N dimensional analysis pipeline + RootInteractive for visualization

Marian Ivanov

Boris Rumyancev - Old C++ visualization
Alperen Yuncu - Python visualization
Martin Kroesen - Machine learning wrappers
Jacek + Sebastian - input for the QA time series analysis use case
Mesut - input /testing of the PID calibration use case

Outlook

Comparison of different methods:

- Thasseography and Shadow projections
- Multidimensional analysis

Multidimensional analysis using **ND pipeline** + **ML interface**

Visualization development - RootInteractive (pythone library)

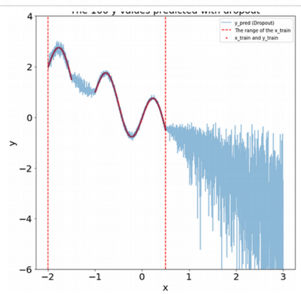
- https://github.com/miranov25/RootInteractive
- **Proposal: To be integrated into aliBuild (AliRoot and O2)**

Interactive visualization of N dimensional parametric space

- Bokeh standalone
- Jupyter notebook and ipywidget
- ML for QA/QC time series analysis/automatic alarms

Machine learning wrappers:

- Predefined algorithm for "typical" use cases
- Measure the uncertainty
- Robust regression and model compression



Tasseography (also known as tasseomancy or tassology) is a divination or fortune-telling method that interprets patterns in tea leaves, coffee ounds, or wine s or 1D histograms

Spring Pouchong tea (Chinese: 包種茶; pinyin: *Bāozhòngchá*) leaves that may be used for tasseography divination

An example of a tea leaf reading showing a dog and a bird on the side of the cup.

https://en.wikipedia.org/wiki/Tasseography

Tasseography (1D examples)

Examples of wrong statements:

"Detector noise did not change":

- **1D conclusion:** 1D mean and rms is "in range"
- **Reality could be:** relative increase of the noise in critical/noisy regions by factor 2-3 not spotted

"DCA resolution is fine":

- **1D conclusion:** TPC σ_{DCA} is 1 cm as usual
- **Reality could be:**
  - DCA resolution at high pt 3-4 times worse (3-4 mm instead of the 1 mm)
  - DCA is biased as function of phi

"TPC-ITS matching is fine":

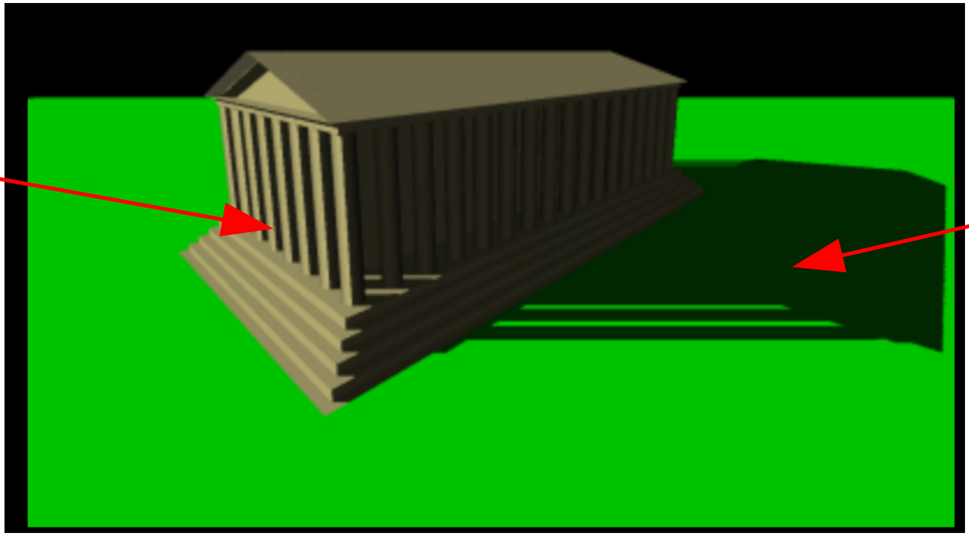
- **1D conclusion:** mean TPC-OTS matching is stable
- **Reality could be:** local spikes in time and space. LHC15o sector 2 misbehaving (~ 0 TPC $\rightarrow$ ITS matching) in some time intervals - due distortion fluctuation

"dEdx bias for pile-up event is small":

- **1D conclussion:** 0.2 sigma bias
- **Reality (4D maps):** up to 2 sigma bias - data not suitable for some category of analysis

Shadow projections (2-Dimensional projections)

Our object
E.g occupancy map from AMORE (3D)



Projection of object - Our current TPC DQM

$$\sigma_{\vec{A} \ominus \vec{A}_{ref}} \leq \sigma_{\vec{A}} (+) \sigma_{\vec{A}_{ref}}$$

Guessing from 2D projection more reliable than Tasseography

- some imagination to be involved (see next slides)

Alarms to be based on some invariance - e.g the difference between the object and referenced object

- after projection impossible
- in my typical cases variance $\boldsymbol{\sigma_{A-Aref}}$ is very often smaller by orders of magnitudes

Shadow projections, alarms and invariants

$$\sigma_{\vec{A}\ominus\vec{A}_{ref}} \leq \sigma_{\vec{A}}(+)\sigma_{\vec{A}_{ref}}$$

Invariance/symmetries in N dimensions (A ref model vector):

- invariance in time (using e.g. reference run)
- invariance in space (e.g. rotation, mirror symmetry)
- data - physical model
- A side/C side, B field symmetry
- smoothness resp. local smoothness

Projections problems (hidden variables):

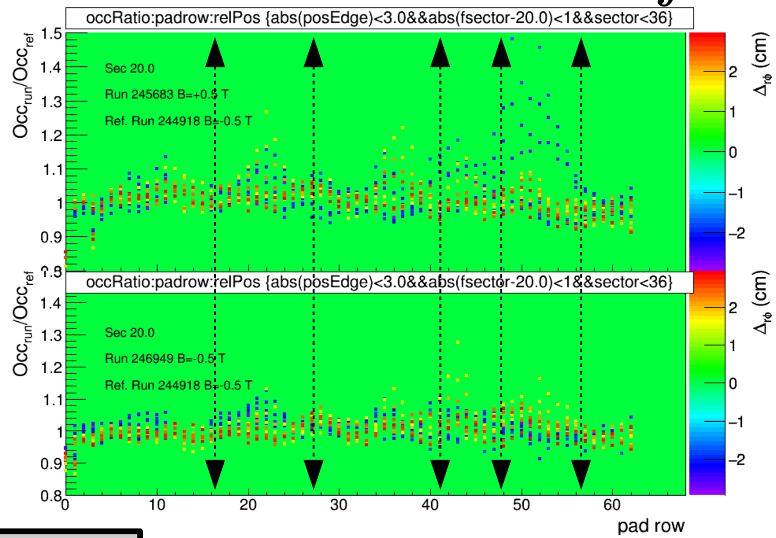
- **Information loss. Intrinsic spread of variable vectors A and A ref is usually significantly bigger than spread of $A\text{-}A_{ref}$**
  - noise map, DCA bias, resolution maps, occupancy maps, sigma invariant mass maps .... as function of 1/pt, θ, occupancy, dEdx
- **Projected vector A depends on the actual distribution of hidden variable**
  - Sometimes misleading results
  - Non trivial interpretation of projected observation

