Millepede Alignment & Calibration

for

Phase 2

Tadeáš Bilka
Charles University, Prague

F2F Tracking Meeting, Dec 5 – 7 , Pisa

- Purpose of this talk:
  - Overview of the status of alignment readiness with focus on Phase 2
- Outline
  - Alignment/Calibration Method
  - Calibration Framework
  - Input Data
  - Calibration Constants
  - Alignment Tests

Alignment/Calibration Method

- General Broken Lines
  - Multiple scattering → additional scattering points in particle trajectory (→ kinks)
  - Global linear least squares fit minimizing all measurement residuals and kinks at once → system of linear equations (A*x = b)
- Millepede II
  - Global linear least squares fit to all track (local) and calibration parameters (global) simultaneously → huge linear system → block matrix algebra → reduced matrix for global parameters only
  - Iterate for non-linearities, all correlations kept in the solution

- Local vs. Global alignment
  - Local: Internal alignment/calibration per constant sub-set
  - Global: Simultaneous calibration of all constants
- The road towards working alignment/calibration

1) Have your DB object on which reconstruction depends, add id/getter/setter/list interface

2) In a RecoHit, calculate the derivatives of residuals w.r.t. to your constants and match them to DB object constants

3) Add your object to GlobalCalibrationManager::initGlobalVector(...)

4) Run on ideal MC (no mis-calibration)

   Alignment should return 0's (within errors)

5) Test with simple misalignment

   Check correct value is returned with correct sign

6) „Global“ misalignment test with some other existing constants

   Check correct relative sign of corrections

7) Study random & systematic misalignment

8) ...

