(Hadronic) interactions in C8

Corsika8
workshop
2022
Heidelberg
felix riehn

Available models

- QGSjet II-04 (HE)
- EPOS-LHC (HE)
- Sibyll 2.3d (HE,DEC)
- URQMD (LE)
- Pythia8 (8.2x (DEC) soon 8.3x (DEC,HE))

In EAS ?

EAS in c8 is an object of type “Cascade”

```
// create the cascade object using the default stack and tracking implementation
setup::Tracking tracking;
setup::Stack<EnvType> stack;
Cascade EAS(env, tracking, sequence, output, stack);
```

Interactions are “Processes” in ProcessSequence

```
// assemble the final process sequence
auto sequence = make_sequence(stackInspect, hadronSequence, decaySequence, cut,
                              emCascade, emContinuous, // trackWriter,
                              observationLevel, longprof);
```

Hadron sequence

```
corsika::sibyll::Interaction sibyll{env};
InteractionCounter sibyllCounted{sibyll};

corsika::urqmd::UrQMD urqmd;
InteractionCounter urqmdCounted(urqmd);

// assemble all processes into an ordered process list
struct EnergySwitch {
  HEPEnergyType cutE_;
  EnergySwitch(HEPEnergyType cutE)
      : cutE_(cutE) {}
  bool operator()(const Particle& p) const { return (p.getKineticEnergy() < cutE_); }
};
auto hadronSequence =
    make_select(EnergySwitch(heHadronModelThreshold), urqmdCounted, sibyllCounted);
auto decaySequence = make_sequence(decayPythia, decaySibyll);
```

Concretely

corsika/modules/ – all implemented processes

BetheBlochPDG.hpp	OnShellCheck.hpp	**sibyll**
conex	ParticleCut.hpp	Sibyll.hpp
CONEX.hpp	**proposal**	StackInspector.hpp
energy_loss	PROPOSAL.hpp	**tracking**
epos	**pythia8**	Tracking.hpp
Epos.hpp	Pythia8.hpp	TrackWriter.hpp
HadronicElasticModel.hpp	**qgsjetII**	**urqmd**
LongitudinalProfile.hpp	QGSJetII.hpp	UrQMD.hpp
ObservationPlane.hpp	Random.hpp	**writers**

corsika/modules/sibyll/ – actual interface code

Decay.hpp	NuclearInteractionModel.hpp	SibStack.hpp
HadronInteractionModel.hpp	ParticleConversion.hpp	
InteractionModel.hpp	Random.hpp	

How to add an interaction process to C8

```cpp
/**
 * @file Sibyll.hpp
 *
 * Includes all the parts of the Sibyll model. Defines the InteractionProcess<TModel>
 * classes needed for the ProcessSequence.
 */

namespace corsika::sibyll {
  /**
   * @brief sibyll::Interaction is the process for ProcessSequence.
   *
   * The sibyll::InteractionModel is wrapped as an InteractionProcess here in order
   * to provide all the functions for ProcessSequence.
   */
  struct Interaction : public InteractionModel, public InteractionProcess<Interaction> {
    template <typename TEnvironment>
    Interaction(TEnvironment const& env)
        : InteractionModel{env} {}
  };

  /**
   * @brief sibyll::NuclearInteraction is the process for ProcessSequence.
   *
   * The sibyll::NuclearInteractionModel is wrapped as an InteractionProcess here in order
   * to provide all the functions for ProcessSequence.
   */
  template <class TNucleonModel>
  class NuclearInteraction
      : public NuclearInteractionModel<TNucleonModel>,
        public InteractionProcess<NuclearInteraction<TNucleonModel>> {
  public:
    template <typename TEnvironment>
    NuclearInteraction(TNucleonModel& model, TEnvironment const& env)
        : NuclearInteractionModel<TNucleonModel>{model, env} {}
  };

} // namespace corsika::sibyll
```