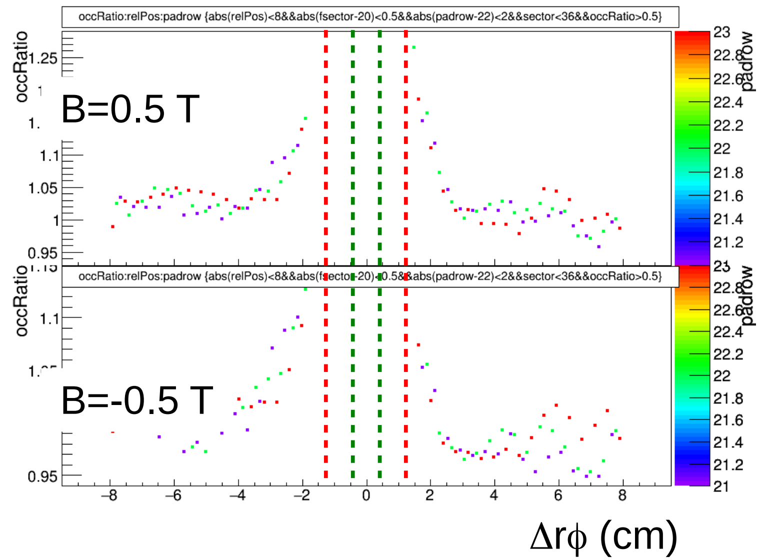
Example usage of N-Dimensional analysis pipeline in detector studies

- distortions and distortion fluctuation strongly mitigated in hardware
- dEdx bias for pile-up events understood and partially mitigated

Space point distortion → Occupancy observable

$$\sigma_{\vec{A}\ominus\vec{A}_{ref}} \leq \sigma_{\vec{A}} (+) \sigma_{\vec{A}_{ref}}$$





Model:

$$\frac{N_{Cl}(IR)}{N_{Cl}(IR=0)} = \overline{\frac{(w+(\Delta_{r\phi}(r_\phi+w/2)-\Delta_{r\phi}(r_\phi-w/2)))}{w}}$$

$$\overline{Z} \approx 125cm$$

$$R = \left(\frac{Occ}{<Occ_{ROC}>}\right)_{IR} \Big/ \left(\frac{Occ}{<Occ_{ROC}>}\right)_{IR=0}$$

Conclusion: Distortion origin in the gap

Increase of occupancy close to the hot-spot region due to space-charge distortion
Very precise measurement of distortion origin - measuring **derivative of distortion** with sub-pad granularity

Without proper normalization to reference effect is invisible.

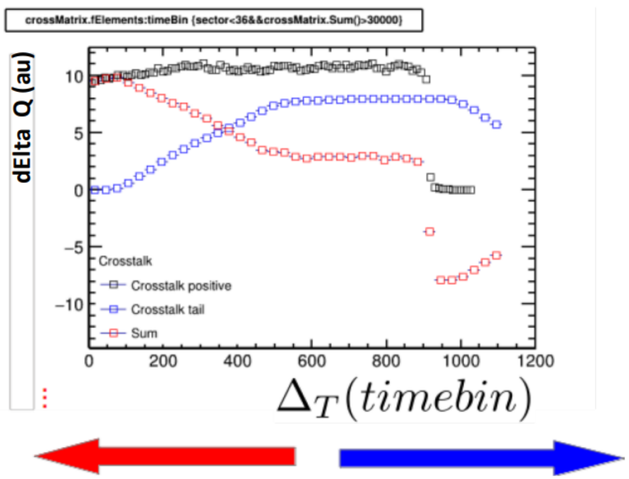
Wrong concussion was made in first analysis

dEdx correction

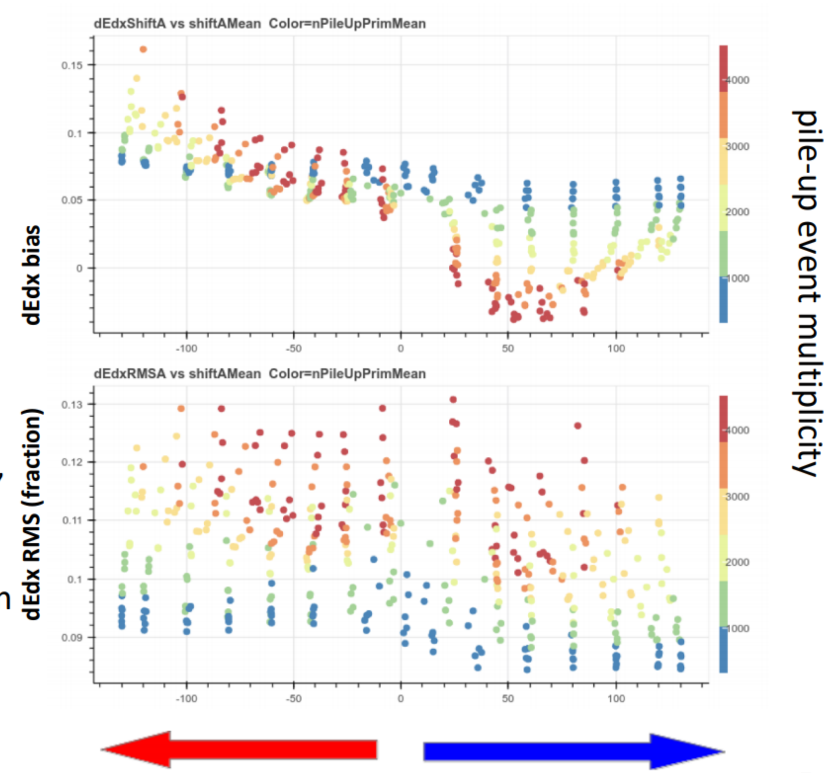
TPC dEdx bias (PbPb 2015/2018)

DPG ALICE • Data Preparation Group

Marian Ivanov



2D slice of 4D correction map (**pz/pt=0, SPD Mult=600**)