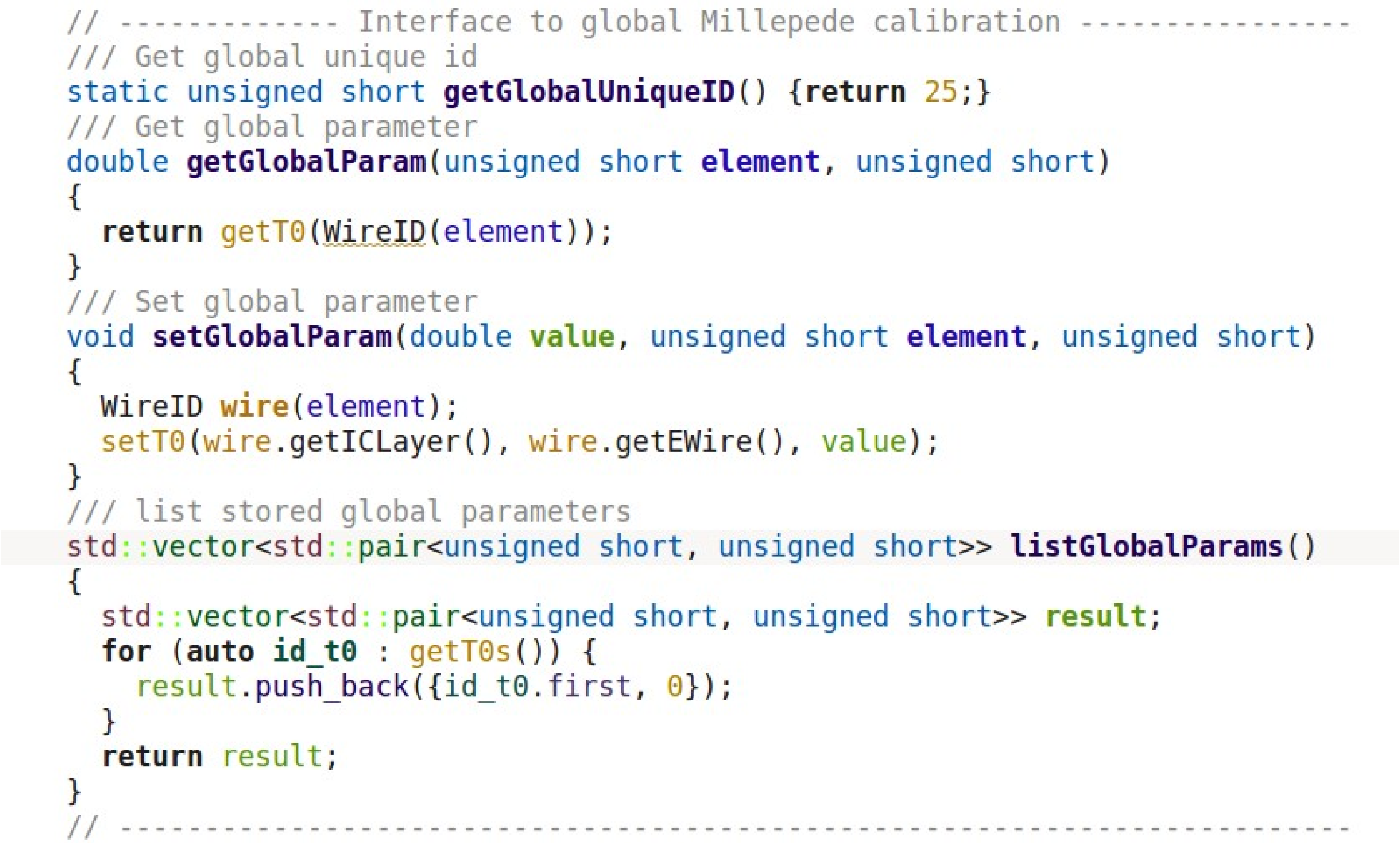
Two ways of extending alignment

- More input types
  - Cosmics, Cosmics @ B = 0 T, vertex/vertex+beam/vertex+mass/... constrained decays, ...
- More parameters
  - Sensor deformations, Lorentz shift for VXD
  - Hierarchy for CDC, KLM
  - CDC calibration

Example: CDC T0 calibration

CDCTimeZeros.cc

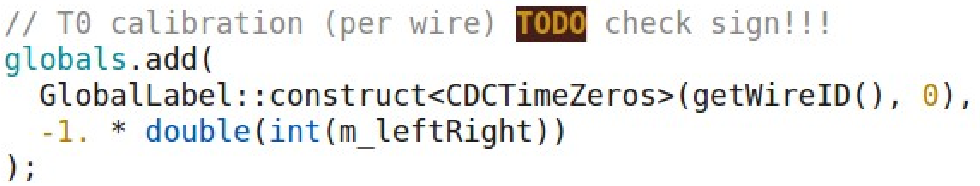
- Mapping constants to numeric labels



```
// ------------- Interface to global Millepede calibration ----------------
/// Get global unique id
static unsigned short getGlobalUniqueID() {return 25;}
/// Get global parameter
double getGlobalParam(unsigned short element, unsigned short)
{
  return getT0(WireID(element));
}
/// Set global parameter
void setGlobalParam(double value, unsigned short element, unsigned short)
{
  WireID wire(element);
  setT0(wire.getICLayer(), wire.getEWire(), value);
}
/// list stored global parameters
std::vector<std::pair<unsigned short, unsigned short>> listGlobalParams()
{
  std::vector<std::pair<unsigned short, unsigned short>> result;
  for (auto id_t0 : getT0s()) {
    result.push_back({id_t0.first, 0});
  }
  return result;
}
// ------------------------------------------------------------------------
```

