Axions
    START_FUNCTION(double)
      ALLOW_MODEL(GeneralALP)
      DEPENDENCY(AxionOscillationTemperature, double)
      DEPENDENCY(T_cmb, double)
  #undef FUNCTION
#undef CAPABILITY
```

Module functions - how?

- **Step 1:** Rollcall header

`MyModuleBit/include/gambit/MyModuleBit/MyModuleBit_rollcall.hpp`

```
// Capability
#define CAPABILITY MyCapability
START_CAPABILITY

  // Module function
  #define FUNCTION MyFunction
  START_FUNCTION(double)

  // Dependencies
  DEPENDENCY(OtherCapability, int)

  // Backend requirement
  BACKEND_REQ(BackendCap, (tag), void, (int&, double&))
  BACKEND_OPTION((MyBackend, 1.0.0), (tag))

  // Models
  ALLOW_MODELS(Model_A, Model_B)
  ALLOW_JOINT_MODEL(Model_C, Model_D)

  #undef FUNCTION
#undef CAPABILITY
```

- **Step 2**: Source file

`MyModuleBit/src/MyModuleBit.cpp`

```
// Signature
void MyFunction(double &result)
{

  // Dependency
  int val = *Pipes::MyFunction::Dep::OtherCapability;

  // Backend requirement
  Pipes::MyFunction::BEreq::BackendCap(val, result);

  // Access to parameters
  double param = *Pipes::MyFunction::Param["par1"];

  // Other pipes
  Pipes::MyFunction::ModelInUse("Model_A");
  Pipes::MyFunction::Downstream::subcaps;
  Pipes::MyFunction::Downstream::neededFor("something");
}
```

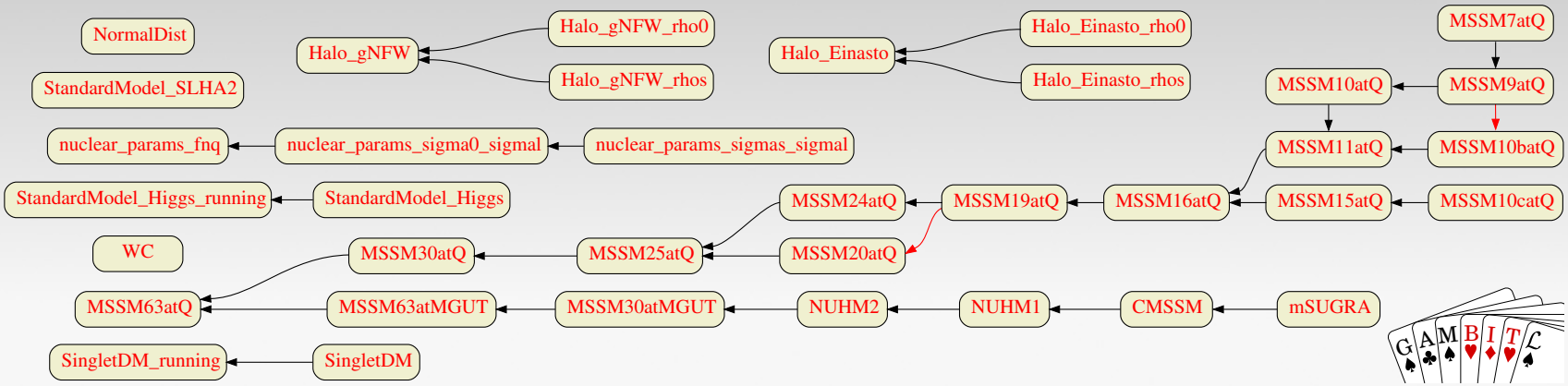
```
void MyFunction(double &result)
{

  using namespace Pipes::MyFunction;

  int val = *Dep::OtherCapability;

  ...