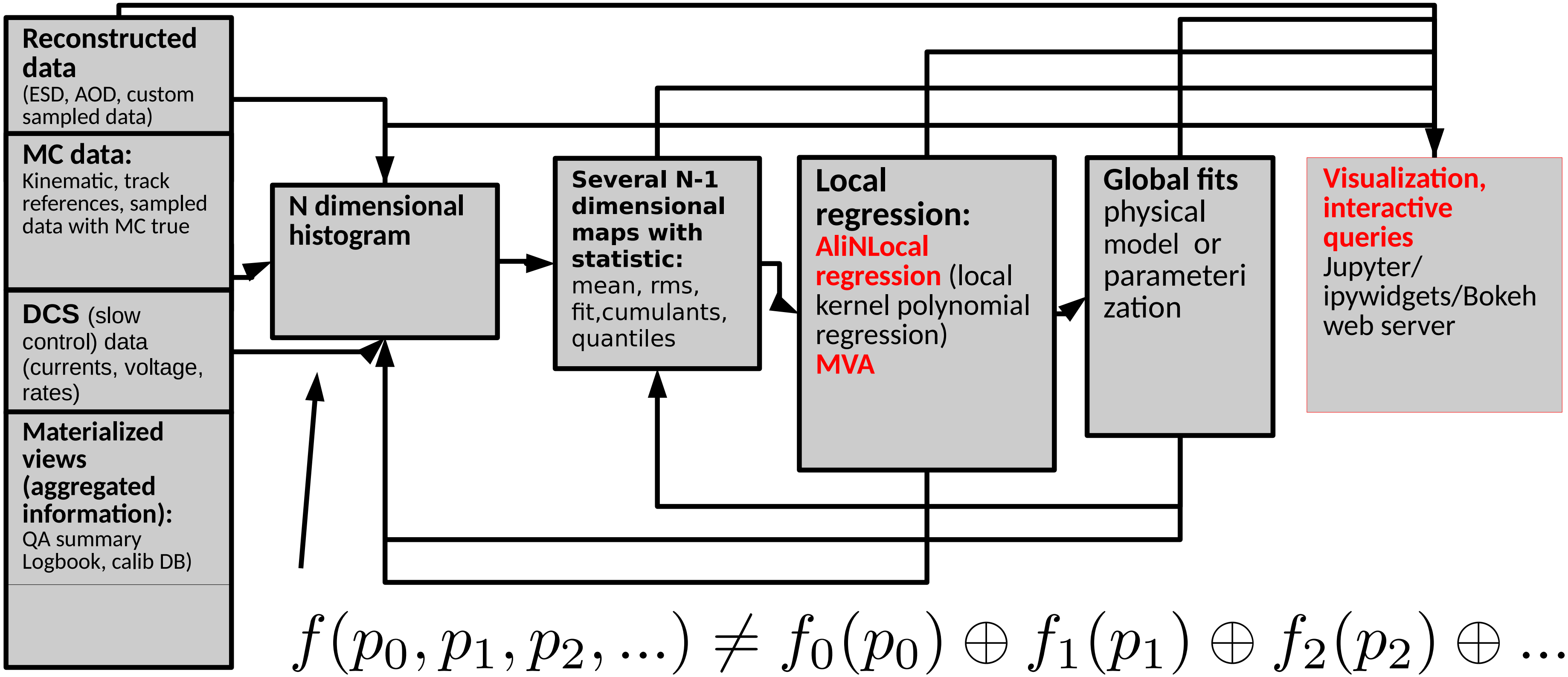
- dEdx measurement influenced by baseline shift(ion tail, crosstalk) and cluster loss bias effect ~ 20-35 % partially corrected in reconstruction
- Calibration tuned for the MB event - **time profile** used in the reconstruction **not correct**
- Strong residual **dEdx bias O(15%-20 %),O(2sigma)** for pile-up events **from the past and future**
  - mean bias over all pile-up times significantly smaller

5

2 dimensional slice of **4 dimensional correction map - 20 % correction**
2D correction map mean correction (projecting all dZ) **~ 0 % - useless**

Multidimensional analysis pipeline

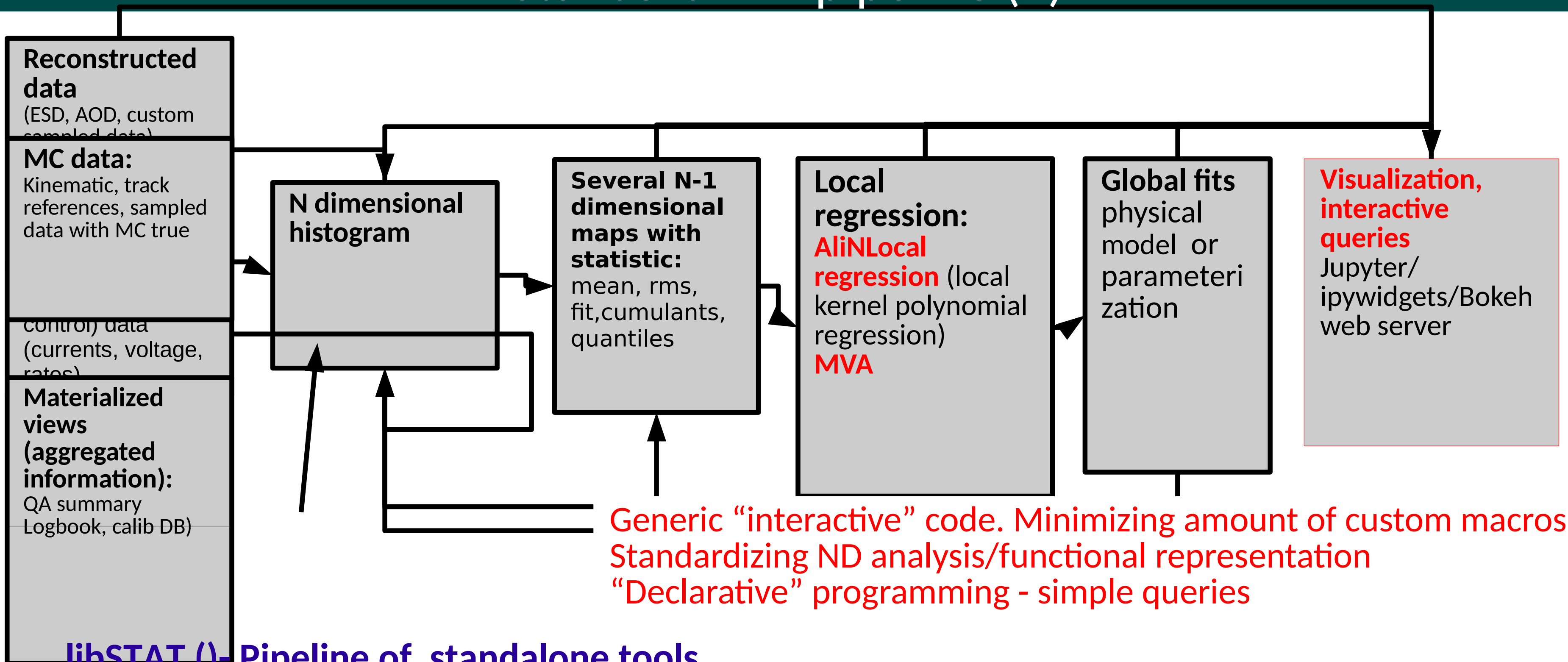
- library (libStat) written in C++
- possible to use in Python
- visualization written in Python - possible to invoke it in C++ (root)



$$f(p_0, p_1, p_2, \ldots) \neq f_0(p_0) \oplus f_1(p_1) \oplus f_2(p_2) \oplus \ldots$$

Standard calibration/performance maps and QA done and interpreted in multidimensional space

- dimensionality depends on the problem to study (and on available resources)
- Data →Histogram → set of ND maps → set of NDlocal regression/TMVA → Global fits
  - Some steps can be skipped, e.g local regression (MVA/AliNDLocal) can be done using unbinned input data
  - Histogramming in case of non sparse data
  - MVA for sparse (going to higher dimensions)



libSTAT ()- Pipeline of standalone tools

- N dimensional histogramming
  - THn/THnSparse - part of root
  - AliTreePlayer::MakeHistogram ….
- THn → Map (tree)
  - part of the TStatToolkit
- Map(Tree) → Local regression
  - AliNDLocalRegression, MVA interface
- Map(Tree) → Global fits (physical models, parameterizations)
  - AliTMinuitToolkit

Curse of dimensionality. MVA/histogramming

https://en.wikipedia.org/wiki/Curse_of_dimensionality

When the dimensionality increases, the volume of the space increases so fast that **the available data become sparse**.

- The volume of a cube grows exponentially with increasing dimension
- The volume of a sphere grows exponentially with increasing dimension
- Most of the volume of a cube is very close to the (d − 1)-dimensional surface of the cube

Effect to be considered. Detector/reconstruction experts to be consulted

- Find relevant dimensions (2-6 dimensions)
- Proper selection of variables (smooth or linear behavior)
  - e.g q/pt instead of pt, occupancy/multiplicity instead of centrality
- Proper binning. In case proper selection of variables, few bins needed

In the following I'm considering properly designed dimensionality/binning of the space

MVA in case of the sparse data (too high dimensions, time series)

Curse of dimensionality (Example performance map)

https://en.wikipedia.org/wiki/Curse_of_dimensionality

When the dimensionality increases, the volume of the space increases so fast that the available data become sparse. This sparsity is problematic for any method that requires statistical significance.