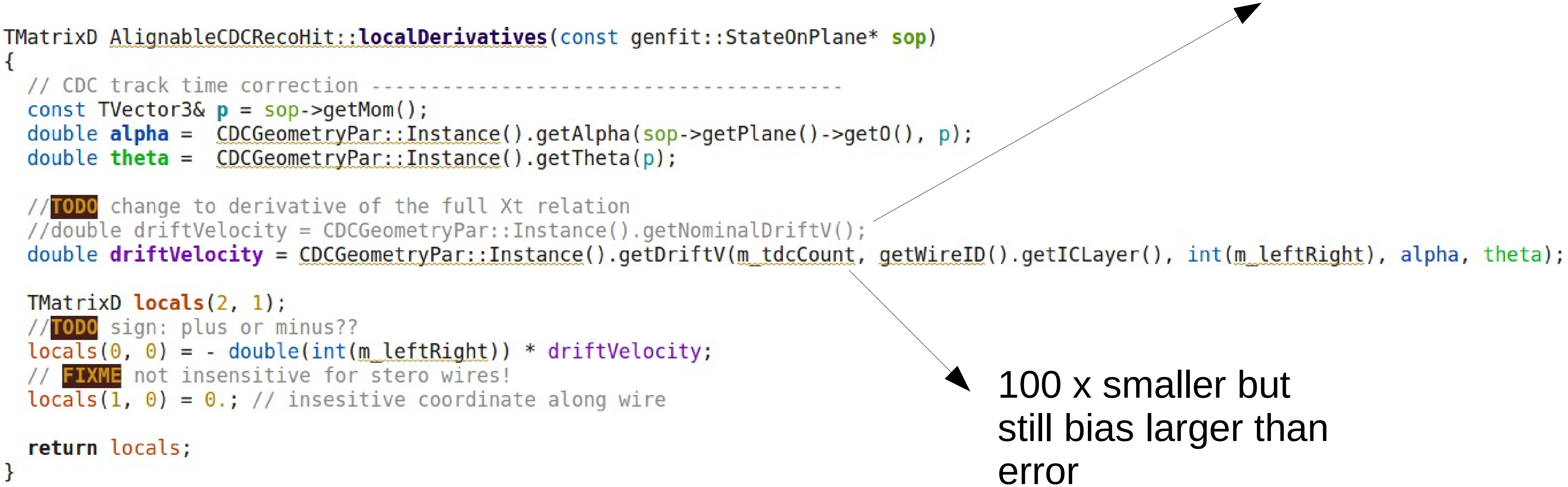
AlignableCDCRecoHit.cc

- Global derivatives



```
// T0 calibration (per wire) TODO check sign!!!
globals.add(
  GlobalLabel::construct<CDCTimeZeros>(getWireID(), 0),
  -1. * double(int(m_leftRight))
);
```

- Local derivatives (track t0)

