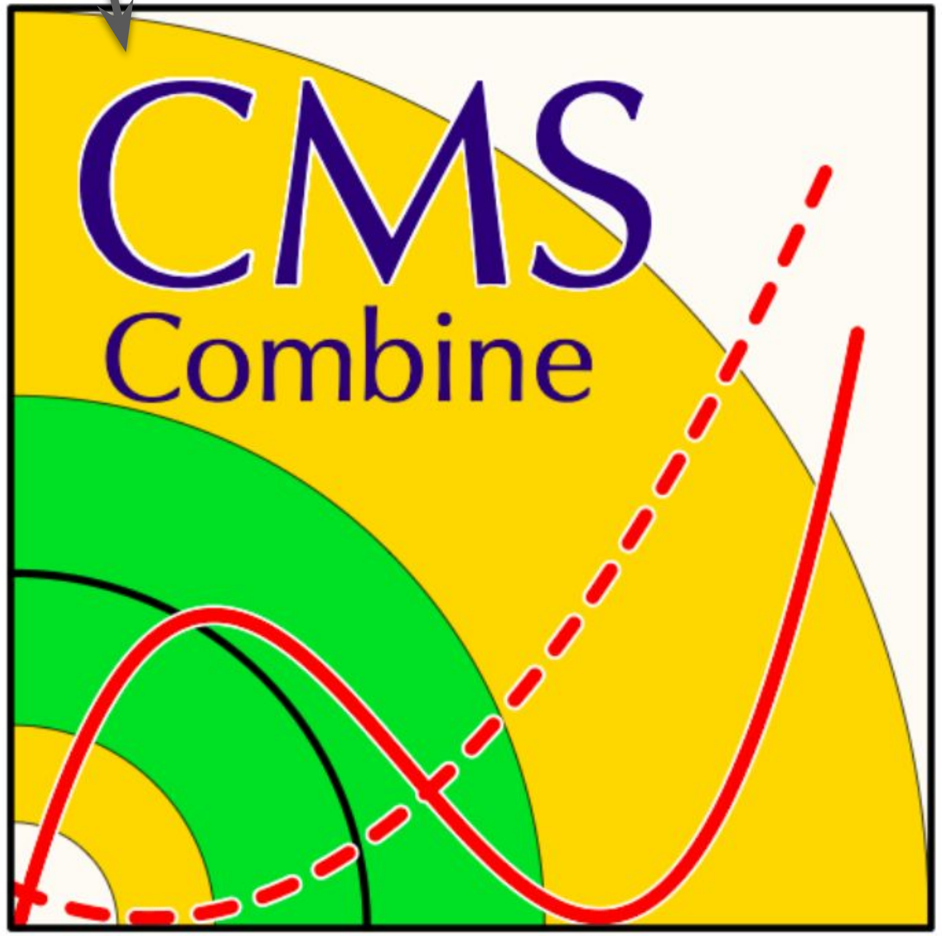
RooFit in CMS: combine



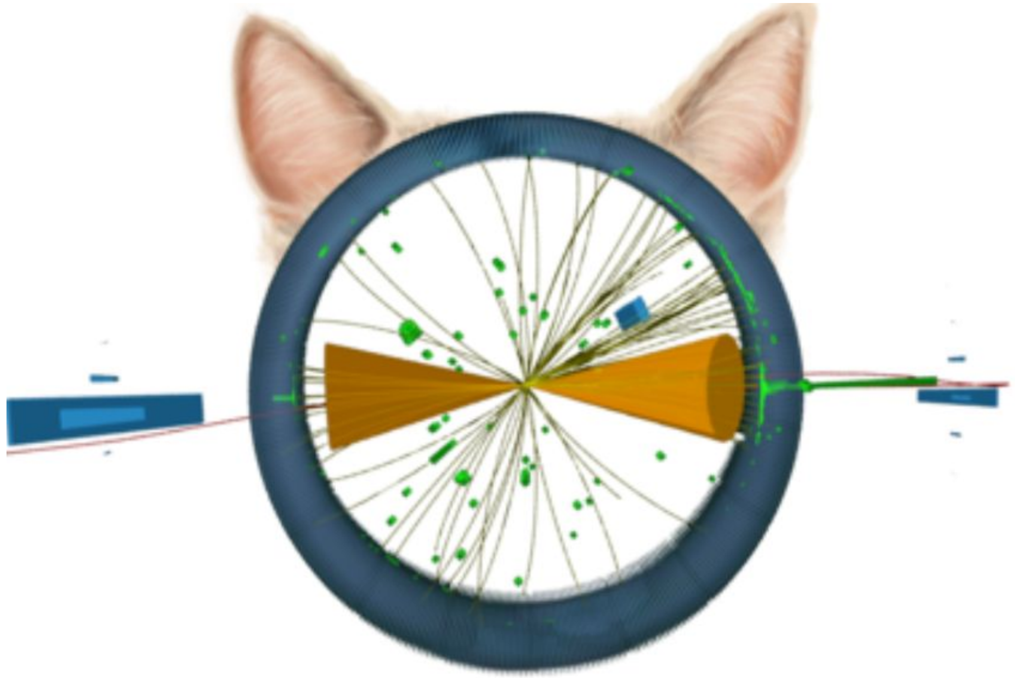
New shiny logo

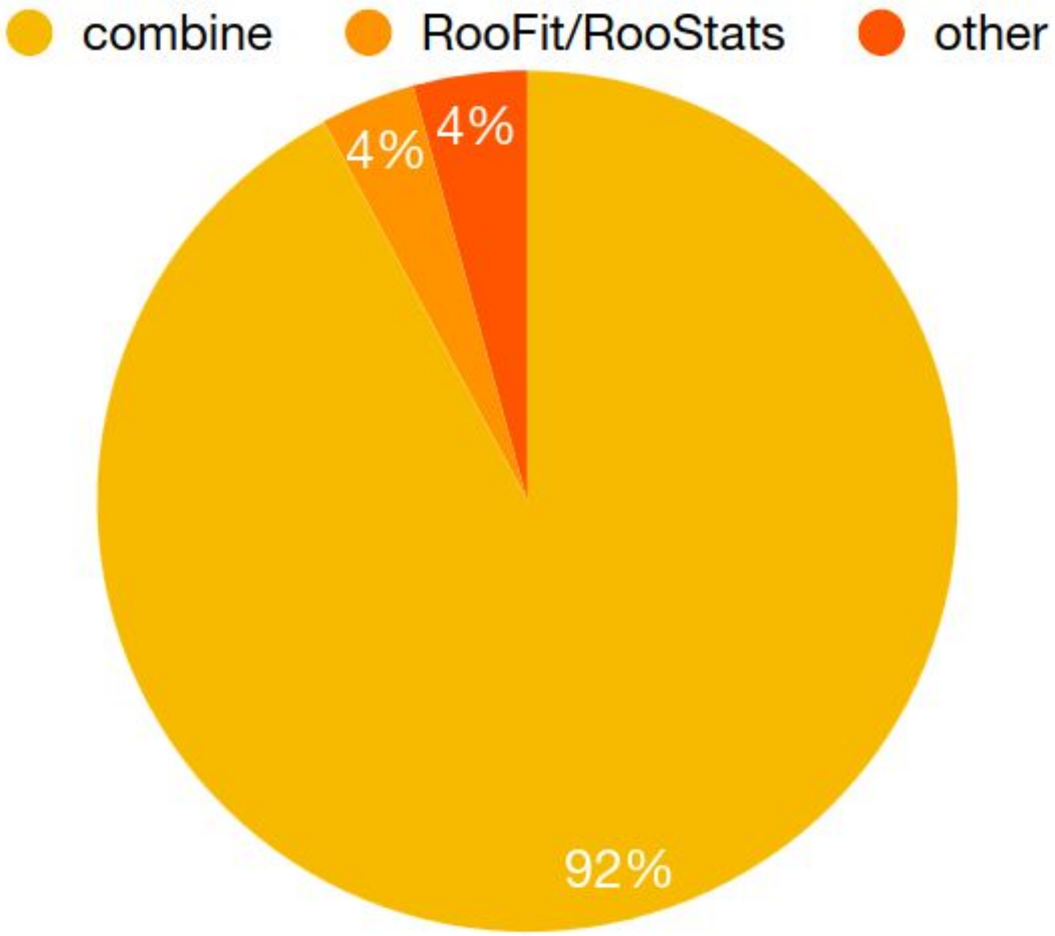
Jon Langford

Slides taken from many people: Aliya, Kyle, Andrew, Nick, Kirill, ...

Combine tool

- Primary software framework for statistical model building & inference in CMS analyses
- Based of RooFit + RooStats packages
  - Construct likelihood function for model of arbitrary complexity
- For many years was supported by only Higgs Combination Group
- Now under remit of Common Analysis Tools (CAT) group
  - Statistical Tools subgroup (Conveners: Kyle Cormier, Aliya Nigamova
  - Continued support by Higgs comb

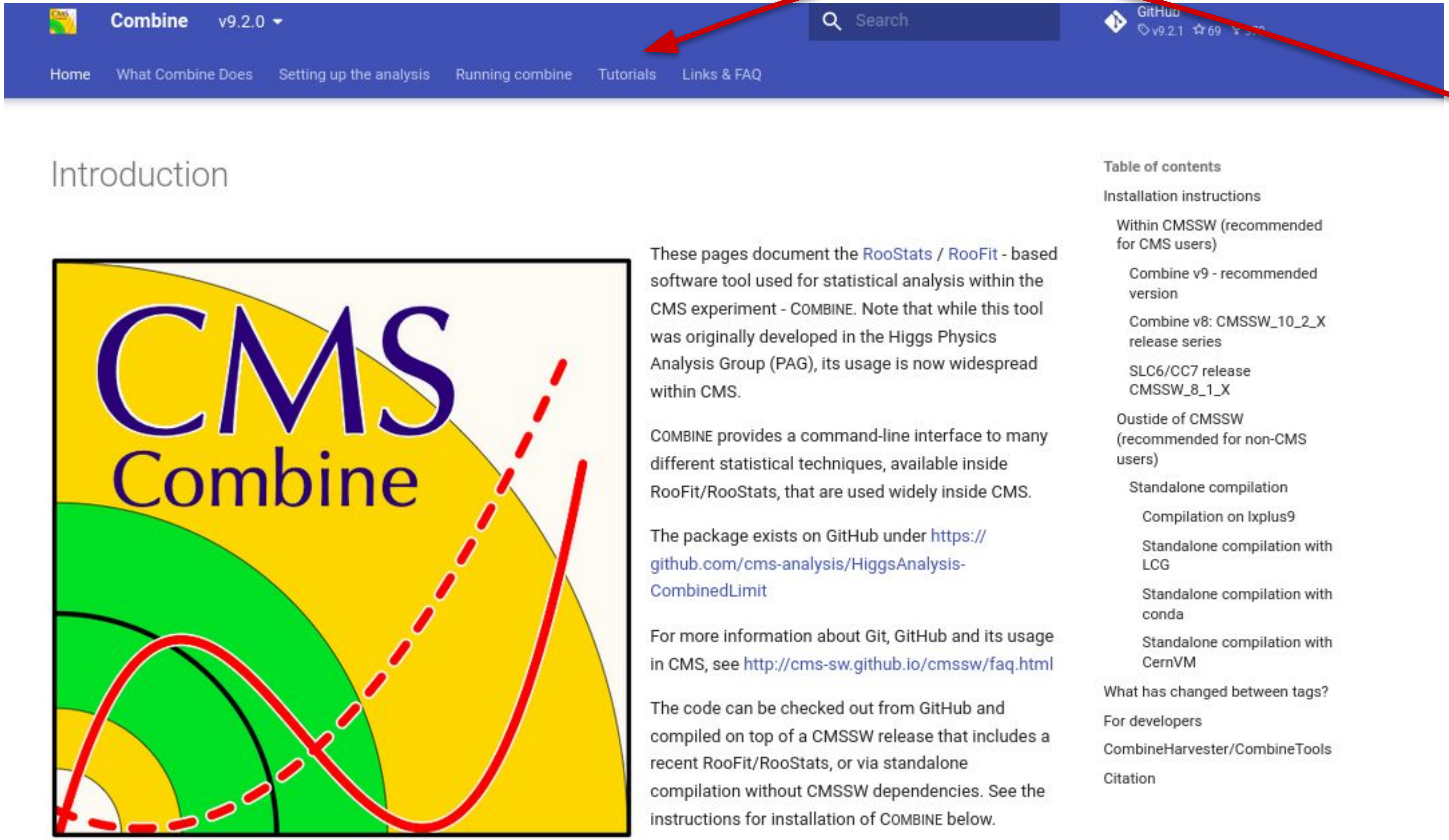




From Statistics Committee Questionnaires 2021-2022

Documentation

- (High quality) documentation + software has been public for a long time



Introduction

These pages document the RooStats / RooFit - based software tool used for statistical analysis within the CMS experiment - COMBINE. Note that while this tool was originally developed in the Higgs Physics Analysis Group (PAG), its usage is now widespread within CMS.

COMBINE provides a command-line interface to many different statistical techniques, available inside RooFit/RooStats, that are used widely inside CMS.

The package exists on GitHub under https://github.com/cms-analysis/HiggsAnalysis-CombinedLimit

For more information about Git, GitHub and its usage in CMS, see http://cms-sw.github.io/cmssw/faq.html

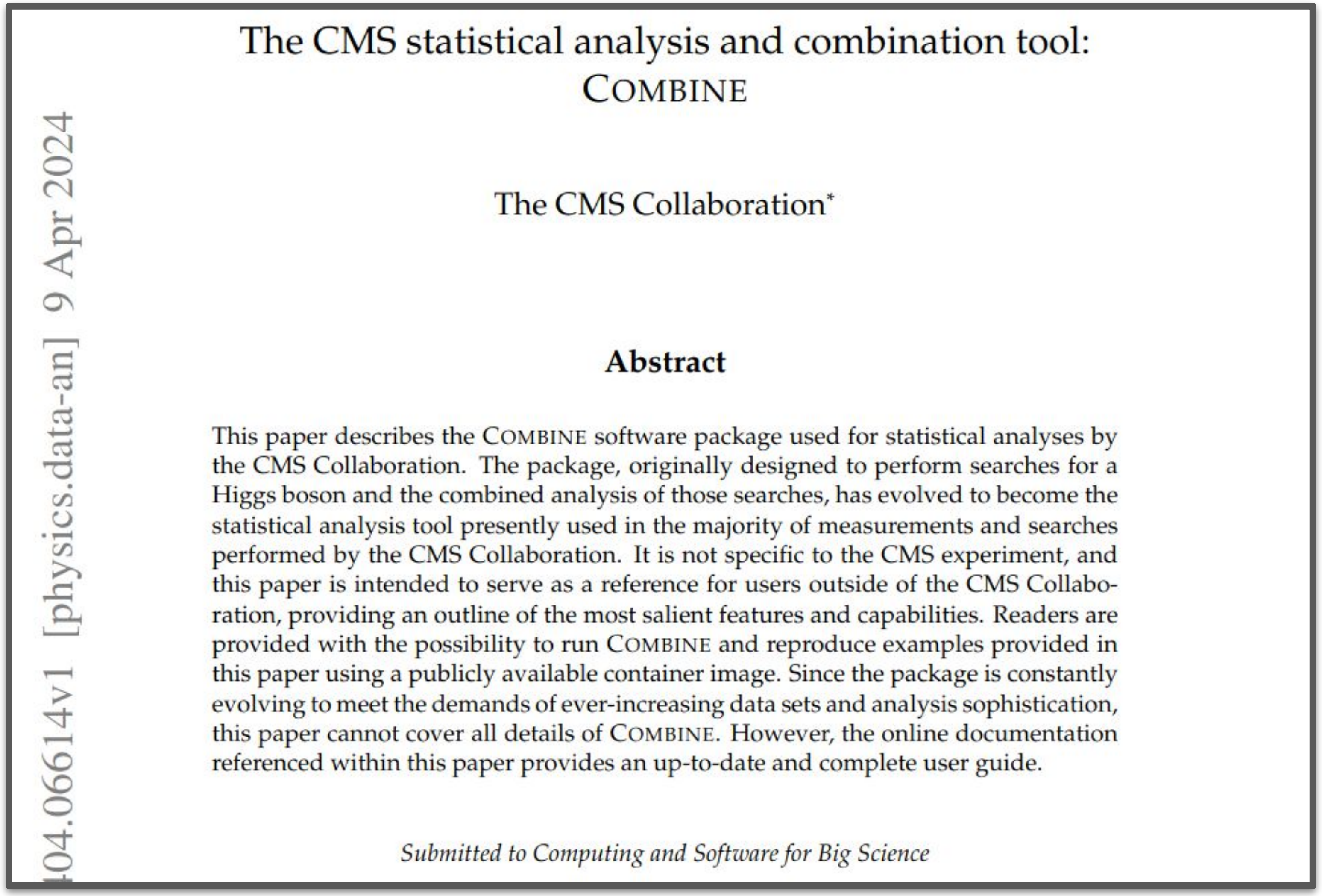
The code can be checked out from GitHub and compiled on top of a CMSSW release that includes a recent RooFit/RooStats, or via standalone compilation without CMSSW dependencies. See the instructions for installation of COMBINE below.