Model namespace

interaction in cascade

Actual model interface

C8 discrete step process in cascade

A composite model: sibyll nuclear interactions

```cpp
/**
 * @file Sibyll.hpp
 *
 * Includes all the parts of the Sibyll model. Defines the InteractionProcess<TModel>
 * classes needed for the ProcessSequence.
 */

namespace corsika::sibyll {
  /**
   * @brief sibyll::Interaction is the process for ProcessSequence.
   *
   * The sibyll::InteractionModel is wrapped as an InteractionProcess here in order
   * to provide all the functions for ProcessSequence.
   */
  struct Interaction : public InteractionModel, public InteractionProcess<Interaction> {
    template <typename TEnvironment>
    Interaction(TEnvironment const& env)
        : InteractionModel{env} {}
  };

  /**
   * @brief sibyll::NuclearInteraction is the process for ProcessSequence.
   *
   * The sibyll::NuclearInteractionModel is wrapped as an InteractionProcess here in order
   * to provide all the functions for ProcessSequence.
   */
  template <class TNucleonModel>
  class NuclearInteraction
      : public NuclearInteractionModel<TNucleonModel>,
        public InteractionProcess<NuclearInteraction<TNucleonModel>> {
  public:
    template <typename TEnvironment>
    NuclearInteraction(TNucleonModel& model, TEnvironment const& env)
        : NuclearInteractionModel<TNucleonModel>{model, env} {}
  };

} // namespace corsika::sibyll
```

Template parameter for nucleon model

The nuclear model

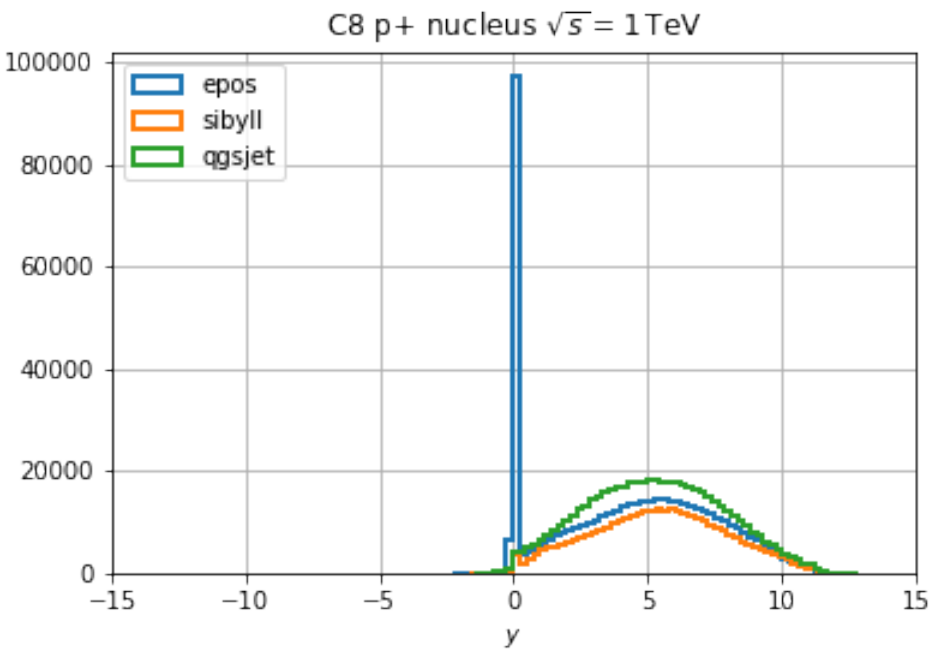
Class InteractionModel

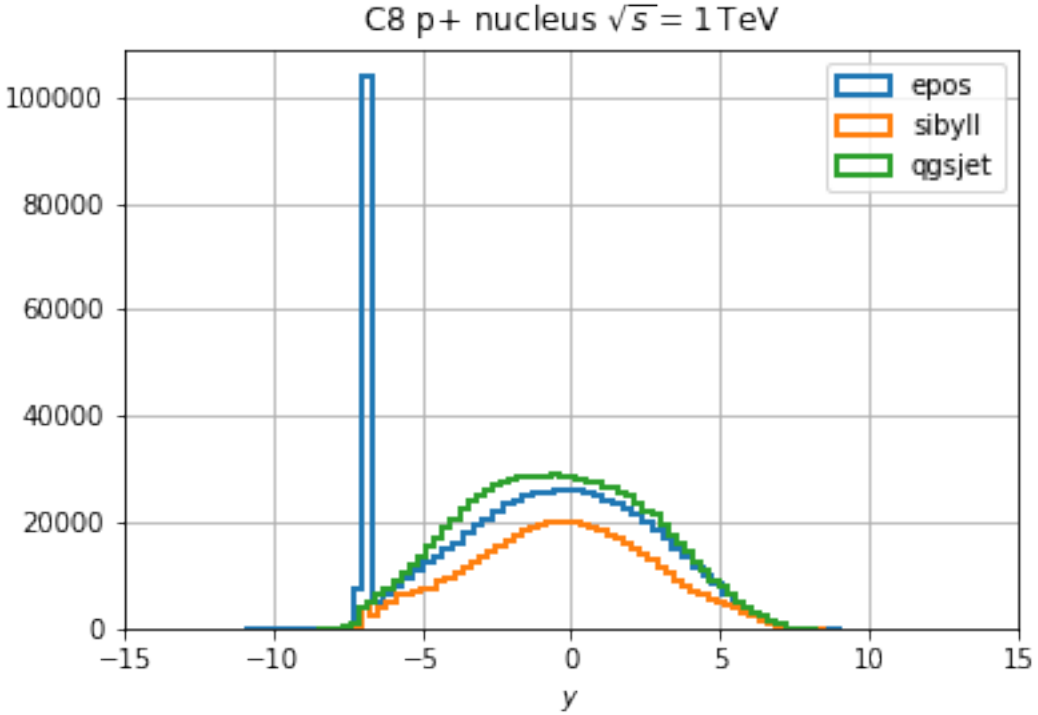
- isValid(projectileId, targetId, sqrtSNN)
  - For internal use, or for direct calls to InteractionModel
  - Inside cascade **NO CHECK** if process is allowed (unselect via cross section)