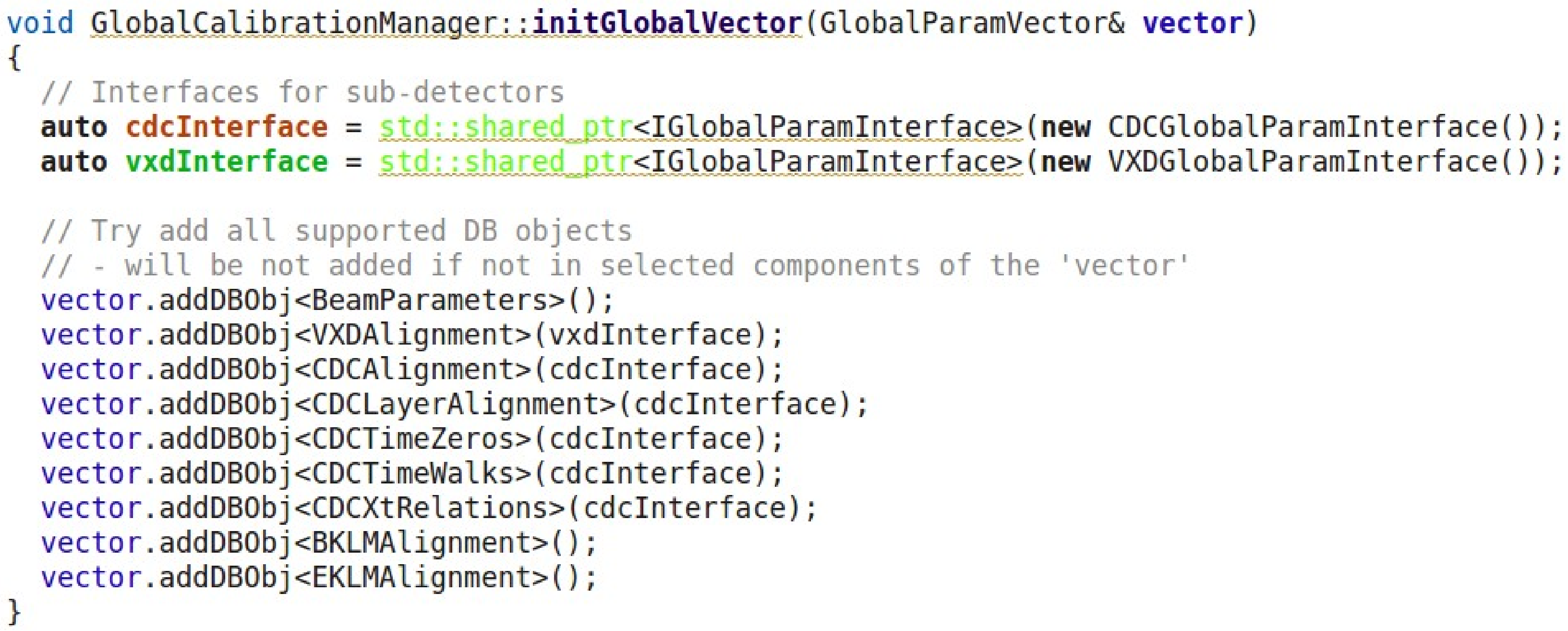


```
TMatrixD AlignableCDCRecoHit::localDerivatives(const genfit::StateOnPlane* sop)
{
  // CDC track time correction ---------------------------------------
  const TVector3& p = sop->getMom();
  double alpha =  CDCGeometryPar::Instance().getAlpha(sop->getPlane()->getO(), p);
  double theta =  CDCGeometryPar::Instance().getTheta(p);

  //TODO change to derivative of the full Xt relation
  //double driftVelocity = CDCGeometryPar::Instance().getNominalDriftV();
  double driftVelocity = CDCGeometryPar::Instance().getDriftV(m_tdcCount, getWireID().getICLayer(), int(m_leftRight), alpha, theta);

  TMatrixD locals(2, 1);
  //TODO sign: plus or minus??
  locals(0, 0) = - double(int(m_leftRight)) * driftVelocity;
  // FIXME not insensitive for stero wires!
  locals(1, 0) = 0.; // insesitive coordinate along wire

  return locals;
}
```

0.1 ns bias

100 x smaller but
still bias larger than
error

Manager.cc



```
void GlobalCalibrationManager::initGlobalVector(GlobalParamVector& vector)
{
  // Interfaces for sub-detectors
  auto cdcInterface = std::shared_ptr<IGlobalParamInterface>(new CDCGlobalParamInterface());
  auto vxdInterface = std::shared_ptr<IGlobalParamInterface>(new VXDGlobalParamInterface());

  // Try add all supported DB objects
  // - will be not added if not in selected components of the 'vector'
  vector.addDBObj<BeamParameters>();
  vector.addDBObj<VXDAlignment>(vxdInterface);
  vector.addDBObj<CDCAlignment>(cdcInterface);
  vector.addDBObj<CDCLayerAlignment>(cdcInterface);
  vector.addDBObj<CDCTimeZeros>(cdcInterface);
  vector.addDBObj<CDCTimeWalks>(cdcInterface);
  vector.addDBObj<CDCXtRelations>(cdcInterface);
  vector.addDBObj<BKLMAlignment>();
  vector.addDBObj<EKLMAlignment>();
}
```