Table of contents

Complete with detailed tutorials

Hot off the press

- arXiv:2404.06614: submitted to Computing and Software for Big Science, citable public document!



404.06614v1 [physics.data-an] 9 Apr 2024

The CMS statistical analysis and combination tool: COMBINE

The CMS Collaboration*

Abstract

This paper describes the COMBINE software package used for statistical analyses by the CMS Collaboration. The package, originally designed to perform searches for a Higgs boson and the combined analysis of those searches, has evolved to become the statistical analysis tool presently used in the majority of measurements and searches performed by the CMS Collaboration. It is not specific to the CMS experiment, and this paper is intended to serve as a reference for users outside of the CMS Collaboration, providing an outline of the most salient features and capabilities. Readers are provided with the possibility to run COMBINE and reproduce examples provided in this paper using a publicly available container image. Since the package is constantly evolving to meet the demands of ever-increasing data sets and analysis sophistication, this paper cannot cover all details of COMBINE. However, the online documentation referenced within this paper provides an up-to-date and complete user guide.

Submitted to Computing and Software for Big Science

Paper ***aims to*** …

- Provide description of general statistical model (observables, pois & nuisance parameters) common to HEP statistical analyses
- Provide description of datacard/input formats for building binary workspaces for statistical models
- Describe common statistical routines available with combine

Paper is ***not*** …

- A replacement for all references – CMS papers should continue to cite original publications (eg CCGV for asymptotic limits, Run-1 combination for combination and fit procedures etc)
- A statistics paper – this paper should not be a reference for statistical methods

Focus on general capabilities of combine (what it is doing under the hood) instead of technical details on how to run the code

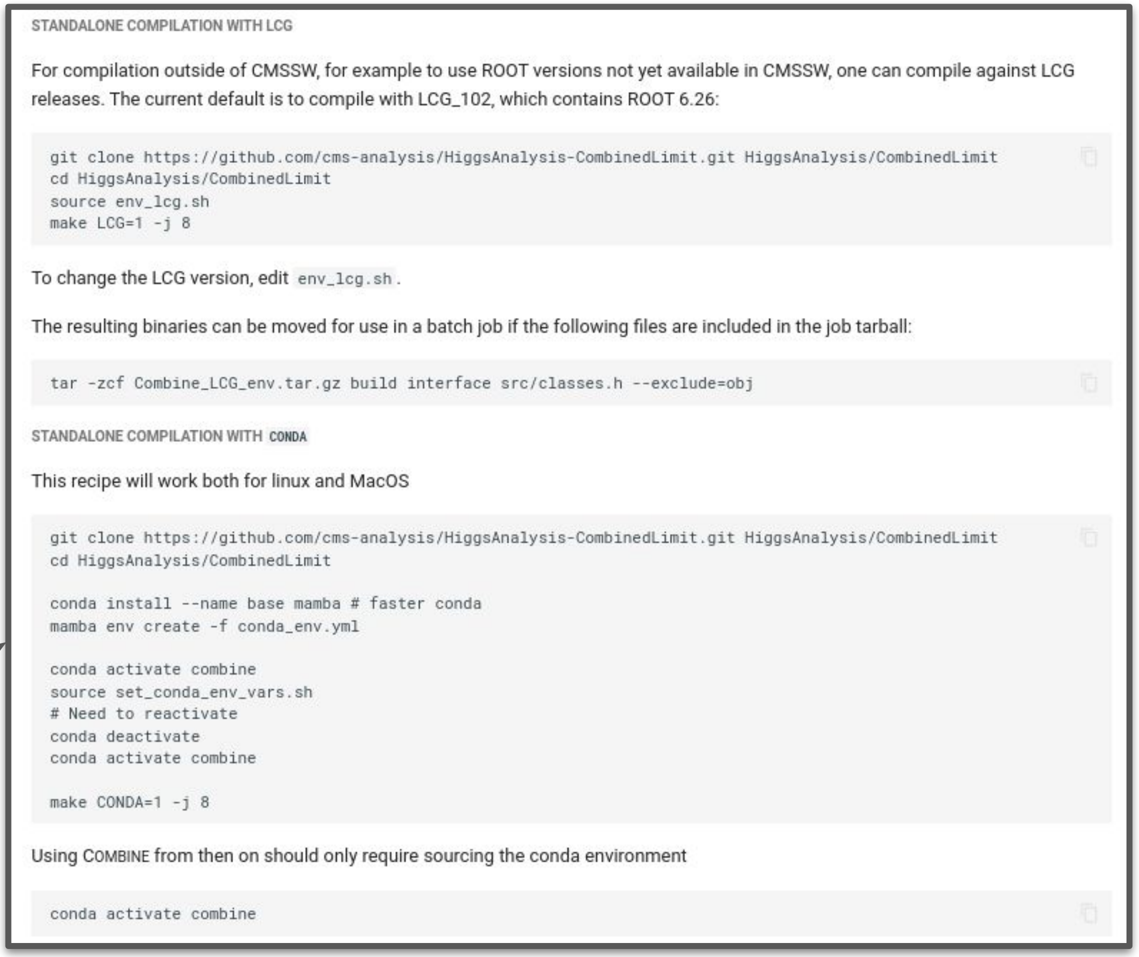
→ The online documentation ***won't*** be replaced.

Installation + dependencies

- Current recommended combine tag is v9.2.1 using ROOT 6.22
- Working branch with ROOT 6.26, using for Run 2 Higgs combination
- Yesterday we worked towards compilation/testing in ROOT 6.30... thanks!
  - **Task: complete today then validate results**
  - Benefit from nice RooFit updates, xRooFit compatibility, ...
  - Aiming for v10 tag in ~month timescale with 6.30 (6.26 Back-Up)
- CMS users can install within CMSSW environment
- For non-CMS users: provide pre-compiled versions as container images from CMS cloud

```
docker run --name combine -it gitlab-registry.cern.ch/cms-cloud/combine-standalone:<tag>
```

- Also standalone compilation options with LCG/conda
- **Today's task: compilation with cmake in** StatAnalysis ?



STANDALONE COMPILATION WITH LCG

For compilation outside of CMSSW, for example to use ROOT versions not yet available in CMSSW, one can compile against LCG releases. The current default is to compile with LCG_102, which contains ROOT 6.26:

```
git clone https://github.com/cms-analysis/HiggsAnalysis-CombinedLimit.git HiggsAnalysis/CombinedLimit
cd HiggsAnalysis/CombinedLimit
source env_lcg.sh
make LCG=1 -j 8
```

To change the LCG version, edit `env_lcg.sh`.

The resulting binaries can be moved for use in a batch job if the following files are included in the job tarball:

```
tar -zcf Combine_LCG_env.tar.gz build interface src/classes.h --exclude=obj
```

STANDALONE COMPILATION WITH `CONDA`

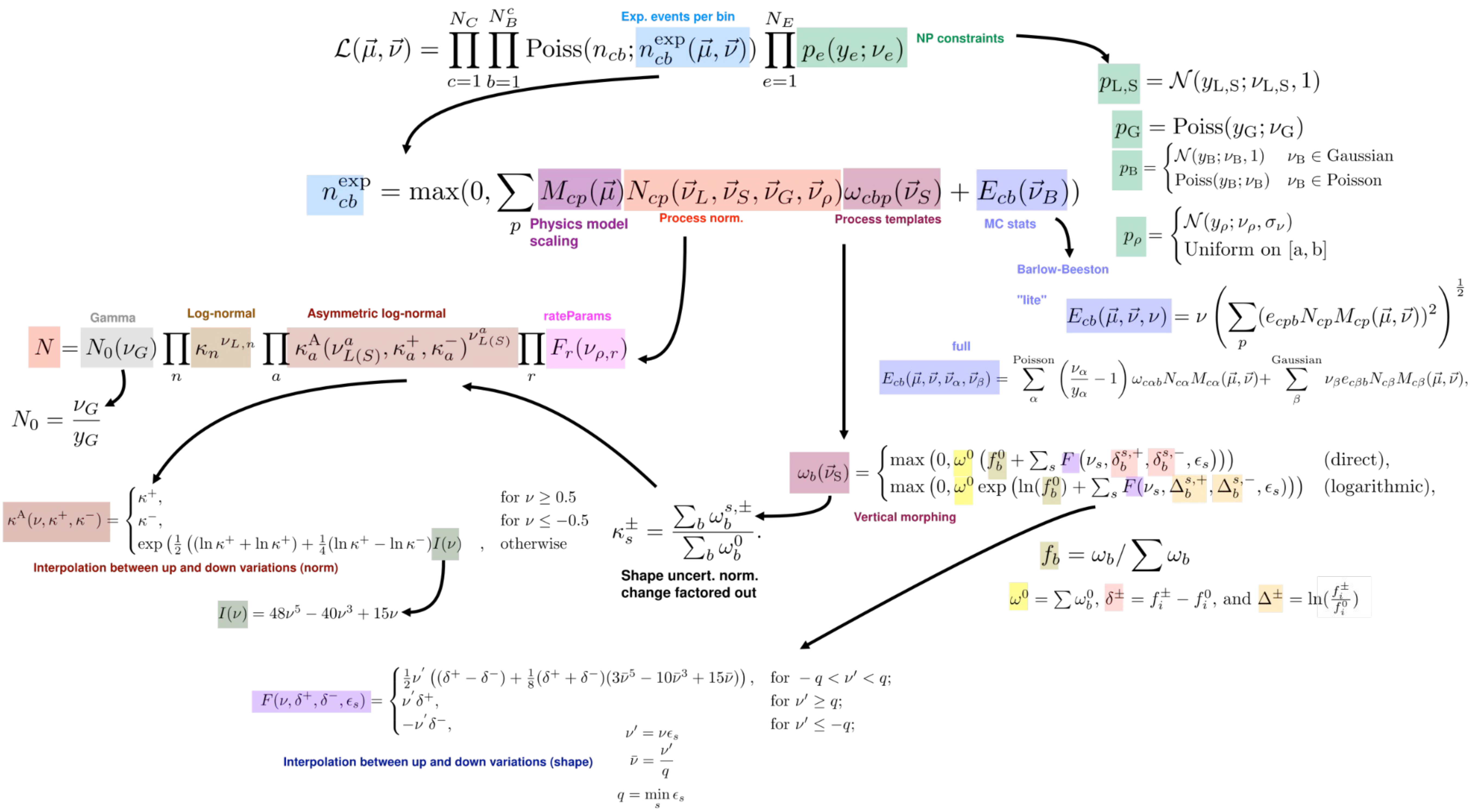
This recipe will work both for linux and MacOS

```
git clone https://github.com/cms-analysis/HiggsAnalysis-CombinedLimit.git HiggsAnalysis/CombinedLimit
cd HiggsAnalysis/CombinedLimit

conda install --name base mamba # faster conda
mamba env create -f conda_env.yml

conda activate combine
source set_conda_env_vars.sh
# Need to reactivate
conda deactivate
conda activate combine

make CONDA=1 -j 8
```

Using COMBINE from then on should only require sourcing the conda environment

```
conda activate combine
```

Likelihood construction (default binned model)



$$\mathcal{L}(\vec{\mu},\vec{\nu}) = \prod_{c=1}^{N_C}\prod_{b=1}^{N_B^c} \mathrm{Poiss}(n_{cb}; n_{cb}^{\mathrm{exp}}(\vec{\mu},\vec{\nu})) \prod_{e=1}^{N_E} p_e(y_e;\nu_e)$$

Exp. events per bin

NP constraints

$$p_{\mathrm{L,S}} = \mathcal{N}(y_{\mathrm{L,S}};\nu_{\mathrm{L,S}},1)$$

$$p_{\mathrm{G}} = \mathrm{Poiss}(y_{\mathrm{G}};\nu_{\mathrm{G}})$$

$$p_{\mathrm{B}} = \begin{cases}\mathcal{N}(y_{\mathrm{B}};\nu_{\mathrm{B}},1) & \nu_{\mathrm{B}} \in \text{Gaussian}\\ \mathrm{Poiss}(y_{\mathrm{B}};\nu_{\mathrm{B}}) & \nu_{\mathrm{B}} \in \text{Poisson}\end{cases}$$

$$p_\rho = \begin{cases}\mathcal{N}(y_\rho;\nu_\rho,\sigma_\nu)\\ \text{Uniform on [a, b]}\end{cases}$$

$$n_{cb}^{\mathrm{exp}} = \max(0, \sum_p M_{cp}(\vec{\mu}) N_{cp}(\vec{\nu}_L,\vec{\nu}_S,\vec{\nu}_G,\vec{\nu}_\rho)\omega_{cbp}(\vec{\nu}_S) + E_{cb}(\vec{\nu}_B))$$

Physics model scaling

Process norm.

