

Distortion summary - Demo of the NDimensional interactive visualization and ML regression

(Bokeh+ipywidgets based)

- 0) Import libraries
- 1) Load csv file with distortion and split them per sector to enable correlation
- 2) Draw distortion as function of other distortions
- 3) Draw Distortion as function of the TRD flux
- 4) Load filter and fit distortion in regions as function of other distortions
 - fit mean and median properties + model "error" estimates -from bootstrapping
 - quantiles tp probe PDF - not yet in the demo
 - inspect fit results - corelation and time series - local data,data-fit, pulls (data-fit)/error

For time series distortion,fit - see the end of the notebook

- distortion vs time
- distortion-fit vs time
- (distortion-fit)"predicted" error vs time

0) Import libraries

- MLpipeline.NDFunctionInterface
 - wrapper on top of Keras and sklearn algorithms for the moment (RandomForest, KNN)
 - main purpose - provide error estimates+ combined estimators (e.g weighted mean)
 - simultaneous regression and comparison
- TTreeHnInteractive.bokehTools
 - interactive visualization of multidimensional data
 - graphics using pandas+Bokeh
 - interface like in our old TTree based interface - but interactive

In [11]:

```
from distortionStudy import *
from TTreeHnInteractive.TTreeHnBrowser import *
from TTreeHnInteractive.bokehTools import *
import matplotlib.pyplot as plt
from MLpipeline.NDFunctionInterface import DataContainer, Fitter
from TTreeHnInteractive.bokehDrawPanda import *
output_notebook()
p3 = figure(plot_width=400, plot_height=250, title="template")
```

(<https://bokeh.pydata.org>) Loading BokehJS ...

1) Load csv file with distortion and split them per sector to enable correlation

- csv file extracted before for combined root trees
 - one liner "AliTreePlayer::selectWhatWhereOrderBy(tree,vars,selection,"",0,10000,"csvroot","data.csv");"
- additional derived variable defined similar as formula "aliases" in tree

In [29]:

```
#AliTreePlayer::selectWhatWhereOrderBy(tree,vars,selection,"",0,10000,"csvroot","distortionAll.csv");
input=os.path.expandvars("$NOTESdata/JIRA/ATO-336/DistortionsTimeSeries/distortionAll.csv")
df=readDataFrame(input)
dfsplitsplit=splitDistortionFrame(df)
print("load csv file", input, df.shape, dfsplitsplit.shape)
dfsplitsplit=dfsplit.query("(iz2x>2 & meanTRDCurrent<0.3 & abs(invTRDCurrentNorm)<0.05 & gascomph20>0")
dfsplitsplit=SetAlias(dfsplitsplit,"H20","gascomph20/100")
dfsplitsplit.head(3)
# list(dfsplitsplit)
```

```
('load csv file', '/data/NOTESdata/alice-tpc-notes//JIRA/ATO-336/DistortionsTimeSeries/distortionAll.csv', (100000, 70), (5555, 92))
```

Out[29]:

yearperiod	fill	run	time	timeStart	timeEnd	bz	ir...	drphiNorm9	drphiNorm16	drphiNorm20	drphiNorm30	meanTRDCurrent	invTRDCurrent	deltaTRDCurrent	invTRDCurrentNorm	deltaTRDCurrentNorm	H2O
89320170	6381282051	1510471394	1510471394	1510472715	1	-0.561.424892	0.902768	0.991176	1.381718	0.901476	0.046659	0.001004	0.002851	0.021528	0.06111	0.880126	
89420170	6381282050	1510471394	1510471394	1510472715	1	-0.561.424892	0.917652	0.999134	1.321197	0.927250	0.046659	0.001004	0.002851	0.021528	0.06111	0.880126	
89820170	6381282051	1510472909	1510472909	1510475194	1	-0.552.678589	0.872641	1.122319	1.341386	0.908417	0.047539	0.001088	0.002928	0.022889	0.06159	0.858448	

