# ValidationsTools

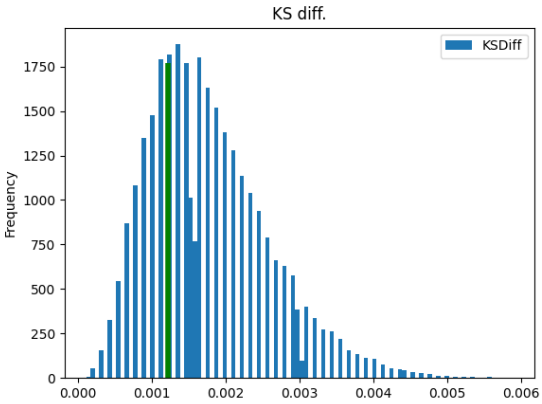
# Table des matières

This repository get all the validations tools such as Kolmogorov-Smirnov (KS Tools) or AutoEncoders Tools (AE).

KS : talk about first dev (1 release vs 1 reference) to (1 reference vs lot of releases). pValues.

From the values of an histogram, we can construct an integrated cumulative curve. From this curve we can take the maximum of the difference between 2 consecutives values. Repeating this operation with a lot of ROOT files (200, 500 or 1000) we can construct a KS curve such here onto the right.

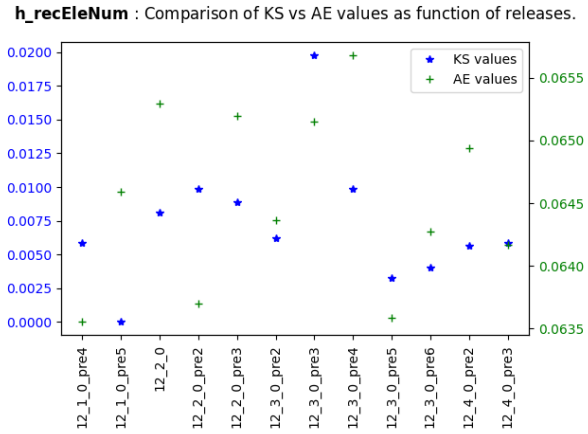
From this curve we can extract a pValue a try to obtain an idea of the validity of the histo and then the release.




AE : analyze on a reference and comparison with a lot of releases. The precedent explanation was made for one release vs 1 reference. Keeping the reference we can compare with a lot of releases (here about 12).

From the ROOT files presented above we can train an autoencoder and try to predict the result of a given entry for one release. Doing this for a tenth of releases we can obtain a comparison, function of the releases, for the pValues or the differences at the end of the AE.

This is shown onto the left here, and extract an idea about the validity of one (or more) release(s).




All thoses files are located into a folder named ValidationsTools which contains at least 5 folders :
- Doc : containing this documentation,
- ZEE_Flow : folder used for the creation of the ROOT files,
- KS_Tools : folder used for the KS pictures and files,
- AutoEncoder : containing the autoencoder tools and results,
- DATA : empty at the installation. Contain the « official » ROOT files, i.e. the releases ROOT files used for the comparison mentioned above.

⚠later create a link to an explanation of the common files used for the ROOT files creation (CommonFiles/path...).
- CommonFiles : with the files which are used by all the tools, KS as AE.

# Launching the installation :

- git clone https://github.com/archiron/ValidationsTools ValidationsTools
- cd ValidationsTools
- chmod 755 *.sh

then launch : . installValidationsTools.sh

This install the library (ChiLib), the release file env (cmsrel $Release $Release) and copy the ZEE_Flow/CCA or ZEE_Flow/LLR files (depending of the site you are working on - the CC or the LLR site) onto the $Release/src/Kolmogorov folder.
$Release is the release you want working with.

# Launching the ROOT files creation :

From the top of the Tools (i.e. ValidationsTools), then launch :
. createROOTFiles.sh
This script launch the creation of the ROOT files, using the step[1-4].py scripts.
You can use own but they have to be similar to the existing ones (see 5).

## Reducing size :

When ALL ROOT files are created, launch (always from the top folder) :
. reduceROOTSize.sh
this will reduce the size of the ROOT files (typically from 150/200 Mo to 1.5/2 Mo), keeping the name of the file.

## Extracting values :

When all the ROOT files are created and reduced, we need to creates 1 file per histo with all the histo values for each ROOT file.
It is the first job of the extractValues[_init].sh scripts.

It can be launched with : . extractValues_init.sh
and create into the RESULTFOLDER (see 6) a lot of text files. All ROOT files are read, and then for each histo, the values of the histo curve is stored into an array, ROOT file after ROOT file.

Once we have all the ROOT files read, the array is stored into a text file dedicated to this histo.
The name is : RESULTFOLDER + 'histo_' + str(leaf) + '_' + '{:03d}'.format(nbFiles) + '.txt' where leaf is the name of the histo, nbFiles the number of the ROOT files (same as NB_EVTS in rootValues.py – see 6).

Example :

| |
|---|
| histo_h_ele_zEff_200.txt |
| histo_h_ele_seedDphi2_200.txt |

Once you have created all those files, you will need to think about comparison. In KS you have a reference release, and onr or more release(s) to compare with (see the beginning of this guide).

# ANNEXE

In that follow, we have the following notation :

al numeric values (int/float) are in **blue**.

## Annexe : ROOT scripts to modify

Here we are talking about the step[1-4].py scripts.

In all stepX.py files, we need to replace the 2 lines :

```
from Configuration.Eras.Era_Run3_cff import Run3

process = cms.Process('SIM',Run3)
```

with :

```
from Configuration.Eras.Era_Run3_cff import Run3

if len(sys.argv) > 1:
print(sys.argv)
print("step 1 - arg. 0 :", sys.argv[0])
print("step 1 - arg. 1 :", sys.argv[1])
print("step 1 - arg. 2 :", sys.argv[2])
print("step 1 - arg. 3 :", sys.argv[3])
print("step 1 - arg. 4 :", sys.argv[4])
ind = int(sys.argv[2])
max_number = int(sys.argv[4])
else:
print("step 1 - rien")
ind = 0
path = ''
max_number = 10 # number of events

max_skipped = ind * max_number

process = cms.Process('SIM',Run3)
```

And also replace :

```
fileName = cms.untracked.string('file:step1.root'),
```

with :

```
fileName = cms.untracked.string('file:step1_' + '%0004d'%max_number + '_' +
'%003d'%ind + '.root'),
```

And for all file name such as step2.root or other.

In the last script (usually step4.py) which generate the DQM file, we have to insert 2 lines between the dqmsave_step line such as :

```
part3 = '/RECO_' + '%0004d'%max_number + '_' + '%003d'%ind
process.dqmSaver.workflow = '/Global/' + 'CMSSW_X_Y_Z' + part3
process.dqmsave_step = cms.Path(process.DQMSaver)
```

# Annexe : paths definitions

RESULTFOLDER : path where you want the created ROOT files are located. It also contain the text files for each histo.

# Annexe : values for ROOT creation

For the creation of the ROOT files we need python files with physics and for those files we need some values.
In general we need a lot files and not only one. So, those file went from **Nbegin = 0** to **Nend=200** or **1000**.
For a lot of tests it can be nice to have smaller values such as :
Nbegin = 20
Nend = 23

Inside the python files we nned to know about the number of events to be used. Most of the « official » runs use **NB_EVTS = 9000**. For some rapid tests we can use :
NB_EVTS = 10