Process templates

MC stats

Barlow-Beeston

"lite"

$$E_{cb}(\vec{\mu},\vec{\nu},\nu) = \nu\left(\sum_p (e_{cpb}N_{cp}M_{cp}(\vec{\mu},\vec{\nu}))^2\right)^{\frac{1}{2}}$$

full

$$E_{cb}(\vec{\mu},\vec{\nu},\vec{\nu}_\alpha,\vec{\nu}_\beta) = \sum_\alpha^{\text{Poisson}}\left(\frac{\nu_\alpha}{y_\alpha}-1\right)\omega_{c\alpha b}N_{c\alpha}M_{c\alpha}(\vec{\mu},\vec{\nu}) + \sum_\beta^{\text{Gaussian}} \nu_\beta e_{c\beta b}N_{c\beta}M_{c\beta}(\vec{\mu},\vec{\nu}),$$

Gamma

Log-normal

Asymmetric log-normal

rateParams

$$N = N_0(\nu_G)\prod_n \kappa_n^{\nu_{L,n}} \prod_a \kappa_a^{\mathrm{A}}(\nu_{L(S)}^a,\kappa_a^+,\kappa_a^-)^{\nu_{L(S)}^a} \prod_r F_r(\nu_{\rho,r})$$

$$N_0 = \frac{\nu_G}{y_G}$$

$$\kappa^{\mathrm{A}}(\nu,\kappa^+,\kappa^-) = \begin{cases}\kappa^+, & \text{for } \nu \geq 0.5\\ \kappa^-, & \text{for } \nu \leq -0.5\\ \exp\left(\tfrac{1}{2}\left((\ln\kappa^+ + \ln\kappa^+) + \tfrac{1}{4}(\ln\kappa^+ - \ln\kappa^-)I(\nu)\right.\right. \quad , & \text{otherwise}\end{cases}$$

Interpolation between up and down variations (norm)

$$I(\nu) = 48\nu^5 - 40\nu^3 + 15\nu$$

$$\kappa_s^\pm = \frac{\sum_b \omega_b^{s,\pm}}{\sum_b \omega_b^0}.$$

Shape uncert. norm. change factored out

$$\omega_b(\vec{\nu}_S) = \begin{cases}\max\left(0,\omega^0\left(f_b^0 + \sum_s F\left(\nu_s,\delta_b^{s,+},\delta_b^{s,-},\epsilon_s\right)\right)\right) & \text{(direct)},\\ \max\left(0,\omega^0\exp\left(\ln(f_b^0) + \sum_s F(\nu_s,\Delta_b^{s,+},\Delta_b^{s,-},\epsilon_s)\right)\right) & \text{(logarithmic)},\end{cases}$$

Vertical morphing

$$f_b = \omega_b/\sum\omega_b$$

$$\omega^0 = \sum\omega_b^0,\ \delta^\pm = f_i^\pm - f_i^0,\ \text{and}\ \Delta^\pm = \ln(\tfrac{f_i^\pm}{f_i^0})$$

$$F(\nu,\delta^+,\delta^-,\epsilon_s) = \begin{cases}\frac{1}{2}\nu'\left((\delta^+-\delta^-) + \frac{1}{8}(\delta^++\delta^-)(3\bar{\nu}^5 - 10\bar{\nu}^3 + 15\bar{\nu})\right), & \text{for } -q < \nu' < q;\\ \nu'\delta^+, & \text{for } \nu' \geq q;\\ -\nu'\delta^-, & \text{for } \nu' \leq -q;\end{cases}$$

$$\nu' = \nu\epsilon_s$$

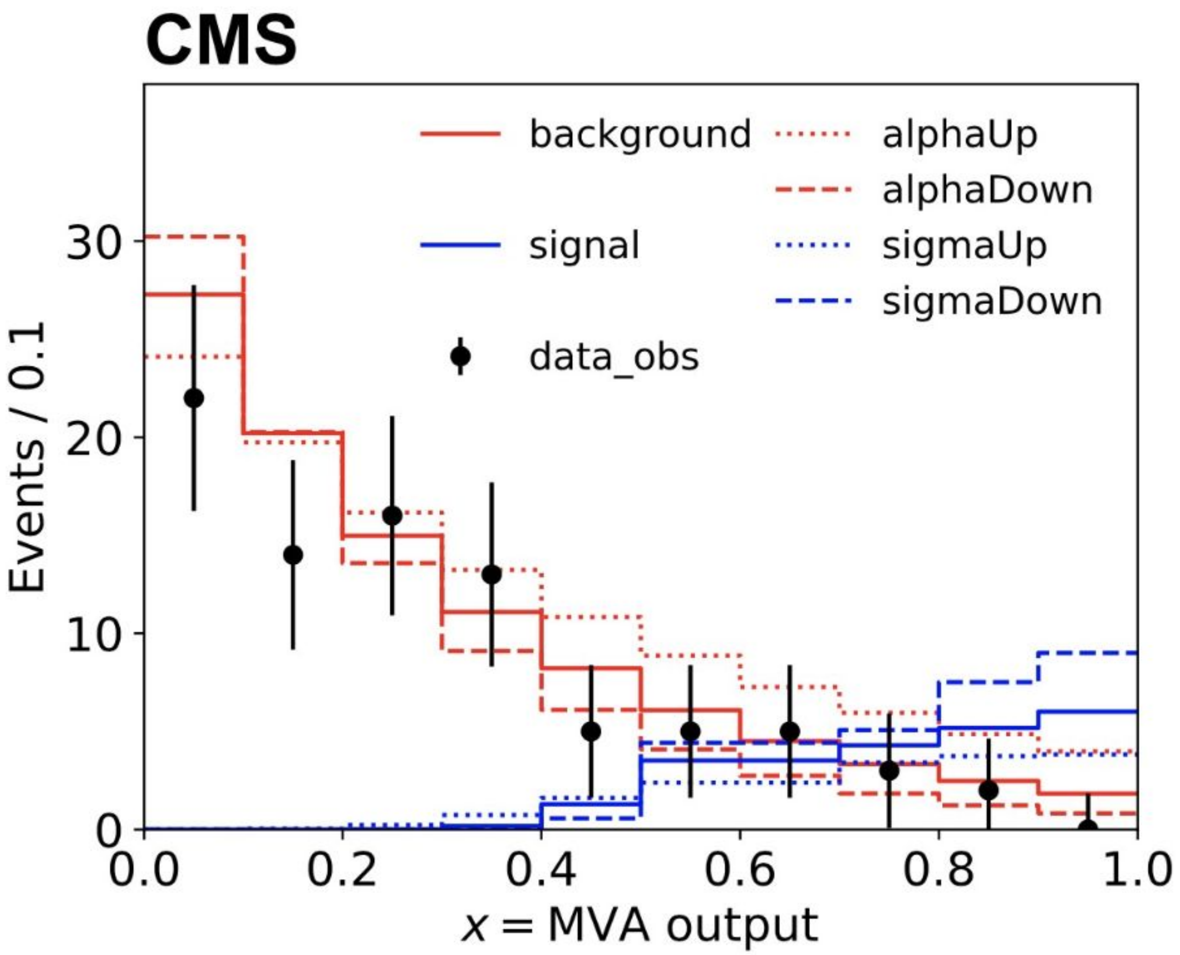
$$\bar{\nu} = \frac{\nu'}{q}$$

$$q = \min_s \epsilon_s$$

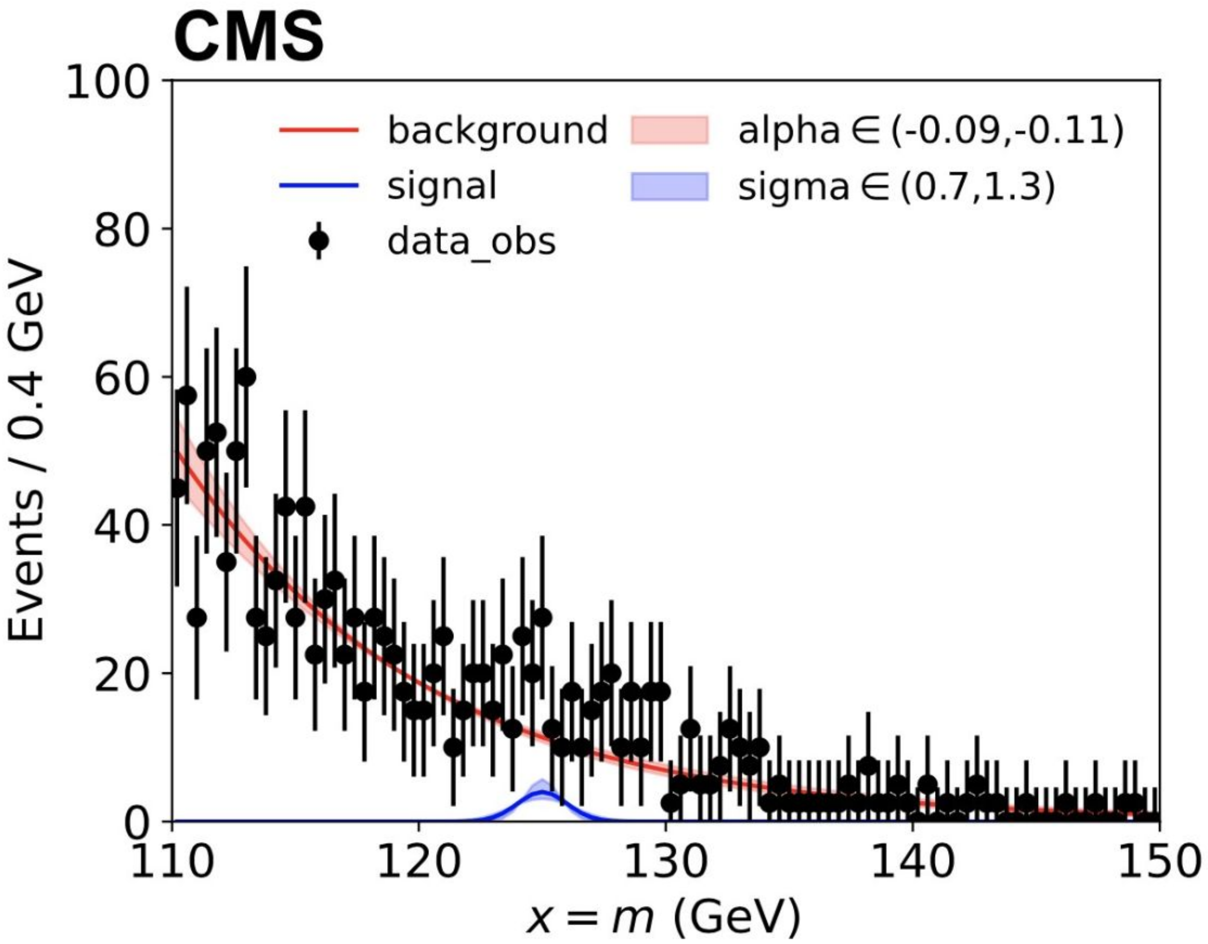
Interpolation between up and down variations (shape)

General purpose tool

- Great advantage of combine is **generality + sheer number of features**
  - Not limited to template (binned) fits
  - Command line tool with many options
  - Quickly go over a number of features...

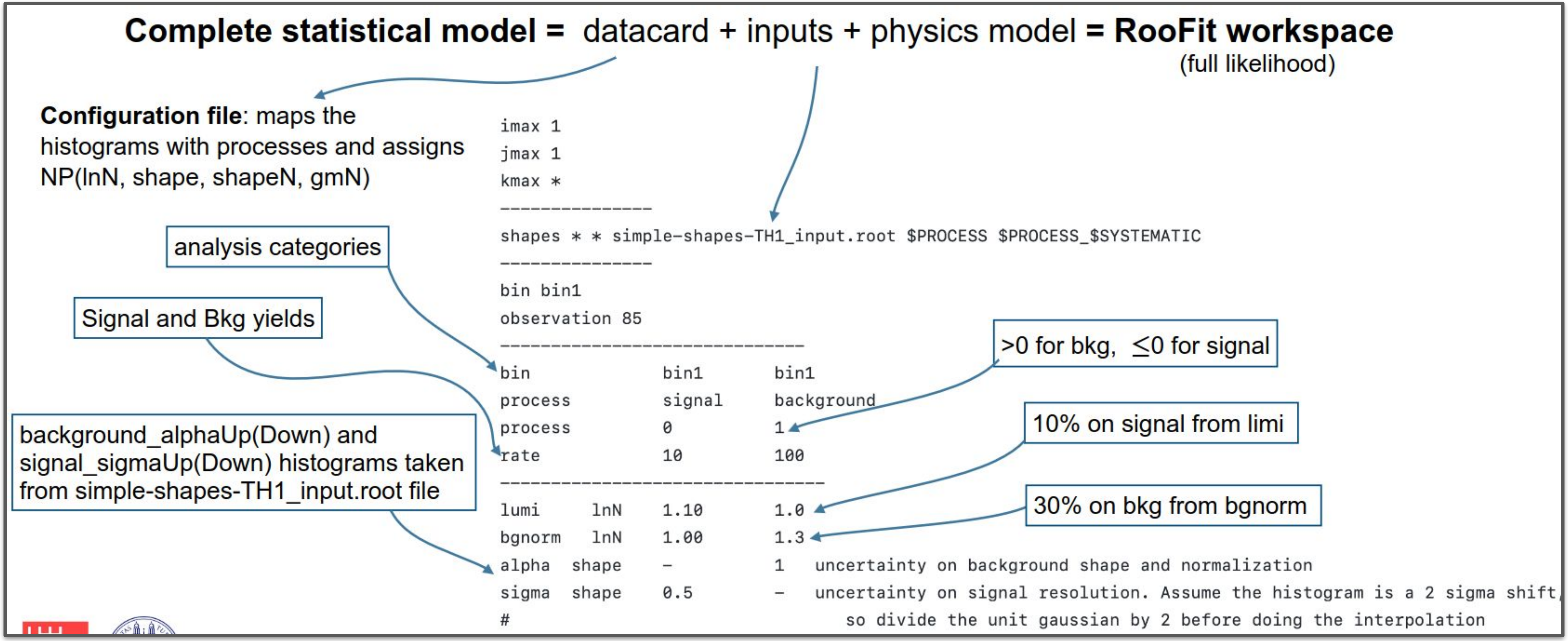


+



Inputs

- Configured by txt datacards...



Complete statistical model = datacard + inputs + physics model **= RooFit workspace**
(full likelihood)

Configuration file: maps the histograms with processes and assigns NP(lnN, shape, shapeN, gmN)

analysis categories

Signal and Bkg yields

background_alphaUp(Down) and signal_sigmaUp(Down) histograms taken from simple-shapes-TH1_input.root file

>0 for bkg, ≤0 for signal

10% on signal from limi

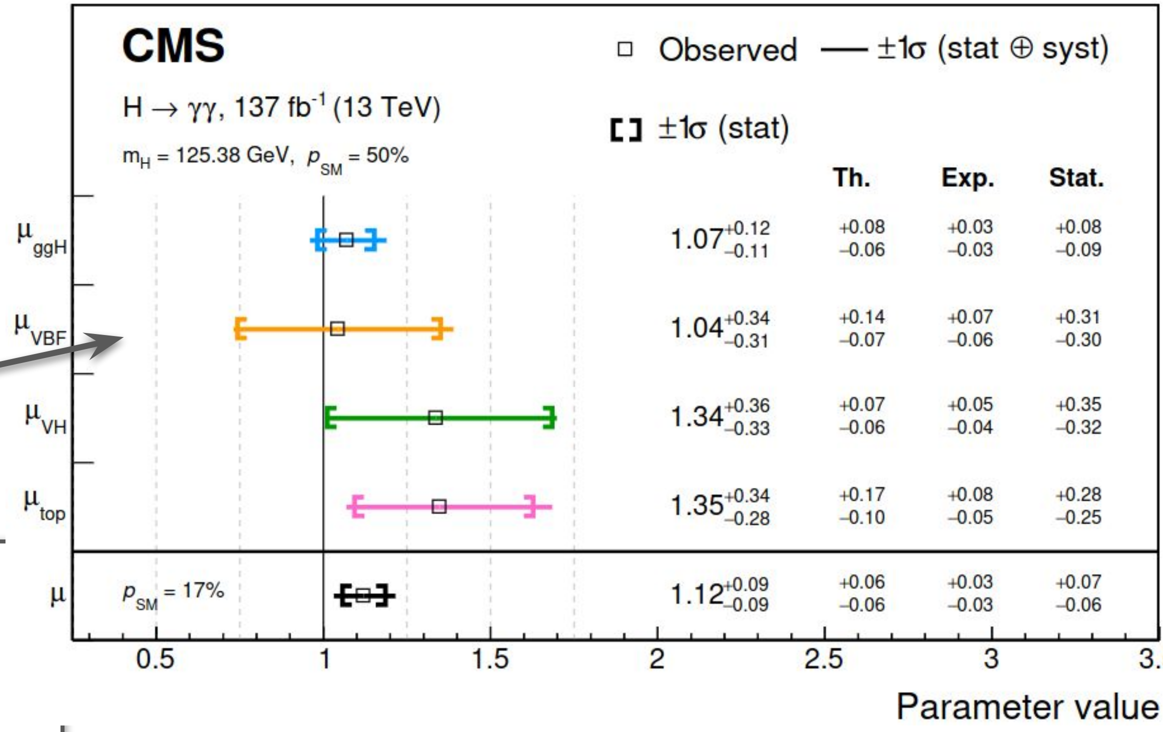
30% on bkg from bgnorm

```
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    shape     -           1    uncertainty on background shape and normalization
sigma    shape     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
```

```
$ text2workspace.py <datacard.txt> -P HiggsAnalysis.CombinedLimit.<PythonFile>:<
  ↪ modelInstance> [--PO <options>]
```

Physics models

- Provide large bank of models for physics interpretations



- Simple process scaling:
  - By default single POI assigned to all processes marked as signal in the datacards (“r”)
  - Multiple POIs can be assigned with [multiSignalModel] :

```
text2workspace.py -P HiggsAnalysis.CombinedLimit.PhysicsModel:multiSignalModel  --PO verbose
--PO 'map=.*/sig1:r_sig1[1,0,10]' --PO ‘map=.*/sig2:r_sig2[1,0,20]' datacard.txt -o ws.root
```

