ATLAS NOTE

6th October 2016

Statistical analysis tool used in ATLAS dijet resonance searches

Lydia Beresford, Katherine Pachal

Abstract

This note describes the the statistical analysis tool which is used in the ICHEP low mass dijet resonance search with $\sqrt{s}$ = 13 TeV data. The tool includes a frequentist 'search phase' which fits a function to the dijet invariant mass distribution and uses the BumpHunter algorithm to search for, and quantify, any local excesses. If no significant excess is observed then the 'limit setting phase' is performed. In this phase a bayesian approach is used to set limits on New Phenomena using code based on the Bayesian Analysis Toolkit (BAT).

Contents

1 Introduction to the package structure and order of running

The structure of the StatisticalAnalysis package is displayed in Figure 1, along with the additional packages that it requires.

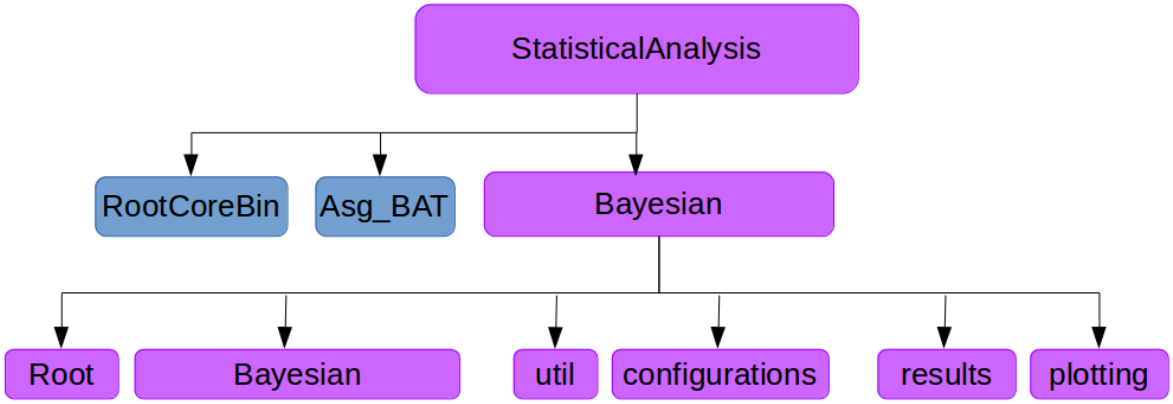

Figure 1: The structure of the StatisticalAnalysis package is displayed (purple boxes), along with the additional packages that it requires (blue boxes).

Within the `Bayesian` directory, the `util` directory contains the following C++ source files:

- SearchPhase.cxx
- SetLimitsOneMassPoint.cxx
- LimitSettingPhase.cxx
- doGaussianLimits.cxx

These are the main files that perform the statistical analysis and they are run in the order shown above. Their corresponding configuration files are stored in the `configurations` folder. The supporting C++ source code containing the classes and functions used in main files are in the `Root` folder with their own header files stored in the `Bayesian` folder. When the main files are run their outputs are stored in the `results` folder, and these outputs are used to make more meaningful plots using the `plotting` package, which is written in python.

Section 2 explains how to download the StatisticalAnalysis package and also how to run it and plot the results. An explanation of each of the main source files is provided in section 7 (search phase) and in section 8 (limit setting phase), these sections can be skipped if you are familiar with the package. Section 3 shows examples of the plots that are produced when following the quick start guide. Section 4 gives common mistakes and the solutions to them. Section 5 and section 6 provide examples of common modifications that can be made to the code for the search phase and limit setting phase, respectively.

2 Quick start guide to downloading and running

A quick start guide is presented to explain how to download and setup the code. How to run the search phase is outlined in section 2.1 and how to run the limit setting phase is outlined in section 2.2, and

how to run the gaussian limit setting is outlined in section 2.3.

Some commands span more than one line due to text wrapping, if lines following initial command are indented then they belong to the initial command and any extraspaces should be removed when copying the command.

Instructions for setting up the code for the **first time**:

Step 1 Starting in a directory of your choice, make a directory for the statistical analysis and enter this directory:

```
mkdir StatisticalAnalysis
cd StatisticalAnalysis
```

Step 2 Set-up ATLAS environment:

```
setupATLAS
```

Step 3 Obtain setup script:

```
svn export svn+ssh://svn.cern.ch/reps/atlasphys-exo/Physics/Exotic/JDM/
    DiJetISR/Run2/Code/StatisticalAnalysis/Bayesian/trunk/scripts/setup-
    StatisticalAnalysis.sh
```

Step 4 Source setup script:

```
source setup-StatisticalAnalysis.sh
```

NOTE: If desired/to remain consistent with high-mass dijet analysis/TLA, can use 'core' code from their svn repository i.e. copy over `Root`, `Bayesian`, and `util`folders from `svn+ssh://svn.cern.ch/reps` your local code.

Step 5 Inputs (read all of this step before downloading inputs):

The code requires:

- A data (or MC) background m_{jj} distribution
- Signal MC m_{jj} distributions
- Theory lines (for plotting)
- Signal templates (for plotting)

If you are re-producing the ICHEP conference version of the analysis then the ICHEP inputs are located in:

```
eos/atlas/atlascerngroupdisk/phys-exotics/jdm/dijet/statsinputs/RunII/
    ICHEP_DijetISR/inputs/
```

This directory should be copied to your local `Bayesian` directory, retaining the name `inputs`. The contents of `inputs` directory is as follows:

- `hist_20160727`(Contains: data background m_{jj} distributions for both analyses)
- `hist_20160801`(Contains: MC background m_{jj} distributions for both analyses)
- `NoTails`(Contains: Z' MC m_{jj} distributions, nominal and JES shifted, chopped to same range in mass that the fit spans. **NOTE**: Used `ChopTails_gjj.py`and `ChopTails.py`scripts to produce these 'chopped' signal templates, user can adjust range to chop in the scripts. This is done so acceptance matches between these templates and those used to calculate theory curves. Theory curves only uses fit range integral over signal cross-section templates, rather than integrating over full signals.)
- `inDir`(Contains: Z' MC m_{jj} distributions, with no mass cut, and chopped to fit range. `inDir`should be moved to `StatisticalAnalysis/Bayesian/scripts/CrystalBall`used to calculate JES uncertainty values for gaussian limits by running `doFitsNoMjj_3Sig.py`script in the same directory)
- `xsecandacceptance` (Contains: Theory line and signal templates)

If using other inputs i.e. non-ICHEP then the usual procedure is to:
Make a directory in the Bayesian package to store your inputs (if it doesn't exist already):

```
cd Bayesian
mkdir inputs
```

The data and MC inputs can be found in the following directory:

```
/eos/atlas/atlascerngroupdisk/phys-exotics/jdm/dijet/inputs/
```

or produced by the user. Note that histograms are required as inputs, NOT trees, so it is not necessary to copy across any `tree` directories. The corresponding theory and signal templates can be found in the

```
eos/atlas/atlascerngroupdisk/phys-exotics/jdm/dijet/statsinputs/RunII/
    inputs/
```

directory, or produced by the user.