Code fragment $AliPhysics_SRC/PWGPP/TPC/macros/performanceFiltered.C

```
//
hisString+=TString::Format("deltaP%d:qPt:tgl:logTracks5:#IsPrim4&&TPCOn&&ITSRefit&&ITSOn01&&nclCut>>hisDeltaP%d_Allv_qPt_tgl_logTracks5(400,%f,%f,200,-5,5,10,-1,1,10,0,10);",iPar,iPar,-rang
hisString+=TString::Format("deltaP%d:qPt:tgl:logTracks5:#IsPrim4&&TPCOn&&ITSRefit&&ITSOn01&&nclCut&&TRDOn>>hisDeltaP%d_TRDv_qPt_tgl_logTracks5(400,%f,%f,200,-5,5,10,-1,1,10,0,10);",iPar,iPa
hisString+=TString::Format("pullP%d:qPt:tgl:logTracks5:#IsPrim4&&TPCOn&&ITSRefit&&ITSOn01&&nclCut>>hisPullP%d_Allv_qPt_tgl_logTracks5(400,-8,8,200,-5,5,10,-1,1,10,0,10);",iPar,iPar);
hisString+=TString::Format("pullP%d:qPt:tgl:logTracks5:#IsPrim4&&TPCOn&&ITSRefit&&ITSOn01&&nclCut&&TRDOn>>hisPullP%d_TRDv_qPt_tgl_logTracks5(400,-8,8,200,-5,5,10,-1,1,10,0,10);",iPar,iPar);
 hisArray = AliTreePlayer::MakeHistograms(chain, hisString, defaultCut,0,maxEntries,200000,15);
```

For the tracking performance studies - histogramming is better option (at first stage)

Resolution/pulls as function of (q/pt,Q,mult) - O (20000) bins

Performance generator (jets flat in q/pt)

- 100 jobs x 50 events x 100 tracks (few hours at GSI farm)
  - Not sparse - O(25 tracks) per bin
  - more in case of bin grouping (parameter in map creation)

Interactive analysis using filtered trees (sampled input flat in pt)

Usage of n-dimensional pipeline

Pipeline with performance maps in N dimensions in form of generic function (TFormula).

- In many cases corresponding physical model or parameterization available

Usage:

- differential QA
- understand/remember detector behavior - physical models
- scan tuning of the reco. parameters (metric diff of perf. maps)
- scan tuning of the MC parameters (metric diff of perf. maps)
- compare differential y data with MC
- provide recipes for optimal cut selections
- provide input/parameterizations for toy MC/fast MC
- feasibility studies
- enable tune on data in N-dimensions - remapping MC → Data
- **enable ML algorithms (tune on data)**

$$f(p_0, p_1, p_2, \ldots) \neq f_0(p_0) \oplus f_1(p_1) \oplus f_2(p_2) \oplus \ldots$$

RootInteractive

RootInteractive

Why RootInteractive?

- RootInteractive is an Python 2 and (soon 3) library for data analysis and visualization. Python already has excellent tools like numpy, pandas, and xarray for data processing, and bokeh and matplotlib for plotting, so why yet another library?

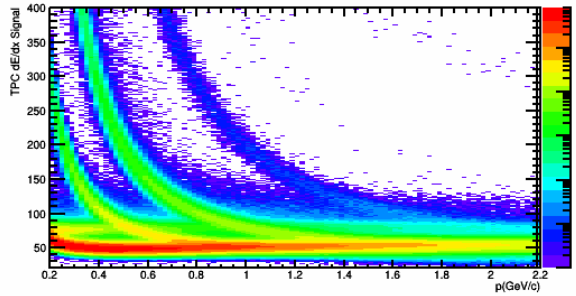
RootInteractive helps you understand your data better, by letting you work seamlessly with both the data and its graphical representation.

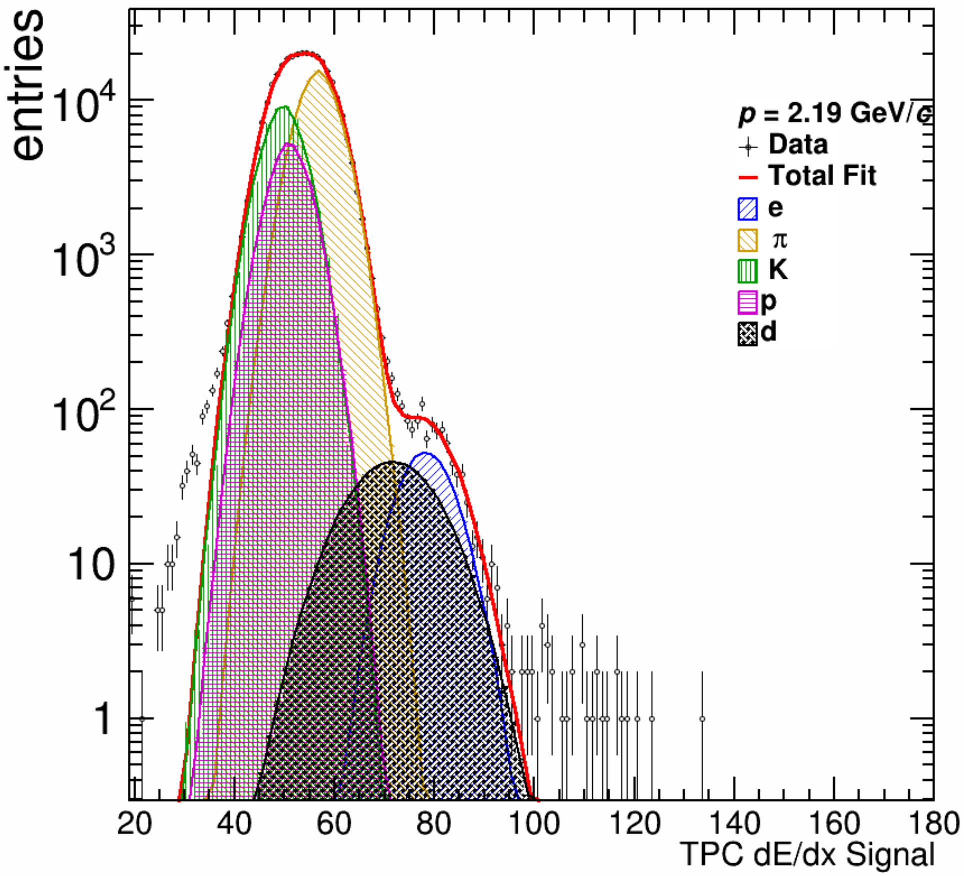
RootInteractive focuses on bundling your data together with the appropriate metadata to support both analysis and visualization, making your raw data and its visualization equally accessible at all times.

With RootInteractive, instead of building a plot using direct calls to a plotting library, you first describe your data with a small amount of crucial semantic information required to make it visualizable, then you specify additional metadata as needed to determine more detailed aspects of your visualization. This approach provides immediate, automatic visualization that can be effortlessly requested at any time as your data evolves, rendered automatically by one of the supported plotting libraries (**such as Bokeh or Matplotlib**).

