Cut based $H \to \gamma\gamma$ analysis example using CMS Open Data - Technical description

Christian Staufenbiel
Leibniz University Hanover

August 31, 2018

Contents

1 Introduction

This is the technical description to the Higgs analysis example using the CMS Open Data [1]. The analysis code is available to the public at GitHub [2].
This example should serve as an introduction to Higgs analysis using the Open Data portal. To compare to already published results as in [1] and [2] the $H \to \gamma\gamma$ decay channel is used in this analysis. Also not the *multivariate analysis* (MVA) is performed, but an alternative cut based analysis. This approach was also used in the 2011 analysis [1].
To run this analysis the usage of CernVM is recommended, as this provides an environment ready for CMS analyses. If you have other resources to run CMS analyses, you can use these as well.
The next two sections we explain how you can setup all needed tools and run the basic analysis. In the following sections we introduce you to the structure of the source code and how to modify specific parts of the analysis.

2 Environment setup

In this section we assume that you are running the analysis on a CernVM.
First we need to setup a working area, where the CMS environment is setup and the code will

[1] http://opendata.cern.ch/
[2] https://github.com/christian512/hgg-2011

be copied to.

```
mkdir WorkingArea
cd WorkingArea
cmsrel CMSSW_5_3_32
cd ./CMSSW_5_3_32/src
cmsenv
```

All tools used in the analysis are ready to use. We can clone the analysis source code from the GitHub repository and compile it.

```
git clone git://github.com/christian512/hgg-2011.git
scram b
```

As a last step of the setup, databases for accessing the datasets (AOD files) from CERN website are linked.

```
cd hgg-2011/Analyzer
ln -sf /cvmfs/cms-opendata-conddb.cern.ch/FT_53_LV5_AN1_RUNA FT_53_LV5_AN1
ln -sf /cvmfs/cms-opendata-conddb.cern.ch/START53_LV6A1 START53_LV6A1
ln -sf /cvmfs/cms-opendata-conddb.cern.ch/FT53_V21A_AN6_FULL FT53_V21A_AN6
ln -sf /cvmfs/cms-opendata-conddb.cern.ch/FT53_V21A_AN6_FULL.db FT53_V21A_AN6_FULL.db
ln -sf /cvmfs/cms-opendata-conddb.cern.ch/FT53_V21A_AN6_FULL FT53_V21A_AN6_FULL
ln -sf /cvmfs/cms-opendata-conddb.cern.ch/START53_V27.db START53_V27.db
ln -sf /cvmfs/cms-opendata-conddb.cern.ch/START53_V27 START53_V27
```

Now we are ready to start a run of the analysis.

3 Running the analysis

The analysis is split into two parts: **Analyzer** and **PostAnalyzer**.
First we need to convert the raw datasets (AOD files) from the CERN server to ntuples, which are stored locally. We provide a simple shell script (`hgg-2011/Analyzer/run.sh`) to do so. Please take a look inside the script before you run it to get a general understanding of how it calls the analyzer. Before we run the analysis we compile the Analyzer again. Note that the shell script can be called with four different arguments to process different datasets or MonteCarlo simulations. Beware that if you run the analysis on the CernVM this process can take weeks. If you have a computer cluster available, which can handle CMS environment as setup above, you need to edit the shell script to submit jobs to the cluster.
During the analyzer run some soft cuts are applied on the raw datasets to only return events, which are interessting for further analysis.

```
cd hgg-2011/Analyzer
scram b
./run.sh 1
./run.sh 2
./run.sh 3
./run.sh 4
```

The analyzer creates `ROOT-ntuples` which we need to move to the PostAnalyzer directory for further analysis.

```
cd hgg-2011/
mv Analyzer/ntuples-data PostAnalyzer
mv Analyzer/ntuples-mc PostAnalyzer
```

To apply the cuts on the `ntuples`, plot corresponding mass distributions and a simplified significance test we provide three C++ scripts to run. The execution of these scripts should not take longer than two minutes (even on CernVM).

```
cd hgg-2011/PostAnalyzer
```

```
./compile.sh
./hggMakeHist
./hggMakePlots
./pvalPlot
```

This creates plots in the director `hgg-2011/PostAnalyzer/plots`, which can be compared to the results of the plots provided in the published results [1] and [2].
If you are interested in improving the analysis or use this as a template for other analysis purposes you can read through the following sections which give a deeper insight of the source code.

