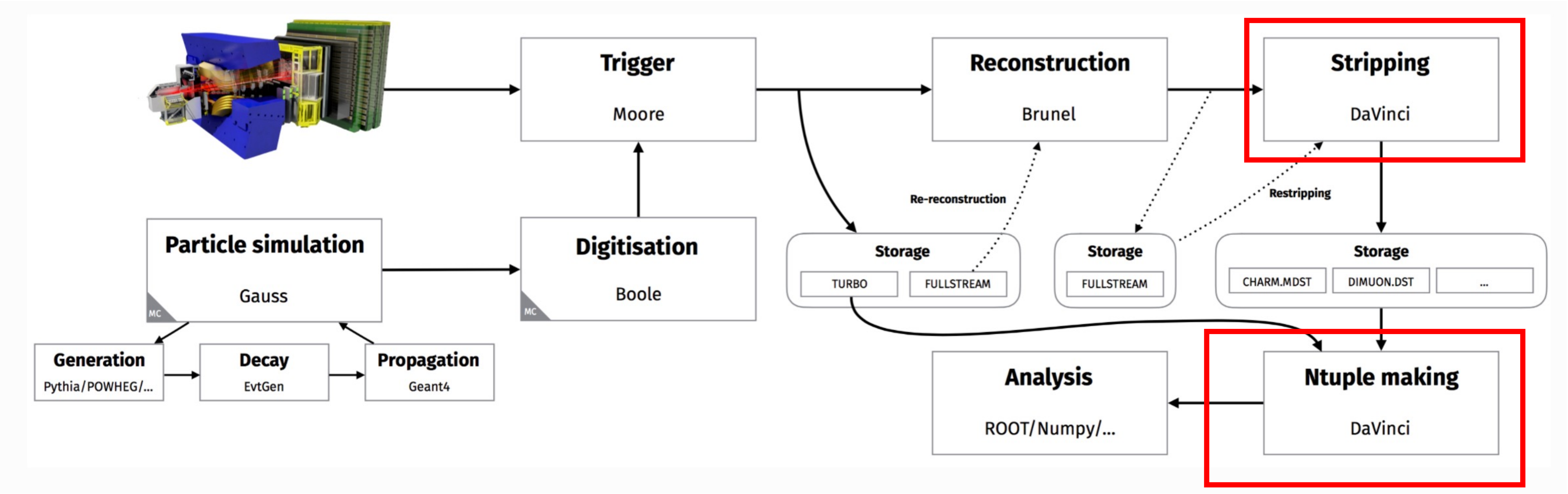
Starterkit tutorial with new DaVinci

Abhijit Mathad, Davide Fazzini and Martina Ferrillo
On behalf of DPA WP3

LHCb StarterKit-For-All Full Run 3 Edition

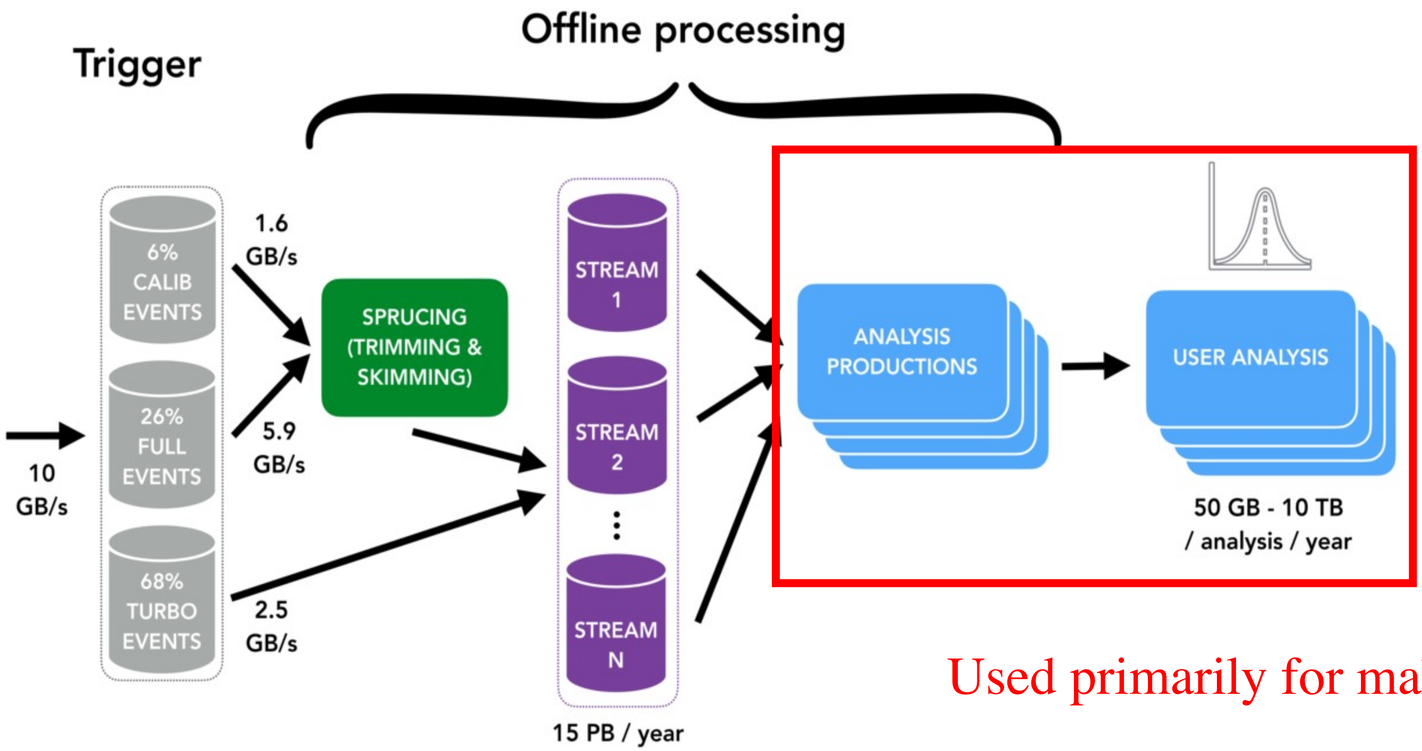
14 March 2022

DaVinci in Run1 and Run2



Used in selection and making nTuples

DaVinci in Run3



Used primarily for making nTuples

- Before we dive in, what algorithms and application existed in Run1/2 for making nTuples?
  - Main ones include: DecayTreeTuple and Bender (less known/popular).

Run1/2: DecayTreeTuple (DTT)

Got loaded with TupleTools that wrote a set of variables to nTuples.

```
tupleB         = DecayTreeTuple("tupleout")
tupleB.Inputs = ["Phys/b2LcMuXB2DMuForTauMuLine/Particles"]
tupleB.Decay  = "[B- -> ^(Lambda_c+ -> ^p+ ^K- ^pi+) ^mu-]CC"
tupleB.addBranches({"Lb" : "[B- -> (Lambda_c+ -> p+ K- pi+) mu-]CC",})
tupleB.ToolList =  [
        "TupleToolKinematic"
      , "TupleToolMCTruth"
      , "TupleToolEventInfo"
]
```

Users had little control over whats written to nTuples at “RunTime” and does not scale well with increase in data.

Could also load it with LoKi functors as strings.

```
LoKi_B=tupleB.Lb.addTupleTool("LoKi::Hybrid::TupleTool/LoKi_B")
LoKi_B.Variables = {
        'DOCAMAX'         : "DOCAMAX",
        'TAU'             : "BPVLTIME()",
        'DIRA_OWNPV'      : "BPVDIRA",
        'FD_CHI2'         : "BPVVDCHI2",
        'ENDVERTEX_CHI2'  : "VFASPF(VCHI2/VDOF)",
        'Corrected_Mass'  : "BPVCORRM"
}
```

New Run 3 event model would break LoKi functors.