Inspired by the **HoloViews project** + **support for the Root/AliRoot**

- possibility to use the code in C++ → in case function to be used in root/C++ parameters by string
- string parsed to python structures internally





dEdx histogrammed in the bins of multiplicity, pz, pz/pt

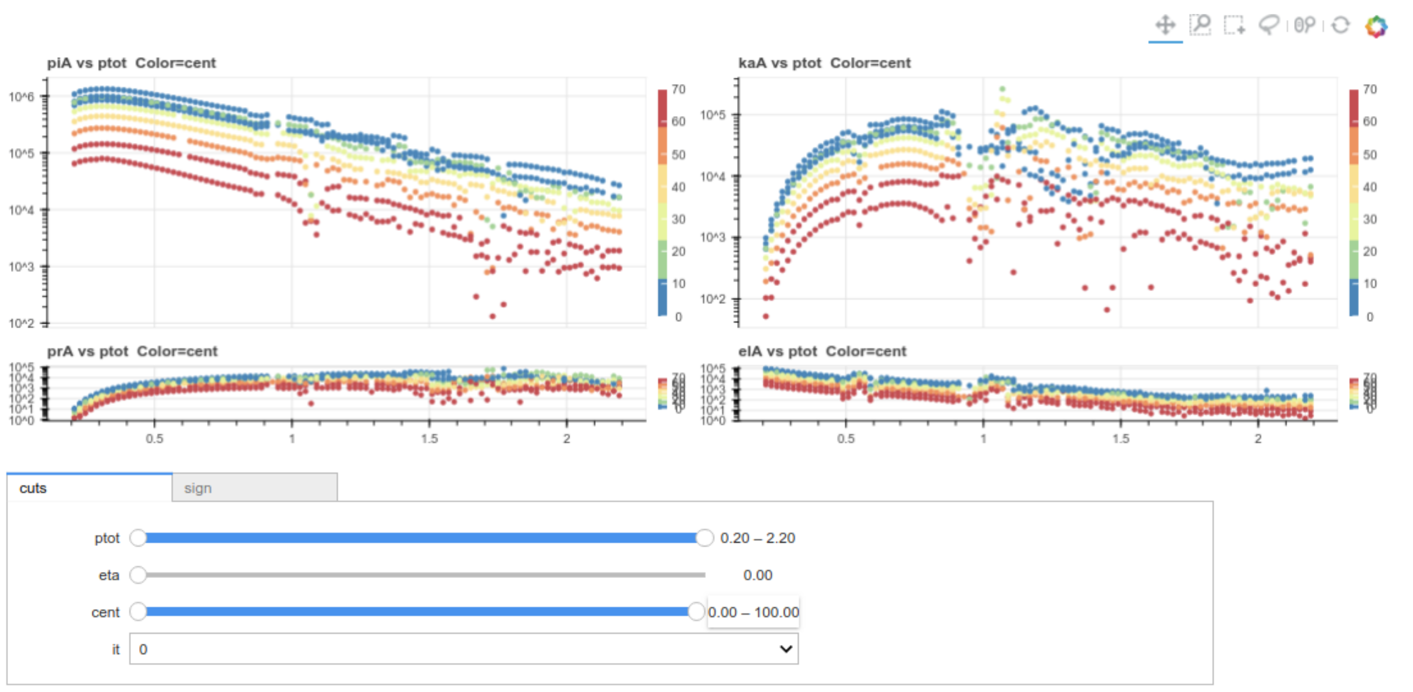
Performance maps extracted from 4 dimensional histograms

- raw particle yield
- dEdx parameters (mean,s curtosis)
- fits in several iterations
- convergence as function iteration additional dimension

Tree/panda interactive application for dummies

3-4 parameters to declare
see Jupyter demo

```
tree.SetAlias("selected","rndm<1.5&&elA>1&&piA>100&&kaA>50&&prA>1")
varDraw="piA:kaA:prA:elA"
tooltips=[('Eta ','@eta'),('Centrality ','@cent'),("pTot","@ptot"),("Iteration","@it")]
layout="((0,1,x_visible=0),(2,3,plot_height=100), \
        plot_width=1200,plot_height=250,y_visible=1)"
widgets="tab.cuts(slider.ptot(0.2,2.2,0.02,0.2,2.2),slider.eta(0,0.8,0.1,0.0), \
slider.cent(0,100,10,0,100),dropdown.it(0,1,2,3,4,5,6)),tab.sign(dropdown.sign(0,1,-1))"
fig=bokehDraw(tree,"selected>0","ptot",varDraw,"cent",widgets,0,size=4,nEntries=100000000,
tooltip=tooltips, y_axis_type="log", layout=layout)
```



Usage in C++

```
void testBokehRender(){
  TString x=importBokeh;
  x+="tree=ROOT.gROOT.GetGlobal(\"tree\") \n";
  x+="aliases=aliasToDictionary(tree)\n";
  x+="base=Node(\"MIPquality_Warning\") \n";
  x+="makeAliasAnyTree(\"MIPquality_Warning\",base,aliases)\n";
  x+="print(RenderTree(base))";
  TPython::Exec(x);
}

void InitData(){
  AliExternalInfo info;
  tree=info.GetTree("QA.TPC","LHC15o","cpass1_pass1");
}
```

Root interactive library can be used from the root session

C++ wrapper for most important functions:

- visualization - e.g. creating of the dashboard from root session
- tree syntax analysis - to build RDataFrame from hierarchy of aliases
- Machine learning wrappers

RootInteractive - Ongoing development

Histogram browsing

Status bar support

Tree - status bar support

Convert static QA pages to bokeh reports

Link histogram and tree/tabular information

Python/C++ wrapper for most important visualization functions

Test (pytest based)

Many examples of the usage of the RootInteractive in separate github

- https://github.com/miranov25/RootInteractiveTest
  - Currently 7 tasks with jupyter notebook
- Root interactive data web server for tutorial and example data

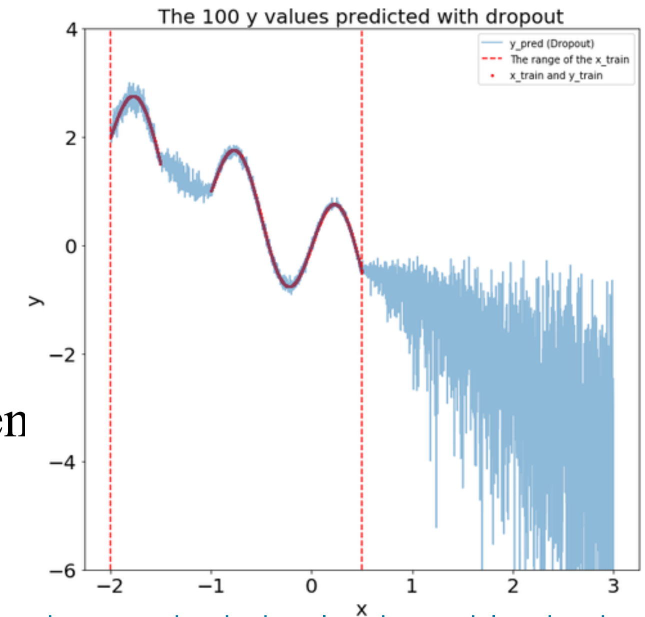
ML framework and QC tools in ALICE.