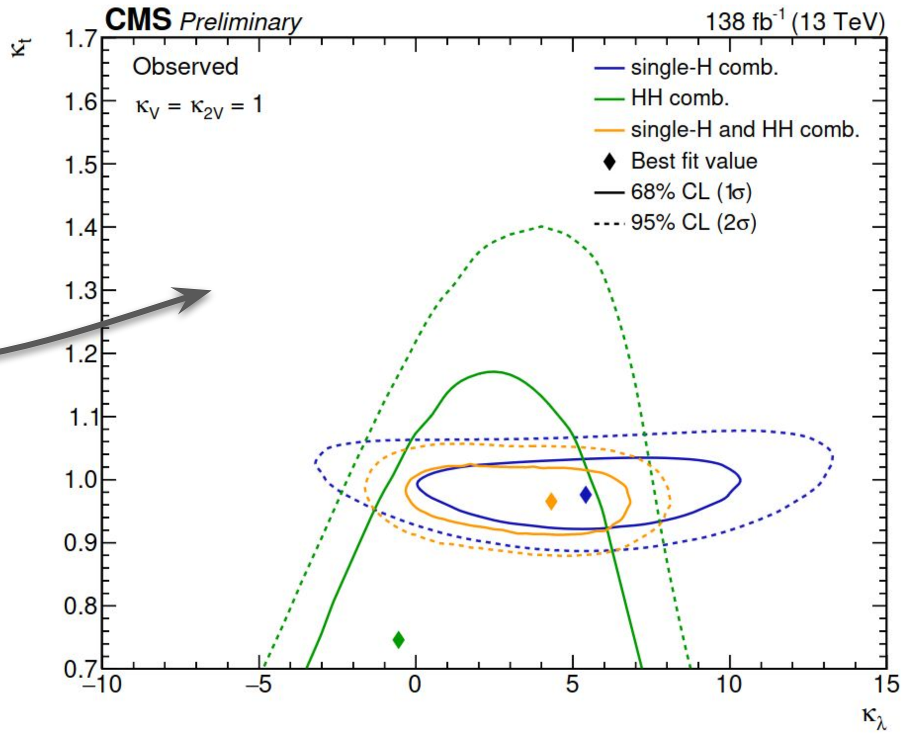
  - Mapping: `--PO 'map=bin/process:parameter'`

- More complicated: Interference, κ, EFT (signal processes are parametrised)
  - Use existing model [location]:

```
text2workspace.py datacard -P HiggsAnalysis.CombinedLimit.PythonFile:modelName
```

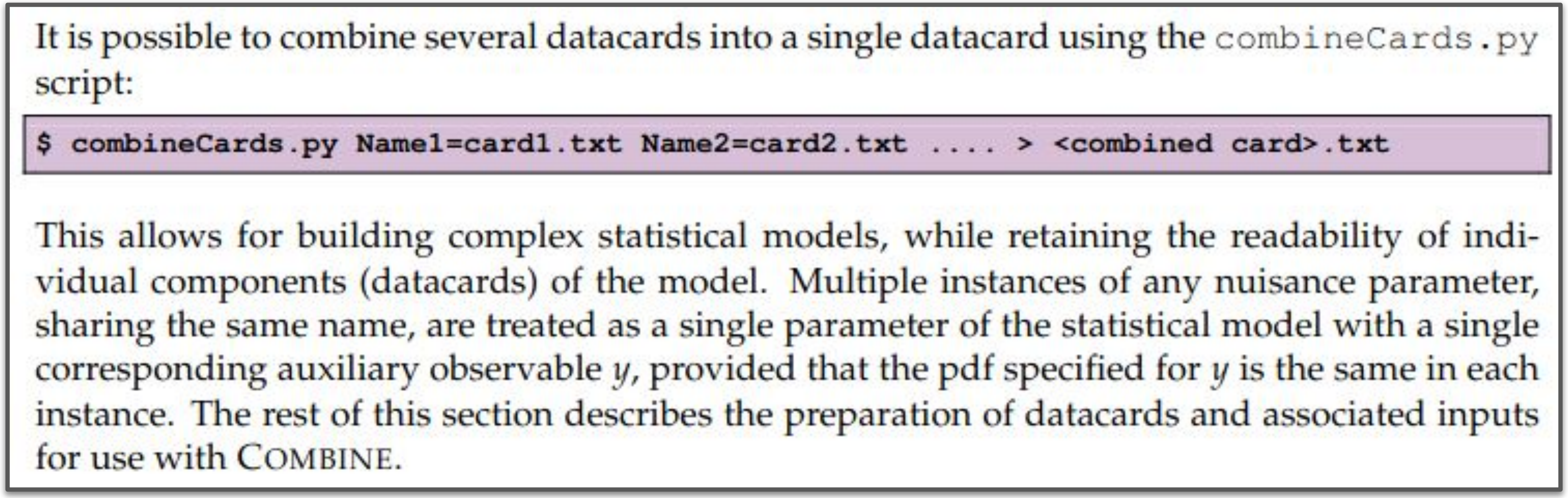
    e.g for Higgs couplings κ-model: `-P HiggsAnalysis.CombinedLimit.HiggsCouplings:c7` [link]

  - Create your own model in CombinedLimit/python directory; example given here on a model with interference: [tutorial], [code]



Combining channels

- Combine makes it very easy to combine channels (categories, regions, analyses, …)
- Does what it says on the tin

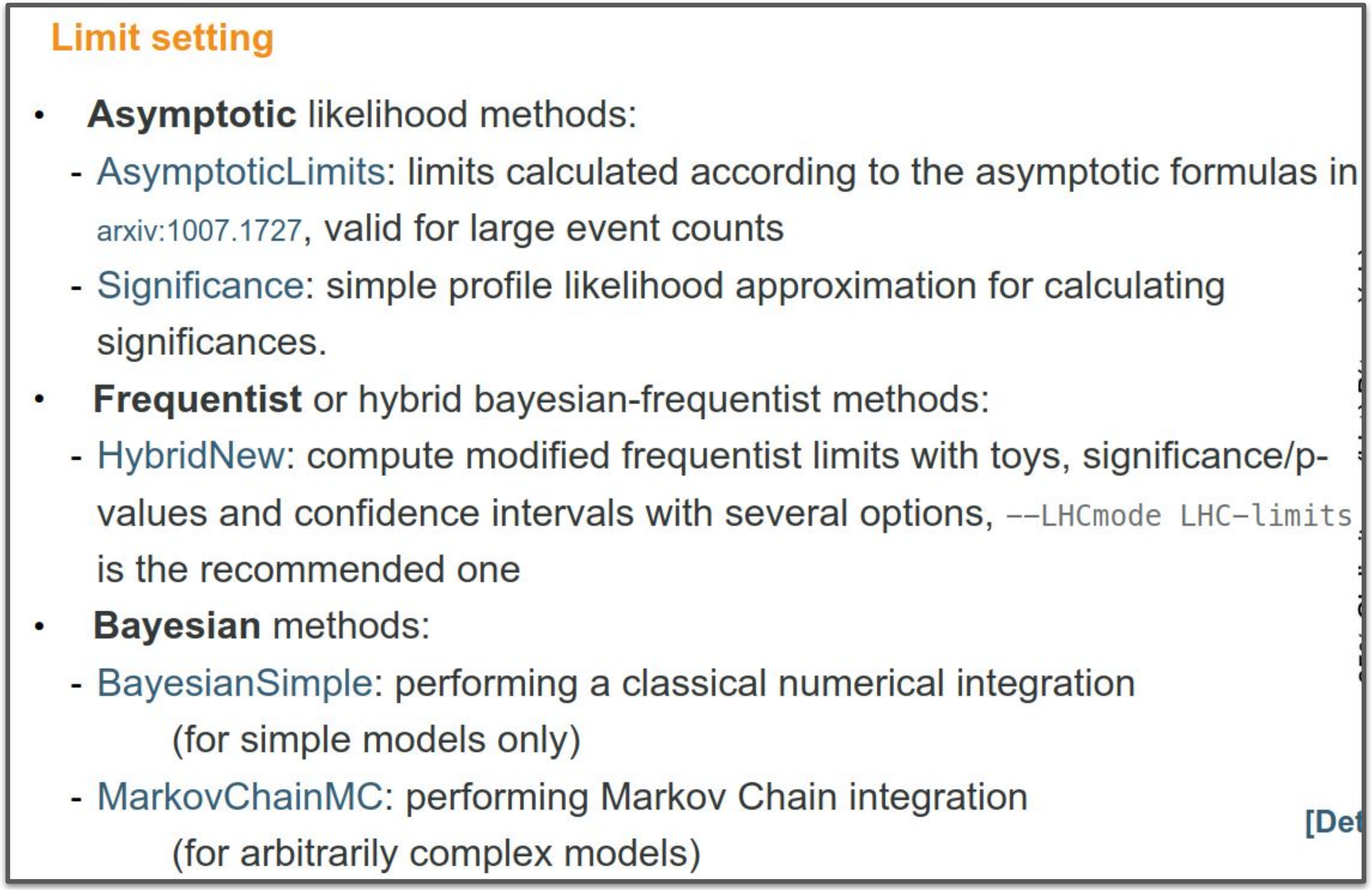


It is possible to combine several datacards into a single datacard using the `combineCards.py` script:

```
$ combineCards.py Name1=card1.txt Name2=card2.txt .... > <combined card>.txt
```

This allows for building complex statistical models, while retaining the readability of individual components (datacards) of the model. Multiple instances of any nuisance parameter, sharing the same name, are treated as a single parameter of the statistical model with a single corresponding auxiliary observable y, provided that the pdf specified for y is the same in each instance. The rest of this section describes the preparation of datacards and associated inputs for use with COMBINE.

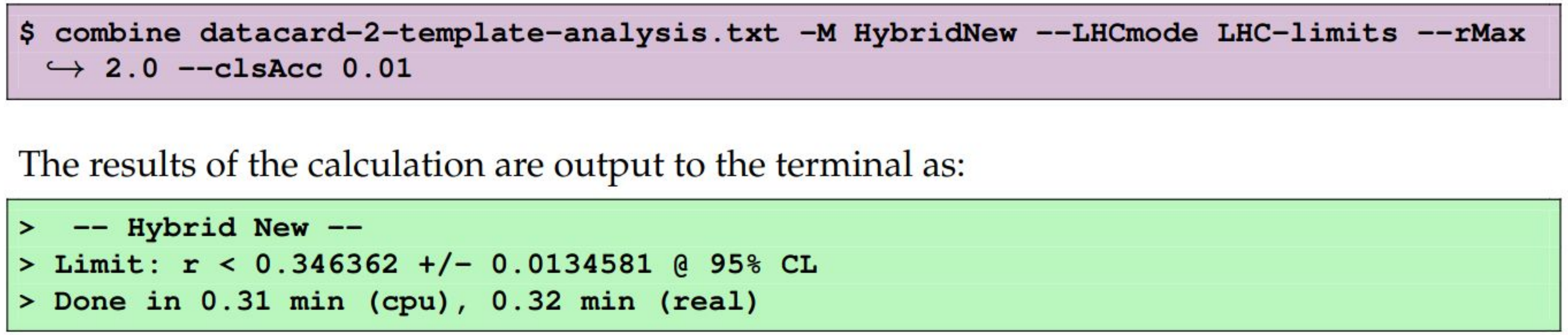
Main methods



Limit setting

- **Asymptotic** likelihood methods:
  - AsymptoticLimits: limits calculated according to the asymptotic formulas in arxiv:1007.1727, valid for large event counts
  - Significance: simple profile likelihood approximation for calculating significances.
- **Frequentist** or hybrid bayesian-frequentist methods:
  - HybridNew: compute modified frequentist limits with toys, significance/p-values and confidence intervals with several options, `--LHCmode LHC-limits` is the recommended one
- **Bayesian** methods:
  - BayesianSimple: performing a classical numerical integration (for simple models only)
  - MarkovChainMC: performing Markov Chain integration (for arbitrarily complex models)

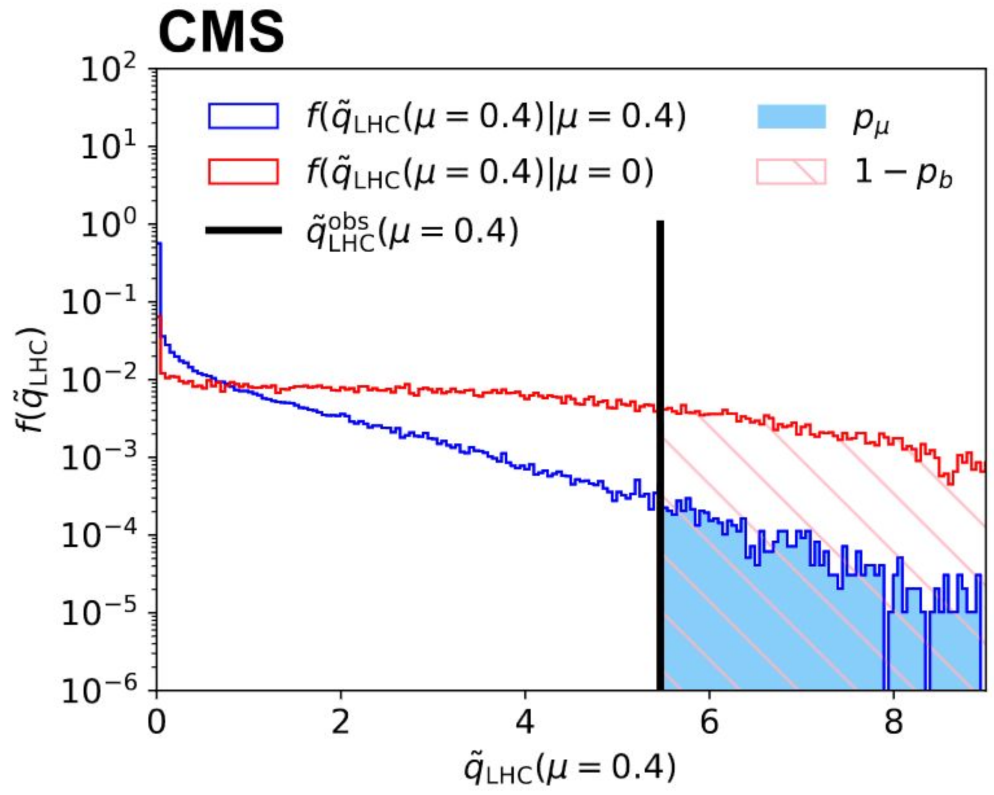
[Det

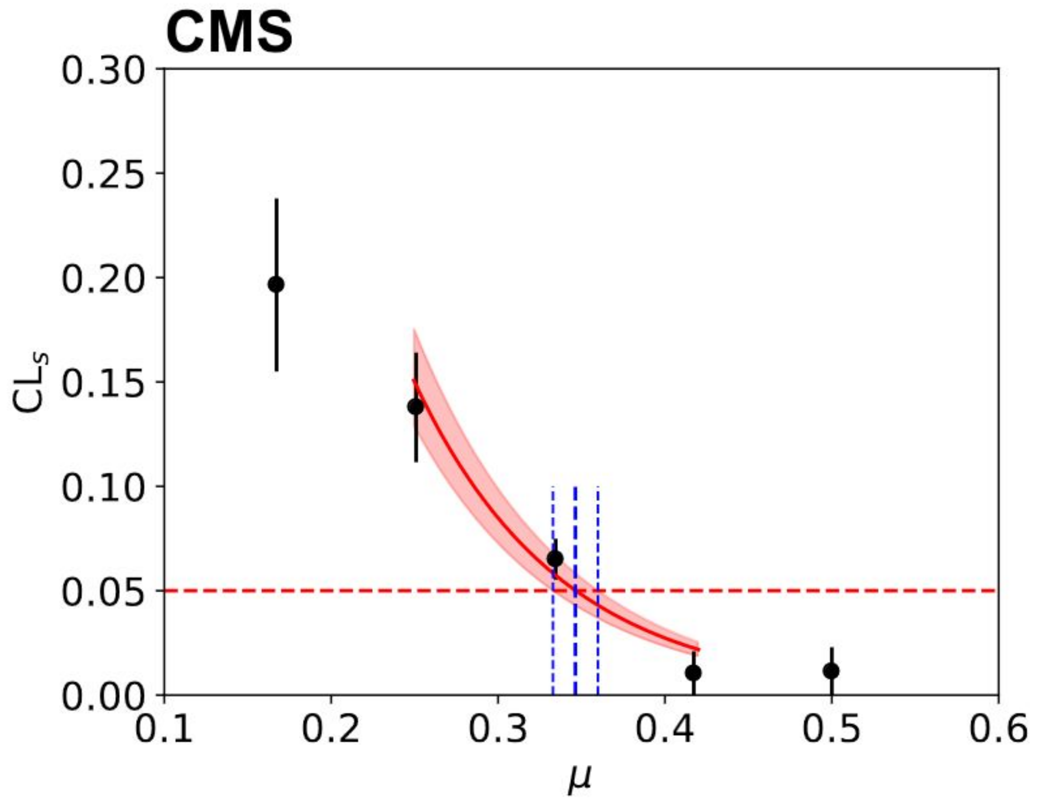


```
$ combine datacard-2-template-analysis.txt -M HybridNew --LHCmode LHC-limits --rMax
 ↪ 2.0 --clsAcc 0.01
```