3 rows × 93 columns

2) Inspect distortion as function of mean distortion

- interactive wrapper using ipywidgets and Bokeh (similar as in old root + but with interactivity)
- distortion in sector 2,4,6,9,16,20,30 as function of mean distortion (40 minutes sampling)
- secondary parameters could be controlled by user defined sliders

In [15]:

```
bokehDrawPanda?
```

```
Init signature: bokehDrawPanda(self, source, query, varX, varY, varColor, sliderString, p, **options)
Docstring: <no docstring>
Init docstring:
:param source:      input data frame
:param query:       query string
:param varX:        X variable name
:param varY:        : separated list of the Y variables
:param varColor:    color map variable name
:param sliderString: : separated sting - list of sliders var(min,max,step, minValue,maxValue)
:param p:           template figure
:param options:    optional drawing parameters
                   - ncols - number fo columns in drawing
                   - commonX=? ,commonY=? - switch share axis
File:   ~/github/RootInteractive/TTreeHnInteractive/bokehDrawPanda.py
Type:  type
```

In [18]:

```
p3 = figure(plot_width=300, plot_height=200, title="template")
vars="drphiSector2:drphiSector4:drphiSector6:drphiSector7:drphiSector9:drphiSector16:drphiSector20:drphiSector30"
sliders="meanTRDCurrent (0, 0.5, 0.05, 0, 1):H20 (0, 5, 0.2, 0, 5):deltaTRDCurrentNorm (-0.0, 0.1, 0.01, -0.2, 0.2)"
plot0=bokehDrawPanda(dfsplitsplit,"time>0","drphiMean",vars,"H2O",sliders,p3,ncols=3, commonX=1)
```

3) Inspect distortion as function of the TRD flux (2015-2016 data)

- demonstration of interactive graphics using Bokeh+ipywidgets
- distortion increasing with flux (TRD current estimators)
 - background characterization - estimated using the radial profile of TRD currents - not explaining data
 - different bands visible - hypotheses -gas mixture?

In [19]:

```
vars="drphiSector2:drphiSector4:drphiSector6:drphiSector7:drphiSector9:drphiSector16:drphiSector20:drphiSector30"
sliders="meanTRDCurrent(0, 0.5, 0.05, 0, 1);H2O(0, 5, 0.2, 0, 5);deltaTRDCurrentNorm(0.0, 0.1, 0.01, -0.2, 0.2)"
plot=bokehDrawPanda(dfsplit,"time>0","meanTRDCurrent",vars,"H2O",sliders,p3,ncols=3, commonX=1)
```

4) Load fitter and fit distortion as function of other distortions

- prepare data
- register regression method (to be done by expert as in our TMVA interface)
- evaluate and register regression

In []:

```
varFit='drphiSector2'
variableX= ['drphiMean','H2O','iz2x','bz','deltaTRDCurrentNorm']
dataContainer = DataContainer(dfsplit, variableX, varFit, [500,500])
fitter = Fitter(dataContainer)

#fitter.Register_Method('RF200','KerasModel', 'Regressor', layout = [200, 10, 10], epochs=100, dropout=0.1, l1=0.1)
fitter.Register_Method('KNN', 'Neighbors', 'Regressor')
fitter.Register_Method('RF', 'RandomForest', 'Regressor', n_estimators=100, max_depth=10)
fitter.Register_Method('RF200', 'RandomForest', 'Regressor', n_estimators=200, max_depth=10)
fitter.Register_Method('KM', 'KerasModel', 'Regressor', layout = [50, 50, 50], epochs=300, dropout=0.4)
fitter.Fit()
test=dataContainer.Test_sample
#fitter.Compress('KM')
for method in ['RF', 'KNN', 'RF200', 'KM']:
    test = fitter.AppendOtherPandas(method,test)
```

In [21]:

```
fitter.printImportance()
```

```
iz2x 0.0022758054138493646
bz 0.008566811875477474
H2O 0.00881670515985294
deltaTRDCurrentNorm 0.010483321545737336
drphiMean 0.9698573560050829
iz2x 0.0022555640558956642
deltaTRDCurrentNorm 0.01028874299003883
bz 0.016483949046866467
H2O 0.016905959155081654
drphiMean 0.9540657847521169
```