}
```

Models - what?

- Extensive model database

SUSY	DM	Cosmo	Others
CMSSM	Scalar Singlet	ΛCDM	SM
NUHM1,2	Fermionic Singlet	$\Delta N_{\rm eff}$	RH neutrinos
MSSM63atQ	Vector Singlet	Power-law	WC
	Axions	inflation	nuisance models

- Parent-daughter hierarchy
- Module functions are activated for each model



Models - how?

- **Step 1**: Declaration

`Models/include/gambit/Models/models/Model_A.hpp`

```
// Model declaration
#define MODEL Model_A
  START_MODEL
  DEFINEPARS(par1,par2,par3) // Up to 10
  DEFINEPARS(par4)
#undef MODEL
```

→ Parent

```
#define MODEL Model_A
#define PARENT Model_B
..
INTERPRET_AS_PARENT_FUNCTION(Model_A_to_Model_B)
#undef PARENT
#undef MODEL
```

→ Friend

```
INTERPRET_AS_X_FUNCTION(Model_C, Model_A_to_Model_C)
```

→ Dependencies

```
INTEPRET_AS_PARENT_DEPENDENCY(aCapability, aType)
INTEPRET_AS_X_DEPENDENCY(Model_C, aCapability, aType)
```

- **Step 2**: *Translation function*

`Models/src/models/Model_A.hpp`

```
#define MODEL Model_A
void MODEL_NAMESPACE::Model_A_to_Model_B(const ModelParameters &myP,
ModelParameters &targetP)
{
  // Set parameters
  targetP.setValue("Par1", myP.getValue("par1"));

  ...
}
#undef MODEL
```

→ Dependencies

```
// Using dependencies
using namespace MODEL_NAMESPACE::Pipes::Model_B_parameters;
aType val = *Dep::aCapability;
```

```
//Shortcut
USE_MODEL_PIPE(Model_B)
aType val = *Dep::aCapability;
```

- **Step 3**: Use it

```
ALLOW_MODELS(Model_B)
```

```
double par1 = *Param["Par1"];
```

Backends - what?

- External tools used to compute some physical quantity
- Interfaced with GAMBIT dynamically
- C, Fortran $\rightsquigarrow$ POSIX `dl`
- C++ $\rightsquigarrow$ BOSS + POSIX `dl`
- Mathematica $\rightsquigarrow$ WSTP
- Python $\rightsquigarrow$ pybind11

CosmoBit

AlterBBN 2.2
DarkAges 1.2.0
MontePythonLike 3.3.0
MultiModeCode 2.0.0
classy 2.9.4

DarkBit

CaptnGeneral 1.0
DDCalc 2.2.0
DarkSUSY 6.2.2
MicrOmegas 3.6.9.2
gamLike 1.0.1

ColliderBit

HiggsBounds 4.3.1
HiggsSignals 1.4
Pythia 8.212

PrecisionBit

FeynHiggs 2.12.0
SUSYHD 1.0.2
gm2calc 1.3.0

SpecBit

FlexibleSUSY 2.0.1
SPheno 4.0.3

FlavBit

SuperISO 3.6

DecayBit

SUSY_HIT 1.5

Backends - how?

- **Step 1**: Build step `cmake/backends.cmake`

```
set(name "MyBackend")
set(ver "1.0")
set(lib "libmybackend")
set(dl "https://...mybackend_v1.0.tgz")
set(md5 "0000000000000000")
set(dir "${PROJECT_SOURCE_DIR}/Backends/installed/${name}/${ver}")
check_ditch_status(${name} ${ver} ${dir})
if(NOT ditched_${name}_${ver})
  ExternalProject_Add(${name}_${ver}
    DOWNLOAD_COMMAND ${DL_BACKEND} ${dl} ${md5} ${dir} ${name} ${ver}
    SOURCE_DIR ${dir}
    BUILD_IN_SOURCE 1
    CONFIGURE_COMMAND ""
    BUILD_COMMAND ${MAKE_PARALLEL} ${lib}.so
    INSTALL_COMMAND ""
  )
  add_extra_targets("backend" ${name} ${ver} ${dir} ${dl} clean)
  set_as_default_version("backend" ${name} ${ver})
endif()
```

→ Patch it

```
set(patchname "${name}_${ver}.diff")
set(patch "${PROJECT_SOURCE_DIR}/Backends/patches/${name}/${ver}/${patchname}")
...
    BUILD_IN_SOURCE 1
    PATCH_COMMAND patch -p1 < ${patch}
    CONFIGURE_COMMAND ""
...
```

→ BOSS it (C++) `BOSS_backend(${name} ${ver})`

→ Dependencies

```
DEPENDS otherBackend_version
DOWNLOAD_COMMAND ${DL_BACKEND} ${dl} ${md5} ${dir} ${name} ${ver}
```

```
set(ditch_if_absent "staticPackage")
check_ditch_status(${name} ${ver} ${dir} ${ditch_if_absent})
```

