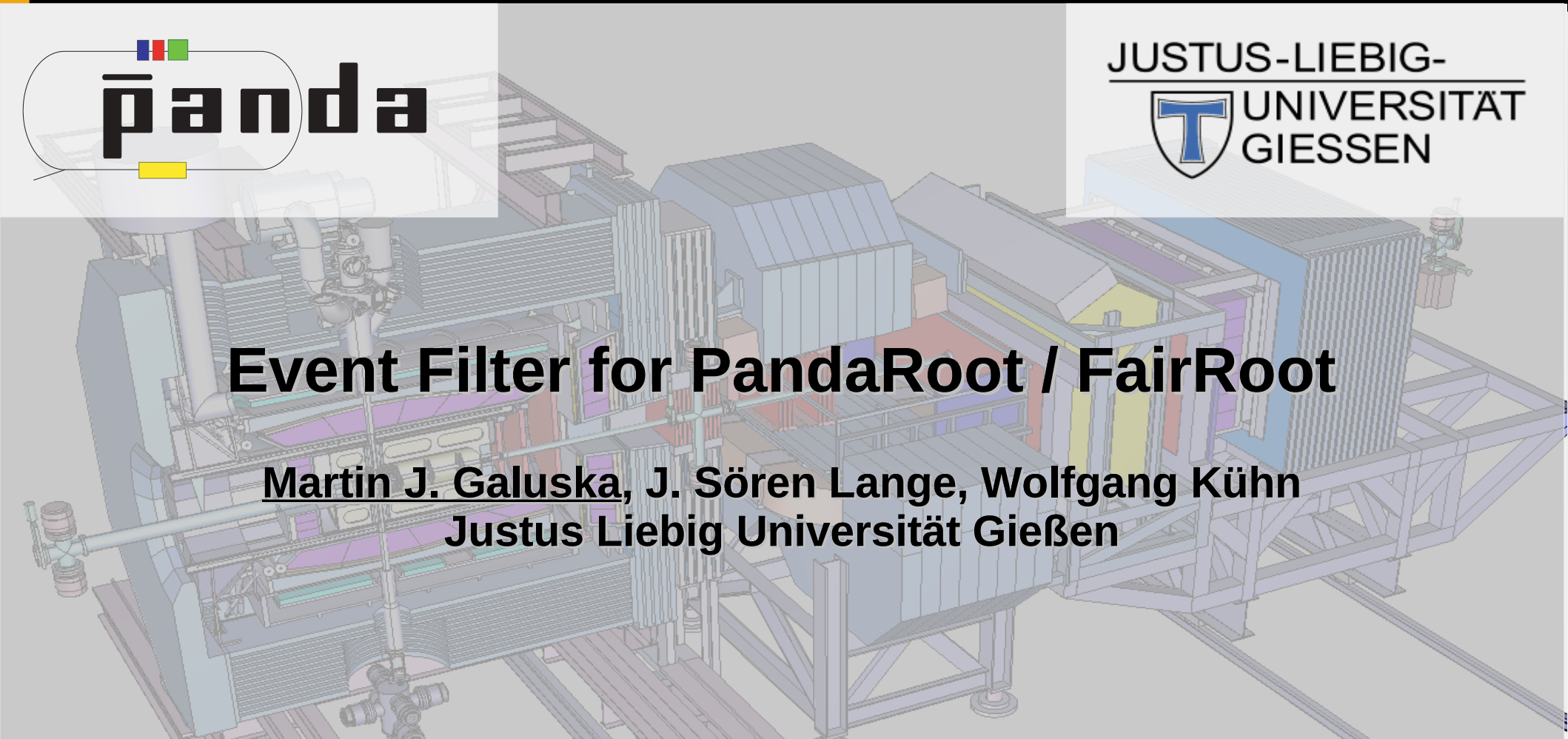
XLVII PANDA Collaboration Meeting (GSI)



Event Filter for PandaRoot / FairRoot

Martin J. Galuska, J. Sören Lange, Wolfgang Kühn
Justus Liebig Universität Gießen

This work was supported in part by BMBF (06GI9107I), HGS-HIRe for FAIR and the LOEWE-Zentrum HICforFAIR.