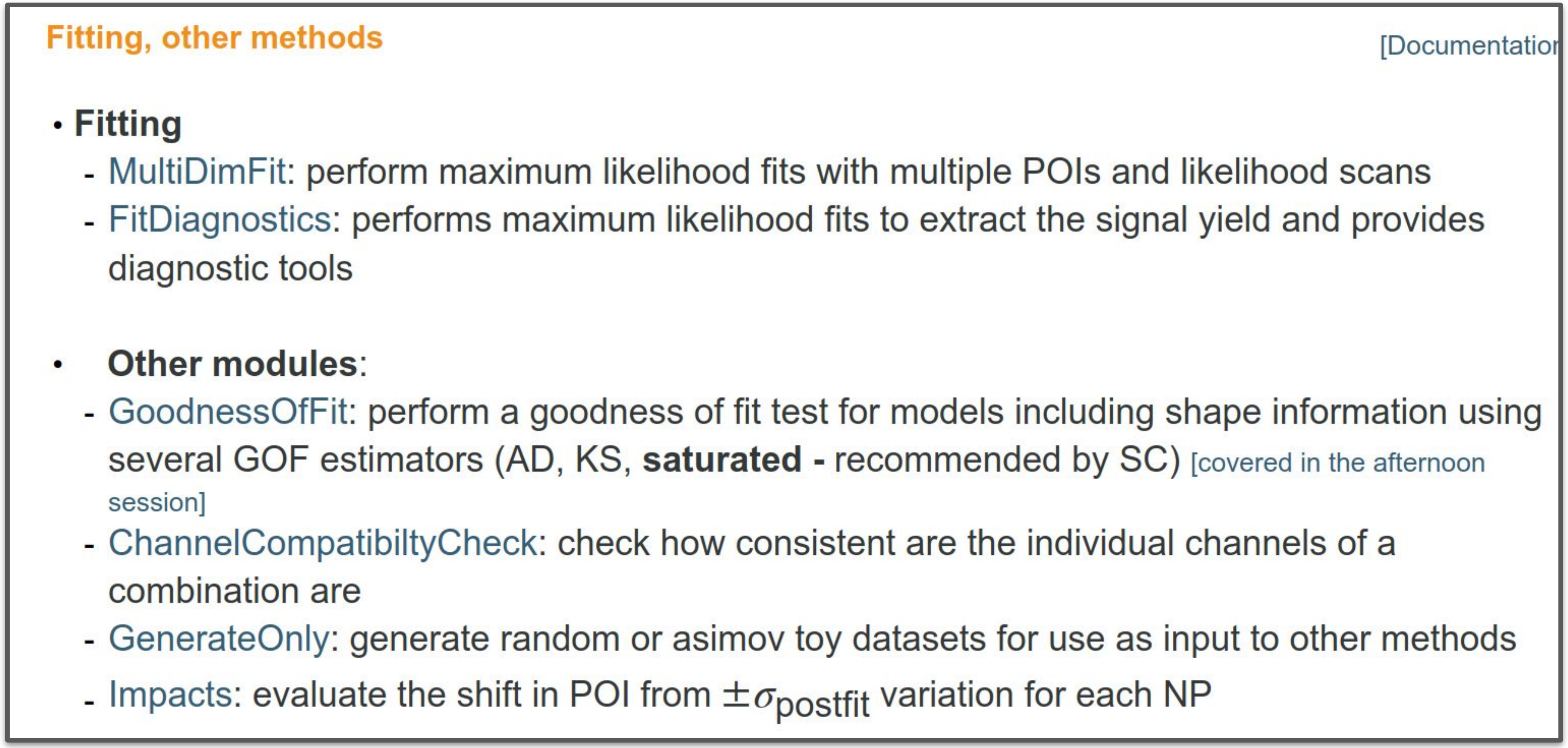
The results of the calculation are output to the terminal as:

```
>  -- Hybrid New --
> Limit: r < 0.346362 +/- 0.0134581 @ 95% CL
> Done in 0.31 min (cpu), 0.32 min (real)
```





Main methods

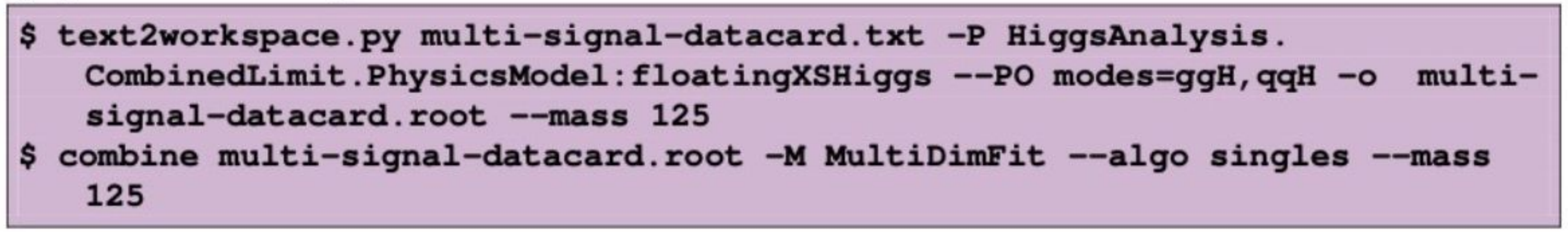


Fitting, other methods

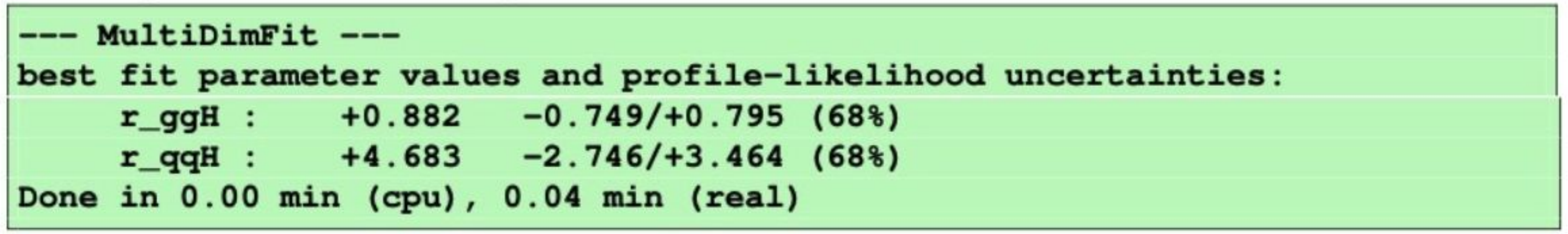
[Documentatio

- **Fitting**
  - MultiDimFit: perform maximum likelihood fits with multiple POIs and likelihood scans
  - FitDiagnostics: performs maximum likelihood fits to extract the signal yield and provides diagnostic tools

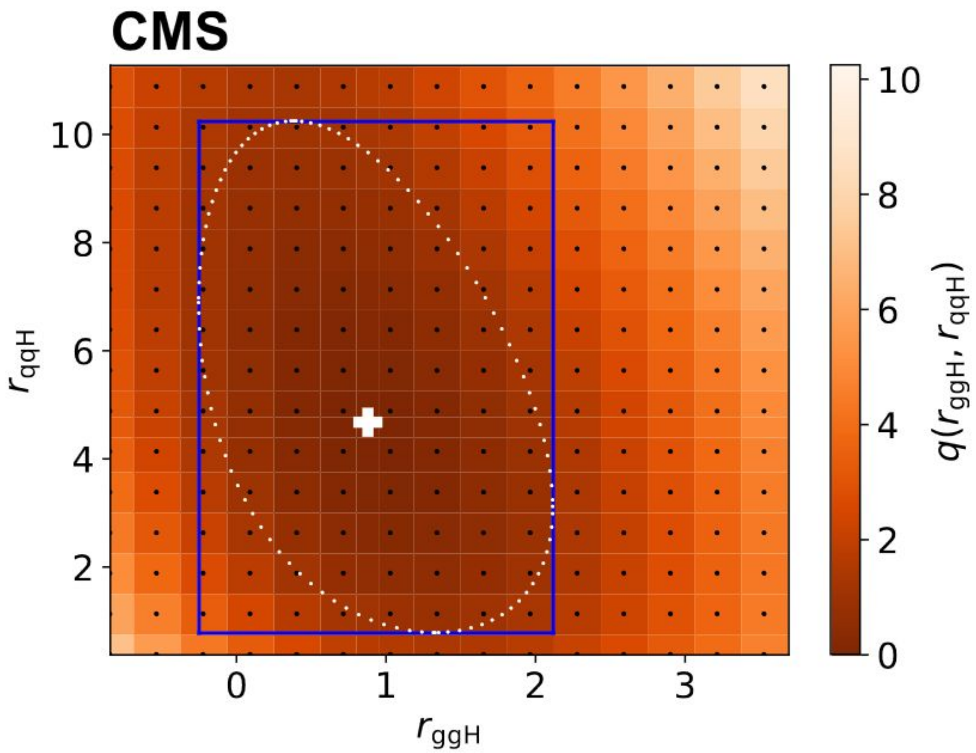
- **Other modules**:
  - GoodnessOfFit: perform a goodness of fit test for models including shape information using several GOF estimators (AD, KS, **saturated -** recommended by SC) [covered in the afternoon session]
  - ChannelCompatibiltyCheck: check how consistent are the individual channels of a combination are
  - GenerateOnly: generate random or asimov toy datasets for use as input to other methods
  - Impacts: evaluate the shift in POI from $\pm\sigma_{\text{postfit}}$ variation for each NP

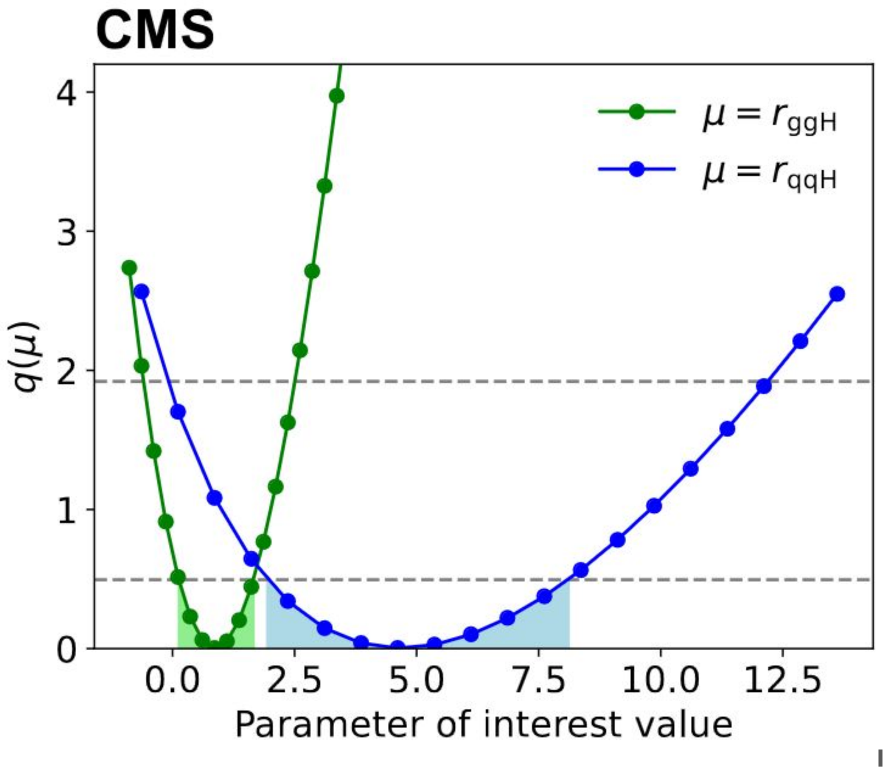


```
$ text2workspace.py multi-signal-datacard.txt -P HiggsAnalysis.
   CombinedLimit.PhysicsModel:floatingXSHiggs --PO modes=ggH,qqH -o  multi-
   signal-datacard.root --mass 125
$ combine multi-signal-datacard.root -M MultiDimFit --algo singles --mass
   125
```



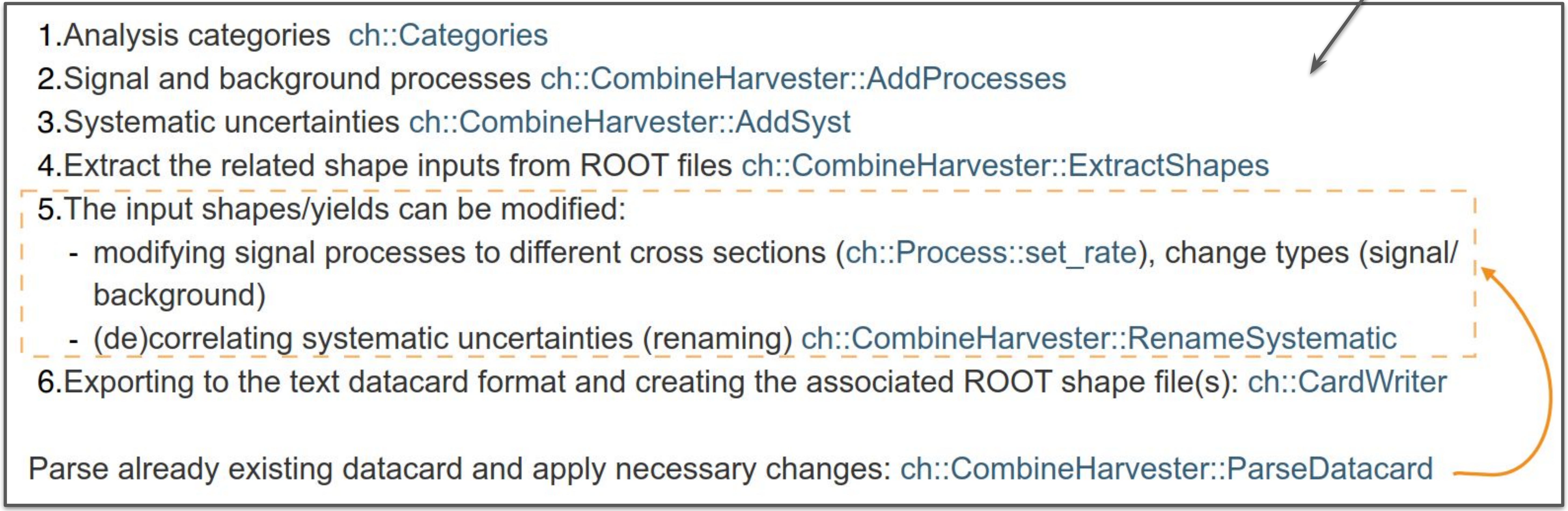
```
--- MultiDimFit ---
best fit parameter values and profile-likelihood uncertainties:
     r_ggH :    +0.882   -0.749/+0.795 (68%)
     r_qqH :    +4.683   -2.746/+3.464 (68%)
Done in 0.00 min (cpu), 0.04 min (real)
```