```
set(required_modules "python_module")
check_python_modules(${name} ${ver} ${required_modules})
```

Backends - how?

- **Step 2**: Frontend header ~~C++~~

Backends/include/gambit/Backends/frontends/MyBackend_1_0.hpp

```
#define BACKENDNAME MyBackend
#define BACKENDLANG CC  // CC, CXX, Fortran, Mathematica, Python
#define VERSION 1.0.0
#define SAFE_VERSION 1_0_0
#define REFERENCE Bibkey

// Begin
LOAD_LIBRARY

// Allow for models
BE_ALLOW_MODELS(Model_A)

...

// End
#include "gambit/Backends/backend_undefs.hpp"
```

→ Backend Variables

```
BE_VARIABLE(MyVar, int, ("myvar_symbol"), "MyVar_Cap")
```

→ Backend Function

```
BE_FUNCTION(MyFunc, void, (double&), ("myfunc_symbol"), "MyFunc_Cap")
```

→ Convenience functions

```
BE_CONV_FUNCTION(MyConv, int, (bool&, double&), "MyConv_Cap")
```

→ Ini dependencies

```
BE_INI_DEPENDENCY(someCap, double)
```

- **Step 3**: *Frontend source*

Backends/src/frontends/MyBackend_1_0.cpp

→ Convenience functions

```
BE_NAMESPACE
{
  int myConv(bool &a, double &b)
  {
    ...
  }
}
END_BE_NAMESPACE
```

→ Ini function

```
BE_INI_FUNCTION
{
  // Scan-level initialisation
  static bool scan_level = true;
  if (scan_level)
  {
    double val = *Dep::someCap;
    ...
  }
  scan_level = false;


}
END_BE_INI_FUNCTION
```

Backends - how?

- **Step 4**: Backend location

`config/backend_locations.yaml.default`

```
MyBackend:
  1.0:         ./Backends/installed/mybackend/1.0/lib/libmybackend.so
```

- **Step 5**: Reference

`config/bibtex_entries.bib`

```
@article{Bibkey,
    author = "Author, The",
    title = "{My Backend}",
    eprint = "xxxx.xxxxxx",
    archivePrefix = "arXiv",
    primaryClass = "hep-ph",
    year = "2022"
}
```

- **Step 6**: Backend requirement

```
#define FUNCTION MyFunction
START_FUNCTION(double)
...

// Backend requirement
BACKEND_REQ(MyFunc_Cap, (tag), void, (double&))
BACKEND_REQ(MyConv_Cap, (tag), int (bool&, double&))
BACKEND_OPTION((MyBackend, 1.0.0), (tag))

...
#undef FUNCTION
```

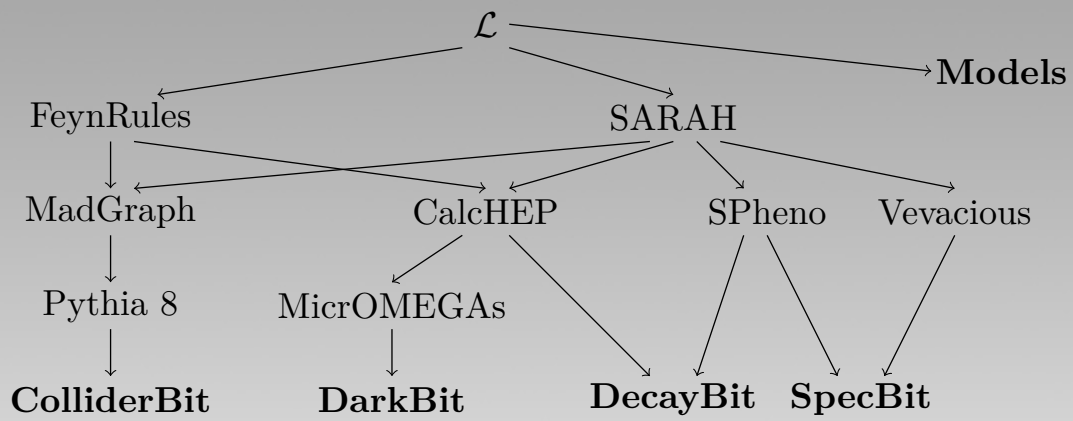
- **Step 7**: *BOSS config file*

`Backends/scripts/BOSS/configs/mybackend_1_0.py`