Measure the uncertainty

Robust regression and model compression

MVA wrapper+AliNDFunctionInterface

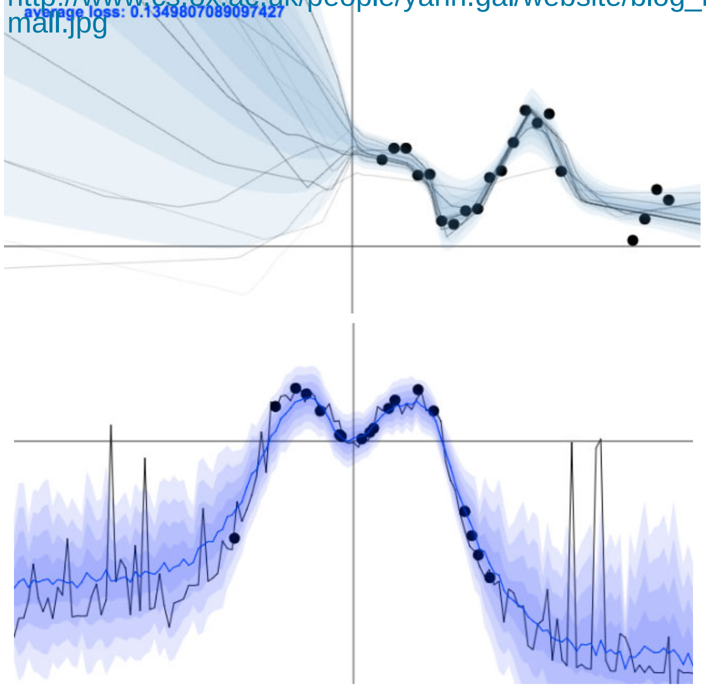
Marian Ivanov, Martin Kroesen



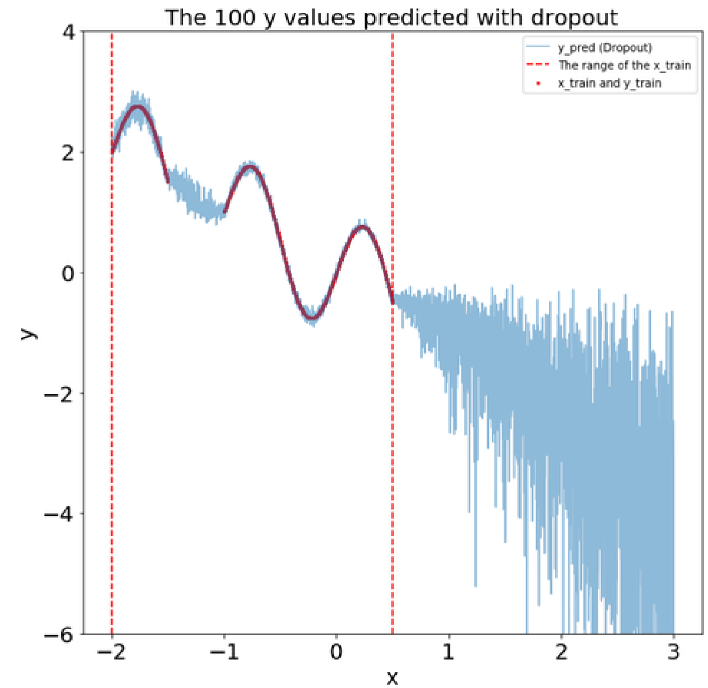
https://fairyonice.github.io/Measure-the-uncertainty-in-deep-learning-models-using-dropout.html

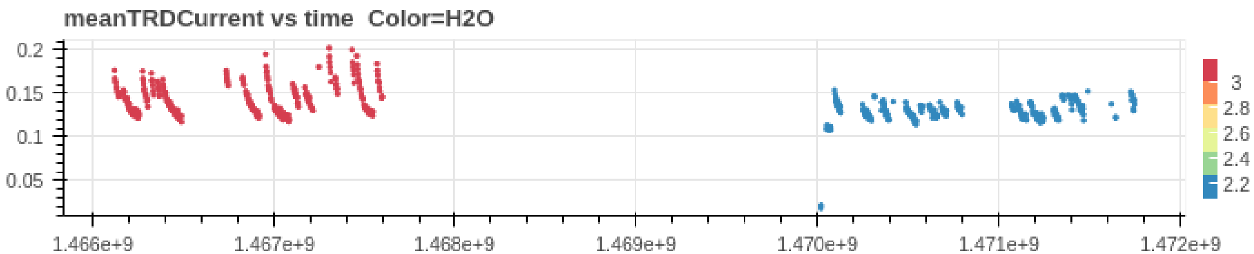
Why Should we Care About Uncertainty?

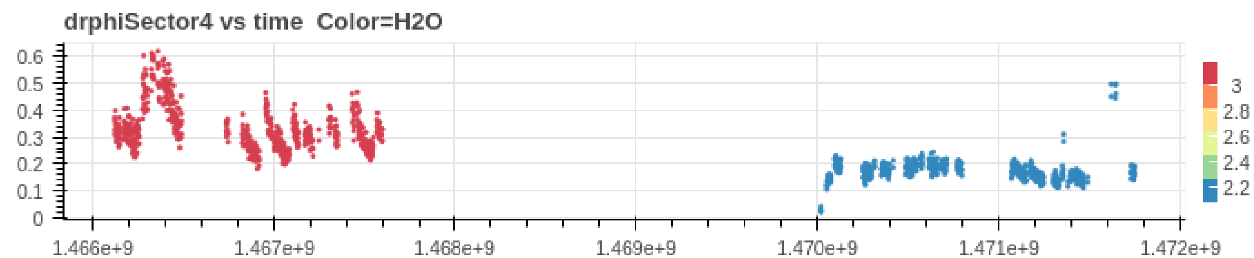
http://www.cs.ox.ac.uk/people/yarin.gal/website/blog_images/reg_demo_small.jpg



https://fairyonice.github.io/Measure-the-uncertainty-in-deep-learning-models-using-dropout.html







Example:
Alice example time series flux, gas composition and distortion

What is the prediction error for non seen data ?

Confidence/prediction intervals

Currently not standard libraries to estimate reducibble and irreducible error of the ML models. Effort only started

**For calibration/QA/data analysis - machine learning has to provide local confidence intervals - **we started to provide wrappers for some algorithms

Confidence Intervals for Scikit Learn Random Forests

- http://contrib.scikit-learn.org/forest-confidence-interval/
- https://github.com/scikit-learn-contrib/forest-confidence-interval
- forestci package
  - This package adds to scikit-learn the ability to calculate confidence intervals of the predictions generated from scikit-learn

Neural network prediction:

- 1: Delta method
- 2: Bayesian method
- 3: Mean variance estimation
- 4: Bootstrap

Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning (https://arxiv.org/abs/1506.02142 - 2015)