CombineHarvester tool

- CombineHarvester: C++ package with python interface to create and modify datacards (proto API)
- Also contains scripts for batch submission of combine jobs (combineTool.py)
- + plotting scripts
- **Future work:** integrate CH functionality within combine + provide complete API



1.Analysis categories ch::Categories

2.Signal and background processes ch::CombineHarvester::AddProcesses

3.Systematic uncertainties ch::CombineHarvester::AddSyst

4.Extract the related shape inputs from ROOT files ch::CombineHarvester::ExtractShapes

5.The input shapes/yields can be modified:

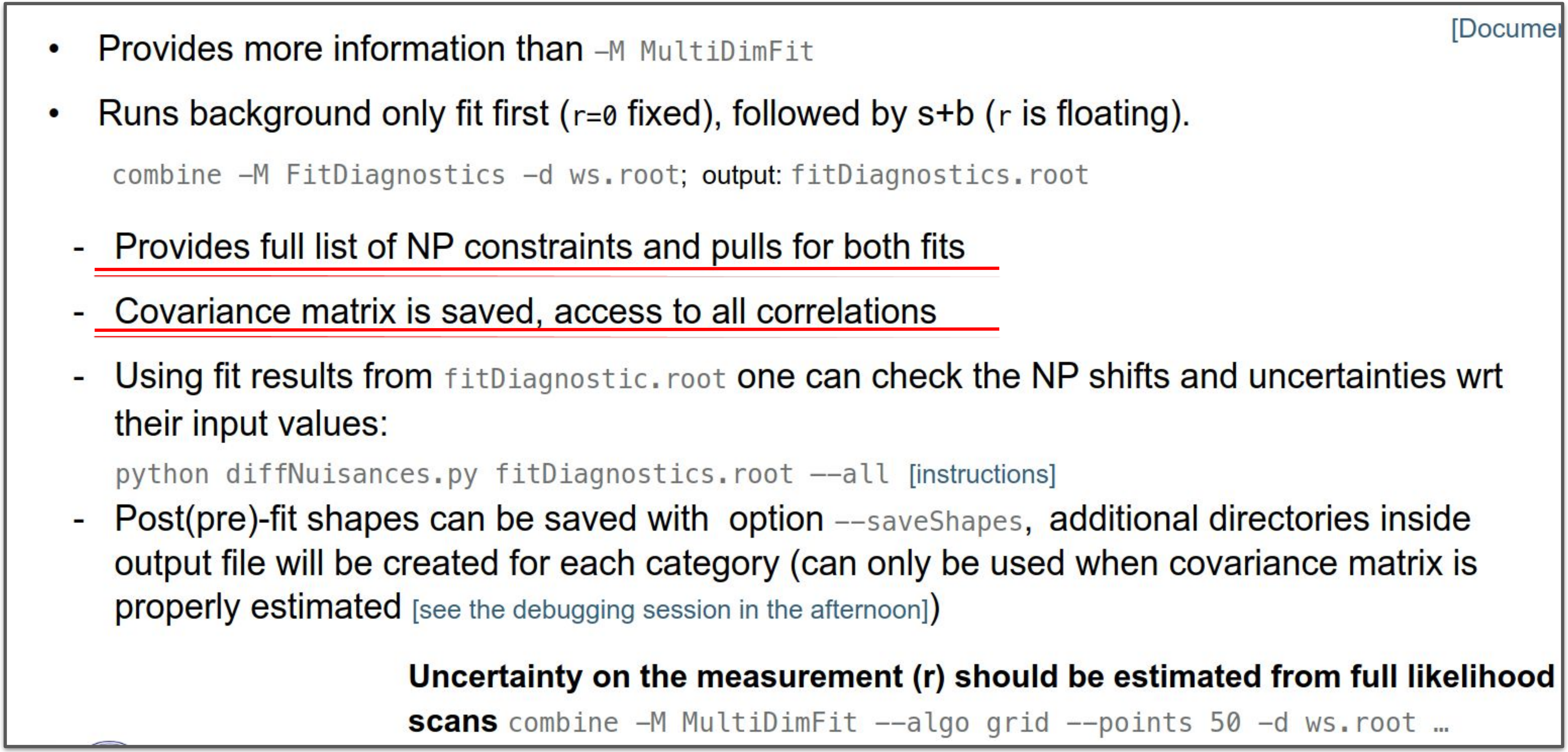
- modifying signal processes to different cross sections (ch::Process::set_rate), change types (signal/ background)
- (de)correlating systematic uncertainties (renaming) ch::CombineHarvester::RenameSystematic

6.Exporting to the text datacard format and creating the associated ROOT shape file(s): ch::CardWriter

Parse already existing datacard and apply necessary changes: ch::CombineHarvester::ParseDatacard

(Non-exhaustive) list of additional features

Additional features: FitDiagnostics



[Docume

- Provides more information than `-M MultiDimFit`
- Runs background only fit first (`r=0` fixed), followed by s+b (`r` is floating).

`combine -M FitDiagnostics -d ws.root`; output: `fitDiagnostics.root`

- Provides full list of NP constraints and pulls for both fits
- Covariance matrix is saved, access to all correlations
- Using fit results from `fitDiagnostic.root` one can check the NP shifts and uncertainties wrt their input values:

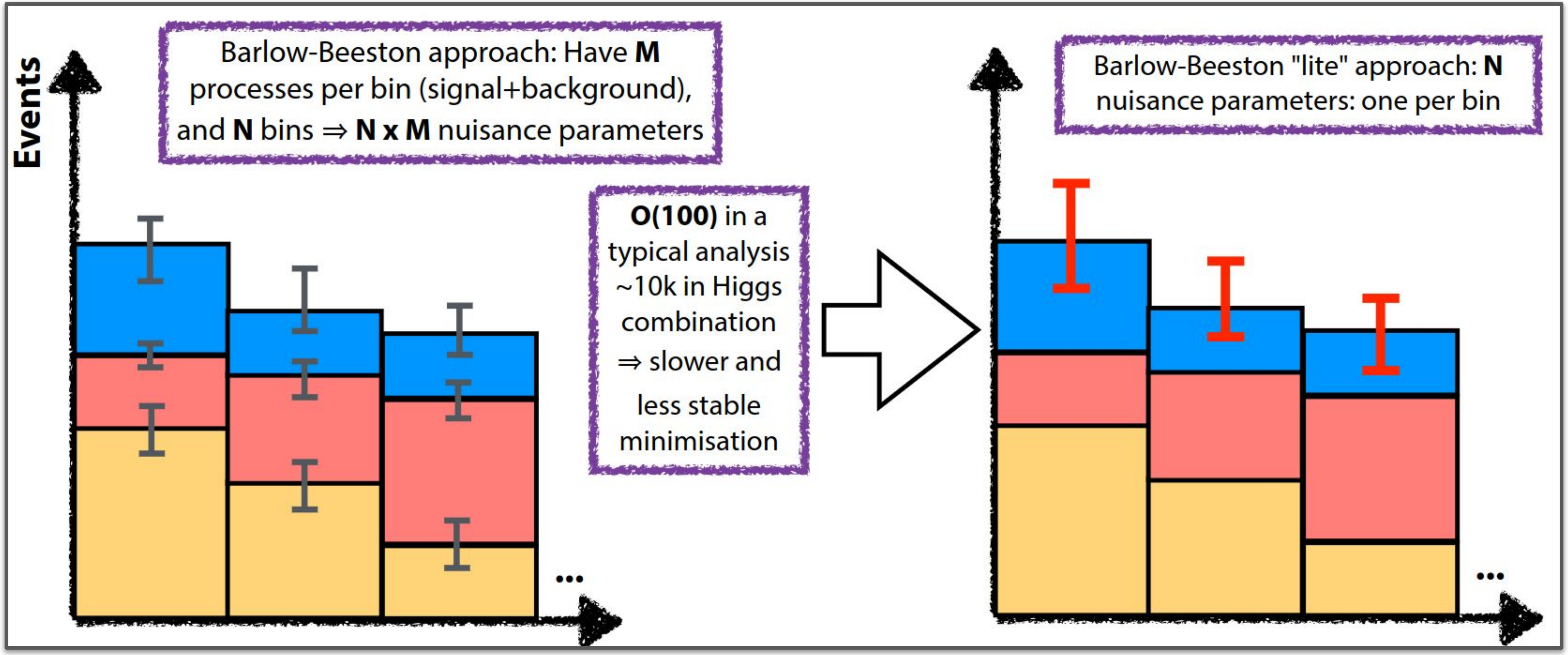
```
python diffNuisances.py fitDiagnostics.root --all
```
[instructions]

- Post(pre)-fit shapes can be saved with option `--saveShapes`, additional directories inside output file will be created for each category (can only be used when covariance matrix is properly estimated [see the debugging session in the afternoon])

Uncertainty on the measurement (r) should be estimated from full likelihood scans `combine -M MultiDimFit --algo grid --points 50 -d ws.root …`

Additional features: autoMCStats

- Feature in combine for incorporating uncertainties due to finite event counts in templates
- Automatically models total uncertainty in each bin with single Gaussian constraint (“lite” approach)
  - Only need to add a single line in datacard to enable
  - Falls back to per-process Poisson if MC stats too low in any particular bin



Barlow-Beeston approach: Have **M** processes per bin (signal+background), and **N** bins ⇒ **N x M** nuisance parameters

O(100) in a typical analysis ~10k in Higgs combination ⇒ slower and less stable minimisation

Barlow-Beeston "lite" approach: **N** nuisance parameters: one per bin

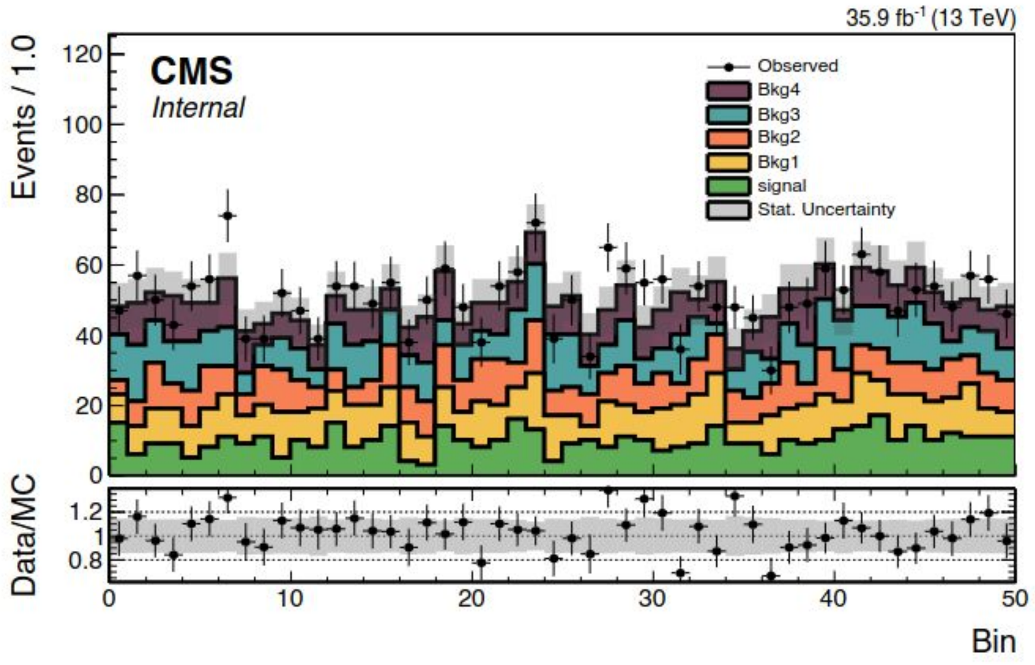
Additional features: autoMCStats

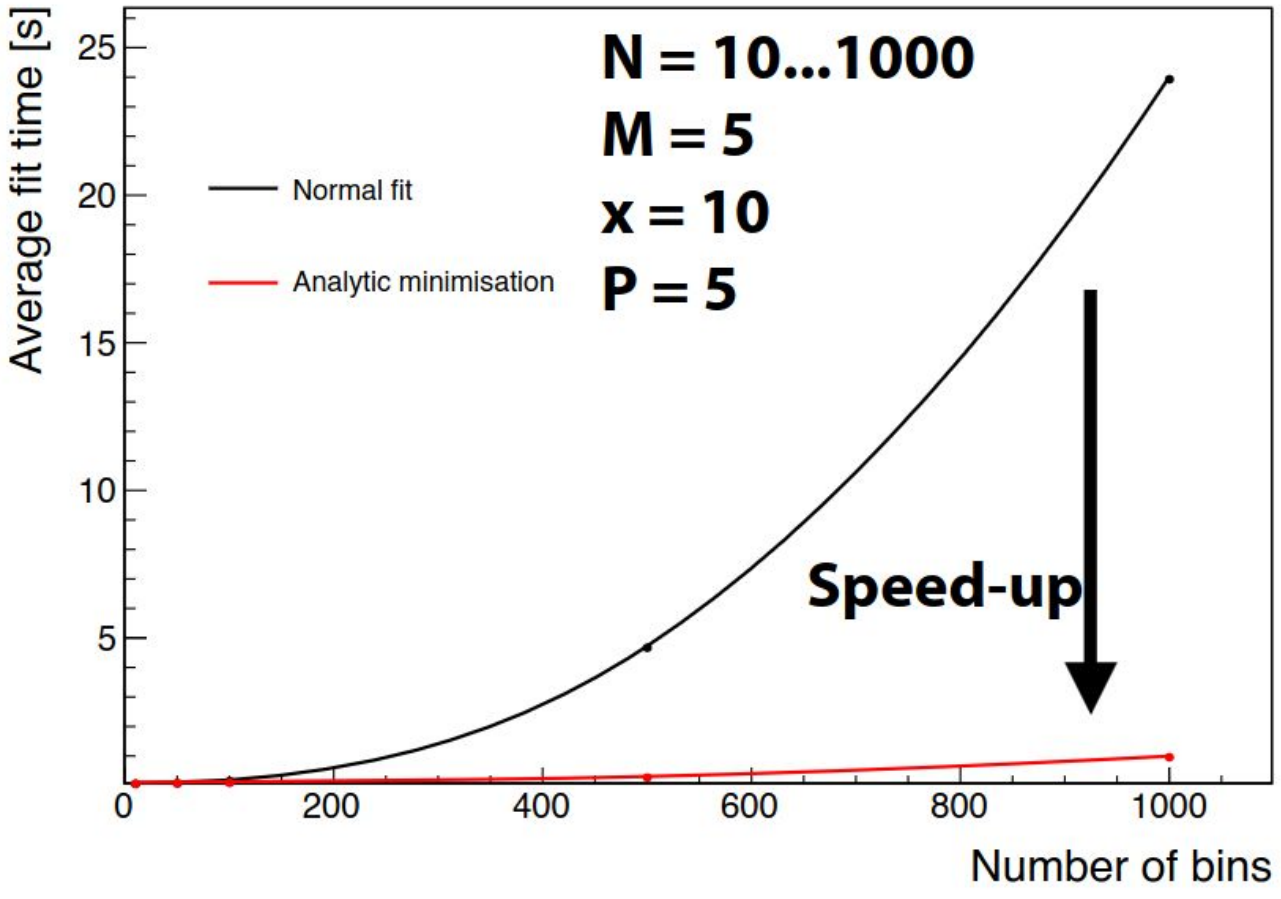
- There is no pruning of uncertainties in this implementation (too error prone) - there will be one nuisance parameter for every populated bin
  - Fitting time can still be long if many bins

- But with the lite approach the maximum likelihood for each parameter is independent of the others and has a simple form that we can solve

- The custom minimizer in combine handles the analytic minimisation of these parameters