```
...
gambit_backend_name      = 'MyBackend'
gambit_backend_version   = '1.0.0'
gambit_backend_reference = 'Bibkey'
gambit_base_namespace    = ''

input_files =
[
  '../../../Backends/installed/mybackend/1.0.0/file1.h',
  '../../../Backends/installed/mybackend/1.0.0/file2.h'
]
include_paths =
[
  '../../../Backends/installed/mybackend/1.0.0/header1.h',
]
base_paths =
[
  '../../../Backends/installed/mybackend/1.0.0/'
]

header_files_to = '../../../Backends/installed/mybackend/1.0.0/include'
src_files_to    = '../../../Backends/installed/mybackend/1.0.0./src'

load_classes = [
    'ClassOne',
    'SomeNamespace::ClassTwo',
]

load_functions = [
    'SomeNamespace::foo(int, SomeNamespace::ClassTwo)'
]

ditch = []
...
```

GUM

- GUM interfaces LLT SARAH and FeynRules with GAMBIT
- Uses existing HEP toolchains



- GAMBIT-compatible outputs from GUM

Generated output	FeynRules	SARAH	Usage in GAMBIT
CalcHEP	✓	✓	Decays, cross-sections
micrOMEGAs (via CalcHEP)	✓	✓	DM observables
Pythia (via MadGraph)	✓	✓	Collider physics
SPheno	✗	✓	Particle mass spectra, decay widths
Vevacious	✗	✓	Vacuum stability

How to use GAMBIT?

Compilation

- Configure
- Build scanners
- Build backends
- Build main gambit

```
cmake ..
make scanners
cmake ..
make -jn backends
make -jn
```

- Useful configuration options
  - → Build mode: -DCMAKE_BUILD_TYPE=Release
  - → Select compilers: -DCMAKE_CXX_COMPILER=g++
  - → Fix paths -DEIGEN3_INCLUDE_DIR=somepath
  - → Turn on/off MPI: -DWITH_MPI=on
  - → Turn on/off packages: -DWITH_ROOT=on, -DWITH_HEPMC=on
  - → Select FS model: -DBUILD_FS_MODELS=None
  - → Other cmake flags: -DCMAKE_CXX_FLAGS=xx
  - → Ditch modules/backends/stuff: -Ditch="NeutrinoBit;Python;Mathematica;DarkSUSY"

Diagnostics

- Can run diagnostics for all backends, scanners, modules, capabilities

```
./gambit backends
```

BACKENDS	VERSION	PATH TO LIB	STATUS	#FUNC	#TYPES	#CTOR
AlterBBN	2.2	Backends/installed/alterbbn/2.2/libbbn.so	absent/broken	6	0	0
CalcHEP	3.6.27	Backends/installed/calchep/3.6.27/lib/libcalchep.so	absent/broken	18	0	0
CaptnGeneral	2.1	Backends/installed/capgen/2.1/gencaplib.so	absent/broken	7	0	0
DDCalc	1.0.0	Backends/installed/ddcalc/1.0.0/lib/libDDCalc.so	absent/broken	36	0	0
	1.1.0	Backends/installed/ddcalc/1.1.0/lib/libDDCalc.so	absent/broken	38	0	0
	1.2.0	Backends/installed/ddcalc/1.2.0/lib/libDDCalc.so	absent/broken	39	0	0
	2.0.0	Backends/installed/ddcalc/2.0.0/lib/libDDCalc.so	absent/broken	50	0	0
	2.1.0	Backends/installed/ddcalc/2.1.0/lib/libDDCalc.so	absent/broken	49	0	0
	2.2.0	Backends/installed/ddcalc/2.2.0/lib/libDDCalc.so	OK	52	0	0
DarkAges	1.2.0	Backends/installed/darkages/1.2.0/DarkAges_1_2_0	absent/broken	1	0	0
DarkSUSY	5.1.3	Backends/installed/darksusy/5.1.3/lib/libdarksusy.so	absent/broken	81	0	0
DarkSUSY_MSSM	6.1.1	Backends/installed/darksusy/6.1.1/lib/libds_core_mssm.so	absent/broken	63	0	0
	6.2.2	Backends/installed/darksusy/6.2.2/lib/libds_core_mssm.so	absent/broken	64	0	0
	6.2.5	Backends/installed/darksusy/6.2.5/lib/libds_core_mssm.so	OK	63	0	0
DarkSUSY_generic_wimp	6.1.1	Backends/installed/darksusy/6.1.1/lib/libds_core_generic_wimp.so	absent/broken	19	0	0
	6.2.2	Backends/installed/darksusy/6.2.2/lib/libds_core_generic_wimp.so	absent/broken	19	0	0
	6.2.5	Backends/installed/darksusy/6.2.5/lib/libds_core_generic_wimp.so	OK	19	0	0
DirectDM	2.2.0	Backends/installed/directdm/2.2.0/directdm	absent/broken	1	0	0
FeynHiggs	2.11.2	Backends/installed/feynhiggs/2.11.2/lib/libFH.so	absent/broken	14	0	0
	2.11.3	Backends/installed/feynhiggs/2.11.3/lib/libFH.so	absent/broken	14	0	0
	2.12.0	Backends/installed/feynhiggs/2.12.0/lib/libFH.so	absent/broken	14	0	0
HiggsBounds	4.2.1	Backends/installed/higgsbounds/4.2.1/lib/libhiggsbounds.so	absent/broken	10	0	0
	4.3.1	Backends/installed/higgsbounds/4.3.1/lib/libhiggsbounds.so	absent/broken	10	0	0
HiggsSignals	1.4	Backends/installed/higgssignals/1.4.0/lib/libhiggssignals.so	absent/broken	12	0	0

YAML file

- Parameters Node

