CBM: PFSimple: configurable package for decays reconstruction

Oleksii Lubynets (GSI, Frankfurt University)
Ilya Selyuzhenkov (GSI)

Meeting of developers and users of the KF code
October 13, 2023

Introduction to Particle Finder Simple (PFSimple)

PFSimple is a physics analysis-oriented (offline) package for decays reconstruction

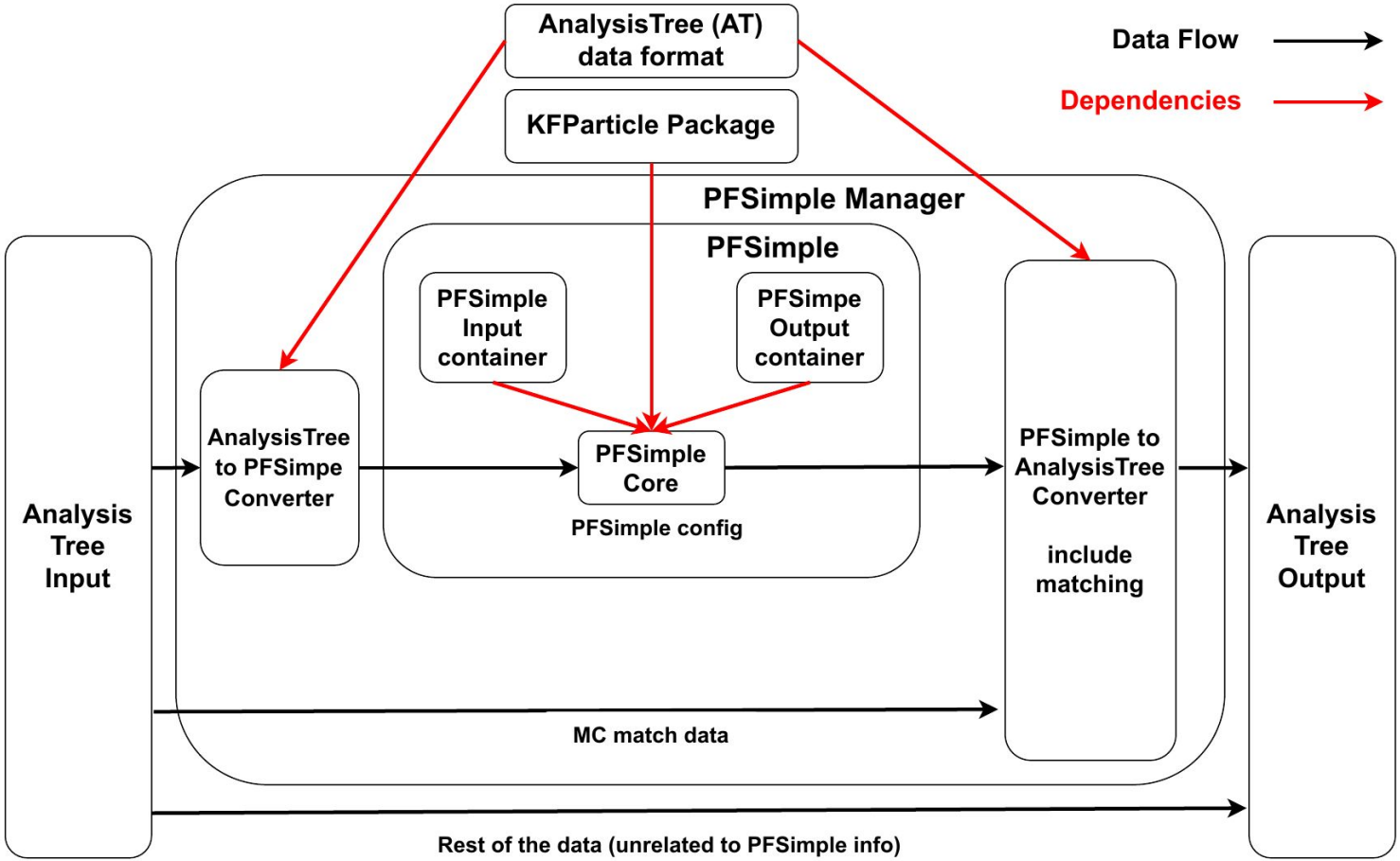
- Uses Kalman Filter mathematics implemented in KFParticle package
  (tracks extrapolation, construction decay candidates from daughter particles, etc.)
- Modular code design:
  - KFParticle (external package)
  - Internal I/O data containers
  - External I/O interface, currently for AnalysisTree (converters to AT for CBM, HADES, NA61, STAR)
  - Core of PFSimple (building decay candidates)
- Flexible configuration by user: each decay channel can be configured individually