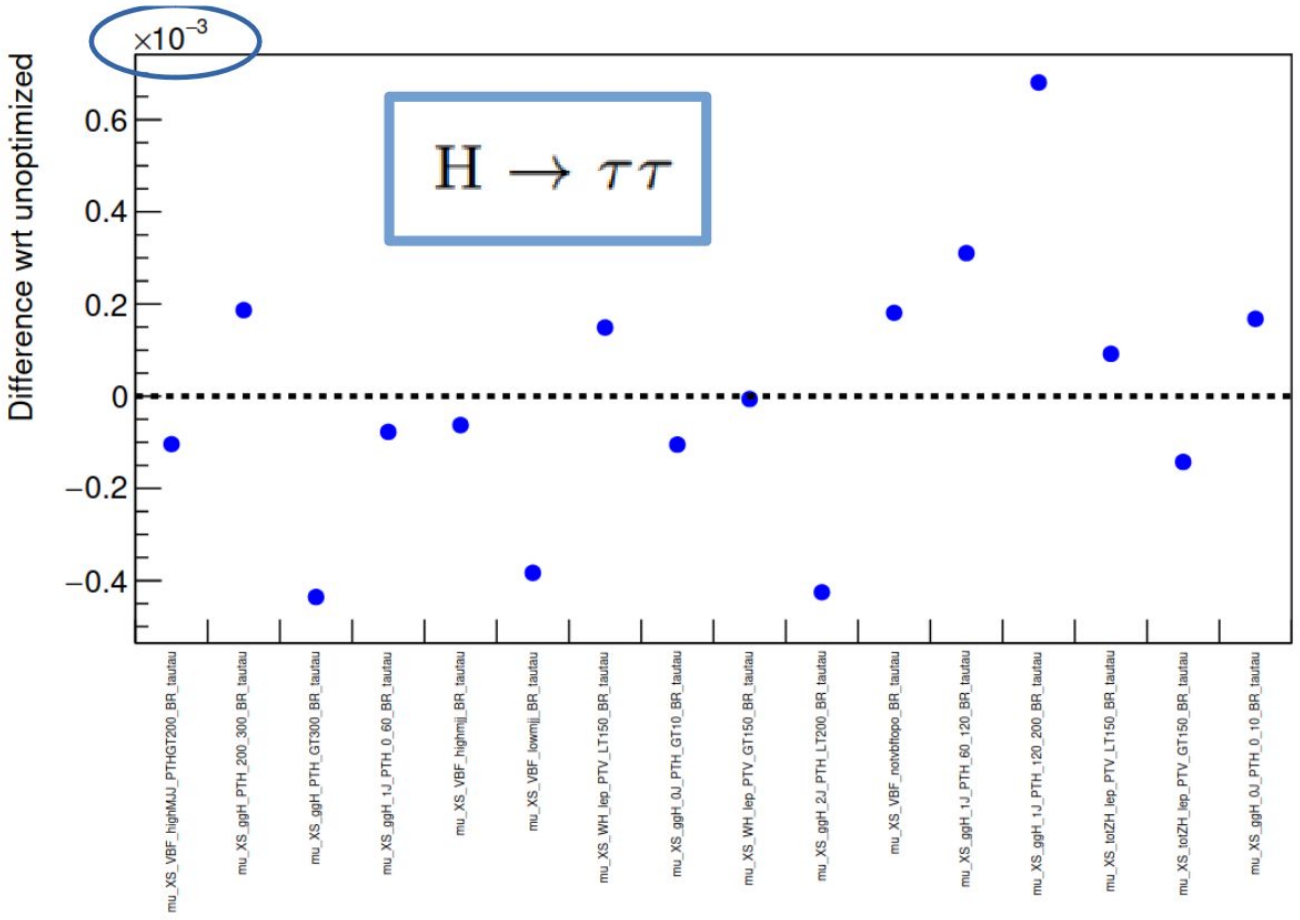
- Large speed-up possible compared to using normal numeric minimisation:





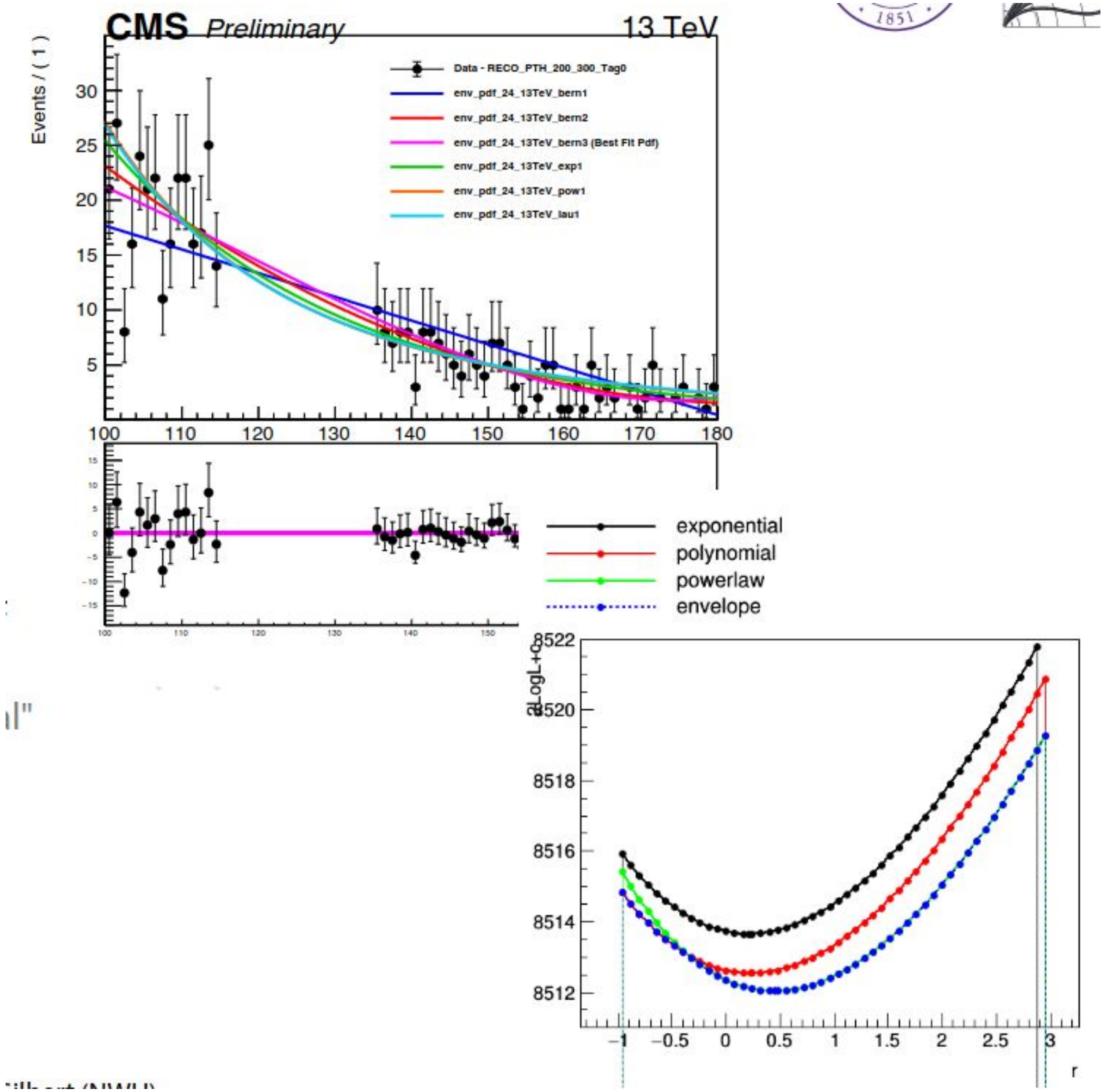
Additional features: CMSHistSum

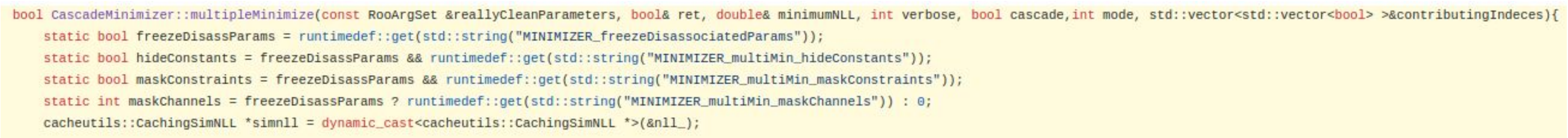
- Memory optimization for binned analyses (PR)
- Standard approach is to treat each process as separate object (histogram) in the workspace
- CMSHistSum: single object encapsulating all processes
    - Keeps track internally of contributions in each bin
    - Dramatically reduces number of separate RooFit objects that go into workspace
- Example: reduces H→$\tau\tau$ memory by 60%, fitting time by 40%
    - Differences in fit results are O(10^{-4}) (<< uncertainty intervals)



Additional features: discrete-profiling

- Discrete profiling widely used in CMS: arXiv:1408.6865
- Introduces discrete nuisance parameters that correspond to choice of pdf for a given process (RooMultiPdf)
- Allow discrete parameter to vary in maximum likelihood fit
    - ○ NLL penalty for additional DoFs
- Gives uncertainty due to uncertainty on choice of PDF functional form
    - ○ Alternative to spurious signal approach
- Minuit does not support fitting discrete parameters
    - ○ Handled directly by combine (CascadeMinimizer)





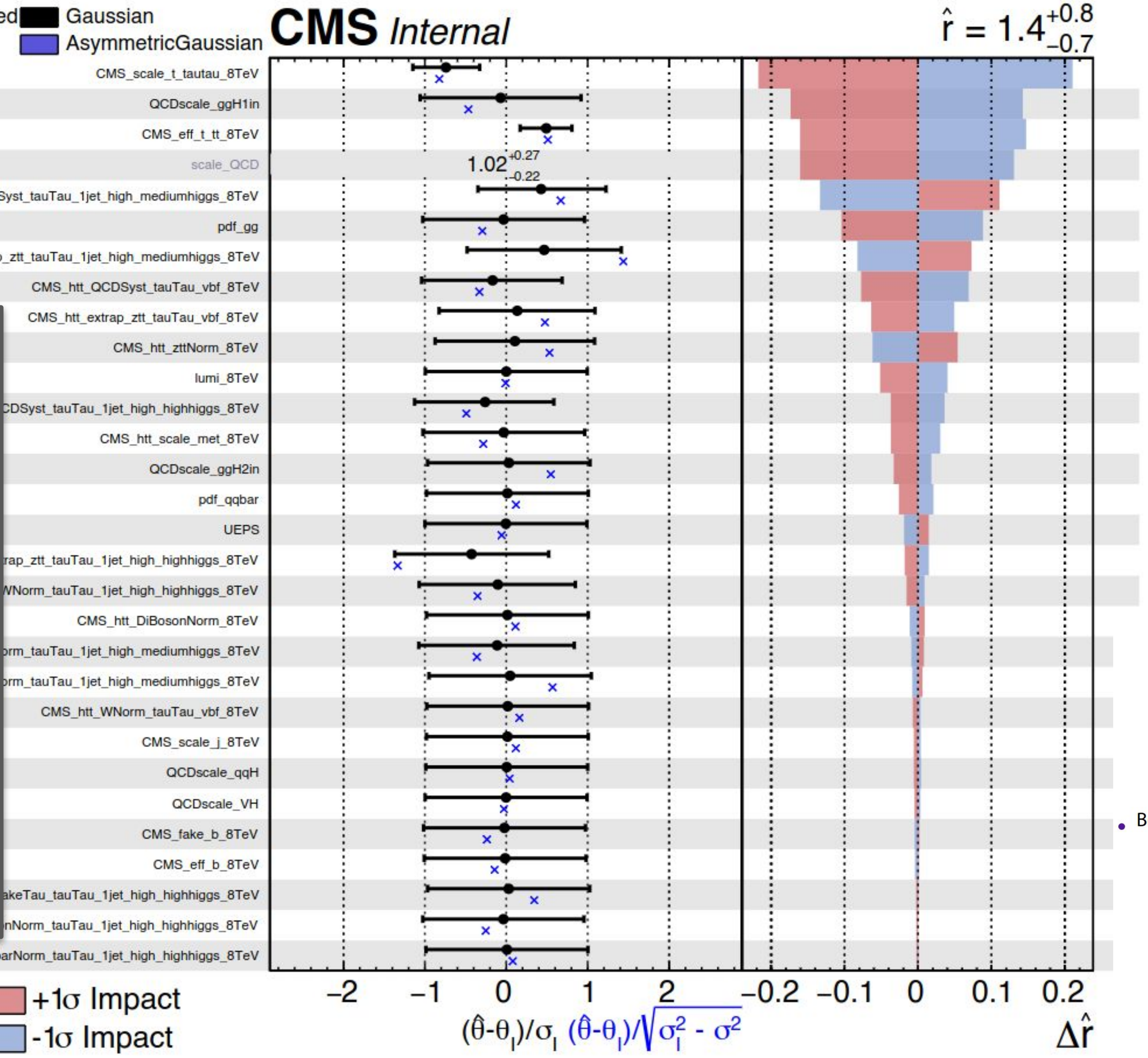
```
bool CascadeMinimizer::multipleMinimize(const RooArgSet &reallyCleanParameters, bool& ret, double& minimumNLL, int verbose, bool cascade,int mode, std::vector<std::vector<bool> >&contributingIndeces){
    static bool freezeDisassParams = runtimedef::get(std::string("MINIMIZER_freezeDisassociatedParams"));
    static bool hideConstants = freezeDisassParams && runtimedef::get(std::string("MINIMIZER_multiMin_hideConstants"));
    static bool maskConstraints = freezeDisassParams && runtimedef::get(std::string("MINIMIZER_multiMin_maskConstraints"));
    static int maskChannels = freezeDisassParams ? runtimedef::get(std::string("MINIMIZER_multiMin_maskChannels")) : 0;
    cacheutils::CachingSimNLL *simnll = dynamic_cast<cacheutils::CachingSimNLL *>(&nll_);
```

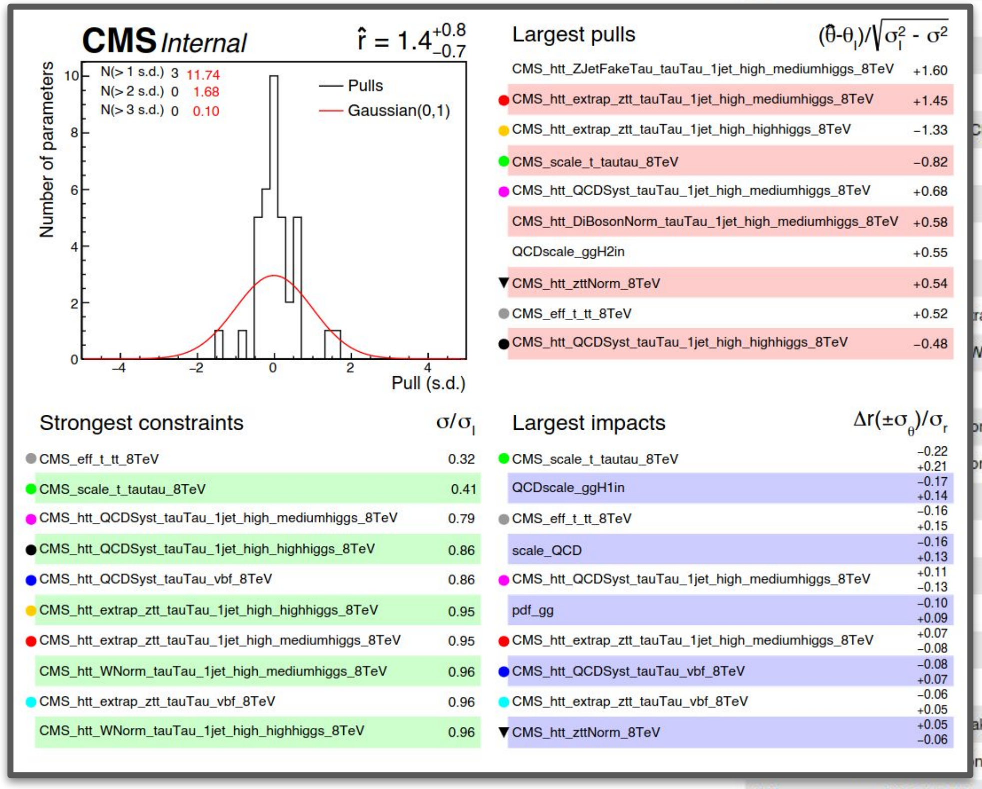
Additional features: impacts

- Combine automates the calculation of impacts for nuisance params

Define impact of NP on POI as the shift in the POI that is induced as the NP is fixed and brought to its +1 and -1 post-fit values