```yaml
Parameters:

  StandardModel_SLHA2:
    alphaS     :  1.18500000E-01
    mBmB       :  4.18000000E+00
    alphainv   :  1.27940010E+02
    mT         :  1.73340000E+02
    GF         :  1.16637870E-05
    mZ         :  9.11876000E+01
    mTau       :  1.77682000E+00
    mNu3       :                0
    mD         :  4.80000000E-03
    mU         :  2.30000000E-03
    mS         :  9.50000000E-02
    mCmC       :  1.27500000E+00
    mE         :  5.10998928E-04
    mMu        :  1.05658372E-01
    mNu1       :                0
    mNu2       :                0
    CKM_lambda:  0.22537
    CKM_A      :  0.814
    CKM_rhobar:  0.117
    CKM_etabar:  0.353
    theta12    :  0.58376
    theta23    :  0.76958
    theta13    :  0.15495
    delta13    :  0
    alpha1     :  0
    alpha2     :  0

  StandardModel_Higgs:
    mH: 125.09

  WC:
    Re_DeltaC7:
      range: [-0.1, 0.1]
    Im_DeltaC7:  0
    Re_DeltaC9:  0
    Im_DeltaC9:  0
    Re_DeltaC10:
      range: [-3, 3]
    Im_DeltaC10: 0
    Re_DeltaCQ1: 0
    Im_DeltaCQ1: 0
    Re_DeltaCQ2: 0
    Im_DeltaCQ2: 0
```

- Printers

(hdf5, ascii, sqlite,

cout, none)

```yaml
Printer:

  printer: hdf5

  options:
    output_file: "WC.hdf5"
    group: "/WC"
```

- Scanners

(diver, multinest,

polychord, minuit2,

twalk, raster, grid)

```yaml
Scanner:

  use_scanner: de

  scanners:

    multinest:
      plugin: multinest
      like:  LogLike
      nlive: 400
      tol: 0.1

    de:
      plugin: diver
      like: LogLike
      NP: 400
      convthresh: 1e-3
```

- Likelihoods

```yaml
ObsLikes:

  # Likelihoods
  - purpose:    LogLike
    capability: b2ll_LL

  - purpose:    LogLike
    capability: b2sgamma_LL
```

- Rules

```yaml
Rules:

  # Use SuperIso instead of FeynHiggs for b->sgamma
  - capability: bsgamma
    function: SI_bsgamma

  # Use SuperIso instead of FeynHiggs for B_s->mumu
  - capability: Bsmumu_untag
    function: SI_Bsmumu_untag
```

- Other

```yaml
Logger:

  redirection:
    [Debug]   : "debug.log"
    [Default] : "default.log"
    [FlavBit] : "FlavBit.log"