4 Structure

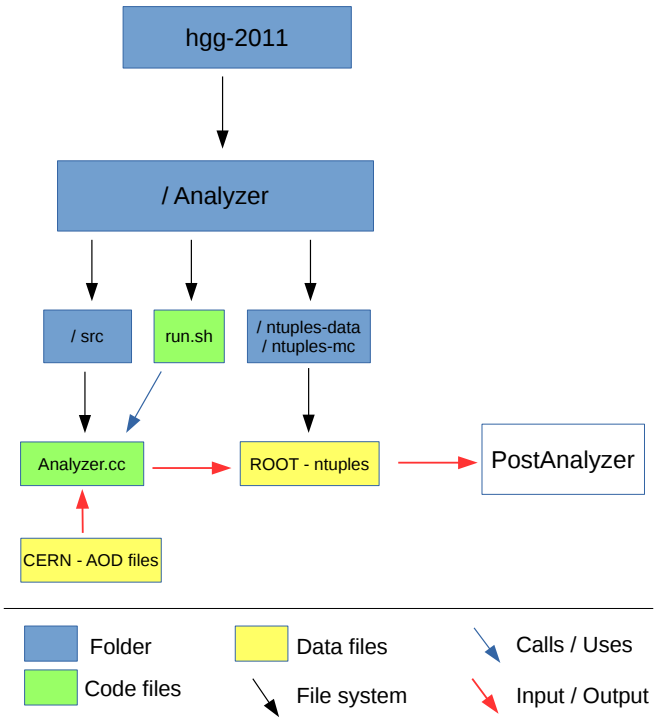

Figure 1: Data structure of `Analyzer`

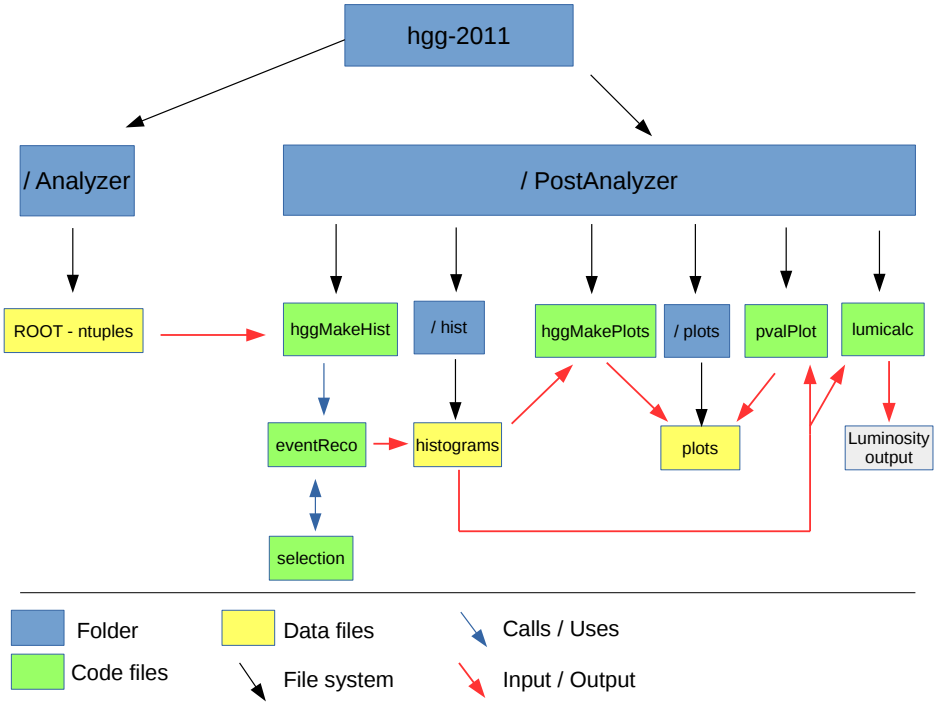

Figure 2: Data structure of `PostAnalyzer`.