Current implementation includes 2-body, 3-body and cascade decays

- Topological variables (χ^2_{prim}, χ^2_{geo}, χ^2_{topo}, L/ΔL, DCA, etc.) are streamed into the output container for further analysis (e.g. save to file, fill histograms, optimize selection criteria with ML)

PFSimple source code and instructions available at github.com/HeavyIonAnalysis/PFSimple
AnalysisTree is a universal, standalone physics analysis-friendly data format: github.com/HeavyIonAnalysis/AnalysisTree

PFSimple modularity



PFSimple configuration example

PFSimple configuration (definition of decays and selection criteria) implemented via compilable C++ code

(possible in future to use YAML/JSON configuration)

Available configurations: 2-body Λ, K^0_S (2-body), hypernuclei (3-body) and Ξ^-, Ω^- (cascades).

Example: $\Lambda \rightarrow p\pi^-$

```
const int pid_mode = 1; // MC-PID

Daughter proton(2212);
Daughter pion(-211);
Mother lambda(3122);

proton.SetCutChi2Prim(3.);  // Daughter-related cuts
pion.SetCutChi2Prim(3.);

lambda.SetCutChi2Geo(3);    // Mother-related cuts
lambda.SetCutDistance(1);
lambda.SetCutLdL(5);

//   proton.CancelCuts();   // Alternative: switch
//   pion.CancelCuts();     // off all the cuts
//   lambda.CancelCuts();

Decay lambda_pi_p("lambda", lambda, {pion, proton});
```

PFSimple configuration example

PFSimple configuration (definition of decays and selection criteria) implemented via compilable C++ code (possible in future to use YAML/JSON configuration)

Available configurations: 2-body Λ, K^0_S (2-body), hypernuclei (3-body) and Ξ^-, Ω^- (cascades).

Example: $\Lambda \to p\pi^-$

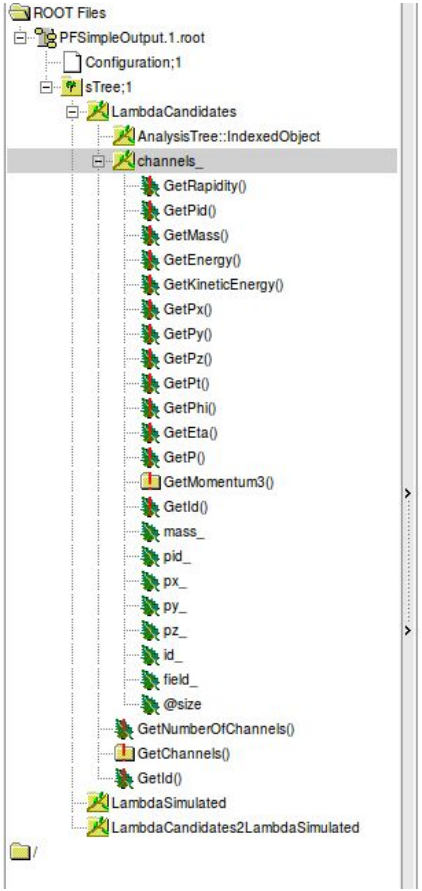
```
const int pid_mode = 1; // MC-PID

Daughter proton(2212);
Daughter pion(-211);
Mother lambda(3122);

proton.SetCutChi2Prim(3.);  // Daughter-related cuts
pion.SetCutChi2Prim(3.);

lambda.SetCutChi2Geo(3);    // Mother-related cuts
lambda.SetCutDistance(1);
lambda.SetCutLdL(5);

//   proton.CancelCuts();   // Alternative: switch
//   pion.CancelCuts();     // off all the cuts
//   lambda.CancelCuts();

Decay lambda_pi_p("lambda", lambda, {pion, proton});
```

$K^0_S \to \pi^+\pi^-$

```
const int pid_mode = 1; // MC-PID

Daughter pion_pos(211);
Daughter pion_neg(-211);
Mother kshort(310);

pion_pos.SetCutChi2Prim(3.);  // Daughter-related cuts
pion_neg.SetCutChi2Prim(3.);

kshort.SetCutChi2Geo(3);    // Mother-related cuts
kshort.SetCutDistance(1);
kshort.SetCutLdL(5);

//   pion_pos.CancelCuts();   // Alternative: switch
//   pion_neg.CancelCuts();     // off all the cuts
//   kshort.CancelCuts();

Decay kshort_pi_p("kshort", kshort, {pion_neg, pion_pos});
```

Other decays can be easily configured, e.g. $D^0 \to \pi^+ K^-$

PFSimple candidates in AnalysisTree container




```
user@user-X55VD:~/cbmdir/working/pfsimple_merge$
 root -l PFSimpleOutput.1.root
root [0]
Attaching file PFSimpleOutput.1.root as _file0..
.
(TFile *) 0x1bcd190
root [1] Configuration->Print()
This is a
The Tree has the following branches:

Branch LambdaCandidates (id=0) consists of:
  floating fields:
   chi2geo (id=5)
   chi2primneg (id=1)
   chi2primpos (id=0)
   chi2topo (id=9)
   cosineneg (id=4)
   cosinepos (id=3)
   cosinetopo (id=8)
   distance (id=2)
   eta (id=-6)
   l (id=6)
   ldl (id=7)
   mass (id=-5)
   p (id=-10)
   pT (id=-2)
   phi (id=-1)
   px (id=-7)
   pxerr (id=13)
   py (id=-8)
   pyerr (id=14)
   pz (id=-9)
   pzerr (id=15)
   rapidity (id=-3)
   x (id=10)
   y (id=11)
   z (id=12)
  integer fields:
   daughter1id (id=3)
   daughter2id (id=4)
   is_signal (id=5)
   isfrompv (id=0)
   nhitsneg (id=2)
   nhitspos (id=1)
   pid (id=-4)
  boolean fields:
```

- Decay kinematics*
  - energy and 3-momentum components
  - inv. mass
- Decay vertex coordinates*

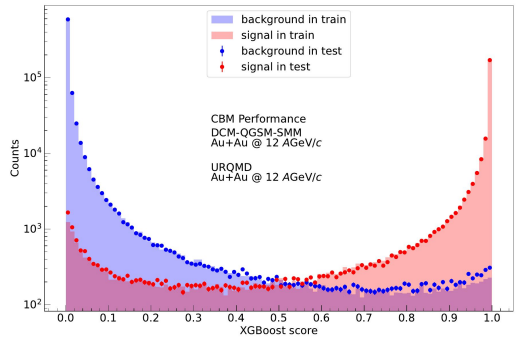
*) Uncertainties of all parameters are provided

- Topological variables
  - X^2_{prim}, X^2_{geo} , X^2_{topo}
  - DCA
  - L/ΔL
  - …
- (If available) MC-matching for mother and daughter particles

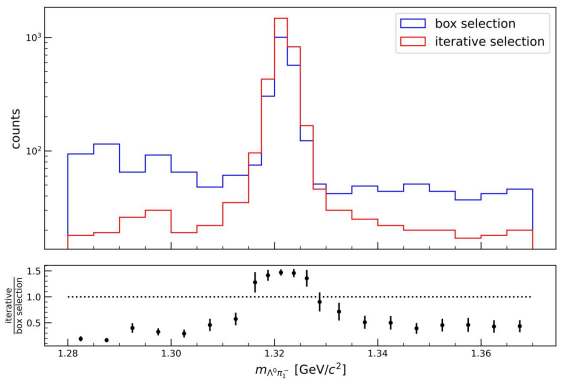
PFSimple application at CBM

(Multi)strange hadrons production: ML selection optimization

S. Khan, PhD thesis @ Uni Tübingen



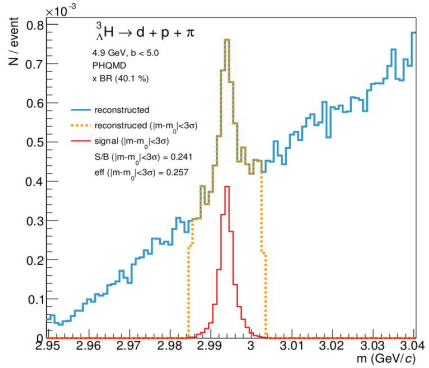
L.-K. Kümmerer, BA thesis @ Uni Heidelberg

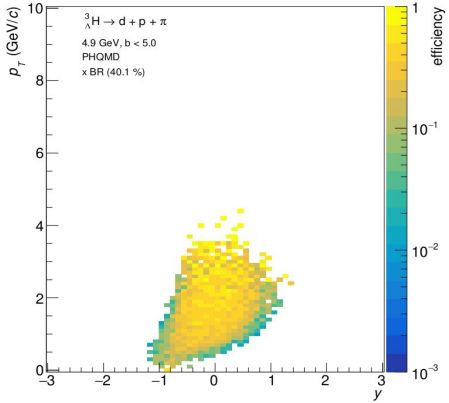


(**top**) XGBoost score for signal and background;
(**bottom**) comparison between ML-optimized and box selection