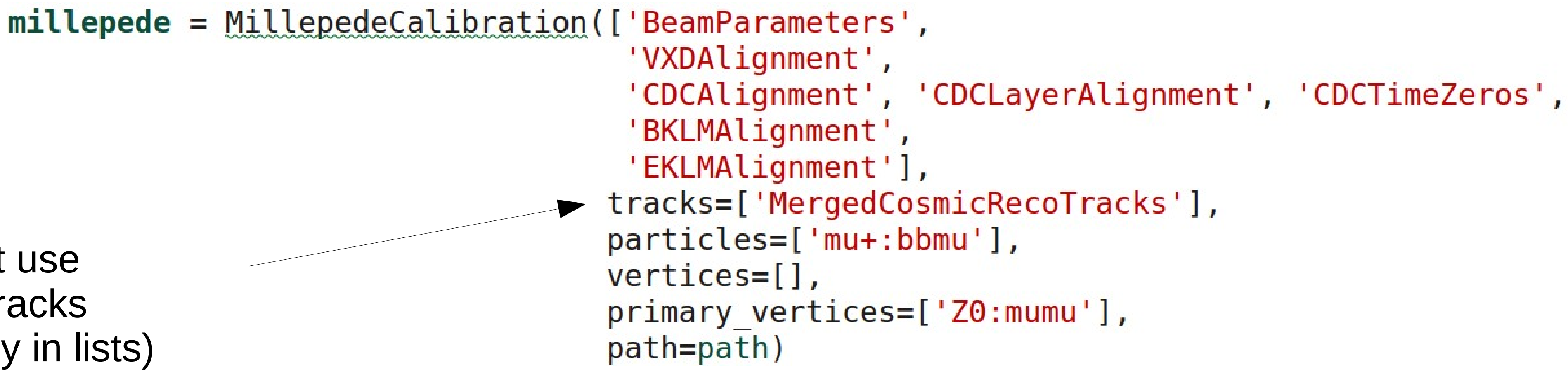
What happened recently

- Tuning, validations, bug-fixes …
- Focus on Phase 2 setup
  - All tests with Calibration Framework – LSF backend nicely running on KEKCC
  - All results here only with di-muons (best established alignment sample)
  - Full reconstruction (no MC)
  - **Beam background overlay**
- Tested/tuned on MC
  - BeamParameters vertex alignment
  - Beast II
    - Half-shell alignment
    - Sensor-by-sensor alignment
  - CDC
    - layer alignment (alternative, not compatible with CDC local approach)
    - T0 calibration per wire (alternative)
    - ...
  - BKLM, EKLM

Since midnight, all statements in this presentation are checked to be valid with beam background

Calibration Framework

- **C**alibration @ **A**lignment **F**ramework = CAF
  - Automated calibration with basf2
  - Collector Modules + Algorithms + Python steering
- Alignment @ Calibration with Millepede II
  - MillepedeCollector + MillepedeAlgorithm
  - MillepedeCalibration helper



```
millepede = MillepedeCalibration(['BeamParameters',
                                  'VXDAlignment',
                                  'CDCAlignment', 'CDCLayerAlignment', 'CDCTimeZeros',
                                  'BKLMAlignment',
                                  'EKLMAlignment'],
                                 tracks=['MergedCosmicRecoTracks'],
                                 particles=['mu+:bbmu'],
                                 vertices=[],
                                 primary_vertices=['Z0:mumu'],
                                 path=path)
```

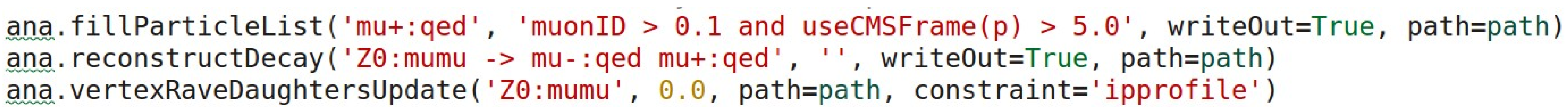
Cannot use
RecoTracks
(already in lists)