- getCrossSection(projectileId, targetId, projectileP4, targetP4)

- doInteraction(stack, projectileId, targetId, projectileP4, targetP4)

(Validation)


C8 p+ nucleus $\sqrt{s} = 1\,\mathrm{TeV}$
epos
sibyll
qgsjet
100000
80000
60000
40000
20000
0
−15
−10
−5
0
5
10
15
y


C8 p+ nucleus $\sqrt{s} = 1\,\mathrm{TeV}$
epos
sibyll
qgsjet
100000
80000
60000
40000
20000
0
−15
−10
−5
0
5
10
15
y

Plans / Todo

- Validation routines
- Sibyll Argon problem
- URQMD cross sections
- CRMC
- Decay overhaul
  - Teach unknown resonances to pythia
  - Improve bookkeeping
- Missing models
  - Sibyll 2.1
  - DPMjet
  - FLUKA
  - Neutrinos
  - Tau decays

How to add an interaction process to C8

```
/**
 * @file Sibyll.hpp
 *
 * Includes all the parts of the Sibyll model. Defines the InteractionProcess<TModel>
 * classes needed for the ProcessSequence.
 */

namespace corsika::sibyll {
  /**
   * @brief sibyll::Interaction is the process for ProcessSequence.
   *
   * The sibyll::InteractionModel is wrapped as an InteractionProcess here in order
   * to provide all the functions for ProcessSequence.
   */
  class Interaction : public InteractionModel, public InteractionProcess<Interaction> {};

  /**
   * @brief sibyll::NuclearInteraction is the process for ProcessSequence.
   *
   * The sibyll::NuclearInteractionModel is wrapped as an InteractionProcess here in order
   * to provide all the functions for ProcessSequence.
   */
  template <class TEnvironment, class TNucleonModel>
  class NuclearInteraction
      : public NuclearInteractionModel<TEnvironment, TNucleonModel>,
        public InteractionProcess<NuclearInteraction<TEnvironment, TNucleonModel>> {
  public:
    NuclearInteraction(TNucleonModel& model, TEnvironment const& env)
        : NuclearInteractionModel<TEnvironment, TNucleonModel>(model, env) {}
  };

} // namespace corsika::sibyll
```

Model namespace

interaction in cascade

Actual model interface

C8 discrete step process in cascade

A composite model: sibyll nuclear interactions

```cpp
/**
 * @file Sibyll.hpp
 *
 * Includes all the parts of the Sibyll model. Defines the InteractionProcess<TModel>
 * classes needed for the ProcessSequence.
 */

namespace corsika::sibyll {
  /**
   * @brief sibyll::Interaction is the process for ProcessSequence.
   *
   * The sibyll::InteractionModel is wrapped as an InteractionProcess here in order
   * to provide all the functions for ProcessSequence.
   */
  class Interaction : public InteractionModel, public InteractionProcess<Interaction> {};

  /**
   * @brief sibyll::NuclearInteraction is the process for ProcessSequence.
   *
   * The sibyll::NuclearInteractionModel is wrapped as an InteractionProcess here in order
   * to provide all the functions for ProcessSequence.
   */
  template <class TEnvironment, class TNucleonModel>
  class NuclearInteraction
      : public NuclearInteractionModel<TEnvironment, TNucleonModel>,
        public InteractionProcess<NuclearInteraction<TEnvironment, TNucleonModel>> {
  public:
    NuclearInteraction(TNucleonModel& model, TEnvironment const& env)
        : NuclearInteractionModel<TEnvironment, TNucleonModel>(model, env) {}
  };

} // namespace corsika::sibyll
```

Template parameter for nucleon model

The nuclear model