Hypernuclei production

S. Gläßel, C. Blume, CBM Progress report 2022

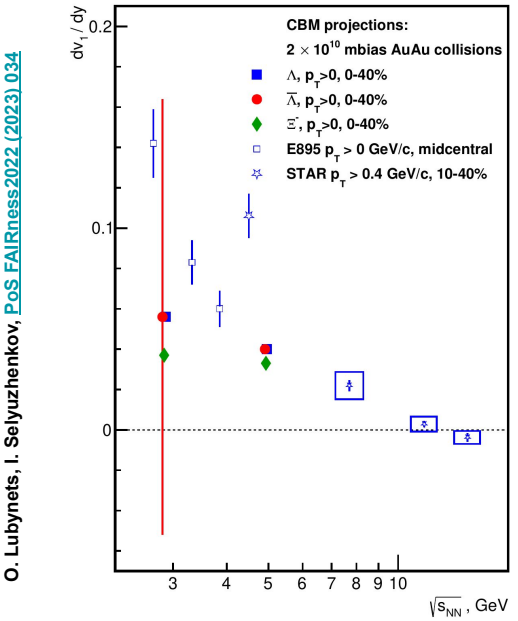




(**top**) Hypertriton invariant mass distribution
(**bottom**) (p_T, y) reconstruction efficiency

Collective flow of strange hadrons

O. Lubynets, I. Selyuzhenkov, PoS FAIRness2022 (2023) 034



Performance for strange hadrons dv_1/dy measurement

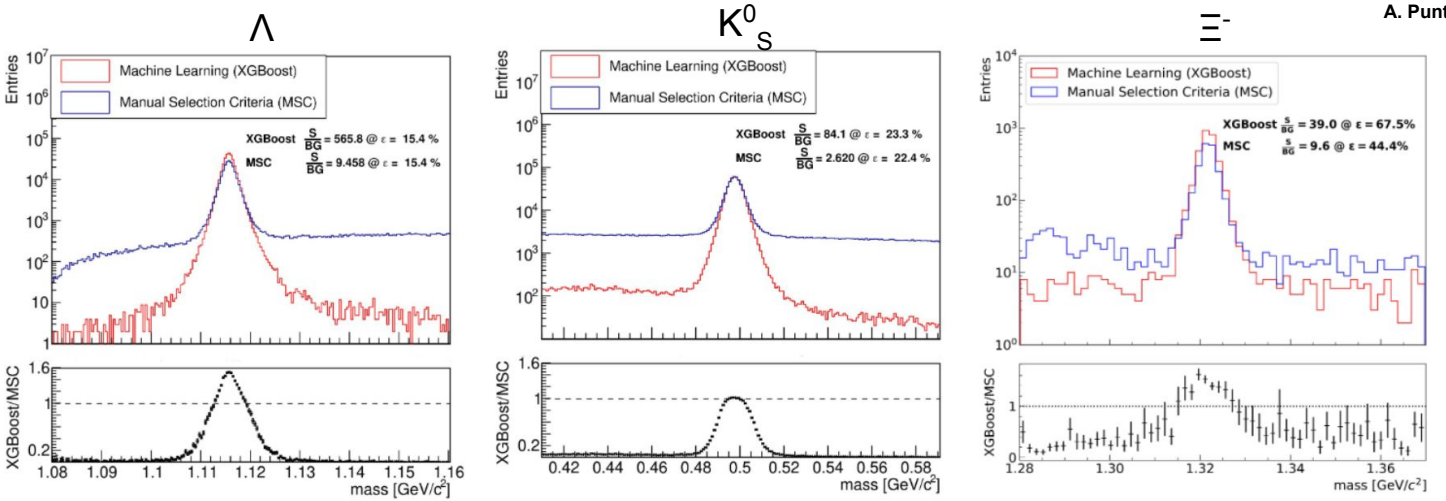
(Multi-)strange hadron analysis with ML

- Boosted Decision Tree (BDT) algorithm - supervised ML
- Improved iterative procedure: 2 BDT models are trained and applied in sequence
  - 1-st model is trained on full (BG dominated) sample: used to suppress overall background without signal loss
  - 2-nd model is trained on reduced BG sample: optimized signal/BG separation
  - 2-step approach provides better performance than a single BDT model

More detailed in BA thesis of L.-K. Kümmerer

BDT vs. box selection

A. Puntke et al., QM2023 poster



BDT selection shows better performance for both background rejection and signal efficiency compared to the box selection