Instructions for setting up the code each **subsequent time**:

Step 1 Setup the package and execute it whenever you log in:

```
cd StatisticalAnalysis
setupATLAS
rcSetup
rc compile
cd Bayesian
. Setup.sh
```

The code should be compiled before running whenever changes to source files or to header files have been madev.

2.1 Performing the search phase

Performing the search phase involves running `SearchPhase.cxx`, which takes the dijet invariant mass histogram as an input. The `Run_gjjSearchPhase.py`script and the `Run_jjjSearchPhase.py`script are used in order to run `SearchPhase.cxx` in the util directory which uses the configuration files `Step1_SearchPhaseNoSyst_gjj_4Par.config`and `Step1_SearchPhaseNoSyst_3Par.config`, respectively, in the configurations directory. The script creates results, plots and log files directories for you and puts all code outputs into the relevant folders in a sub-directory with a name of your choice (specified by folderextension in `Run_gjjSearchPhase.py`or in `Run_jjjSearchPhase.py`). The γ + jet jet (gjj) analysis and the 3 jet (jjj) analysis each have separate scripts which must be run in order to perform the analysis, the example below is only for the gjj analysis, but the steps are the same for the jjj analysis, using its own scripts and files.

Step 1 The user should modify the fields in the *****User specifies ***** section of `Run_gjjSearchPhase.py` to suit their inputs and requirements (Currently set-up to re-produce ICHEP result).

Step 2 If necessary, the user should modify any fields in `Step1_SearchPhaseNoSyst_gjj_4Par.config` that are not over-written by `Run_gjjSearchPhase.py`, any fields that are over-written are labelled in the configuration file, e.g. fit function starting parameters (Currently set-up to re-produce ICHEP result).

Step 3 Run the script as follows:

```
python Run_gjjSearchPhase.py
```

Results can be found in: `results/Step1_SearchPhase/folderextension` where `plotextension` is a name specified by the user in the `Run_gjjSearchPhase.py` script. Plots are stored in the `plotting/SearchPhase/plots/folderextension`. For examples of some of the plots that should be produced, see section 3.1, these can be compared to the plots that you've just obtained.

Note: To change the ATLAS label displayed on the plots (e.g. Internal, Preliminary, etc.), modify the number in the following line `myPainter.setLabelType(2)` in `plotting/SearchPhase/plotSearchPhase_gjj.py`, and re-run `Run_gjjSearchPhase.py` with only `doPlotting`set to true.

Aside: To run the search phase manually the following commands can be used:

```
SearchPhase --config configurations/Step1_SearchPhaseNoSyst_gjj_4Par.
   config
python plotting/SearchPhase/plotSearchPhase_gjj.py
```

2.2 Performing the limit setting phase

Running the limit setting phase for models involves first running `setLimitsOneMassPoint.cxx` and then `LimitSettingPhase.cxx`.`setLimitsOneMassPoint.cxx` takes the output root file from the search phase as an input, along with histograms of the nominal and JES shifted signal MC mjj spectra, for each mass point. The outputs of `setLimitsOneMassPoint.cxx` are a root file for each mass point. The single mass points are then scaled by luminosity and graphed in `LimitSettingPhase.cxx`.

The Run_gjjLimits_Splits.pyand the Run_jjjLimits_Splits.pyscript is used in order to run setLimitsOneMassPoint.cxx in the util directory which uses the configuration file Step2_setLimitsOneMassPoin the Step2_setLimitsOneMassPoint_ZPrimemR.configfiles, respectively, in the configurations directory, and to run LimitSettingPhase.cxx in the util directory which uses the configuration file Step3_LimitSettingPha the Step3_LimitSettingPhase_ZPrimemR_gSM0p30.configfiles, respectively, for example, in the configurations directory. The script creates results, plots and log files directories for you and puts all code outputs into the relevant folders in a sub-directory with a name of your choice (specified by plotextension in Run_gjjLimits_Splits.pyor Run_jjjLimits_Splits.py). The γ + jet jet (gjj) analysis and the 3 jet (jjj) analysis each have separate scripts which must be run in order to perform the analysis, the example below is only for the gjj analysis, but the steps are the same for the jjj analysis, using its own scripts and files.

Step 1 The user should modify the fields in the *****User specifies ***** section of Run_gjjLimits_Splits.py to suit their inputs and requirements (Currently set-up to re-produce ICHEP result).

Step 2 If necessary, the user should modify any fields in Step2_setLimitsOneMassPoint_gjj_XXX.config and Step3_LimitSettingPhase_gjj_XXX_YYY.config (where XXX is the signal Model and YYY is the coupling) that are not over-written by Run_gjjLimits_Splits.py, any fields that are over-written are labelled in the configuration files (Currently set-up to re-produce ICHEP result).

Step 3 Run the script as follows:

```
python Run_gjjLimits_Splits.py
```

The first time the script is run, only dosetLimitsOneMassPoint should be set to True and doLimitSettingPhase and doPlotting are set to False, such that only setLimitsOneMassPoint.cxx is run.

This step should be run for each signal and coupling you are setting limits on!

To re-produce the ICHEP result, run once with the 'gSM0p10' part in the Signal information section uncommented, then comment this part out and uncomment the 'gSM0p20' part and re-run, etc. until all coupling values have been run on. Once all batch jobs are completed then dosetLimitsOneMassPoint should be set to False, doLimitSettingPhase and doPlotting are set to True, such that only LimitSettingPhase.cxx and plotLimitSetting_gjj.py are run, and again this is run twice, once for each signal and coupling type.

Results can be found in: results/Step2_setLimitsOneMassPoint/plotextension and results/Step3_LimitSettingPhase/plotextension where plotextension is a name specified by the user in the Run_gjjLimits_Splits.py script. Plots are stored in the plotting/LimitSettingPhase/plots/plotextension and all Log files are stored in StatisticalAnalysis/LogFiles/plotextension. For examples of some of the plots that should be produced, see section 3.2, these can be compared to the plots that you've just obtained, also the limit values and bump low and high edge values are provided for comparison.