HIC for FAIR
Helmholtz International Center

HGS-HIRe for FAIR
Helmholtz Graduate School for Hadron and Ion Research

Bundesministerium für Bildung und Forschung

Motivation / History

- Our group wanted to run a **full PandaRoot simulation** of **specific rare background events** which should be produced with PndDpmDirect.
- I started writing a pretty flexible **filter for PndDpmDirect**.
- Stefano suggested that such a filter could be **useful also for other event generators** and that we should think about **implementing it inside FairPrimaryGenerator**.
- Discussion on PandaRoot Forum:
  http://forum.gsi.de/index.php?t=msg&goto=15374&rid=1493&S=ad3eb0b32fc82b24f8fc5ea7cd91adae#msg_15374

  (Maybe we will be able to access it some day again from outside of GSI...)

Desirable Features of an Event Filter

- Filter events produced by **any event generator**...
  - Implementation in FairPrimaryGenerator (or derived task)
- … based on
  - **User specified PDG codes** (+ momentum or geometrical requirements)
  - **Charge** (+ momentum or geometrical requirements)
  - **Invariant masses**
  - **Vertices**
- Accept only events of the desired multiplicities (min. and max.)
- Inform user how many events were produced by the generator to reach the number of accepted events.
- Avoid infinite loops
  - If no acceptable event can be found (in a user-defined amount of tries), accept the latest ”random” event and warn the user

Example: Imagine that...

- ... we want to generate events using PndDpmDirect
- … we want to run the full PandaRoot simulation for very specific events
- … time is short
- … we are only interested in events that have:
  - at least 2 e- OR pi-
    - with a p_T within [1.0, 3.0] GeV/c
    - and with a p within [1.5, 3.0] GeV/c
  - at most 4 e+ OR pi+
    - with a p_z within [0.5, 4.0] GeV/c
  - at least 4 and at most 6 pos. Charged particles
    - with theta within $[20.0, 90.0]^0$
    - and phi within $[10.0, 80.0]^0$
  - at least 3 and at most 10 gamma
    - With E>2 GeV

Example (Assuming We Already Had Such a Filter)

- In sim macro:

```
FairPrimaryGenerator* primGen = new FairPrimaryGenerator();
primGen->SetEventMeanTime(10);
fRun->SetGenerator(primGen);

// here you put the generator that you want to use, let's say DpmDirect
PndDpmDirect *Dpm= new PndDpmDirect(mom,1);
primGen->AddGenerator(Dpm);

// The standard behaviour is the same as before (i.e. no event filtering)

// now add some filters on multiplicities of charged/neutral particles or pdg codes within certain momentum regions
primGen->AddFilterMin( 2, 11, -211 ); // request at least 2 e- OR pi-
primGen->SetPtRange(1.0, 3.0); // with a p_T within [1.0, 3.0] GeV/c
primGen->SetPRange(1.5, 3.0); // and with a p within [1.5, 3.0] GeV/c

primGen->AddFilterMax( 4, -11, 211 ); // request at most 4 e+ OR pi+
primGen->SetPzRange(0.5, 4.0); // with a p_z within [0.5, 4.0] GeV/c

primGen->AddFilterMinMax(4,6,"PLUS"); // request at least 4 and at most 6 pos. Charged particles
primGen->SetThetaRange(20.0, 90.0); // with theta within [20.0, 90.0]^0
primGen->SetPhiRange(10.0,80.0); // and phi within [10.0, 80.0]^0

primGen->AddFilterMinMax( 3, 10, 22 ); // request at least 3 and at most 10 gamma
primGen->SetPRange(2.0, 9999.0); // and with a p within [2.0, 9999.0] GeV/c
```

Setting Parameters and Getting Info

- You can tell PndDpmDirect how often it should try to find a suitable event before giving up:
```
primGen->SetFilterMaxTries(99999);
```
- and the filter can tell you at the end of the simulation macro how many events the generator simulated (in total) to get the number of filtered events that you wanted...
```
  cout << primGen->GetNumberOfSimulatedEvents() << " events were simulated in dpm\n";
```
- … as well as how many events reached the limit of tries without success:
```
  // should be 0 if filter is applicable and SetFilterMaxTries is not too low
  cout << primGen->GetNumberOfFilterFailedEvents() << " unsuccessful attempts to find an event that suits your filters\n\n";
```
- Note: The number of generator runs and the number of failed filterings will also be written into the root output file.