Alignment Input Sample Types: Status

	Example	Note	Status
RecoTracks	Cosmic mu	Also needed with B=0	OK Mixing of cosmic and std reco? How to mix B=0 runs?
Particles	Single muons		OK, selection?
Vertices	Lambda, K_S		Not tested
Primary Vertices	ee->mumu J/Psi->mumu	vertex+beam constraint Alignment of beam vertex	OK Not tested
Two Body Decays	J/Psi->mumu	mass+vertex constraint	TODO
Primary Two Body Decays	ee->mumu	Calibration of beam kinematics	TODO

Di-muon Sample + Phase II Background

- 100 x 1k ee->mumu events from KKGenInput + Phase 2 BG
- Use analysis package to reconstruct decays



```
ana.fillParticleList('mu+:qed', 'muonID > 0.1 and useCMSFrame(p) > 5.0', writeOut=True, path=path)
ana.reconstructDecay('Z0:mumu -> mu-:qed mu+:qed', '', writeOut=True, path=path)
ana.vertexRaveDaughtersUpdate('Z0:mumu', 0.0, path=path, constraint='ipprofile')
```

- Select particle lists for collection of calibration data
- Vertex+beam constrained re-fit of the dimuon combined object
- Parameters: 64 (Beast II) + 3 (BeamParameters)
  - fix Layer1
  - fix w, beta for all sensors
- Misalignment: 100 um in shifts, 1 mrad in angles (random)
- Residual misalignment < 3 um, < 0.1 mrad

Alignment with Background

- Several tests with background overlay
  - /home/belle/staric/public/basf2/release-01-branch/samples/phase2/BgforOverlay-*.root
- Alignment still works!
  - But close to point where it will get into some troubles
  - Large fraction of rejected tracks

```
55901 = number of records
Sum(Chi^2)/Sum(Ndf) =  9186837.2  / (  8365533 -  67 ) =   1.0982

             with correction for down-weighting      1.17755294

 Data rejected in last iteration:   118  (rank deficit/NaN),  0  (Ndf=0),  1  (huge)
          12773  (large)
```

Beast II Alignment Reference System

- Fixing by CDC?
  - No! Does not converge ~40 um systematic shift after 5 **iterations** in V(Z) (N.B. CDC z-resolution)
  - for U: systematics < 10 um
- Fix Layer 1 (both PXD?)
  - Systematics < 4 um
- Fix only PXD 1.1.1?
  - 6 um systematics in U
- Constraints?
  - Needs some update to work with CAF (transport of constraint equations)

Testing the alignment

- Iterations
  - With all the realistic track finder, clusters, …
  - We have to iterate (non-linearities)
  - In misaligned tests, often even 6th iteration still improves the result
  - Iterations stop if average parameter correction/error < 1 (or maximum # iterations is reached)
- CAF
  - Makes iterations easy
  - But when something fails/to hunt bugs... less convenient

„Half-shell“ alignment for Beast II

- Example:

  alignment/examples/phase2/sampleMuMu.py

  alignment/examples/phase2/alignBeast2.py

- Fix PXD, align SVD
- Align both – not tested yet (only CDC as reference)

How Data will reach alignment?

- More an issue for CAF running the alignment
- Tag calibration events and write out?
  - Online/Express/Offline difference
- Currently CAF assumes single pre-collector path for each calibration
  - Need to run „analysis“ to select alignment tracks and decays for all channels at once...
  - Conditional paths for different event types?
  - Change CAF to allow for different paths/collector setup for different file types (how to change the user interface?)

Conclusion – what still has to be done

- Alignment works
  - Ready for Phase 2
- The worflow still has limitations/uncertainities
  - Sample combination
  - Sample selection/skim
  - Run by run calibration
- New developments mostly not crucial for alignment in Phase 2

Thank you for your attention!