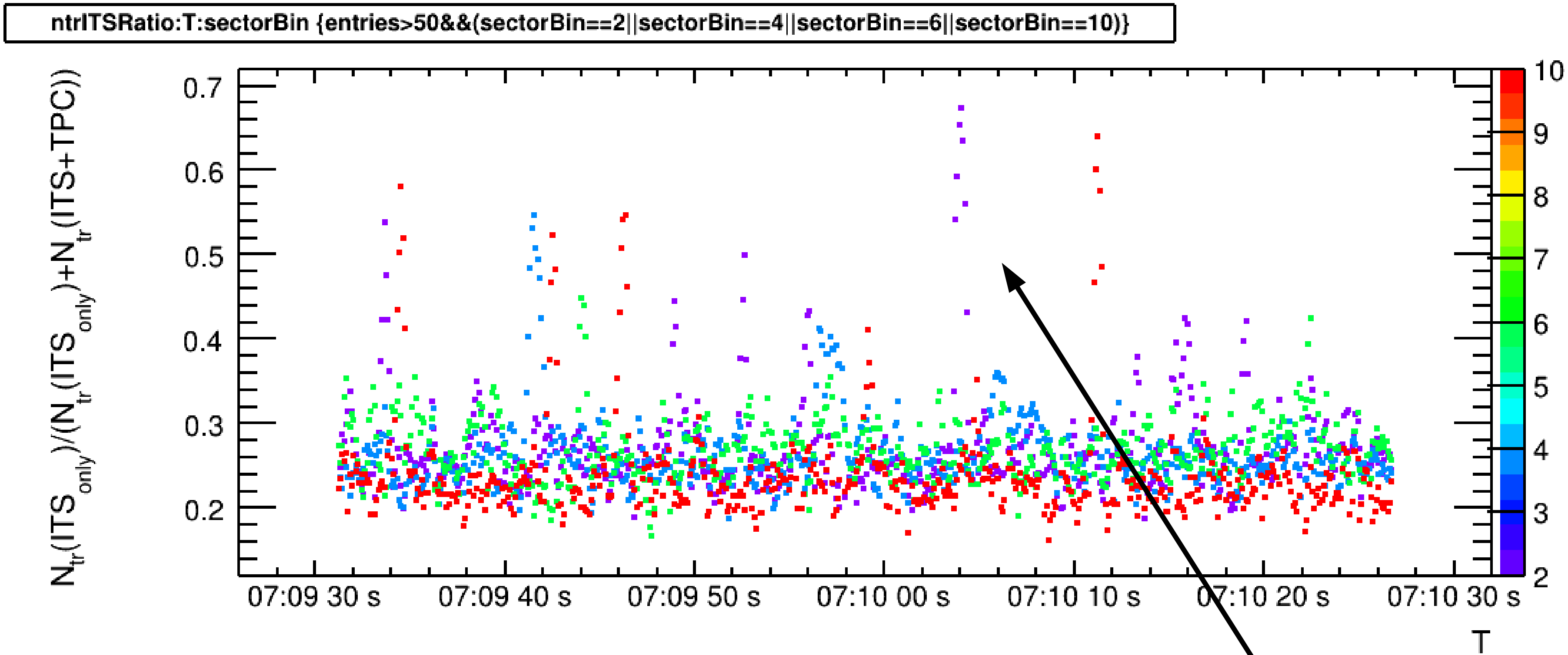
- *test-time dropout can be seen as Bayesian approximation to a Gaussian process related to the original network*

Bootstrap approach

- provides “prediction” intervals for all methods

Time interval QA + time series
Run2 example of investigation/problems

Correlated TPC/ITS efficiency loss. **Normal run 246148**



Time series QA O(0.1s)

Killer effect for many analysis

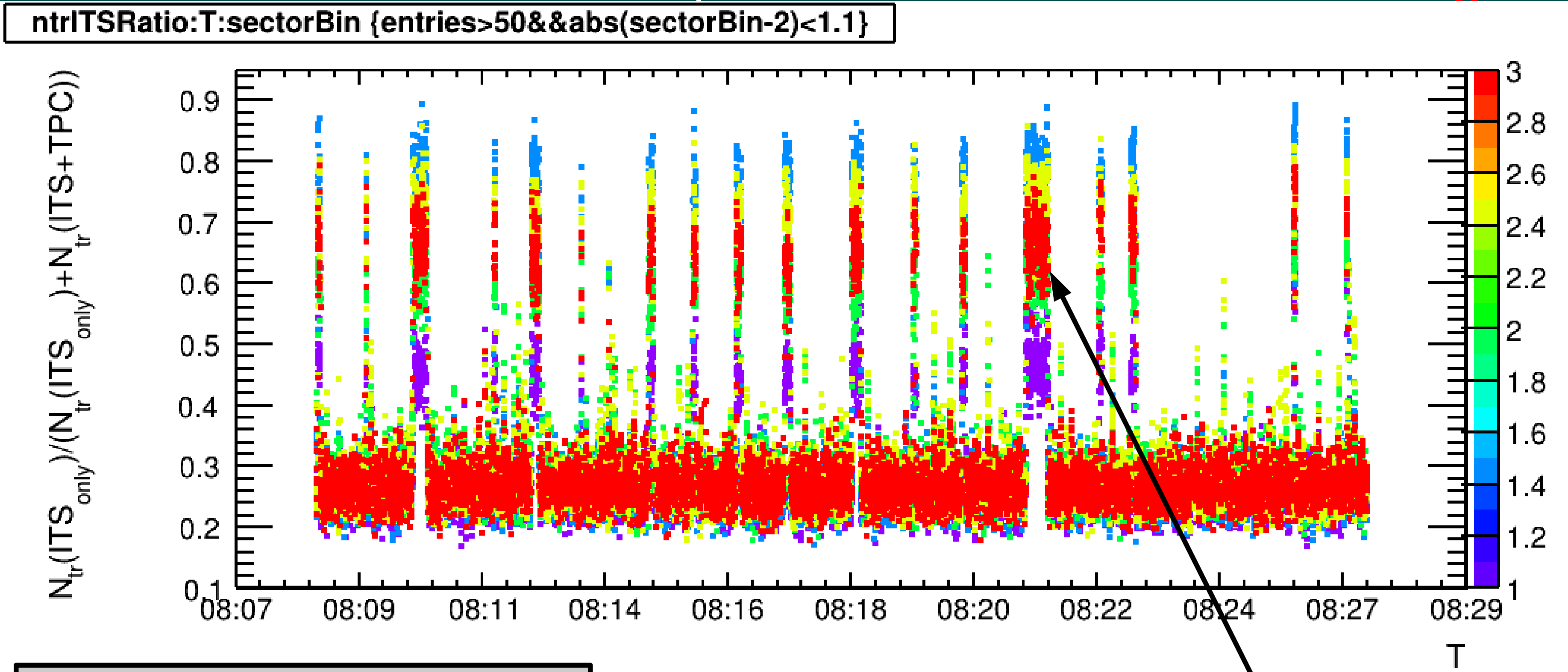
In particular time intervals **O(0.5) seconds** distortion increased

- locally worse resolution and matching **TPC-ITS efficiency**

Distortion independent- see time position of spikes

- **sector bins 2, 4, 6, 10**

Correlated TPC/ITS efficiency loss. Problematic run 246272 begin



Time series QA O(0.1s)

Regular structure observed at sector boundary 2 in the run (246272 and also some others)

Outliers in **matching efficiency** related to time intervals

- Looks like regular position O(min) spacing
- Irregular amplitude, probability and duration
  - begin of run (higher IR) - bigger probability longer duration

Killer effect for many analysis

ML for the time bin based QA

Classification problem:

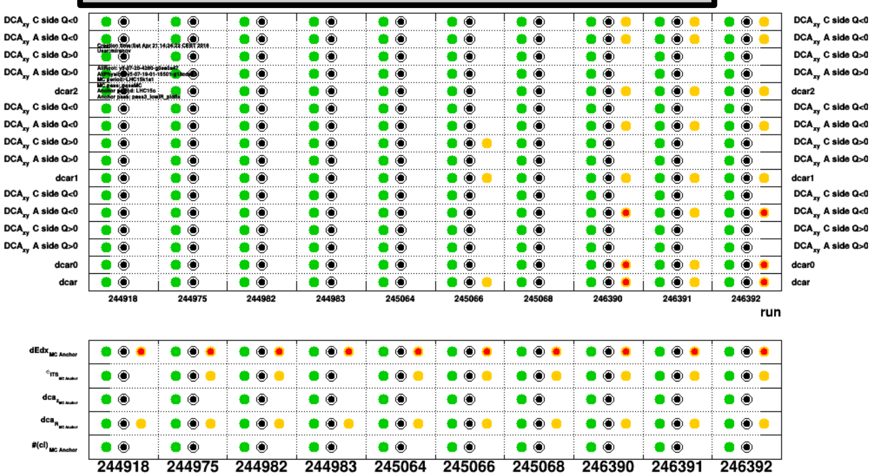
- Anomalies/Outliers in the performance QA
  - usually [|value-expectedValue|<n sigma
- Find most relevant features in the other observables
  - Maps: currents, distortion, local multiplicity, matching efficiency, chi2, Ncl, resolution ....
  - Derived “invariant” variables:
    - e.g RMS of current map/phi averaged map/scaled maps
    - distortion map/phi averaged map
    - local discontinuities in time and space
    - “physics” acceptable performance

Explain/find hardware origin of anomalies (hierarchy of alarms)

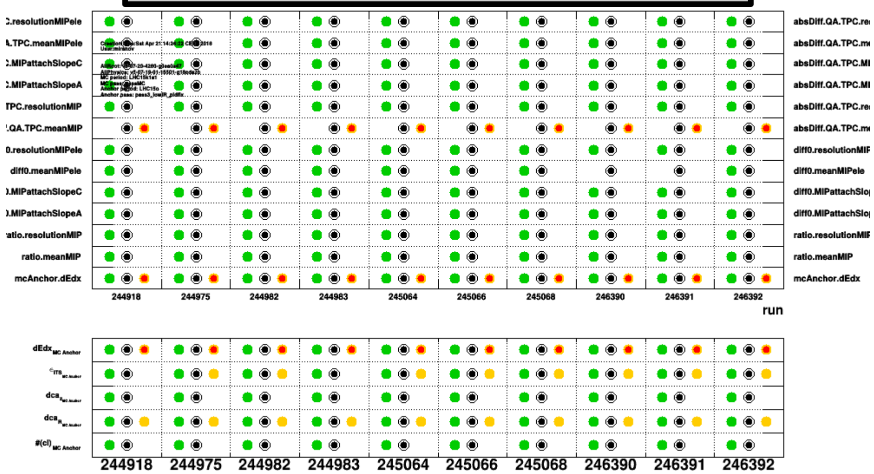
Training data:

- current maps, and distortion maps (at GSI) can be exported to alien
- time bin based QA currently available only for few run

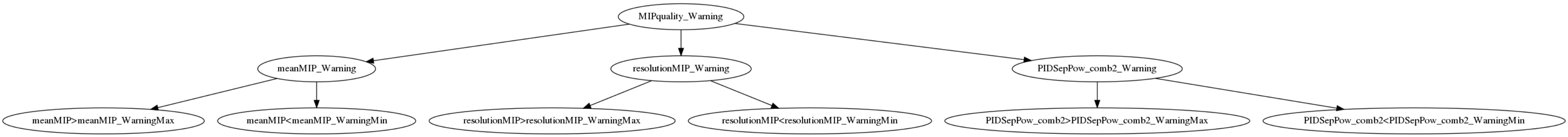
DCAr alarms decomposition



dEdx alarms decomposition



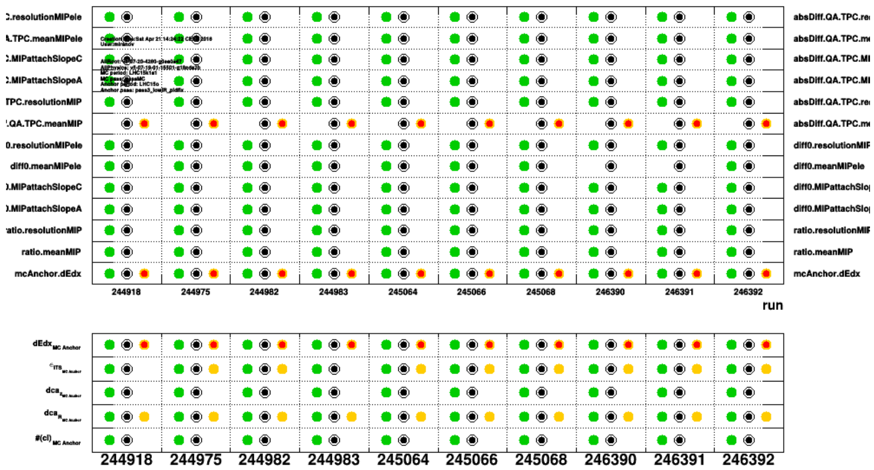
Current dEdx automatic alarms decomposition for the dEdx

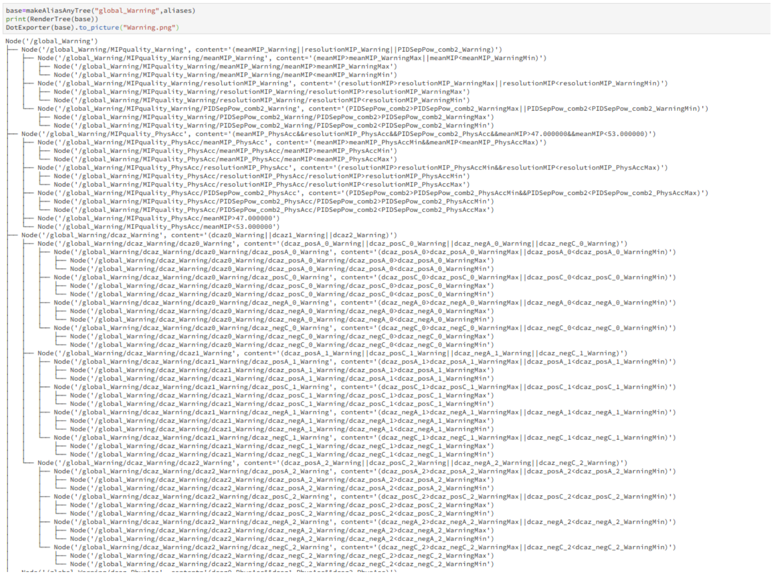


2018 data - QA extracted merge in 10 minutes interval:

- [JIRA] (ADQT-3) Classification of TPC QA Run2 data based on ML techniques
- https://alice.its.cern.ch/jira/browse/ADQT-3
- More data → possibility to use ML technique
  - Investigating auto-encoder to use to better define expected values and resolution

dEdx alarms decomposition





Conclussion

RootInteractive visualization library for interactive visualization of tabular data (TTree and pandas) and mutlidimensional histogramming data

- easy to create/configure/modify interactive dashboard
  - Jupyter notebooks as a template for interactive data expolaration/troubleshouting
  - regularly updated with calibration/QA/reconstruction tasks
- Library already usable for many use cases
- library further developed = new functionality to be added
  - major relaese with improved Ndimensional histogramming next week

RootInteractive - machine learning interface

- Goal -make ML as easy as standart fits
- provide good unncertainty estimators
- test cases : dEdx calibration, QA alarms,

Proposal:

Enable RootInteractive library in the aliBuild (AliRoot and O2)