First Implementation in PndDpmDirect

I have written a fully working test version of the filter
(implemented in PndDpmDirect)
which is uploaded at:

https://subversion.gsi.de/trac/fairroot/browser/pandaroot/development/mgaluska/PndDpmDirectWithFilter

It does not include all features yet!

Instructions:
Just download the two files in the above folder,
replace the according files in
pgenerators
and recompile PandaRoot!

Current Implementation in PndDpmDirect – Adding Filter

- Example usage (in sim macro):

```
FairPrimaryGenerator* primGen = new FairPrimaryGenerator();
primGen->SetEventMeanTime(10);
fRun->SetGenerator(primGen);

PndDpmDirect *Dpm= new PndDpmDirect(mom,1);
primGen->AddGenerator(Dpm);

// The standard behaviour is the same as before (i.e. no event filtering)

// now add some filters on multiplicities of pdg codes
Dpm->AddFilterMinMax( 1, 5, 11, -211 ); // request at least 1 and at most 5 e- OR pi-
Dpm->AddFilterMinMax( 1, 5, -11, 211 ); // request at least 1 and at most 5 e+ OR pi+
Dpm->AddFilterMinMax( 3, 9999, 22 ); // request at least 3 and at most 9999 gamma
```

Current Implementation in PndDpmDirect – MaxTries

- You can tell PndDpmDirect how often it should try to find a suitable event before giving up:
  Dpm->SetFilterMaxTries(99999);
- and it can tell you at the end of the simulation macro how many events dpm simulated in total to get the number of filtered events that you wanted as well as how many events reached the limit of tries without success:

```
  cout << Dpm->GetNumberOfSimulatedEvents() << " events were simulated in dpm\n";
  cout << Dpm->GetNumberOfFilterFailedEvents() << " unsuccessful attempts to find an event that suits your filters\n\n";
```

On the Filter Usage: A Word of Caution

- The event filter is a versatile, but dumb tool
- As a potential user remember that:
- The event filter **only looks at primary particles** produced by the event generator
  - It does not include secondaries
  - It does not take material effects / interaction with detector / tracking / etc. into account
- **Be careful** when you want to extract some physical meaning from an analysis of filtered events
  - In case of the DPM generator, the user should analyse ALL events from DPM with sufficient statistics and based on the results, (s)he can decide that mainly certain event topologies contribute (after cuts) to the background events for the analysed channel
  - Once such specific events were identified, the filter can be useful in saving simulation time and computer ressources

Current / Planned Features for the Event Filter

Already implemented / Missing

- Filter events based on
  - Multiplicities [min. and max.]
- Criteria
  - PDG code(s) / Charge (neutral / charged / + / –) / Invariant masses (of arbitrary PDG code combinations) / Vertices
  - Momentum (total / transversal / z) [in lab system]
  - Geometry (theta, phi, (pseudo-)rapidity) [in lab system]
  - Geometry + momenta in center of mass system [for arbitrary 4 vector]
- Any other criteria?
  - Suggestions from audience are welcome!

Summary

- The ground work is done.
  - The event filter works for PndDpmDirect.
    - Main functionality is present.
    - Many features are still missing.
  - The filter can be obtained from my svn development directory.
    https://subversion.gsi.de/trac/fairroot/browser/pandaroot/development/mgaluska/PndDpmDirectWithFilter
- Further implementation plans were discussed with the computing team, esp. S. Spataro and R. Kliemt.
  - We found a master student, Katja Kleeberg, who wants to continue my implementation and add functionality to the filter. (Note that this will not be her master thesis project. It is one module of her curriculum.)
  - I will supervise the project. R. Kliemt has offered his suggestions.

Thank You!