Run1/2: Bender

Users wrote their own algorithms and made nTuples.
Lot of flexibility: LoKi functors could be *directly* imported into python and operated on Particles (ThOr functors cannot do this [yet]).

```python
from Bender.Main import *
from Bender.MainMC import *
import math
from ROOT import Double, TLorentzVector, TVector3, TRotation, TLorentzRotation, TMath
from Xib2pKK.ThreeBodyKinematics import ThreeBodyKinematics

# Begin Xib2pKKReco Class Definition
class Xib2pKKReco(AlgoMC) :
    """
    Algorithm to perform ntupling (and potentially offline selection) for
    Xib2pKK analyses.
    Reads stripped candidates from the DST and stores useful info in an ntuple.
    """
    def __init__( self, name, simulation = False, signalmc = False, mc_daughters = [],
        super(Xib2pKKReco,self).__init__( name, **kwargs )
```

New Run 3 event model would break LoKi functors.

Run 3: FunTuple

FunTuple is a Gaudi Functional nTuple algorithm developed in C++

```
#import FunTuple
from FunTuple import FunTuple_Particles as Funtuple
#Define an instance of FunTuple
my_tuple = Funtuple(
    name="Tuple",
    tuple_name="DecayTree",
    fields=fields,
    variables=variables,
    inputs=d02kpi_data)
```

Name of Algorithm name or Tdirectory name

Name of the TTree in the ROOT file

Different decay branches

Variables (ThOr or LoKi functors) to be added to each decay branch.

Input data from Sprucing line.

How to get and run DaVinci?

Before we dive into FunTuple, how can I setup and run the new DaVinci?

Onto Davide Fazzini…

Link to the examples gone through are here.

Example0: Basic FunTuple example with LoKi and ThOr functors

The objectives of this example include:

- Basic configuration of `FunTuple`.
- Defining a collection of `LoKi` and `ThOr` functors (`FunctorCollection`).
- Inspecting C++ string representation of `ThOr` functors.
- Loading particles in the event from `.dst` onto Transient Event Store (TES) location.
- Usage of event filter (mainly to get over a technical hurdle).

Onto the live coding session…

For hand-on later:

- Try to plot the variables on which you have applied cuts, do they match your cuts!

Link to the examples gone through are here.

Example1: Decay tree components, special field name and LoKi preamble

The objectives of this example include:

- Configuring the `fields` attribute of `FunTuple`: Map between user defined field name and different decay tree components.
- Usage of special field name `ALL`.
- Defining a LoKi preamble for a complex LoKi functor to be used in `FunctorCollection`.
- Overiding the options set in `jobopts.yaml` and `dataprops.yaml`.

Onto the live coding session…

For hand-on later [$B_c^+ \rightarrow B_s^0\pi^+$ with $B_s^0 \rightarrow J/\psi(\rightarrow \mu^+\mu^-)\phi(\rightarrow K^+K^-)$]:

- Use LoKi preamble feature and find sum of PT of all basic particles in the decay tree with LoKi (use SUMTREE) adding to B_c^+ *field* name.
- Add to B_c^+ *field* name the ThOr functor F.SUM(F.PT).
- Do they match? Why not?

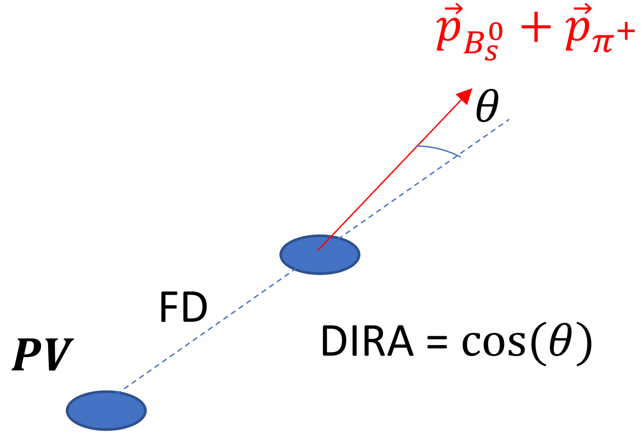
Link to the examples gone through are here.

Example2: ThOr functors for nTupling: Data dependence, arguments and various return types

The objectives of this example include:

- Loading primary vertices (PVs) onto TES, which is passed to data dependent `ThOr` functors in `FunTuple` e.g. `F.BPVIPCHI2(pvs)`.
- Functors returning three and four vectors e.g. `F.BPVFDVEC(pvs)`.
- Usage of functors taking other various arguments such as other functors ents e.g. `F.CHILD(1,func)`, `F.SUM(func)`, `F.SUMCOMB(func,Indices)`, `F.MASSWITHHYPOTHESES(('K+', 'K-'))`.
- Basic math with functor returning scalars e.g. `CHILD_2(F.END_VZ) - F.END_VZ`.

Onto the code…



For hand-on later [$B_c^+ \to B_s^0\pi^+$ with $B_s^0 \to J/\psi(\to \mu^+\mu^-)\phi(\to K^+K^-)$]:

- Plot flight distance (FD) and DIRA wrt best PV for B_c^+ and B_s^0.
- Calculate maximum PT of ϕ daughter using F.CHILD functor starting from B_c^+.

Link to the examples gone through are here.

Example3: Usage of pre-defined functorcollections, storing trigger and event-level information

The objectives of this example include:

- Usage of pre-defined `functorcollections`, inspecting and manipulating them before loading it onto `FunTuple`.
- Exploring few simple methods of `FunctorCollection` class.
- Storing event-level information with `functorcollection` e.g. RunNumber, EventNumber, etc.
- Storing trigger (`Hlt2` and `Sprucing`) information with `functorcollection` e.g. line decisions, Trigger Configuration Key (TCK) (SideNote: Storing decisions of Hlt1 not yet possible as `Hlt1DecReport` decoder not yet available).

Onto the code...

For hand-on later [$B_c^+ \to B_s^0 \pi^+$ with $B_s^0 \to J/\psi(\to \mu^+\mu^-)\phi(\to K^+K^-)$]:

- Make your own functorcollection that replaces TupleToolGeometry with ThOr functors (Essentially a function returning FunctorCollection object).
- When you are done send in a merge request! 😁

Link to the examples gone through are here.

Example3: Usage of pre-defined functorcollections, storing trigger and event-level information

The objectives of this example include:

- Usage of pre-defined `functorcollections`, inspecting and manipulating them before loading it onto `FunTuple`.

INPUT are particles from Hlt2Starterkit_Bs0ToKmKpMumMup_SP_Line. Trigger line decisions below:

```
(bool) true
root [2] TTreeName->GetEntries("Hlt2_Hlt2Starterkit_Bs0ToKmKpMumMup_SP_LineDecision==1")
(long long) 1393
root [3] TTreeName->GetEntries("Hlt2_Hlt2Starterkit_Bs0ToJpsiPhi_PR_LineDecision==1")
(long long) 1227
```

For hand-on later [$B_c^+ \to B_s^0\pi^+$ with $B_s^0 \to J/\psi(\to \mu^+\mu^-)\phi(\to K^+K^-)$]:

- Make your own functorcollection that replaces TupleToolGeometry with ThOr functors (Essentially a function returning FunctorCollection object).
- When you are done send in a merge request! 😁

Link to the examples gone through are here.