KeyValues:

  default_output_path: "runs/WC_lite"

  debug: true
```

Running GAMBIT

- 2D Wilson coefficient fit

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

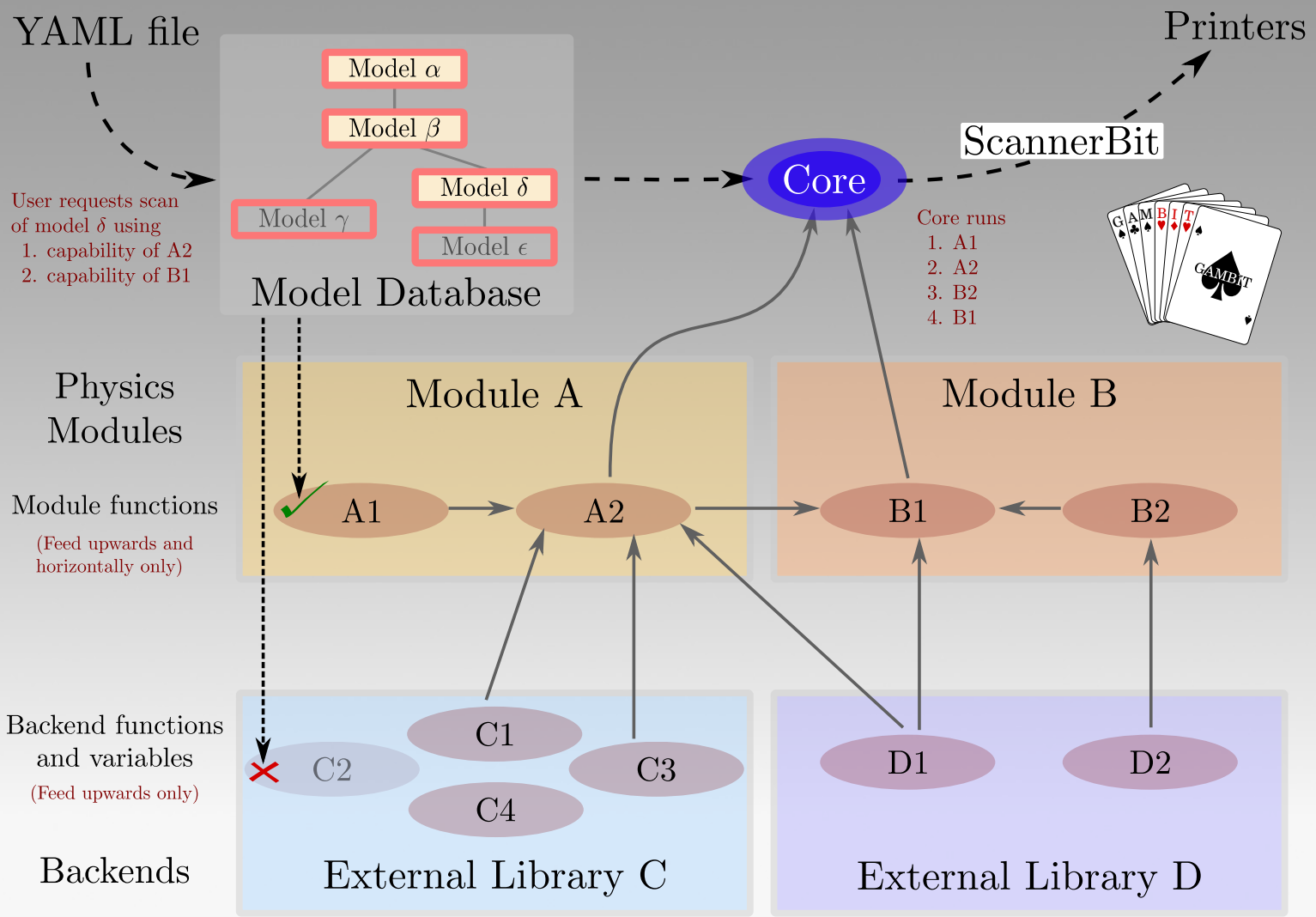
- Free parameters: ΔC_7 `Re_DeltaC7`
  ΔC_{10} `Re_DeltaC10`

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

- Run yaml file

```
./gambit -f yaml_files/WC_lite.yaml
```

An example run



Results

- Samples are written in output path
- Plotting tool `pippi`
- Run pippi
- Results

```
runs/WC_lite/samples/WC.hdf5
```

```
make get-pippi
```

```
./pippi/pippi yaml_files/WC_lite.pip
```