NOTE: Currently `plotting/LimitSettingPhase/plotLimitSetting_gjj.py`divides the limit curves, uncertainty bands and theory curve by the photon signal efficiency values in `EffDict`in the script. The values in this dictionary should be recalculated and updated for any future iterations of the analysis. You will notice that in `plotting/LimitSettingPhase/plotLimitSetting.py`the dictionary is commented out and un-used, this is because for the jjj analysis, jet reconstruction is $\sim$ 100 % so it is not necessary to divide by the efficiency.

Note: To change the ATLAS label displayed on the plots (e.g. Internal, Preliminary, etc.), modify the number in the following line `myPainter.setLabelType(2)` in `plotting/LimitSettingPhase/plotLimitSetting_gjj.py`, and re-run `Run_gjjLimits_Splits.py`with only `doPlotting`set to true for each signal type.

Aside: To run the limit setting manually the following commands can be used, for example:

```
setLimitsOneMassPoint --config configurations/
    Step2_setLimitsOneMassPoint_gjj_ZPrimemR.config --mass 250

LimitSettingPhase --config configurations/LimitSetting/
    Step3_LimitSettingPhase_ZPrimemR_gSM0p20.config

python plotLimitSetting_gjj.py
```

2.3 Performing the gaussian limit setting phase

Running the limit setting phase for gaussian signals involves running `doGaussianLimits.cxx`. `doGaussianLimits.c` takes the output root file from the search phase as an input, the outputs of `doGaussianLimits.cxx` are a root file for each mass point and gaussian width. The single mass points are then scaled by luminosity and plotted in `plotting/LimitSettingPhase/plotGaussians.py`or `plotting/LimitSettingPhase/plotGauss`

The `calculateGaussianLimits_gjj.py`and the `calculateGaussianLimits_jjj.py`scripts are used in order to run `doGaussianLimits.cxx` in the util directory which uses the configuration file `Step4_GenericGaussians_gjj.config`and the `Step4_GenericGaussians.config`files, respectively, in the configurations directory. The script creates results, plots and log files directories for you and puts all code outputs into the relevant folders in a sub-directory with a name of your choice (specified by plotextension in `calculateGaussianLimits_gjj.py`or `calculateGaussianLimits_jjj.py`). The γ + jet jet (gjj) analysis and the 3 jet (jjj) analysis each have separate scripts which must be run in order to perform the analysis, the example below is only for the gjj analysis, but the steps are the same for the jjj analysis, using its own scripts and files.

Step 1 The user should modify the fields in the *****User specifies ***** section of `calculateGaussianLimits_gj` to suit their inputs and requirements (Currently set-up to re-produce ICHEP result).

Step 2 If necessary, the user should modify any fields in `Step4_GenericGaussians_gjj.config`that are not over-written by `calculateGaussianLimits_gjj.py`,any fields that are over-written are labelled in the configuration files (Currently set-up to re-produce ICHEP result). Note, gaussians don't use JES templates, they use values specified in the config file. For the ICHEP result flat

values were calculated using the following script: `scripts/CrystalBall/doFitsNoMjj_3Sig.py`uncertainty values should be updated for each iteration of the analysis.

Step 3 Run the script as follows:

```
python calculateGaussianLimits_gjj.py
```

The first time the script is run, only `doGaussianLimits` should be set to True and `doPlotting` are set to False

Note: Currently script only set up to work on Oxford (torque) batch, so user can either run jobs locally, or set up to run on lxplus batch system, like is done in other Run scripts. In order to plot results `doGaussianLimits`should be set to False, `doPlotting`is set to True, such that only `plotGaussians_gjj.py`is run.

Results can be found in: `results/Step4_GaussianLimits/plotextension`where `plotextension` is a name specified by the user in the `calculateGaussianLimits_gjj.py` script. Plots are stored in the `plotting/LimitSettingPhase/plots/plotextension` and all Log files are stored in `StatisticalAnalysis/LogFiles/plotextension`. For examples of some of the plots that should be produced, see section 3.2, these can be compared to the plots that you've just obtained.

NOTE: Currently `plotting/LimitSettingPhase/plotGaussians_gjj.py`divides the limit curves, uncertainty bands and theory curve by the average photon signal efficiency value from different width Z' signals. The value should be recalculated and updated in all places with the comment `Dividingby0.81` for any future iterations of the analysis. You will notice that in `plotting/LimitSettingPhase/plotGaussians.py`the limits are not divided by efficiency, this is because for the jjj analysis, jet reconstruction is $\sim$ 100 % so it is not necessary to divide by the efficiency.

Note: To change the ATLAS label displayed on the plots (e.g. Internal, Preliminary, etc.), modify the number in the following line `myPainter.setLabelType(2)` in `plotting/LimitSettingPhase/plotGaussians_gjj.py`, and re-run `calculateGaussianLimits_gjj.py` only `doPlotting`set to true for each signal type.

3 ICHEP plots for comparison

Examples of the plots and values that should be produced when following the above instructions are shown below, for both the search phase (Section 3.1) and the limit setting phase (Section 3.2) . **Note** that some of your plots and values might vary slightly in appearance to the ones shown due to the use of pseudo-experiments and Markov Chain Monte Carlo in the statistical code, which can give slightly different results each time the code is run.

3.1 Search phase plots

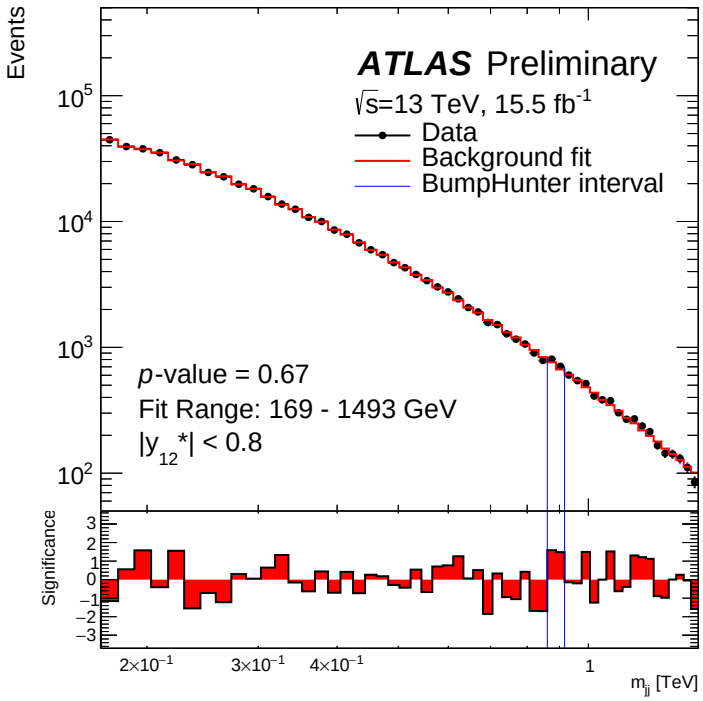

Figure 2: figure1.pdf

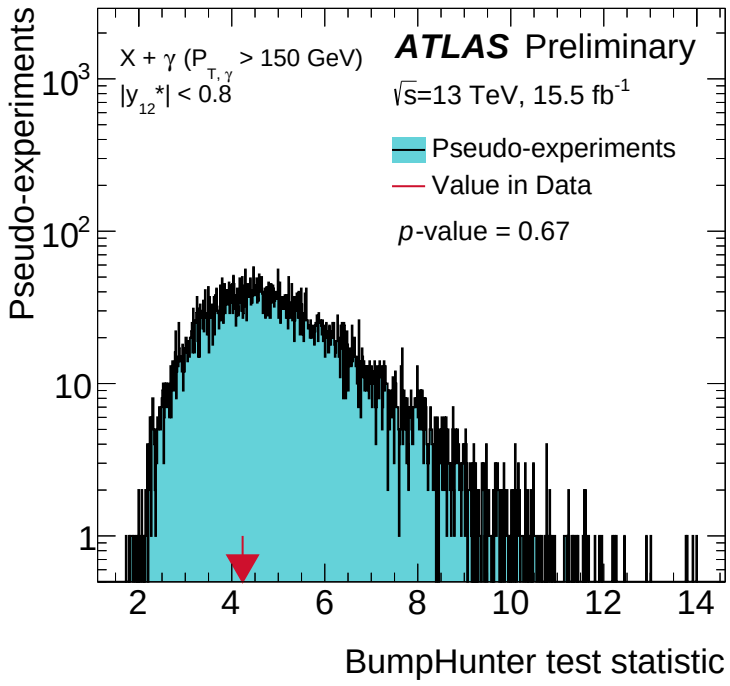

Figure 3: bumpHunterStatPlot.pdf

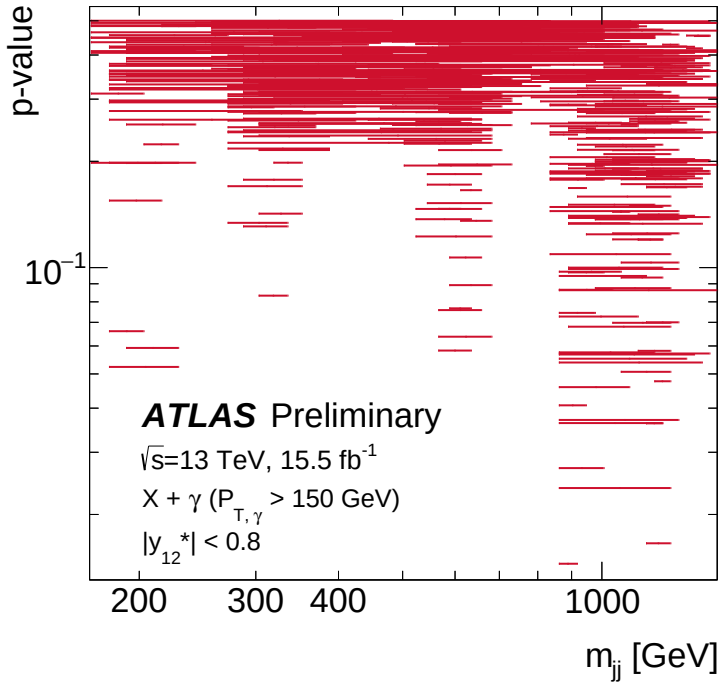

Figure 4: bumpHunterTomographyPlot.pdf

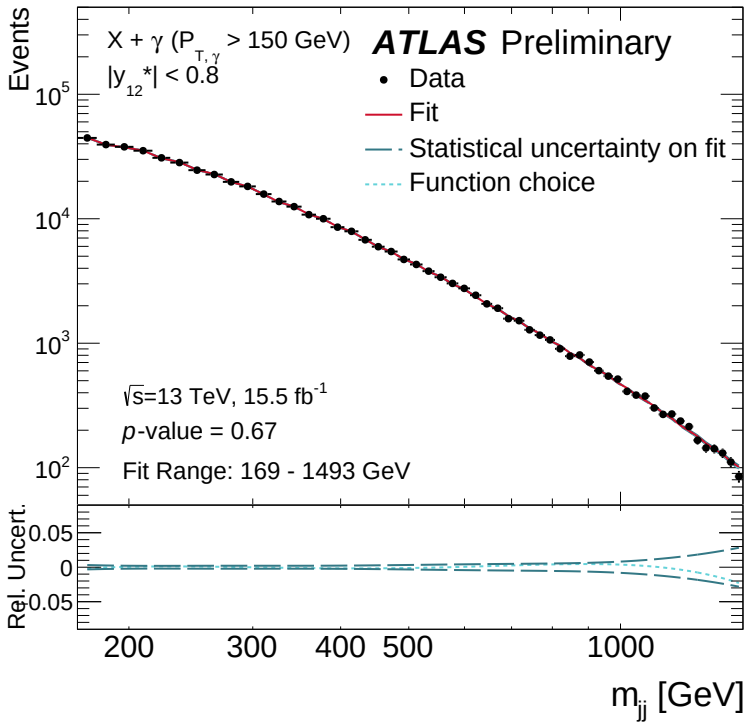

Figure 5: compareFitQualityAndFitChoice_Asymm_WithRatio.pdf

3.2 Limit setting plots

Paper values for comparison:
Bump low, high edges are 861 - 917 GeV

Signal ZPrime gSM0p20
Observed limit at 95% CL: 315.54
Expected limit at 95% CL: 374.28

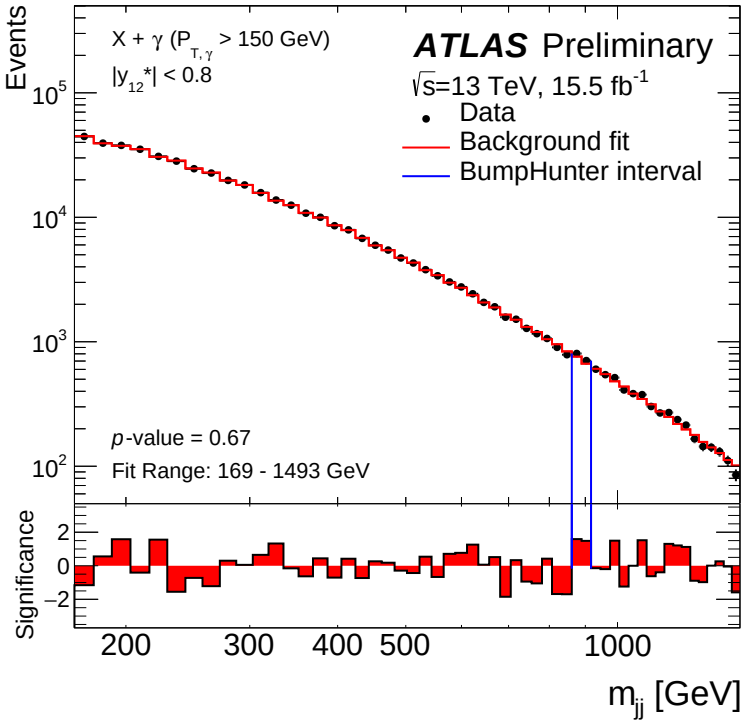

Figure 6: FancyFigure1WithFitLabels_NoSignals.pdf

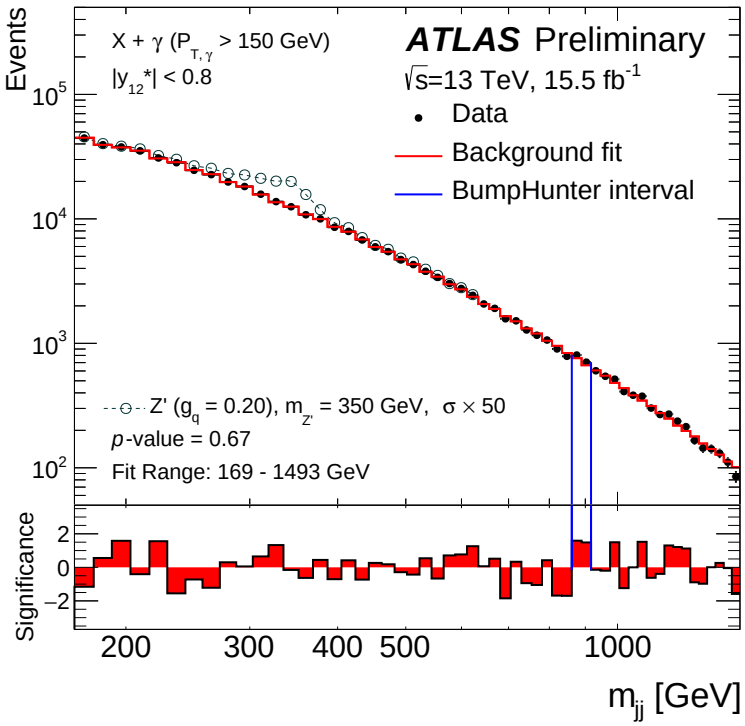

Figure 7: FancyFigure1WithFitLabels_ZPrime0p20.pdf

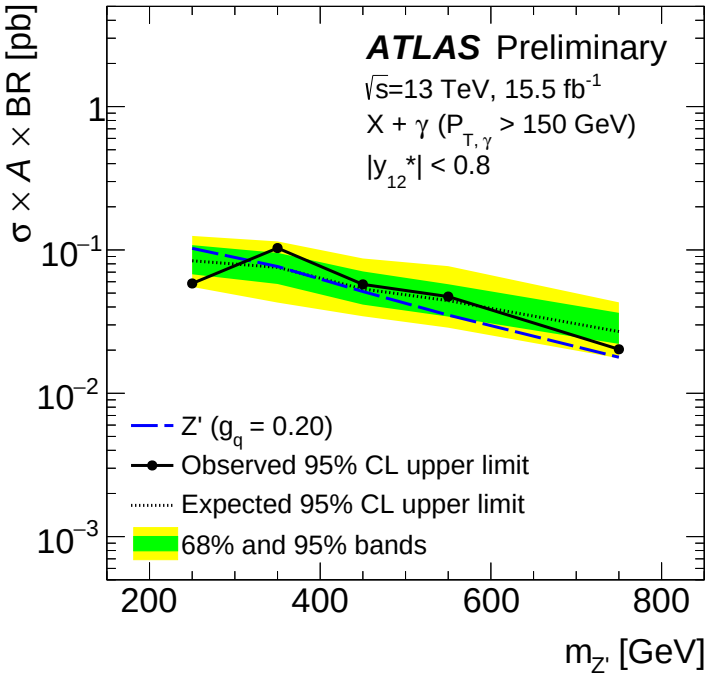

Figure 8: Limits_ZPrime0p20.pdf

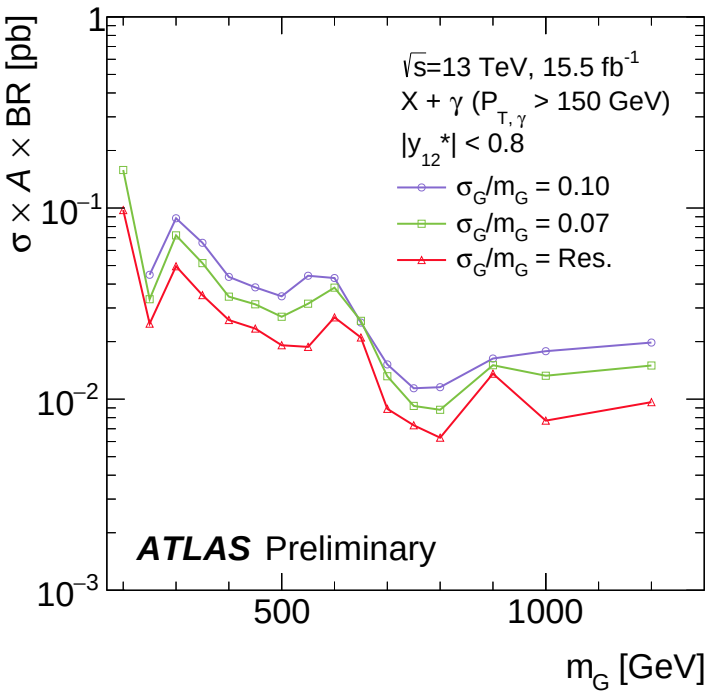

Figure 9: GenericGaussians.pdf

4 Common mistakes

Error message when plotting:

`Tell ImportError: No module named art.morisot`

You forgot to run the setup script to set the python path etc. Solution:

`. Setup.sh`

HERE onwards needs to be updated in the note!!

5 Examples of modifications that can be made to the code (search phase)

5.1 Selecting the range and initial fit function parameters for your mass spectrum

The range (`minXForFit`and `maxXForFit`) are specified in `SearchPhase.config`as shown in appendix B, and by default if these are set to -1 then the start and end of the data are used. Note that only bins which fully contain the specified range are used e.g. if 1099 is specified for minXForFit then the next bin edge above this (e.g. 1100 GeV) is where the fit will begin from.

There is no definite way in which the initial fit function parameters should be chosen, but some tips will be presented here.

- As a starting point, `Parameters.txt` in the `configurations` directory provides some existing start parameters, so you can search for parameters for a sample with a luminosity similar to the one you are using.

- It should be noted that the first parameter for both the 3 and 4 parameter fit functions control the normalisation, so for example, the parameters for the 1 inv fb histograms could be used with the first parameter scaled to correspond to the luminosity you are using.
- If good start parameters are found for the 3 parameter fit, then the 'alternative' fit (4 parameter function) the same initial start parameters as the 3 parameter function can be used, with the following two changes, parameter 3 should be set to the negative value of what was used for the 3 parameter fit (due to the way the 3 and 4 parameter functions are defined) and parameter 4 should be set to 0.
- If reasonable start parameters are found then run the search phase once to obtain the parameter values after fitting. These should be used as your new initial start paramters in the future (for this sample).

5.2 Using a different fit function

6 Examples of modifications that can be made to the code (limit setting phase)

Adding in mass points and signals. If the user wishes to add another signal, other than QStar, then a new configuration file should be made e.g. `Step2_setLimitsOneMassPoint_WStar.config`, where WStar is the name chosen by the user, and in `Run_Limits.py`this signal and it's mass points should be added to the Signals dictionary, in the same way as for QStar, but using the name chosen by the user, e.g. WStar.

7 Search phase explanation

The aim of the search phase is to establish whether there are any statistically significant excesses present in the measured dijet invariant mass spectrum. In order to do this the expected background from Standard Model particles, our *background only hypothesis*, is determined by fitting the spectrum with a smooth function. The function is chosen such that it does not accommodate the type of deviations which would be introduced by a resonance reference paper or write about it. The fitting procedure is described in more detail in section Section 7.2.

Once the fit has been performed, the resulting background only hypothesis is compared to our spectrum using a *test statistic*. A test statistic is used as this provides us with a single value which increases monotonically with decreasing compatibility, giving us a way to quantify the compatibility between our spectrum and the hypothesis. A variety of test statistics are available for analysers to use, and these are discussed in more detail in section 7.3.

By calculating the *p-value* ref of a test statistic we answer the question "What is the probability of observing data at least as extreme as the measured spectrum, given that the Standard Model is true?", and we can reject the background only hypothesis if the p-value is lower than an agreed cutoff value. The rejection of the background only hypothesis would indicate that the hypothesis is not sufficient to describe the data and may point to new physics. An explanation about how this is implemented in the code is provided in section 7.4 . A more practical discussion about the inputs to the code and the weighting of events in our spectrum will now be presented. mention/expl freq vs bayesian and freq search phase and bayesian limit setting, and why? LOOK http://arxiv.org/pdf/1101.0390v2.pdf !!!!!!

7.1 Search phase inputs and weighting

The input to the search phase is a root Ntuple containing a dijet invariant mass spectrum after analysis selection and the `SearchPhase.config` configuration file which is described in appendix B. The main file which performs the search phase is the `SearchPhase.cxx` file in the `util` directory. This file starts off by reading in the information from the configuration file and accessing the input Ntuple which contains the dijet invariant mass spectrum.

If the input dijet invariant mass spectrum was produced using of multiple triggers with different prescales, then the spectrum will contain weighted events and the contents of a bin may be substantially different from the number of raw events that were used to fill it. In order to retain information at both levels, both at the weighted level and at the level of the raw events, the dijet invariant mass spectrum contained in the Ntuple is passed to the class `MjjHistogram`,and an instance of this class called `theHistogram` is then created. The source file for `MjjHistogram` is in the `Root` directory and the corresponding header file is in the lower level `Bayesian` directory.

The `MjjHistogram` class is a wrapper and contains three key histograms, which were produced from the input dijet invariant mass spectrum. These histograms are as follows:

- The basic data histogram i.e. the weighted dijet invariant mass spectrum.
- The effective histogram, which has a value in each bin equal to the equivalent statistical power of the events in that bin in the basic histogram. explain more and explain how calculated!
- The weights histogram, which has a value in each bin equal to the effective weight of each event in that bin in the effective histogram, and is used to convert between basic and effective histograms.

explain where each used? and how calc eff histo and weights Throughout the code `theHistogram` is used and is passed to all user-accessible functions such that the correct form can be accessed by the program. All comparisons between the observed spectrum and a hypothesis makes use of the effective statistics in the observed spectrum and an appropriately scaled version of the hypothesis, unless otherwise stated. stat uncerts! expl weighting below above 1TeV?.

7.2 Fitting a smooth function to the data

The background is parameterised by fitting a smooth functional form to the observed dijet invariant mass spectrum. In the 8 TeV dijet resonance a four parameter fit function that was chosen, and it has the following form: XXX use equation mode

where x XXX s is the ratio of the dijet invariant mass to the center of mass energy and the p i are free parameters. The number of parameters and their default values are read in from the `SearchPhase.config` file and they are pushed back into the `paramDefaults` vector. The starting point for the fit `minXForFit` is also read in from the configuration file, and is stored as `minX`in. The fit picks the low edge of the bin that contains the position `minXForFit` and fits down to that value. If `minXForFit` is within a bin that is below the first bin with data then `minX`is reset to the first bin with data in the code. If `maxXForFit` is not specified in the configuration file then the end of the data is used instead.

The next step is to pass these values to the fit function. In order to do this an instance of the class `FourParamFitFunction` is created called `theMjjFitFunction`which takes in the inputs `minXForFit`,maxXForFit and Ecm, which are defined in the `SearchPhase.config` file. `FourParamFitFunction` is defined in the `MjjFitFunction.h` file in the lower level `Bayesian` directory, and this is where the fit functions

are defined. For more information about creating your own fit fuction refer to section XXX. All fits that are created are daughter classes of the class `MjjFitFunction`which is defined in the files XXX ..

7.3 Available test statistics

Now the fit has been performed, we have a background hypothesis which we can compare to the observed spectrum. As mentioned in section 7, a test statistic is used in order to quantify the compatibility between the hypothesis and the observed spectra, as it provides us with a single value which increases monotonically with decreasing compatibility. Three different values are used as test statistics in the search phase.

7.3.1 The BumpHunter

The first statistical test which is created in `SearchPhase.cxx` is the BumpHunter ref, via the line:
`MjjBumpHunter theBumpHunter;`
`theBumpHunter` is an instance of the class `MjjBumpHunter`, which is defined in `MjjBumpHunter.cxx` in the `Root` directory, with it's corresponding header file in the lower level `Bayesian` directory. The BumpHunter starts with a two bin-window, set by the line `theBumpHunter.SetMinBumpWidth(2);`, and it shifts its location such that all two bin-windows (i.e. neighbouring two bins) have been tested. The size of the window is then increased by one bin and the process is repeated, this continues until the bin-window range is half the number of bins in the full observed spectrum where max bin range set?.

7.4 p-value determination

show these plots or final paper ones OR ones from reduced code and compare to paper!!!! locations in code? refer to change fit function section? put other test statistic p val plots in appendix?

7.5 Representing significance bin-by-bin

7.6 The resulting background only hypothesis

8 Limit setting phase explanation

In the case that there is no statistically significant excess found in the dijet invariant mass spectrum, then the limit setting phase is performed. This phase is carried out in order to set limits on the rate of production of a new particle or process. Each new particle that is considered is described by a particular model, which describes the shape of the resonance the presence of this particle would create, and also the cross section for the production of this particle for given particle masses. When setting limits on this new particle or process, the question that we ask for each particle mass is as follows; "What is the probability that this particle with this mass exists, given our observed data". In order to answer this question an upper limit on the number of 'new particle events' that could be contained in the data is determined. This upper limit is then converted into the maximum allowed cross-section for the new particle at each mass. interpolate? The intersection between this 'observed' limit and the nominal cross-section for the new particle or process gives the highest mass of this new particle that we can exclude 95 percent level?. The details of the limit setting are described below, together with references to where each step is carried out in the code.

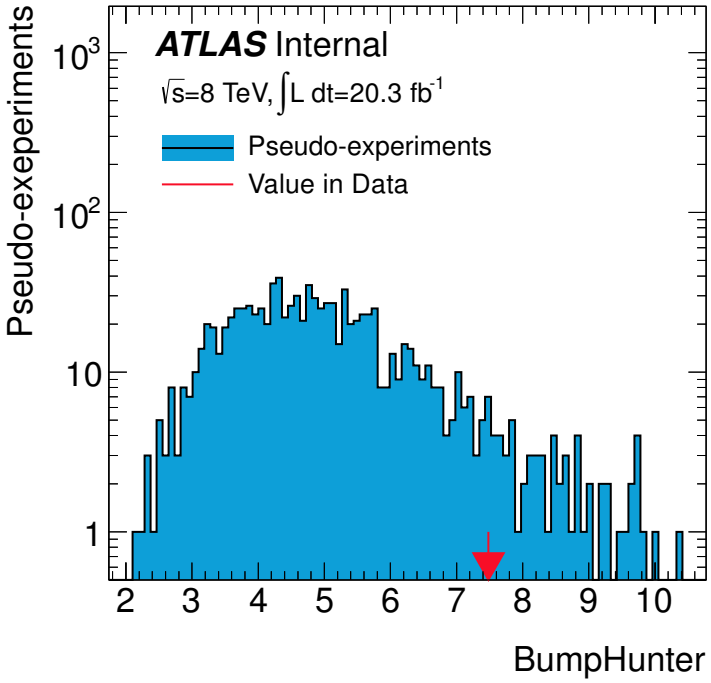

Figure 10: An example ATLAS figure2 UPDATE TO REDUCED VERSIONchange lumi etc and labels.

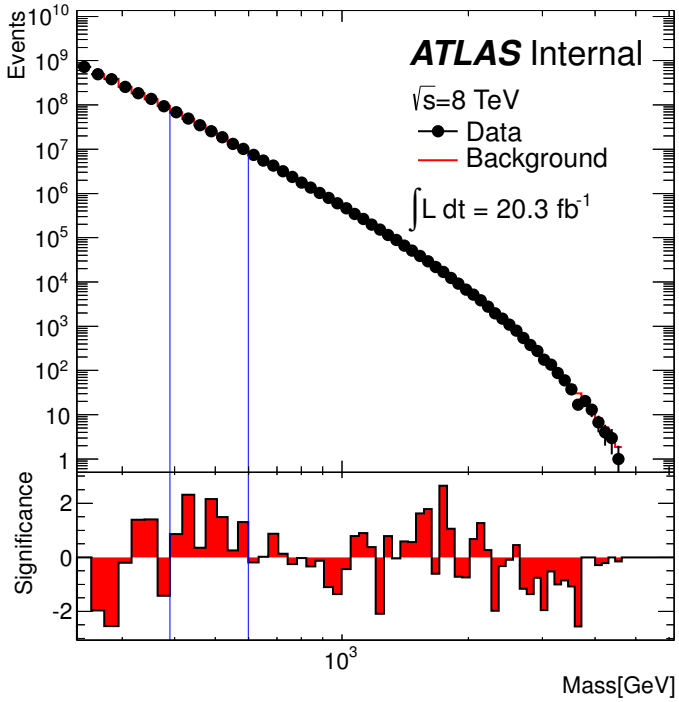

Figure 11: An example ATLAS figure. UPDATE TO REDUCED VERSION. change lumi etc and labels!

9 Summary and conclusion

Acknowledgements

Appendices

A Provided inputs to the code

- Data15_C_fGRL_JetCalibCorrection_20150721/dataHistograms.PeriodC.root:The input Ntuple to the code, which contains the dijet invariant mass histogram after analysis cuts.
- xsecandacceptance : Directory containing CrossSectionsFor13Plotting.root and Templates_QStarQBH_1fb.root . The former contains the theory line for the limit plot and the latter contains the signal templates used for the 'Fancy_Figure' plots of the mjj spectrum with overlaid signal bumps.