Example4: MC truth association and background category algorithm

The objectives of this example include:

- Configuring the MC association and background category algorithm (`MCTruthAndBkgCatAlg`) to build a relation table. For MC association, the table is essentialy a map between reconstructed particle and "truth" particle (MCParticle).
- Usage of the relations table and `ThOr` functor handling such table (e.g. `F.MAP_INPUT(func, RelTable`) to get truth information and background category.
- Also explore `functorcollections` such as `MCKinematics`, `MCHeirarchy`, etc.

Onto the code…

For hand-on later [$B_c^+ \to B_s^0 \pi^+$ with $B_s^0 \to J/\psi(\to \mu^+\mu^-)\phi(\to K^+K^-)$]:

- Plot the m($B_s^0\pi^+$) with and without truth-matched candidates.

Link to the examples gone through are here.

Example5: Decay Tree Fitter (DTF) algorithm

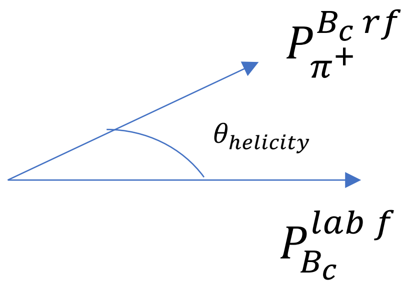
The objectives of this example include:

- Configuring the Decay Tree Fitter algorithm (`DecayTreeFitterAlg`) to build a relation table i.e. map between B candidate and refitted B candidate.
- Usage of the relations table and `ThOr` functor (e.g. `F.MAP_INPUT(func, RelTable`) to get refitted information of the candidate.
- Defining different instances of DTF algorithm with mass constraints, primary vertex constraint (Note: The primary vertex constraint expects `v1` vertices, but in this sample only `v2` available (?). Also defining DTF instance with different mass hypothesis substitution is WIP.)

Onto the code…

For hand-on later [$B_c^+ \to B_s^0\pi^+$ with $B_s^0 \to J/\psi(\to \mu^+\mu^-)\phi(\to K^+K^-)$]:

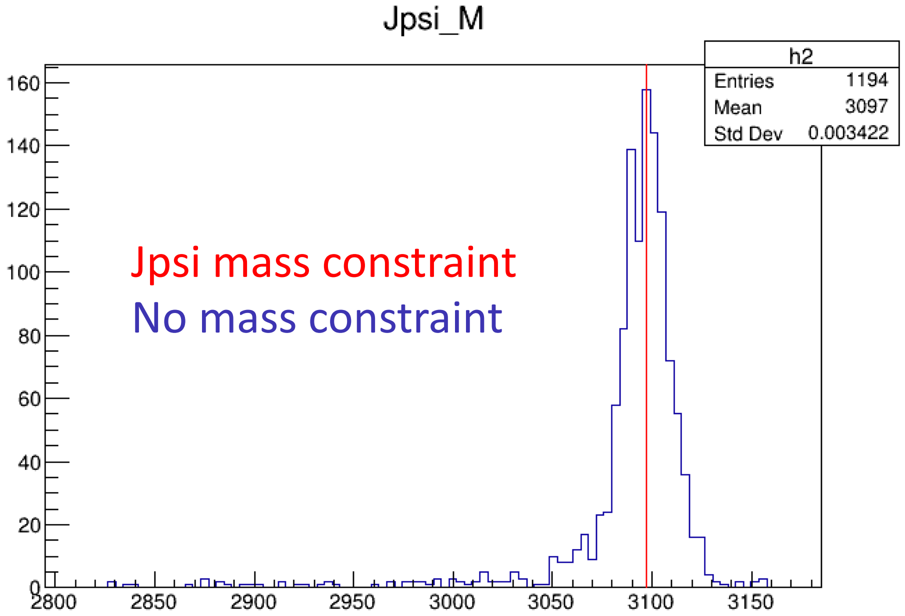
- Compare helicity angle of pion with and w/o Bc mass constraint.



Link to the examples gone through are here.