4.1) Bokeh visualization of regression results

In [22]:

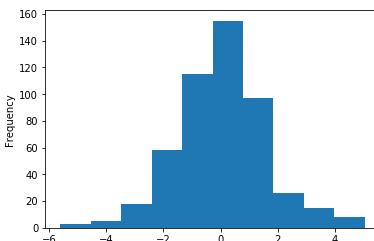
```
p3 = figure(plot_width=400, plot_height=250, title="drphiSector")
plot.drawColzArray(test, " abs (invTRDCurrentNorm)<0.05&year<2017", varFit,"RF:RF200:KNN:KM","H2O",p3,ncols=2)
plot.drawColzArray(test, " abs (invTRDCurrentNorm)<0.05&year<2017", "RF", "RF:RF200:KNN:KM","H2O",p3,ncols=2)
```

In [23]:

```
fitter.AppendStatPandas("RF",test)
fitter.AppendStatPandas("RF200",test)
test=SetAlias(test,"pullRF_2","(drphiSector2-RFMedian)/RFRMS")
test=SetAlias(test,"pullRF200_2","(drphiSector2-RF200Median)/RF200RMS")
drawColzArray(test,"abs(invTRDCurrentNorm)<0.05&year<2017", "RF", "RFMedian","H2O",p3,ncols=3)
test['pullRF_2'].plot.hist()
```

Out[23]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f9ccce8be10>
```



TRD current model

- make regression of the distortion
 - flux (TRD current estimator)
 - background estimators
 - gas composition
- export fit errors

In [30]:

```
variableX= ['meanTRDCurrent','deltaTRDCurrent','bz','bkglMean', 'bkgl2Mean','gascompH2O']
variableX= ['meanTRDCurrent','deltaTRDCurrentNorm','bz','H2O','iz2x', 'bsign','gascompCO2']
variableX= ['meanTRDCurrent','deltaTRDCurrentNorm','bz','iz2x', 'bsign']
x = DataContainer(dfsplit, variableX, ['drphiSector4'], [500,500])
fitter = Fitter(x)
fitter.Register_Method('KNN', 'Neighbors', 'Regressor')
fitter.Register_Method('RF', 'RandomForest', 'Regressor', n_estimators=100, max_depth=10)
fitter.Register_Method('RF200', 'RandomForest', 'Regressor', n_estimators=200, max_depth=10)
#list(variableX)
fitter.Fit()
for method in ['RF', 'KNN', 'RF200']:
    dfsplit = fitter.AppendOtherPandas(method,dfsplit)
fitter.printImportance()
```

In []:

```
fitter.AppendStatPandas("RF",dfsplit)
fitter.AppendStatPandas("RF200",dfsplit)
dfsplit=SetAlias(dfsplit,"pullRF","(drphiSector2-RFMedian)/RFRMS")
dfsplit=SetAlias(dfsplit,"deltaRF","(drphiSector2-RFMedian)")
p = figure(plot_width=500, plot_height=300, title="drphiSector2")
plot.drawColzArray(dfsplit, " trdMeanMedianL0<1 & abs (invTRDCurrentNorm)<0.05&year<2017", "meanTRDCurrent","drphiSector4:RF200","H2O",p,commonX=1,commonY=1)
```

Time series distortion, fit

- distortion vs time
 - distortion-fit vs time
 - (distortion-fit) "predicted" error vs time

In [37]:

```

ptime = figure(plot_width=700, plot_height=150, title="template")
sliders = "meanTRDCurrent(0,0.3,0.03,0,1):H2O(0,0.5,0.2,0,5):deltaTRDCurrentNorm(-0.0,0.1,0.01,-0.2,0.2)""
plot0=bokehDrawPanels(dfsplit,"trdMeanMedianLoc1 & abs(invTRDCurrentNorm)<0.05&year>2017","time","drphiSector4:deltaRF:pullRF:meanTRDCurrent","H2O",sliders,ptime,ncols=1, commonX1=
```

In []:

