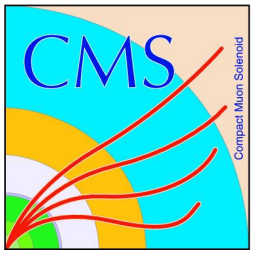


Introduction to the Combine Tool

29th Nov. 2017

Andrew Gilbert (KIT)
Giacomo Ortona (LLR)
David Sperka (University of Florida)
Nick Wardle (CERN)

Overview of Combine

- Combine is a RooStats based command line tool which executes different statistical methods (compute limits/significance, make fits, etc.)
- Internally, there are two main components
  - ➔ text2workspace: a python module which converts a textual datacard into a RooStats model (“workspace”)
    - Datacards can also be used to load user created workspaces
  - ➔ C++ code which is responsible for the statistical methods
- Highly customizable: anything which is possible in RooStats is possible in the Combine package
- Supported by a HIG PAG Subgroup: Hcomb

[Much more information on the Combine Tool gitbook]

The commands in the blue boxes on each slide provide a hands on commands which can be run to follow along.

```
cd HiggsAnalysis/CombinedLimit/data/tutorials/counting/
```

Datacard: Simple Counting Experiment

```
# Simple counting experiment, with one signal and one background process
# Extremely simplified version of the 35/pb H->WW analysis for mH = 200 GeV,
# for 4th generation exclusion (EWK-10-009, arxiv:1102.5429v1)
imax 1  number of channels
jmax 1  number of backgrounds
kmax 2  number of nuisance parameters (sources of systematical uncertainties)
------------
# we have just one channel, in which we observe 0 events
bin         bin1
observation 0
------------
# now we list the expected events for signal and all backgrounds in that bin
# the second 'process' line must have a positive number for backgrounds, and 0 for signal
# then we list the independent sources of uncertainties, and give their effect (syst. error)
# on each process and bin
bin           bin1   bin1
process       ggh4G  Bckg
process         0      1
rate           4.76  1.47
------------
deltaS  lnN    1.20    -    20% uncertainty on signal
deltaB  lnN      -   1.50   50% uncertainty on background
```

- The “datacard” is a textual input which combine uses to create a RooStats model (“workspace”)

```
cat simple-counting-experiment.txt
```

Datacard: Simple Counting Experiment

```
# Simple counting experiment, with one signal and one background process
# Extremely simplified version of the 35/pb H->WW analysis for mH = 200 GeV,
# for 4th generation exclusion (EWK-10-009, arxiv:1102.5429v1)
imax 1  number of channels
jmax 1  number of backgrounds
kmax 2  number of nuisance parameters (sources of systematical uncertainties)
------------
# we have just one channel, in which we observe 0 events
bin         bin1
observation 0
------------
# now we list the expected events for signal and all backgrounds in that bin
# the second 'process' line must have a positive number for backgrounds, and 0 for signal
# then we list the independent sources of uncertainties, and give their effect (syst. error)
# on each process and bin
bin           bin1   bin1
process       ggh4G  Bckg
process         0      1
rate           4.76  1.47
------------
deltaS  lnN    1.20     -    20% uncertainty on signal
deltaB  lnN      -    1.50   50% uncertainty on background
```

- First block specifies the number of channels, backgrounds, nuisances
- Can also use “*” and combine can determine on the fly, but specifying explicitly can help spot mistakes

```
cat simple-counting-experiment.txt
```

Datacard: Simple Counting Experiment

```
# Simple counting experiment, with one signal and one background process
# Extremely simplified version of the 35/pb H->WW analysis for mH = 200 GeV,
# for 4th generation exclusion (EWK-10-009, arxiv:1102.5429v1)
imax 1  number of channels
jmax 1  number of backgrounds
kmax 2  number of nuisance parameters (sources of systematical uncertainties)
------------
# we have just one channel, in which we observe 0 events
bin         bin1
observation 0
------------
# now we list the expected events for signal and all backgrounds in that bin
# the second 'process' line must have a positive number for backgrounds, and 0 for signal
# then we list the independent sources of uncertainties, and give their effect (syst. error)
# on each process and bin
bin           bin1   bin1
process       ggh4G  Bckg
process         0      1
rate           4.76  1.47
------------
deltaS  lnN    1.20    -    20% uncertainty on signal
deltaB  lnN      -   1.50   50% uncertainty on background
```

- Second block labels the channel (here it is called “bin1”)
- Specifies the number of of observed events

```
cat simple-counting-experiment.txt
```

Datacard: Simple Counting Experiment

```
# Simple counting experiment, with one signal and one background process
# Extremely simplified version of the 35/pb H->WW analysis for mH = 200 GeV,
# for 4th generation exclusion (EWK-10-009, arxiv:1102.5429v1)
imax 1  number of channels
jmax 1  number of backgrounds
kmax 2  number of nuisance parameters (sources of systematical uncertainties)
------------
# we have just one channel, in which we observe 0 events
bin         bin1
observation 0
------------
# now we list the expected events for signal and all backgrounds in that bin
# the second 'process' line must have a positive number for backgrounds, and 0 for signal
# then we list the independent sources of uncertainties, and give their effect (syst. error)
# on each process and bin
bin           bin1   bin1
process       ggh4G  Bckg
process         0      1
rate           4.76  1.47
------------
deltaS  lnN    1.20     -    20% uncertainty on signal
deltaB  lnN      -    1.50   50% uncertainty on background
```

- Third block specifies the number of expected events (“rate”) for each process
- Two process lines: one gives a label to each process, and the second a number which if ≤0 denotes signal and if >0 denotes background
  - ➔ Signal processes will be given a free floating normalization parameter

```
cat simple-counting-experiment.txt
```

Datacard: Simple Counting Experiment

```
# Simple counting experiment, with one signal and one background process
# Extremely simplified version of the 35/pb H->WW analysis for mH = 200 GeV,
# for 4th generation exclusion (EWK-10-009, arxiv:1102.5429v1)
imax 1  number of channels
jmax 1  number of backgrounds
kmax 2  number of nuisance parameters (sources of systematical uncertainties)
------------
# we have just one channel, in which we observe 0 events
bin         bin1
observation 0
------------
# now we list the expected events for signal and all backgrounds in that bin
# the second 'process' line must have a positive number for backgrounds, and 0 for signal
# then we list the independent sources of uncertainties, and give their effect (syst. error)
# on each process and bin
bin           bin1   bin1
process       ggh4G  Bckg
process         0      1
rate           4.76  1.47
------------
deltaS  lnN    1.20     -     20% uncertainty on signal
deltaB  lnN      -    1.50    50% uncertainty on background
```

- Final block specifies the nuisance parameters affecting the processes
- A label and prior distribution (e.g. Log-normal, Gamma, Uniform) are given
- Nuisances with different names are uncorrelated (and a single nuisance is correlated across all processes which it affects)

```
cat simple-counting-experiment.txt
```

From Datacard to RooWorkspace

- The text2workspace.py script converts the textual datacard into a RooWorkspace, defining the Likelihood function used for the statistical methods
- The RooWorkspace contains variables, pdf's, functions, datasets, etc.
- One can inspect the workspace to see how the likelihood has been constructed

```
RooWorkspace(w) w contents

variables
---------
(deltaB,deltaB_In,deltaS,deltaS_In,n_obs_binbin1,r)

p.d.f.s
-------
SimpleGaussianConstraint::deltaB_Pdf[ x=deltaB mean=deltaB_In sigma=1 ] = 1
SimpleGaussianConstraint::deltaS_Pdf[ x=deltaS mean=deltaS_In sigma=1 ] = 1
RooProdPdf::modelObs_b[ pdf_binbin1_bonly ] = 0.229925
RooProdPdf::modelObs_s[ pdf_binbin1 ] = 0.00196945
RooProdPdf::model_b[ modelObs_b * nuisancePdf ] = 0.229925
RooProdPdf::model_s[ modelObs_s * nuisancePdf ] = 0.00196945
RooProdPdf::nuisancePdf[ deltaS_Pdf * deltaB_Pdf ] = 1
RooPoisson::pdf_binbin1[ x=n_obs_binbin1 mean=n_exp_binbin1 ] = 0.00196945
RooPoisson::pdf_binbin1_bonly[ x=n_obs_binbin1 mean=n_exp_binbin1_bonly ] = 0.229925

functions
--------
RooAddition::n_exp_binbin1[ n_exp_binbin1_proc_ggh4G + n_exp_binbin1_proc_Bckg ] = 6.23
RooAddition::n_exp_binbin1_bonly[ n_exp_binbin1_proc_Bckg ] = 1.47
ProcessNormalization::n_exp_binbin1_proc_Bckg[ thetaList=(deltaB) asymmThetaList=() otherFactorList=() ] = 1.47
ProcessNormalization::n_exp_binbin1_proc_ggh4G[ thetaList=(deltaS) asymmThetaList=() otherFactorList=(r) ] = 4.76

datasets
--------
RooDataSet::data_obs(n_obs_binbin1)

named sets
----------
ModelConfig_GlobalObservables:(deltaS_In,deltaB_In)
ModelConfig_NuisParams:(deltaS,deltaB)
ModelConfig_Observables:(n_obs_binbin1)
ModelConfig_POI:(r)
ModelConfig_bonly_GlobalObservables:(deltaS_In,deltaB_In)
ModelConfig_bonly_NuisParams:(deltaS,deltaB)
ModelConfig_bonly_Observables:(n_obs_binbin1)
ModelConfig_bonly_POI:(r)
POI:(r)
globalObservables:(deltaS_In,deltaB_In)
nuisances:(deltaS,deltaB)
observables:(n_obs_binbin1)

generic objects
---------------
RooStats::ModelConfig::ModelConfig
RooStats::ModelConfig::ModelConfig_bonly
RooArgSet::discreteParams
```

```
text2workspace.py simple-counting-experiment.txt
root -l simple-counting-experiment.root
root [1] RooWorkspace* w = (RooWorkspace*)_file0->Get("w")
root [2] w->Print()
```

Likelihood From the RooWorkspace

- Lets start from the pdf called model_s in the workspace, which is the full likelihood including the signal, and expand as much as we can:
  - For brevity, I won't always copy the full name of every object

```
L = model_s

L = modelObs_s * nuisancePdf

L = pdf_binbin1 * deltaS_Pdf * deltaB_Pdf

L = Poisson[x=n_obs mean=n_exp]
    * Gauss[x=deltaS mean=deltaS_In sigma=1] * Gauss[x=deltaB mean=deltaB_In sigma=1]

L = Poisson[x=n_obs mean=(n_exp_ggh4G + n_exp_Bckg)]
    * Gauss[x=deltaS mean=0 sigma=1] * Gauss[x=deltaB mean=0 sigma=1]

L = Poisson[x=n_obs mean=( n_nom_ggh4G*f(deltaS)*r + n_nom_Bckg*f(deltaB))]
       * Gauss[x=deltaS mean=0 sigma=1] * Gauss[x=deltaB mean=0 sigma=1]
```

- Which is the single bin example of a generic likelihood function:

$$L(r,\vec{\theta}) = \prod_i \frac{[r\cdot s_i(\vec{\theta}) + b_i(\vec{\theta})]^{n_i}}{n_i!} e^{-[r\cdot s_i(\vec{\theta}) + b_i(\vec{\theta})]} \prod_\kappa e^{-\frac{1}{2}\theta_\kappa^2}$$

  - It has three parameters: r, deltaS, and deltaB. "r" is the POI and is unconstrained (except by the observed data), while deltaS and deltaB are nuisance parameters which have external constraints. These three parameters are jointly fitted to get the value of "r".

Datacard: Realistic Counting Experiment

```
# Simple counting experiment, with one signal and a few background processes
# Simplified version of the 35/pb H->WW analysis for mH = 160 GeV
imax 1  number of channels
jmax 3  number of backgrounds
kmax 5  number of nuisance parameters (sources of systematical uncertainties)
------------
# we have just one channel, in which we observe 0 events
bin bin1
observation 0
------------
# now we list the expected events for signal and all backgrounds in that bin
# the second 'process' line must have a positive number for backgrounds, and 0 for signal
# then we list the independent sources of uncertainties, and give their effect (syst. error)
# on each process and bin
bin             bin1 bin1  bin1  bin1
process         ggH  qqWW  ggWW  others
process          0     1     2     3
rate           1.47  0.63  0.06  0.22
------------
lumi    lnN    1.11     -   1.11     -    lumi affects both signal and gg->WW (mc-driven). lnN = lognormal
xs_ggH  lnN    1.16     -      -     -    gg->H cross section + signal efficiency + other minor ones.
WW_norm gmN 4     -   0.16     -     -    WW estimate of 0.64 comes from sidebands: 4 events in sideband times 0.16
xs_ggWW lnN       -      -   1.50     -    50% uncertainty on gg->WW cross section
bg_others lnN     -      -      -  1.30   30% uncertainty on the rest of the backgrounds
```

- Realistic datacards will have more processes and nuisance parameters, but we will end up with a similar likelihood function:

$$L(r,\vec{\theta}) = \prod_i \frac{[r\cdot s_i(\vec{\theta})+b_i(\vec{\theta})]^{n_i}}{n_i!} e^{-[r\cdot s_i(\vec{\theta})+b_i(\vec{\theta})]} \prod_\kappa e^{-\frac{1}{2}\theta_\kappa^2}$$

```
cat realistic-counting-experiment.txt
```

Combination of Multiple Channels

```
imax 3  number of channels
jmax *  number of backgrounds ('*' = automatic)
kmax *  number of nuisance parameters (sources of systematical uncertainties)
------------
# three channels, each with it's number of observed events
bin          e_tau mu_tau e_mu
observation   517   540   101
------------
# now we list the expected events for signal and all backgrounds in those three bins
# the second 'process' line must have a positive number for backgrounds, and 0 for signal
# for the signal, we normalize the yields to an hypothetical cross section of 1/pb
# so that we get an absolute limit in cross section in units of pb.
# then we list the independent sources of uncertainties, and give their effect (syst. error)
# on each process and bin
bin             e_tau  e_tau  e_tau   mu_tau  mu_tau  mu_tau   e_mu   e_mu   e_mu
process         higgs    ZTT    QCD   higgs     ZTT     QCD   higgs    ZTT  other
process           0      1      2       0       1       2       0      1      2
rate            0.34    190    327    0.57     329     259    0.15     88     14
------------
lumi    lnN     1.11     -      -      1.11     -       -      1.11     -    1.11   A 11% lumi uncertainty, a
tauid   lnN     1.23   1.23     -      1.23   1.23      -       -       -     -     The infamous 23% tau id u
ZtoLL   lnN       -    1.04     -        -    1.04      -       -     1.04   -      4% uncertainty on lumi*Z
effic   lnN     1.04   1.04     -      1.04   1.04      -      1.04   1.04  1.04    4% uncertainty on effici
QCDel   lnN       -      -    1.20       -      -       -       -       -     -    20% uncertainty on QCD in
QCDmu   lnN       -      -      -        -      -     1.10      -       -     -    10% uncertainty on QCD in
other   lnN       -      -      -        -      -       -       -       -    1.1   10% uncertainty on non-Z
```

- Additional channels can be added easily (different “bin” label, same process labels)
  - ➔ Defining conventions for process/nuisance labels will make your life easier
- Can be done by hand or using a tool: `combineCards.py ch1.txt ch2.txt ch3.txt > comb.txt`

```
cat realistic-multi-channel.txt
```

Shape Experiment: Binned

```
imax 1
jmax 1
kmax *
---------------
shapes * * simple-shapes-TH1_input.root $PROCESS $PROCESS_$SYSTEMATIC
---------------
bin bin1
observation 85
-------------------------------
bin             bin1       bin1
process         signal     background
process         0          1
rate            10         100
---------------------------------
lumi     lnN    1.10       1.0
bgnorm   lnN    1.00       1.3
alpha  shapeN2    -          1   uncertainty on background shape and normalization
sigma  shapeN2    0.5        -   uncertainty on signal resolution. Assume the histogram is a 2 sigma shift,
#                                so divide the unit gaussian by 2 before doing the interpolation
```

shapes *process channel file histogram [histogram_with_systematics]*

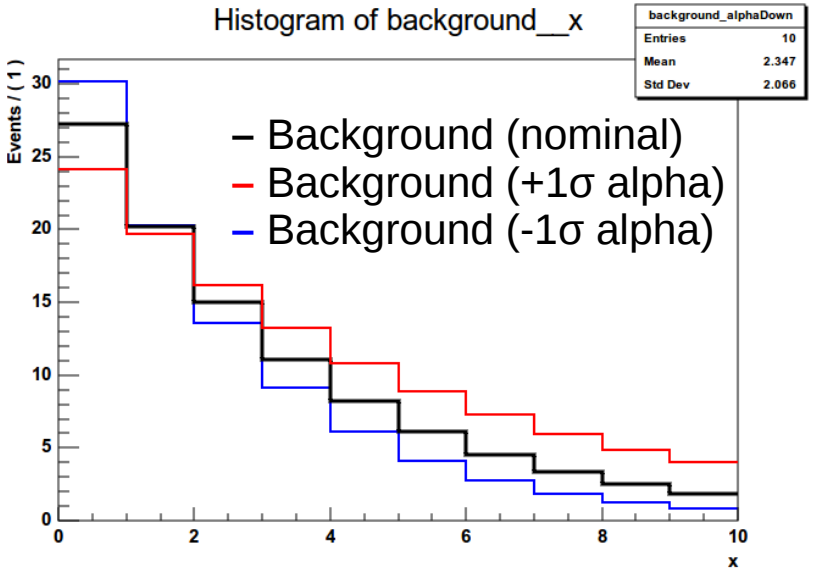
- In a shape experiment, a line pointing to a .root file is added
- For a binned experiment this .root file contains the shapes (histograms, PDFs, etc.) for the nominal distrubtion, systematic variations, and observed data
- Normalization of histograms can be used to simultaneously vary shape and rate

```
root [1] _file0->ls()
TFile**         data/benchmarks/shapes/simple-shapes-TH1.root
 TFile*         data/benchmarks/shapes/simple-shapes-TH1.root
  KEY: TH1F     signal;1        Histogram of signal__x
  KEY: TH1F     signal_sigmaUp;1        Histogram of signal__x
  KEY: TH1F     signal_sigmaDown;1      Histogram of signal__x
  KEY: TH1F     background;1    Histogram of background__x
  KEY: TH1F     background_alphaUp;1    Histogram of background__x
  KEY: TH1F     background_alphaDown;1  Histogram of background__x
  KEY: TH1F     data_obs;1      Histogram of data_obs__x
  KEY: TH1F     data_sig;1      Histogram of data_sig__x
```

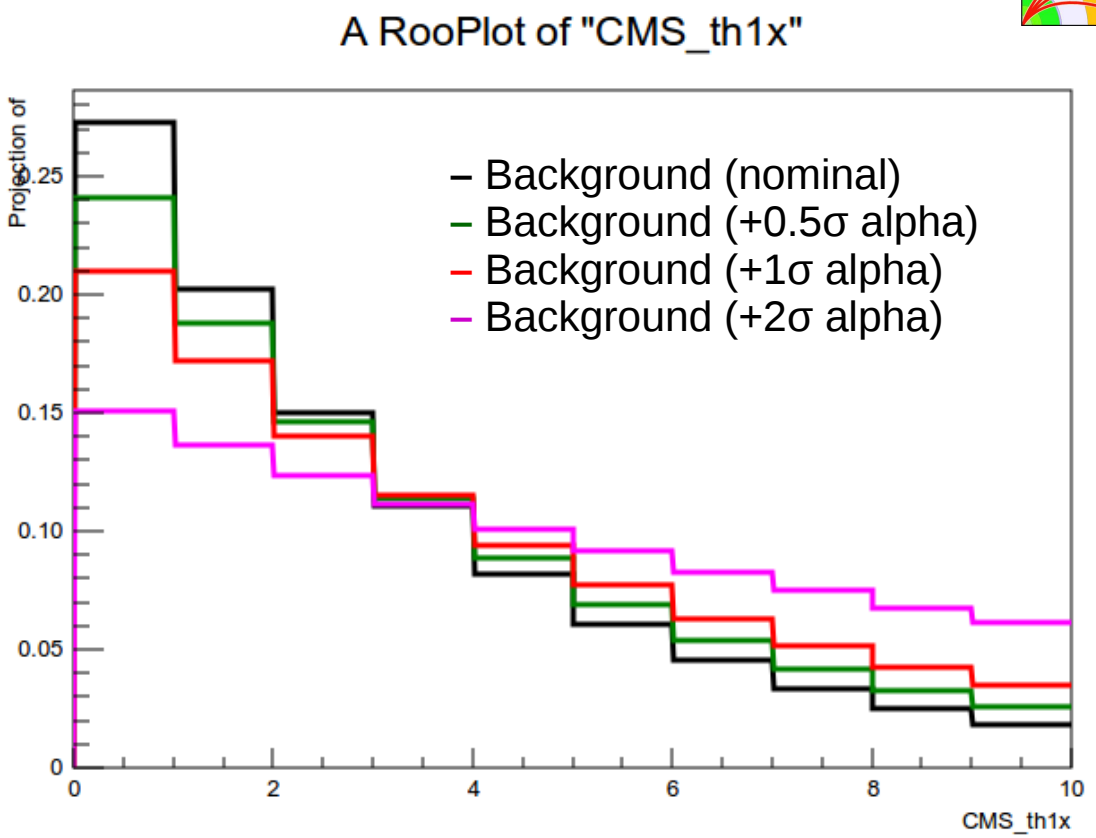
```
cd ../shapes/
cat simple-shapes-TH1.txt
root -l input-shapes-TH1.root
root [1] _file0->ls()
```

Shape Experiment: Morphing

- Given the histograms corresponding to +/- 1 sigma for a particular nuisance parameter as input...



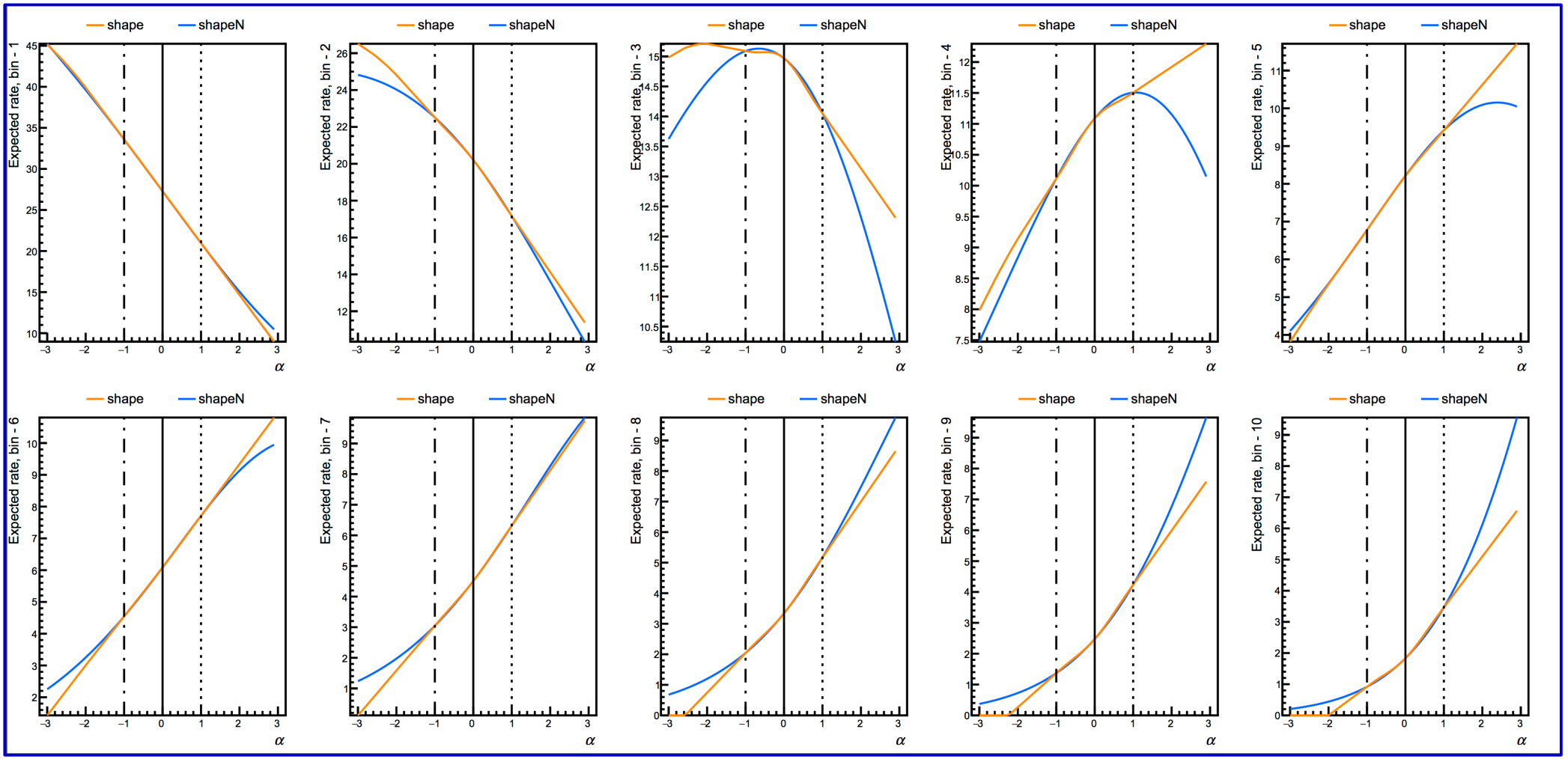
- ...Combine creates a morphing pdf which provides a shape for any value of the nuisance parameter




```
text2workspace.py simple-shapes-TH1.txt
root -l simple-shapes-TH1.root
RooWorkspace* w = (RooWorkspace*)_file0->Get("w")
RooAbsPdf* shapeBkg = (RooAbsPdf*)w->pdf("shapeBkg_bin1_background_morph")
RooRealVar* th1x = (RooRealVar*)w->var("CMS_th1x")
RooPlot* plot = th1x->frame()
RooRealVar* alpha = (RooRealVar*)w->var("alpha")
alpha->setVal(0.0)
shapeBkg->plotOn(plot,LineColor(1))
alpha->setVal(0.5)
shapeBkg->plotOn(plot,LineColor(3))
alpha->setVal(1.0)
shapeBkg->plotOn(plot,LineColor(2))
alpha->setVal(2.0)
shapeBkg->plotOn(plot,LineColor(6))
plot->Draw()
```

Shape Experiment: Morphing

- “shape” will result in a quadratic interpolation (within +/-1 sigma) and a linear extrapolation (beyond +/-1 sigma) of the fraction of events in each bin.
- “shapeN” will instead base the interpolation on the logs of the fraction in each bin.
- Usually they are similar between +/- 1 sigma, but can differ beyond



Shape Experiment: MC Stat. Uncs.

- It is common in shape experiments with templates to have uncertainties coming from the limited MC statistics.
- These uncertainties can be automatically added by the combine tool
- Achieved by adding a single line to the datacard:

```
QCDscale_VH             lnN     1.04        -           1.04        -
QCDscale_ggH1in         lnN     -           -           -           1.205
QCDscale_ggH2in         lnN     -           -           -           -
QCDscale_qqH            lnN     -           1.012       -           -
UEPS                    lnN     1.025       1.025       1.025       0.975
lumi_8TeV               lnN     1.026       1.026       1.026       1.026
pdf_gg                  lnN     -           -           -           1.097
pdf_qqbar               lnN     1.02        1.036       1.02        -
* autoMCStats 0
```

*= apply all channels in the card, or specify the name of the channel explicitly. 0 = threshold for number of effective events for Poisson modelling

- When running text2workspace.py will get a summary of the bin contents and errors and the resulting parameters that have been created:

```
------------------------------------------------------------
2          1.602526        0.123658        total sum
2          1.602526        0.123658        excluding marked processes
2          168.000000      12.961481       Unweighted events, alpha=0.009539
  => Total parameter prop_binhtt_tt_2_8TeV_bin2[0.00,-7.00,7.00] to be gaussian constrained
------------------------------------------------------------
3          6.645564        0.551605        total sum
3          6.643344        0.551601        excluding marked processes
3          145.000000      12.041595       Unweighted events, alpha=0.045831
  => Total parameter prop_binhtt_tt_2_8TeV_bin3[0.00,-7.00,7.00] to be gaussian constrained
------------------------------------------------------------
4          8.937501        0.614375        total sum
4          8.513046        0.612402        excluding marked processes
4          193.000000      13.892444       Unweighted events, alpha=0.046308
  => Total parameter prop_binhtt_tt_2_8TeV_bin4[0.00,-7.00,7.00] to be gaussian constrained
------------------------------------------------------------
```

- N.B.: important that TH1::GetBinError gives the actual uncertainty that should be added!
- When running combine add the option --X-rtd MINIMIZER_analytic to enable analytic minimization of the bin-wise stat uncertainties
  - ➔ Will speed up the fit if you have many bins

Shape Experiment: Parametric

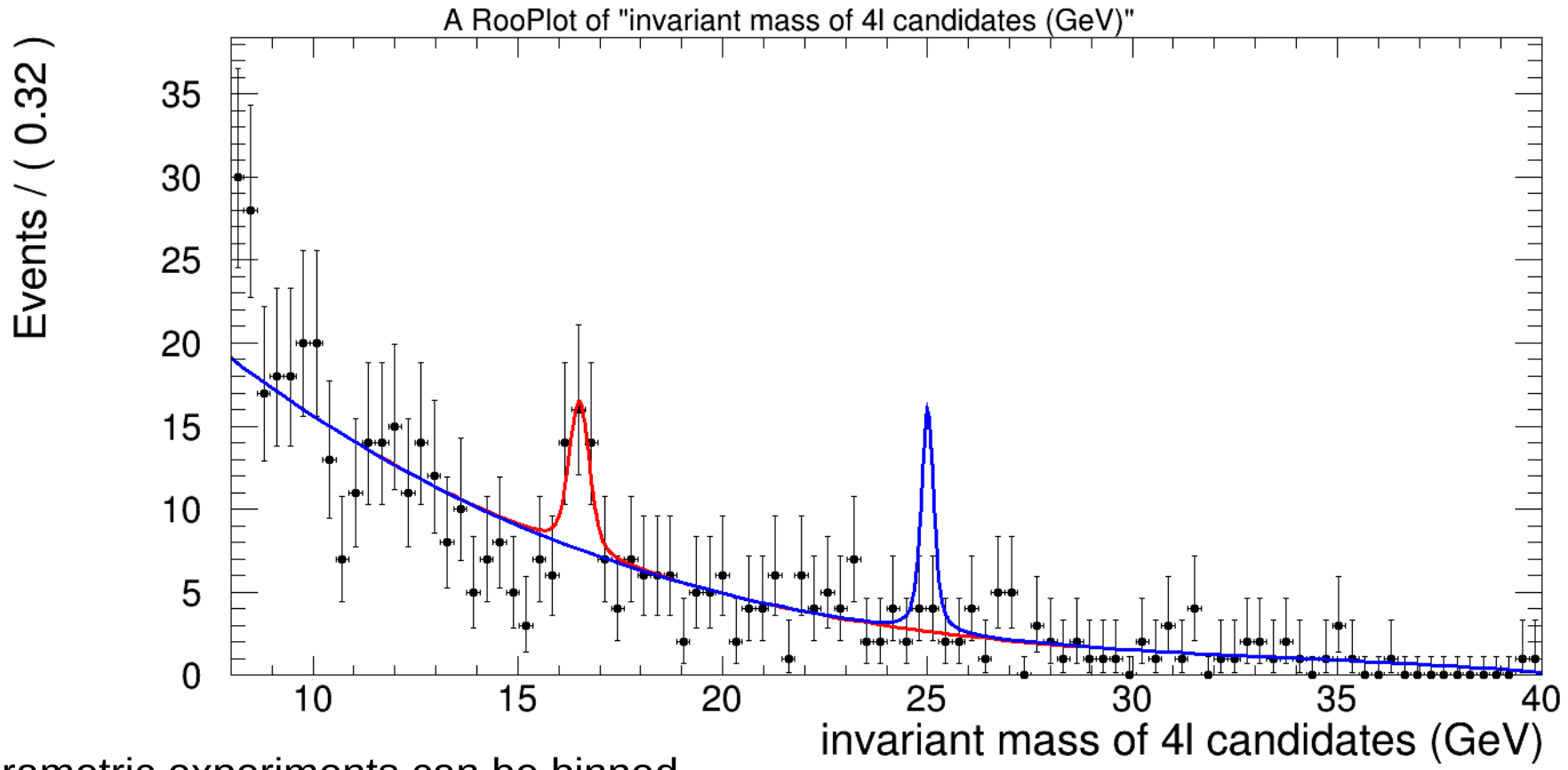
```
# Simple parametric shape based on a RooChebyChev background with freely floating normalisation and a RooVoigtian signal model.
# To run, make sure to add the option `-m 30` since the MH parameter is contained in the input workspace

imax 1
jmax 1
kmax *
---------------
shapes * * simple-shapes-parametric_input.root w:$PROCESS
---------------
bin bin1
observation -1
-------------------------------
bin          bin1       bin1
process      sig        bkg
process      0          1
rate         1          1
--------------------------------
lumi    lnN  1.1        1.0
sigma   param 1.0       0.1
```

- In a parametric shape experiment, the “shapes” line points to a .root file which has a RooWorkspace which contains RooAbsPdf’s that describe the shape of each process

```
RooWorkspace(w) w contents

variables
---------
(MH,bkg_norm,cc_a0,cc_a1,cc_a2,rmdj,vogian_sigma,vogian_width)

p.d.f.s
-------
RooChebychev::bkg[ x=rmdj coefList=(cc_a0,cc_a1,cc_a2) ] = 2.6243
RooVoigtian::sig[ x=rmdj mean=MH width=vogian_width sigma=vogian_sigma ] = 0.000639771

datasets
--------
RooDataSet::data_obs(rmdj)
```

```
cat simple-shapes-parametric.txt
root -l simple-shapes-parametric_input.root
RooWorkspace* w = (RooWorkspace*)_file0->Get("w")
w->Print()
```

Shape Experiment: Parametric



- Parametric experiments can be binned or unbinned (binned faster for larger datasets)
- How nuisance parameters affect the shape for a process can be defined in the workspace
- After converting to a workspace, one can draw the total signal+background PDF for any set of parameter values

```
root -l simple-shapes-parametric.root
RooWorkspace* w = (RooWorkspace*)_file0->Get("w")
RooRealVar* rmdj = w->var("rmdj")
RooPlot* plot = rmdj->frame()
RooDataSet* data_obs = (RooDataSet*)w->data("data_obs")
data_obs->plotOn(plot)
RooAbsPdf* pdf_binbin1 = (RooAbsPdf*)w->pdf("pdf_binbin1")
RooRealVar* r = (RooRealVar*)w->var("r")
RooRealVar* vogian_sigma = (RooRealVar*)w->var("vogian_sigma")
RooRealVar* MH = (RooRealVar*)w->var("MH")
r->setVal(20.0); MH->setVal(16.5); vogian_sigma->setVal(0.2)
pdf_binbin1->plotOn(plot,RooFit::LineColor(2))
r->setVal(40.0); MH->setVal(25.0); vogian_sigma->setVal(0.1)
pdf_binbin1->plotOn(plot,RooFit::LineColor(4))
plot->Draw()
```

Likelihood Formalism

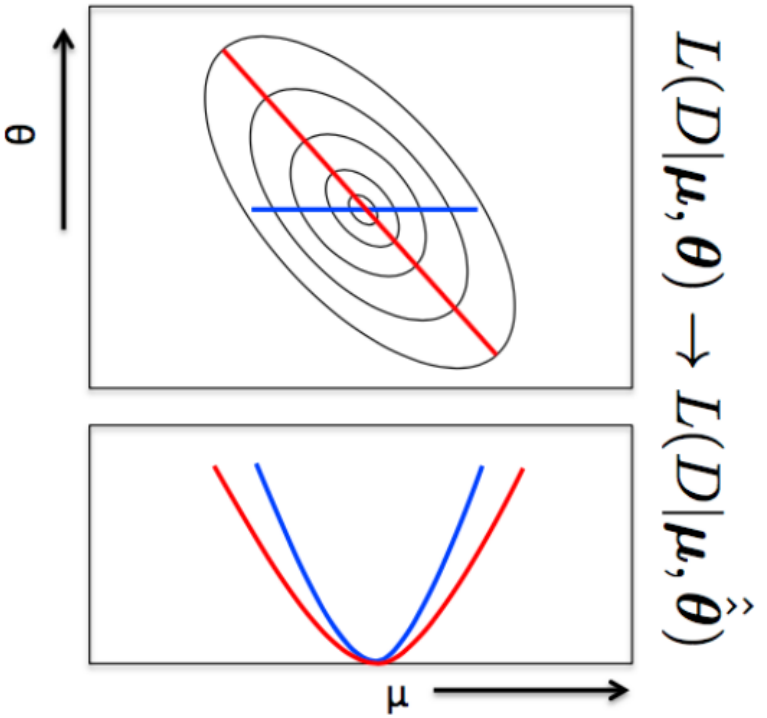
$$L(D|\boldsymbol{\mu},\boldsymbol{\theta}) = \prod_n Prob\left(d_n|\sum_{i,f}\mu_i\mu^f S^f_{i,n}(\boldsymbol{\theta}) + \sum_k B_k(\boldsymbol{\theta})\right) \times Gauss(\tilde{\boldsymbol{\theta}}|\theta)$$

- The Likelihood is a function of the parameters of interest (e.g. μ) and nuisance parameters which account for the systematic uncertainties
- Systematic uncertainties are “profiled”, i.e. fitted, at each value of the POI:
- Likelihood ratio provides the optimal test between two hypotheses (Neyman-Pearson Lemma), so the profile LR should be nearly optimal

$$\lambda(\mu) = \frac{L(\mu,\hat{\hat{\boldsymbol{\theta}}})}{L(\hat{\mu},\hat{\boldsymbol{\theta}})}$$

← Maximize L at μ

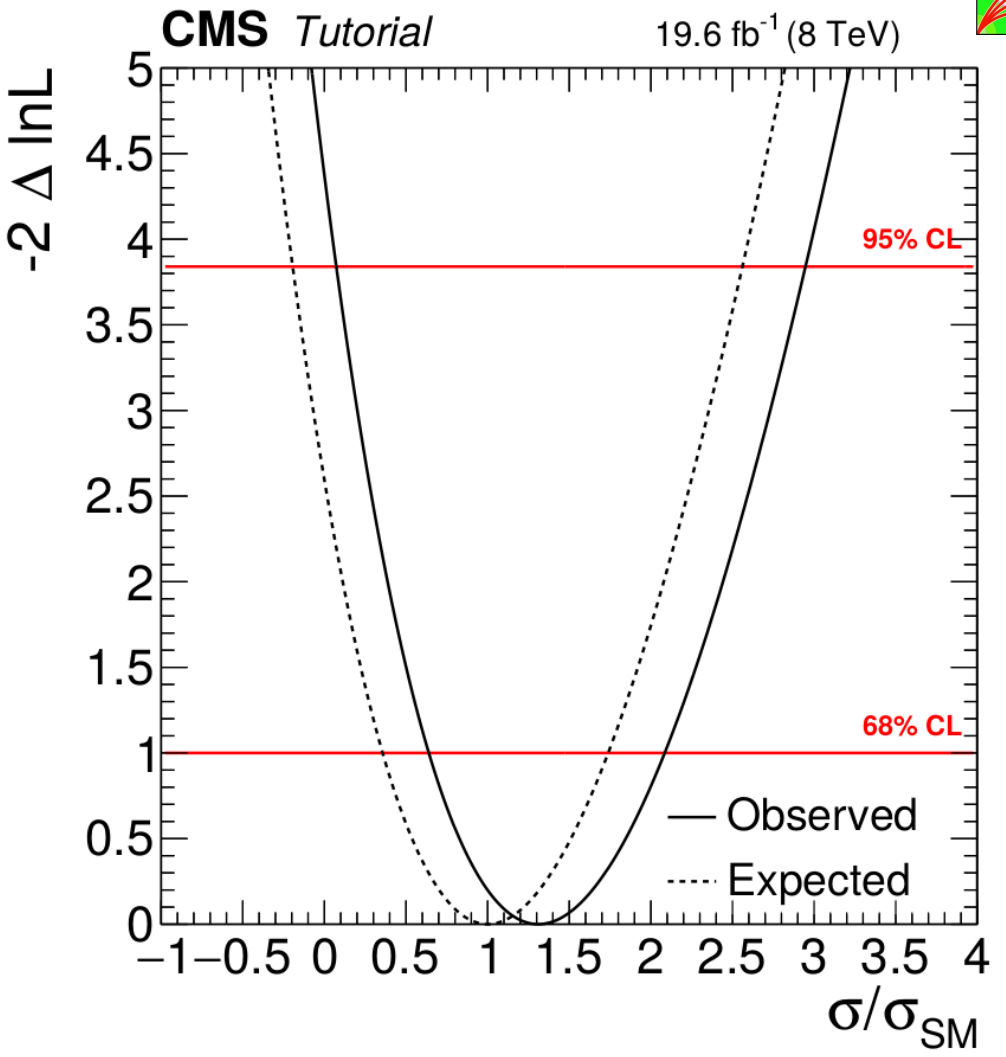
← Maximize L



- For computing limits/significance, need to know the distribution of the test statistic
- For large N, -2 ln λ(μ) tends to a non-central χ^2 distribution
- An “Asimov” dataset, where all parameters are equal to their expected outcomes, can be used to estimate the non-centrality parameter arxiv:1007.1727

Maximum Likelihood Fit

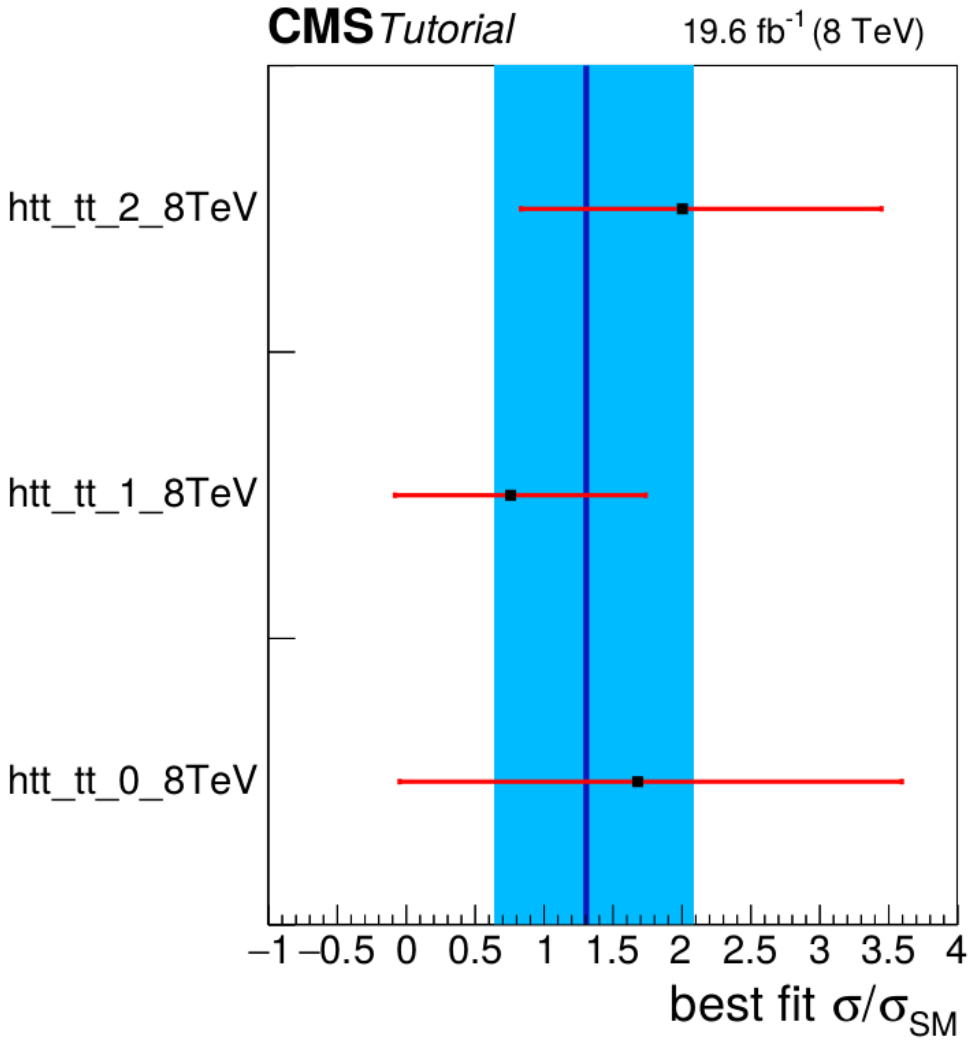
- Combine provides a Maximum Likelihood Fit method for extracting the (expected) best-fit and uncertainties for the POIs
  - ➔ Can produce likelihood scans for the POI
- --robustFit option with algo=singles can be used to find the 68% crossing automatically
- Expected signal strength can be obtained by using an asimov dataset (-t -1)
- Adding --toysFreq option will use nuisance parameters at their best fit to the observed data for the generation of the asimov dataset (a-posteriori expected)




```
cd ../htt/125/
combine -n Obs -M MultiDimFit -m 125 htt_tt.txt --algo=grid --points 300 --setParameterRanges r=-1.0,4.0
combine -n Exp -M MultiDimFit -m 125 htt_tt.txt -t -1 --expectSignal=1 --algo=grid --points 300 --setParameterRanges r=-1.0,4.0
wget https://raw.githubusercontent.com/nucleosynthesis/HiggsAnalysis-CombinedLimit/combine_tutorial_SWAN/combine_tutorials_2016/combine_intro/plotMuScan.py
wget https://raw.githubusercontent.com/nucleosynthesis/HiggsAnalysis-CombinedLimit/combine_tutorial_SWAN/combine_tutorials_2016/combine_intro/tdrStyle.py
python plotMuScan.py
combine -n Obs -M MultiDimFit -m 125 htt_tt.txt -t -1 --expectSignal=1 --algo=singles --robustFit=1 --setParameterRanges r=-1.0,4.0
combine -n Obs -M MultiDimFit -m 125 htt_tt.txt -t -1 --toysFreq --expectSignal=1 --algo=singles --robustFit=1 --setParameterRanges r=-1.0,4.0
```

Channel Compatibility

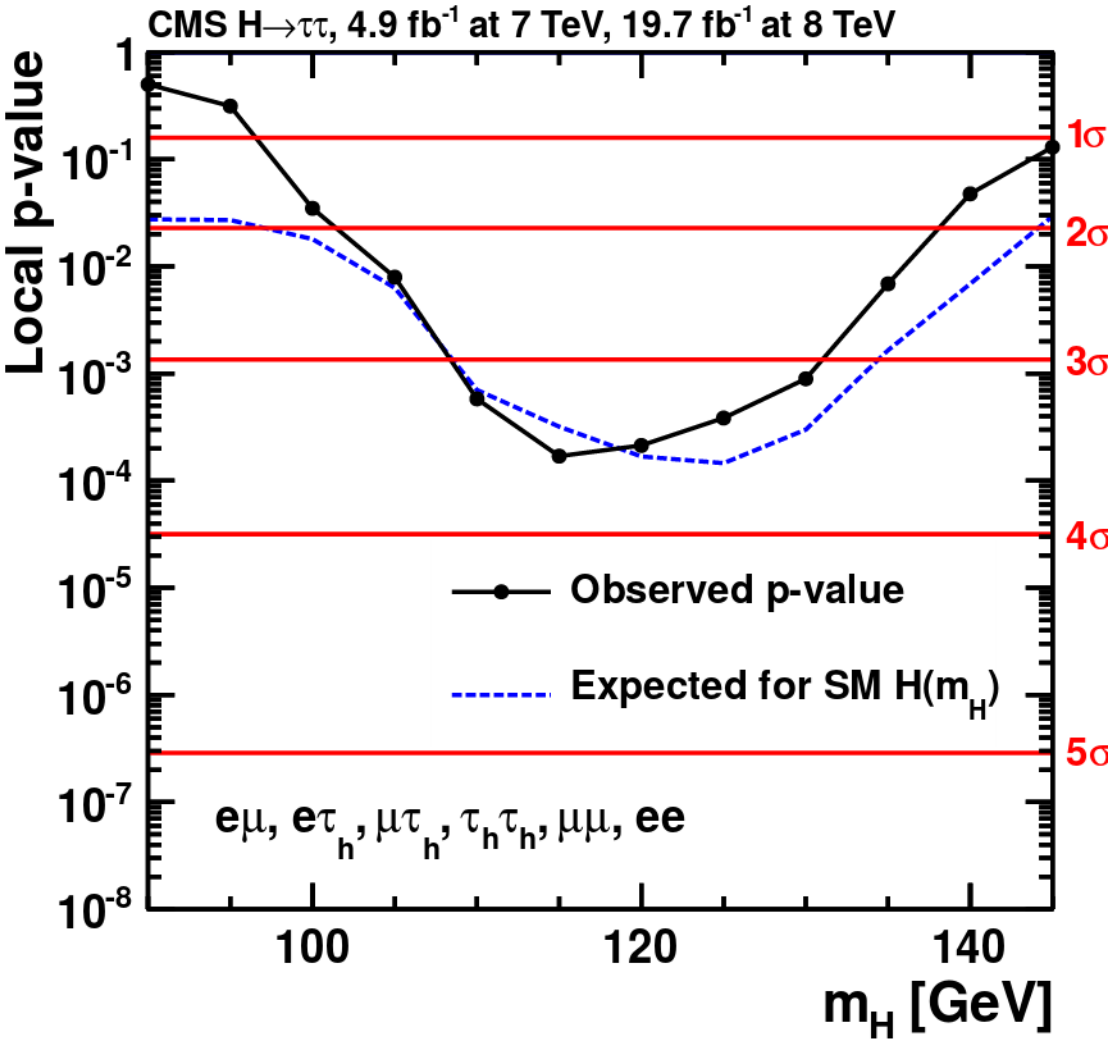
- ChannelCompatibilityCheck method can be used to fit an independent signal strength in each analysis category
  - ➔ All categories are fit simultaneously with full correlation of nuisance parameters
  - ➔ This is not the same as fitting each category completely separately!
- A chi2 like variable is also printed, which can be turned into a p-value by running on toy datasets




```
combine -m 125 -M ChannelCompatibilityCheck htt_tt.txt --saveFitResult --rMin -1 --rMax 4
wget
https://raw.githubusercontent.com/nucleosynthesis/HiggsAnalysis-CombinedLimit/combine_tutorial_SWAN/combine_tutorials_2016/combine_intro/cccPlot.py
python cccPlot.py
```

Extracting Significance

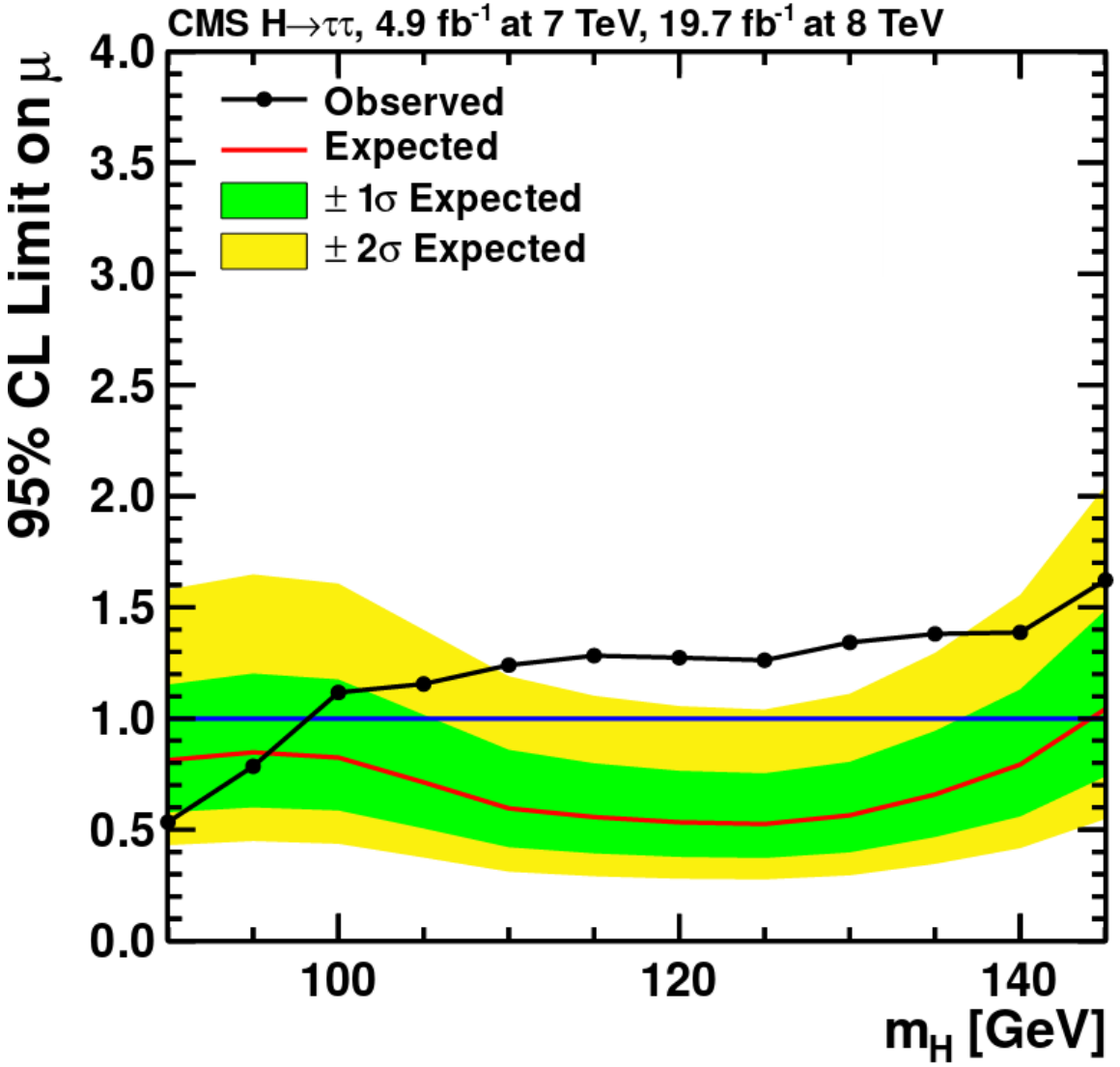
- Significance is calculated using a ratio of profiled likelihoods, one in which the signal strength is set to 0 and the other in which it is free to float, with the nuisance parameters profiled separately
- The distribution of this test-statistic can be determined in the Asymptotic limit. The significance can therefore be calculated very quickly.
- "Significance" method in combine
  - ➔ Equivalent to evaluating √(-2lnL) at r=0
- The expected significance can be computed from an Asimov dataset of signal+background.
  - ➔ Again, can choose a-priori or a-posteriori expected significance




```
combine -M Significance --signif -m 125 htt_tt.txt -t -1 --expectSignal=1 --toysFreq
combine -M Significance --signif -m 125 htt_tt.txt
```

Asymptotic Frequentist Limits

- In case of no observed signal, can set a limit on the signal strength
- The Asymptotic limit is the most commonly used method (its fast).
- The limit calculation relies on an asymptotic approximation of the distributions of the LHC test-statistic, which is based on a profile likelihood ratio, under signal and background hypotheses
  - No need to generate frequentist toys to build the distribution of the test statistic.
  - Toy based limits are also supported with “HybridNew”
- It fits the data first and uses this as the starting point to determine these distributions. This is the fully frequentist prescription. You can remove this by either adding the options --run blind or --noFitAsimov.




```
combine -M AsymptoticLimits -m 125 htt_tt.txt --run blind
combine -M AsymptoticLimits -m 125 htt_tt.txt --run both
```