Example5: Decay Tree Fitter (DTF) algorithm

The objectives of this example include:



terAlg)

INPUT(f

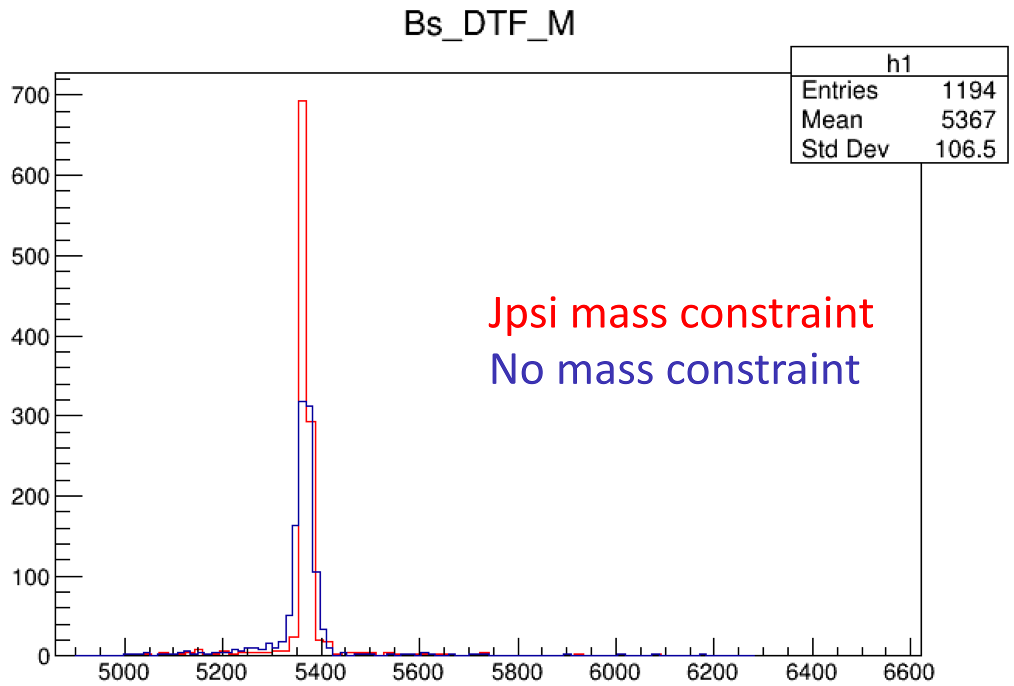
straints,

). Also c

he cc

Fo

$J/\psi($



f

- Compare helicity angle of pion with and w/o Bc mass constraint.

$P_{B_c}^{lab\ f}$

Link to the examples gone through are here.

Example6: Defining different instances of FunTuple when analysing outputs of multiple selection lines

The objectives of this example include:

- Defining different instances of FunTuple to return different `TDirectory` in the output root file.
- Changes to the function returning user algorithm "sequence" to allow for this.

Onto the code…

For hand-on later [$B_c^+ \to B_s^0 \pi^+$ with $B_s^0 \to J/\psi(\to \mu^+\mu^-)\phi(\to K^+K^-)$]:

- Write one ROOT file with two TDirectories one corresponding to each of the spruced line you wrote.

Link to the examples gone through are here.

ThOr functors I

In the beginning, you will find that the required ThOr functor for your information is missing. You may have to write them yourself!

List of all available ThOr functors can be found here.
List of required ThOr functors can be found here.

What are they?

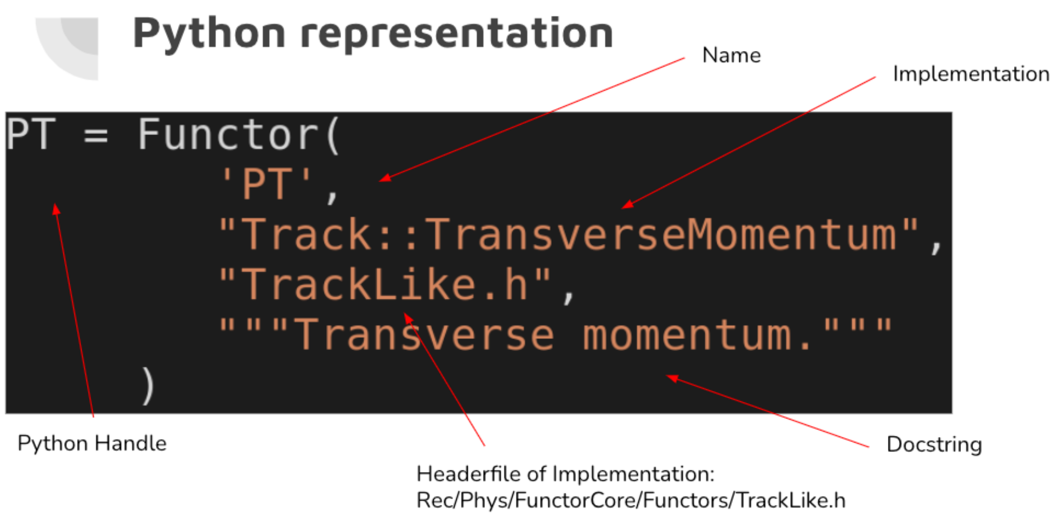
ThOr functors II

See a dedicated talk by Niklas in hackathon and checkout the documentation by Alex.

Link

- Python representations of Functors in Rec/Phys/FunctorCore/python/Functors/__init__.py
- C++ implementations in Rec/Phys/FunctorCore/Functors/*.h

Link



ThOr functors III

See a dedicated [talk](#) by Niklas in hackathon and checkout the [documentation](#) by Alex.

[Link](#)

- Python representations of Functors in Rec/Phys/FunctorCore/python/Functors/__init__.py
- C++ implementations in Rec/Phys/FunctorCore/Functors/*.h

[Link](#)

C++ implementation

either **Function** or **Predicate**:
return data or bool?

template type:
LHCb::Particle,
v2::Particle,
LHCb::Track,
SoA Track

branch impl.
depending on
API

```cpp
struct TransverseMomentum : public Function {
  template <typename Data>
  auto operator()( Data const& d ) const {
    if constexpr ( Sel::Utils::has_tracklike_API<Data> )
      return d.pt( Sel::Utils::get_kinematic_state( d ) );
    else
      return Sel::Utils::deref_if_ptr( d ).pt();
  }
};
```

Helper functions extracting the data..
Make sure your code is agnostic of vector width, if the data structure supports vectorization (SoA Track, v2::Particle)

There are also data dependent wrapper and adapter.
Checkout Niklas talk. If you want to write one get it touch.

Handling various functor output types in FunTuple

- On top of basic C++ types, FunTuple can handle various output types of ThOr functors (e.g. 3-vectors, 4-vectors, SymMatrix, std::vector, std::map<std::string, std::any>) , etc).

For hand-on later:

- Write a functor to store rapidity of the particle.
- And send in an MR! 

Where do I find examples?

- The examples covered in the starter kit can be found here.

- Currently examples live in DaVinciExamples (tests in DaVinciTests).
- Specific examples:
  - Running over HLT2 output, see example.
  - Running over Sprucing output, see example.
  - Running with MCParticles, see example.
  - Using MC association, see example.
  - Usage of Decay tree fitting, see example.
  - Tests of available ThOr functors, see example.
  - Job configuration the standard way (gaudirun.py), see example.

Summary and hands-on

- Presented various features of new DaVinci configuration and FunTuple.
- For documentation see here (will be revised).
- Not everything is set in stone yet. Expect things to change.
- If in doubt ask them on [Mattermost]!

- For hands-on try to take the sprucing output of Bc-> Bs0 pi+ and try to reproduce the examples gone through here.


ANY QUESTIONS?
makeameme.org