B Step1_SearchPhase.config

```
##############################################
#                                            #
#  Config file for Bayesian                  #
#                                            #
##############################################

#set all the parameters of your analysis
#IMPORTANT: don't leave spaces after the parameters!


########################################
# input/output
########################################

# This contains the data spectrum which will be analysed
# Value overwritten if use Run_SearchPhase.py
inputFileName ./inputs/Data15_C_fGRL_JetCalibCorrection_20150721/dataHistograms.PeriodC.roo

# Value overwritten if use Run_SearchPhase.py
dataHist Nominal/mjj_Data_PeriodC_0p072fb

# Value overwritten if use Run_SearchPhase.py
outputFileName ./results/Step1_SearchPhase/Data15_C_fGRL_JetCalibCorrection_20150721/Step1_

########################################
# general
########################################

# Center-of-mass energy of the spectrum studied
# Value overwritten if use Run_SearchPhase.py
Ecm        13000.0

# Number of pseudoexperiments to run
```

```
# Matches 8 TeV paper value
nPseudoExp 10000

########################################
# fitting
########################################

# To use min of data put -1 (Use 1099 so fit starts from bin above, i.e. from 1100 GeV)
minXForFit       1099

# use default: maximum of data
maxXForFit       -1

# 13 TeV 3 param fit function:
# this will be nominal function for now.
functionCode 9
nParameters  3

# For Up to Period C4 data (72 inv pb)
parameter1   0.00307775
parameter2   15.1199
parameter3   -4.57371

# 13 TeV 4 param fit function:
doAlternateFunction     true
alternateFunctionCode 4
alternateNParameters 4

# For Up to Period C4 data (72 inv pb)
altparameter1   69.3597
altparameter2   27.3806
altparameter3   -1.67095
altparameter4   -1.0643

doPValWithSysts true

doPEOnData false
```

Comments: If `outputFileName` is changed then it should be changed in the limit setting configuration files too as this file is an input to the limit setting phase. expl nPseudoExp `minXForFit` is the centre of bin value, but the fit actually picks the low edge of this bin and fits down to that value. `maxXForFit` is not specified as the end of the data is used instead. The fit function parameters are currently set up to be used with the four parameter fit function as described in section XXX. If the user wishes to switch to the five parameter fit function, also described in section XXX then `nParameters` and `parameter5` should be uncommented, for further details about switching fit function see section XXX.

C Search phase results explanation

SearchPhase_results.root contains the following histograms:
basicData:Input dijet invariant mass histogram
normalizedData:Input dijet invariant mass histogram normalized by bin width check and where used?
theFitFunction:The function fitted to the dijet invariant mass histogram
basicBkgFrom4ParamFit:Same as theFitFunction, but in histogram form ask/how decide binning, match original mjj binning?
normalizedBkgFrom4ParamFit:basicBkgFrom4ParamFit normalized by bin width
residualHist:XXX
relativeDiffHist:XXX
sigOfDiffHist:XXX
logLikelihoodStatHistNullCase:XXX
logLOfFitToData:XXX
bumpHunterStatHistNullCase:XXX
bumpHunterStatOfFitToData:XXX
bumpHunterTomographyFromPseudoexperiments:XXX
bumpHunterPLowHigh:XXX
fittedParameters:XXX

D Step2_setLimitsOneMassPoint_QStar.config

```
 ##############################################
#                                              #
#  Config file for Bayesian                    #
#                                              #
##############################################

#set all the parameters of your analysis
#IMPORTANT: don't leave spaces after the parameters!


########################################
# input/output
########################################

# This input to limit setting phase must be an output from the search phase
# Value overwritten if use Run_SearchPhase.py
dataFileName ./results/Step1_SearchPhase/Data15_C_fGRL_JetCalibCorrection_20150721/Step1_Se

dataHist          basicData

# Value overwritten if use Run_SearchPhase.py
signalFileName inputs/QBH_20150715/1fb/QBH%d_1fb.root
```

```
# Value overwritten if use Run_SearchPhase.py
nominalSignalHist mjj_QBH%d_1fb_Nominal

# Value overwritten if use Run_SearchPhase.py
outputFileName ./results/Step2_setLimitsOneMassPoint/Data15_C_fGRL_JetCalibCorrection_20150

# Put LogFiles in this folder
# Value overwritten if use Run_SearchPhase.py
plotDirectory .

# If you want to keep the BAT output plots with a distinguishable name, specify it here
plotNameExtension QStar

# Name of signal for retrievals etc
signame   QStar

########################################
# general
########################################

# Value overwritten if use Run_SearchPhase.py
Ecm   13000.0

minXForFit   1099

nParameters       3

########################################
# for limits
########################################

nSigmas   3.

doExpected   true

# Lydia changed 10 to 100
nPEForExpected   100

########################################
# Background
########################################

doFitError   true

nFitsInBkgError   100
```

```
########################################
# Lumi
########################################

doLumiError       true
# 9% lumi error
luminosityErr   0.09

########################################
# Function choice
########################################

doFitFunctionChoiceError true

nFitFSigmas       1

alternateFunctionCode 4
alternateNParameters 4
# For Up to Period C4 data (72 inv pb)
altparameter1   69.3597
altparameter2   27.3806
altparameter3   -1.67095
altparameter4   -1.0643

########################################
# Beam energy systematic
########################################

doBeam  false

BeamFile          ./inputs/BeamUncertainty/AbsoluteBEAMUncertaintiesForPlotting.root

########################################
# JES
########################################

doJES   true

##-------------------------------------##

useMatrices       false

nominalJES         #matrix_mjj_TotalUncertainty_05

nComponents       1
name1   1
```

```
##--------------------------------------##

useTemplates      true

# Value overwritten if use Run_SearchPhase.py
nominalTemplateJES mjj_QStar%d_1fb_Nominal

nComponentsTemp 3
nameTemp1         mjj_QStar%d_1fb_JET_GroupedNP_1
nameTemp2         mjj_QStar%d_1fb_JET_GroupedNP_2
nameTemp3         mjj_QStar%d_1fb_JET_GroupedNP_3

##--------------------------------------##
# nJES is number of extensions +1
nJES   13
extension1        __3down
extension2        __2down5
extension3        __2down
extension4        __1down5
extension5        __1down
extension6        __0down5
extension7        __0up5
extension8        __1up
extension9        __1up5
extension10       __2up
extension11       __2up5
extension12       __3up
```

Individual systematics can be turned on or off by changing e.g. `doFitError` from true to `false`.

E Limit setting phase results expanation

F References