The most important parts of the used data structure are summarized in figures 1 and 2. As stated before in the `Analyzer.cc` only soft precuts are applied to reduce the number of events in the `ntuples`. The cuts (in final version) are applied in the `selection.h`. `eventReco.h` delivers a framework for easier access and data storage of the histograms. Also it helps with analyzing data and MC - simulations. `hggMakePlots.cxx` uses the histograms and create plots for the $m_{\gamma\gamma}$ mass distribution. These plots serve as an example and can be optimized for several use cases. In the current state, these can be directly compared to plots from the papers.

5 Modifying files

If you want to edit the cuts, which we applied, the two interesting files are `Analyzer.cc` and `selection.h`.
In `Analyzer.cc`:
The file contains several functions, which are called at specific events. The most important function is `analyze()` which is called at each event. This function uses `SelectPhotons()`, which applies the soft-cuts to variables. Thus this is the main point to edit the physics part of this code. We use flags to refer to different dataset/mc-signals. These flags are tested in `if`-condition and then the corresponding soft cuts are applied.
In `selection.h`:
We here have several functions which are called for the specific datasets. These functions are called within `SelectHgg()`. Cuts are applied in the functions and `histograms` are created from the `ntuples`.
The `histograms` are then used to create the result plots.
Important: After you edited the files you need to compile them again. For the `Analyzer.cc` that is done by the command `scram b` which can be called from anywhere inside `hgg-2011/`. For the three scripts in `hgg-2011/PostAnalyzer/` this can be done by invoking `./compile.sh`. This script will also create all needed folders.

6 Adding more variables to the `ntuples`

To expand the analysis and use more variables these need to be added to the `ntuples`. As this requires a rerun of the `Analyzer`, variables should be chosen wisely beforehand.
In the analyzer a local variable/array is created for each variable in the `ntuples`. Afterwards this variable is added to a `ROOT - Tree`, which is then stored as the `ntuple`-file. For example below the implementation of the ratio of hadronic- to electromagnetic-energy ($\frac{H}{E}$) is shown in the `Analyzer.cc`.

```
// +++++ Analyzer.cc +++++
class Analyzer{
    //...
    private:
        //...
        //creating local array
        float _phHadronicOverEm[_maxNph];
        //...
}
Analyzer::Analyzer(...)
{
    //...
    //Add variable to tree branch
    _tree->Branch("phHadronicOverEm", _phHadronicOverEm, "phHadronicOverEm[Nph]/F");
    //...
}

int Analyzer::SelectPhotons(...)
{
    //...
    //Set the array values
    _phHadronicOverEm[_Nph] = it->hadronicOverEm();
    //...
}
```

To use this variables then in the `PostAnalyzer` we need to add them to the `Tree` which we use there. This is done by editing the file `hgg-2011/PostAnalyzer/tree.h`. Here again an example for the $\frac{H}{E}$ variable is shown.

```
// +++++ tree.h +++++
class ZTree {
public:
    //...
    //create local variable
    Float_t          phHadronicOverEm[maxNph];
    //...
    //Add tree branch
    TBranch        *b_phHadronicOverEm;
    //...
}

void ZTree::Init(...)
{
    //...
    // Add local variable to tree
    fChain->SetBranchAddress("phHadronicOverEm", phHadronicOverEm, &b_phHadronicOverEm);
    //...
}
```

The variable is now available at the tree and can be used for example in the `selection.h` file to provide cuts.

```
//+++++ selection.h +++++
//...
double SelectPh11(...)
{
    //...
    // Example for accessing a tree variable
    // preselTree is a ZTree-pointer here
    if( (preselTree->phHadronicOverEm[ph] > 0.082 && phClass == 3)
    //...
}
```

7 Luminosity calculation

To calculate the luminosity we provide an additional python script `PostAnalyzer/lumicalc.py`. This file needs the luminosity files `2011lumi.txt` and `2012lumi.txt`, which can be downloaded from the Open Data Portal[3][4] and are also provided in the GitHub repository.
It is important to note that you need to **change the trigger selection** here, when you changed it in the Analyzer or PostAnalyzer.

References

[1] CMS Collaboration. Search for the standard model Higgs boson decaying into two photons in pp collisions at $\sqrt{s} = 7$ TeV. *Physics Letters B*, 710(3):403–425, Apr 2012.

[2] CMS Collaboration. Observation of the diphoton decay of the higgs boson and measurement of its properties. *The European Physical Journal C*, 74(10), Oct 2014.

[3] http://opendata.cern.ch/record/1051
[4] http://opendata.cern.ch/record/1052