Also see parameter constraint relative to input uncertainty





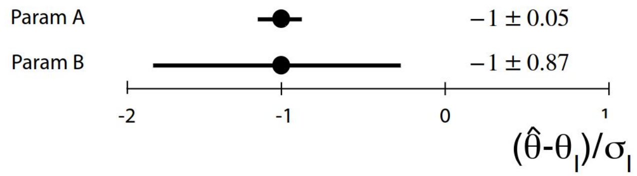
Largest pulls	$(\hat{\theta}-\theta_I)/\sqrt{\sigma_I^2 - \sigma^2}$
CMS_htt_ZJetFakeTau_tauTau_1jet_high_mediumhiggs_8TeV	+1.60
● CMS_htt_extrap_ztt_tauTau_1jet_high_mediumhiggs_8TeV	+1.45
● CMS_htt_extrap_ztt_tauTau_1jet_high_highhiggs_8TeV	−1.33
● CMS_scale_t_tautau_8TeV	−0.82
● CMS_htt_QCDSyst_tauTau_1jet_high_mediumhiggs_8TeV	+0.68
CMS_htt_DiBosonNorm_tauTau_1jet_high_mediumhiggs_8TeV	+0.58
QCDscale_ggH2in	+0.55
▼ CMS_htt_zttNorm_8TeV	+0.54
● CMS_eff_t_tt_8TeV	+0.52
● CMS_htt_QCDSyst_tauTau_1jet_high_highhiggs_8TeV	−0.48

Strongest constraints	σ/σ_I
● CMS_eff_t_tt_8TeV	0.32
● CMS_scale_t_tautau_8TeV	0.41
● CMS_htt_QCDSyst_tauTau_1jet_high_mediumhiggs_8TeV	0.79
● CMS_htt_QCDSyst_tauTau_1jet_high_highhiggs_8TeV	0.86
● CMS_htt_QCDSyst_tauTau_vbf_8TeV	0.86
● CMS_htt_extrap_ztt_tauTau_1jet_high_highhiggs_8TeV	0.95
● CMS_htt_extrap_ztt_tauTau_1jet_high_mediumhiggs_8TeV	0.95
CMS_htt_WNorm_tauTau_1jet_high_mediumhiggs_8TeV	0.96
● CMS_htt_extrap_ztt_tauTau_vbf_8TeV	0.96
CMS_htt_WNorm_tauTau_1jet_high_highhiggs_8TeV	0.96

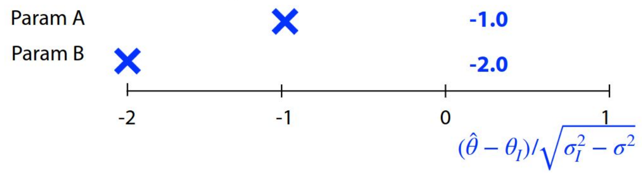
Largest impacts	$\Delta r(\pm\sigma_\theta)/\sigma_r$
● CMS_scale_t_tautau_8TeV	−0.22 +0.21
QCDscale_ggH1in	−0.17 +0.14
● CMS_eff_t_tt_8TeV	−0.16 +0.15
scale_QCD	−0.16 +0.13
● CMS_htt_QCDSyst_tauTau_1jet_high_mediumhiggs_8TeV	+0.11 −0.13
pdf_gg	−0.10 +0.09
● CMS_htt_extrap_ztt_tauTau_1jet_high_mediumhiggs_8TeV	+0.07 −0.08
● CMS_htt_QCDSyst_tauTau_vbf_8TeV	−0.08 +0.07
● CMS_htt_extrap_ztt_tauTau_vbf_8TeV	−0.06 +0.05
▼ CMS_htt_zttNorm_8TeV	+0.05 −0.06

New summary

+ Fitted parameter pulls

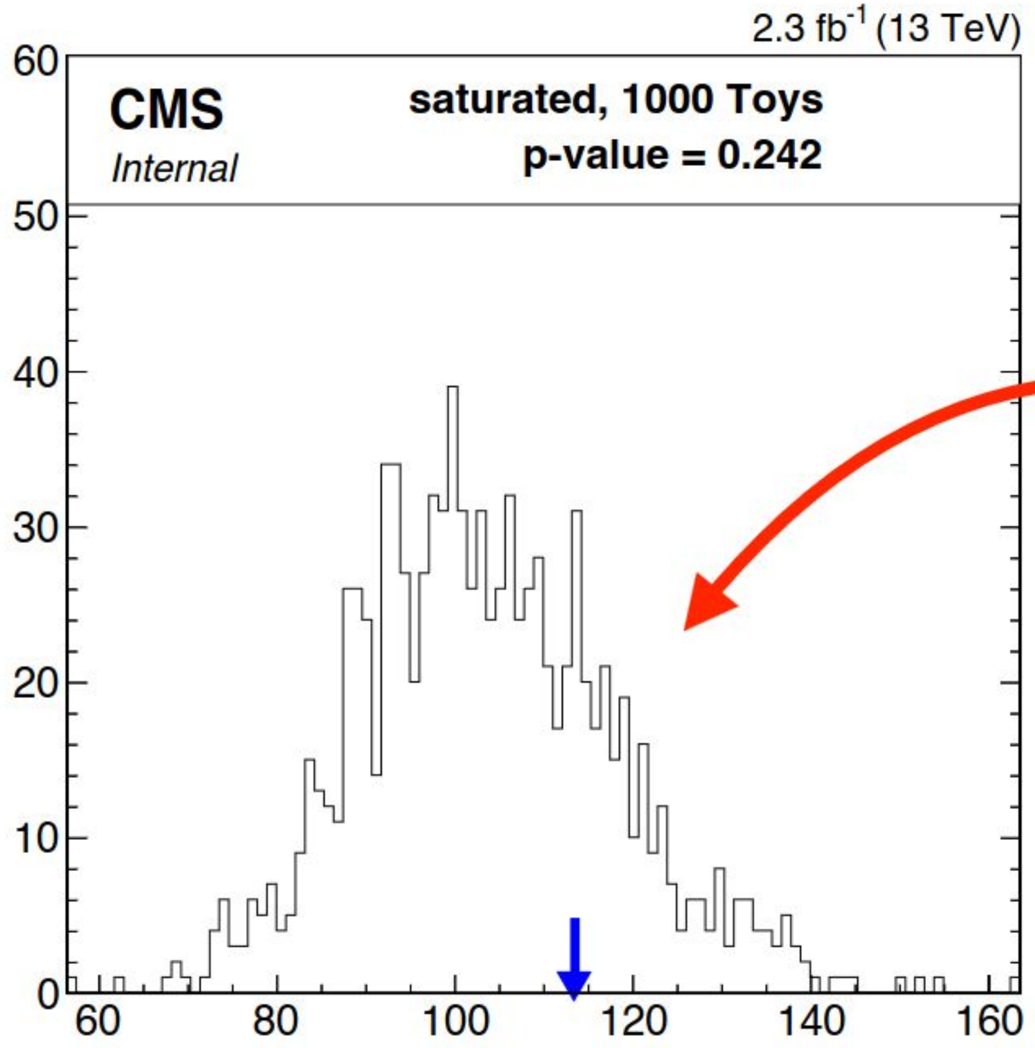


- Better definition from [XXX]: $(\hat{\theta} - \theta_I)/\sqrt{\sigma_I^2 - \sigma^2}$

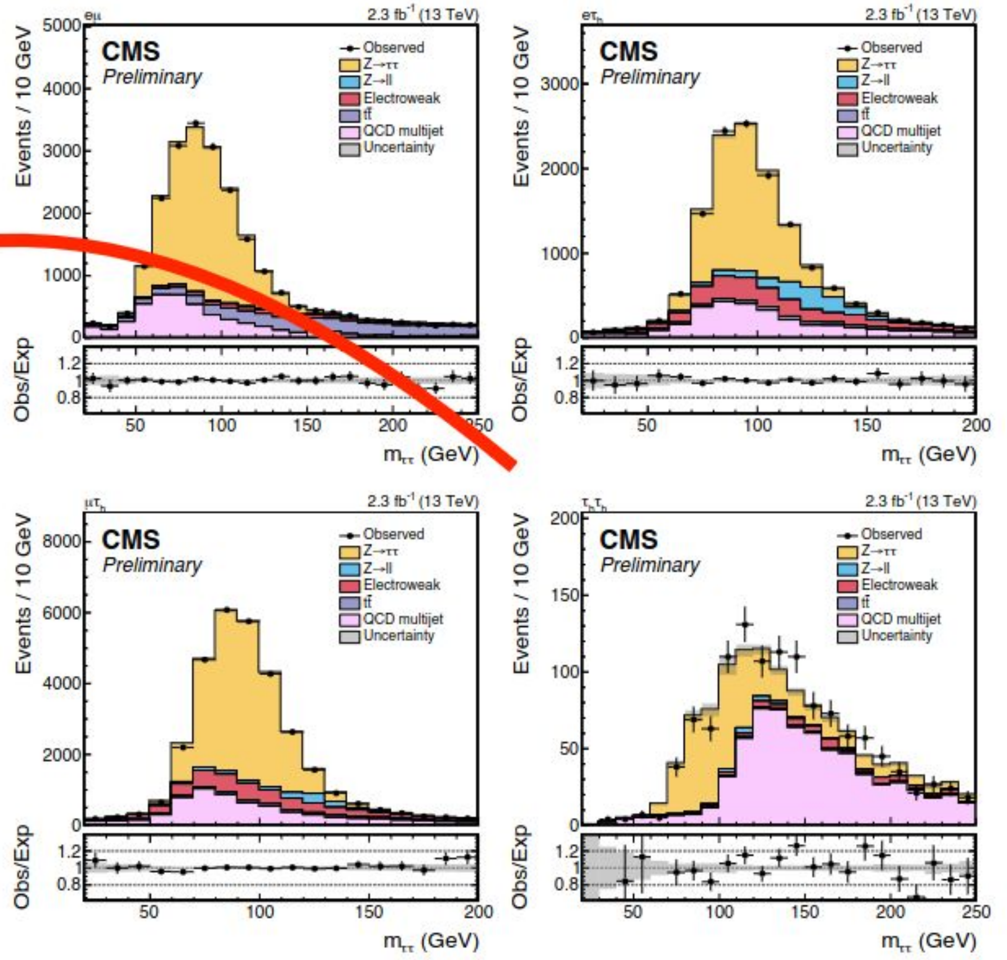


Additional features: goodness-of-fit

- Support for calculating saturated model, Kolmogorov-Smirnov and Anderson-Darling test statistics
  - Combine's toy generation routines used for building up expected distributions

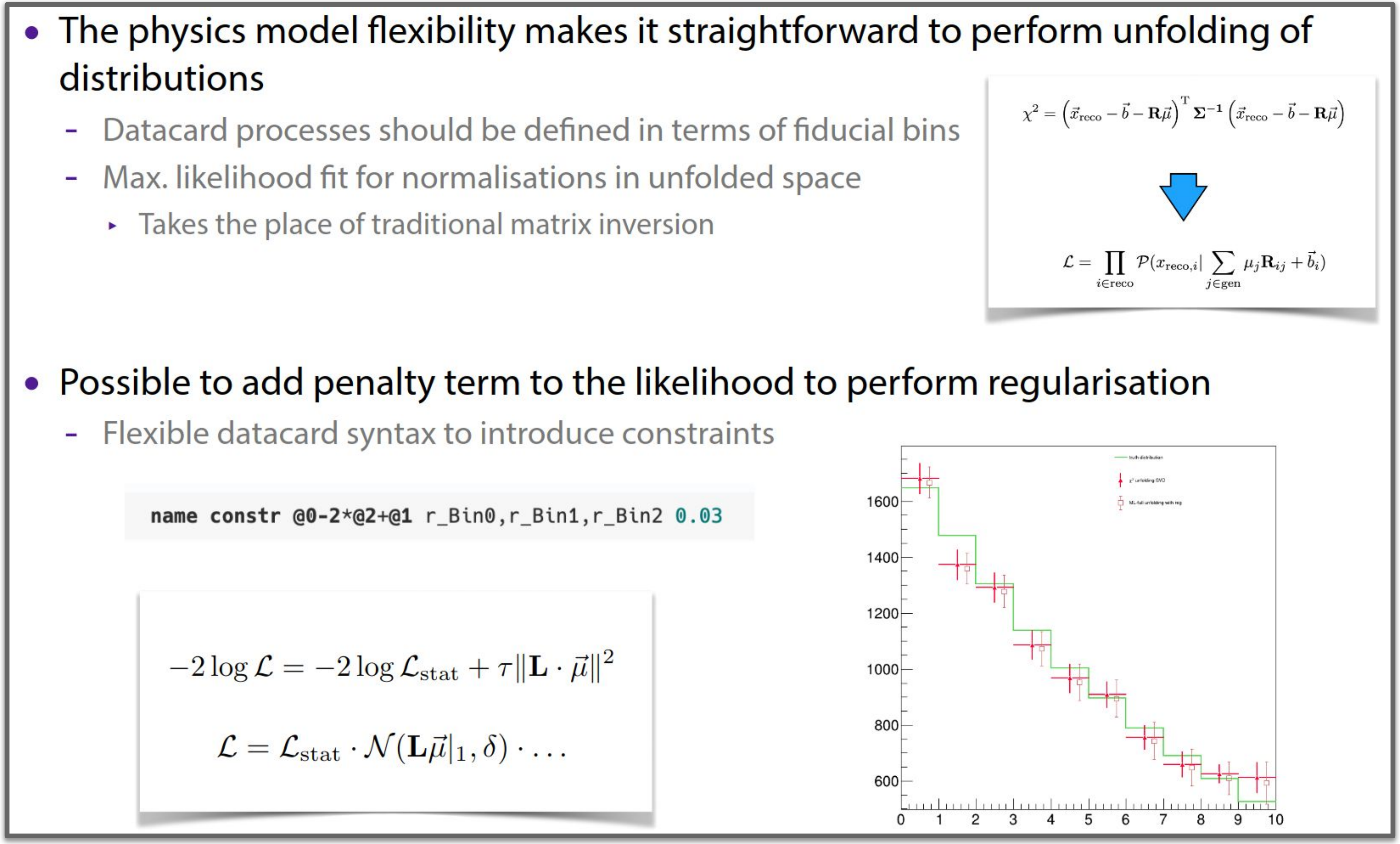


Saturated model: $-2\ln\lambda = 2\sum_i f_i - d_i + d_i \ln(d_i/f_i).$



Additional features: unfolding

Details, Tutorial



- The physics model flexibility makes it straightforward to perform unfolding of distributions
  - Datacard processes should be defined in terms of fiducial bins
  - Max. likelihood fit for normalisations in unfolded space
    - Takes the place of traditional matrix inversion

$$\chi^2 = \left(\vec{x}_{\text{reco}} - \vec{b} - \mathbf{R}\vec{\mu}\right)^{\mathrm{T}} \mathbf{\Sigma}^{-1} \left(\vec{x}_{\text{reco}} - \vec{b} - \mathbf{R}\vec{\mu}\right)$$

$$\mathcal{L} = \prod_{i \in \text{reco}} \mathcal{P}(x_{\text{reco},i} | \sum_{j \in \text{gen}} \mu_j \mathbf{R}_{ij} + \vec{b}_i)$$

- Possible to add penalty term to the likelihood to perform regularisation
  - Flexible datacard syntax to introduce constraints

```
name constr @0-2*@2+@1 r_Bin0,r_Bin1,r_Bin2 0.03
```

$$-2\log\mathcal{L} = -2\log\mathcal{L}_{\text{stat}} + \tau||\mathbf{L}\cdot\vec{\mu}||^2$$

$$\mathcal{L} = \mathcal{L}_{\text{stat}} \cdot \mathcal{N}(\mathbf{L}\vec{\mu}|_1, \delta) \cdot \ldots$$

- We would like to improve the interface/number of features here

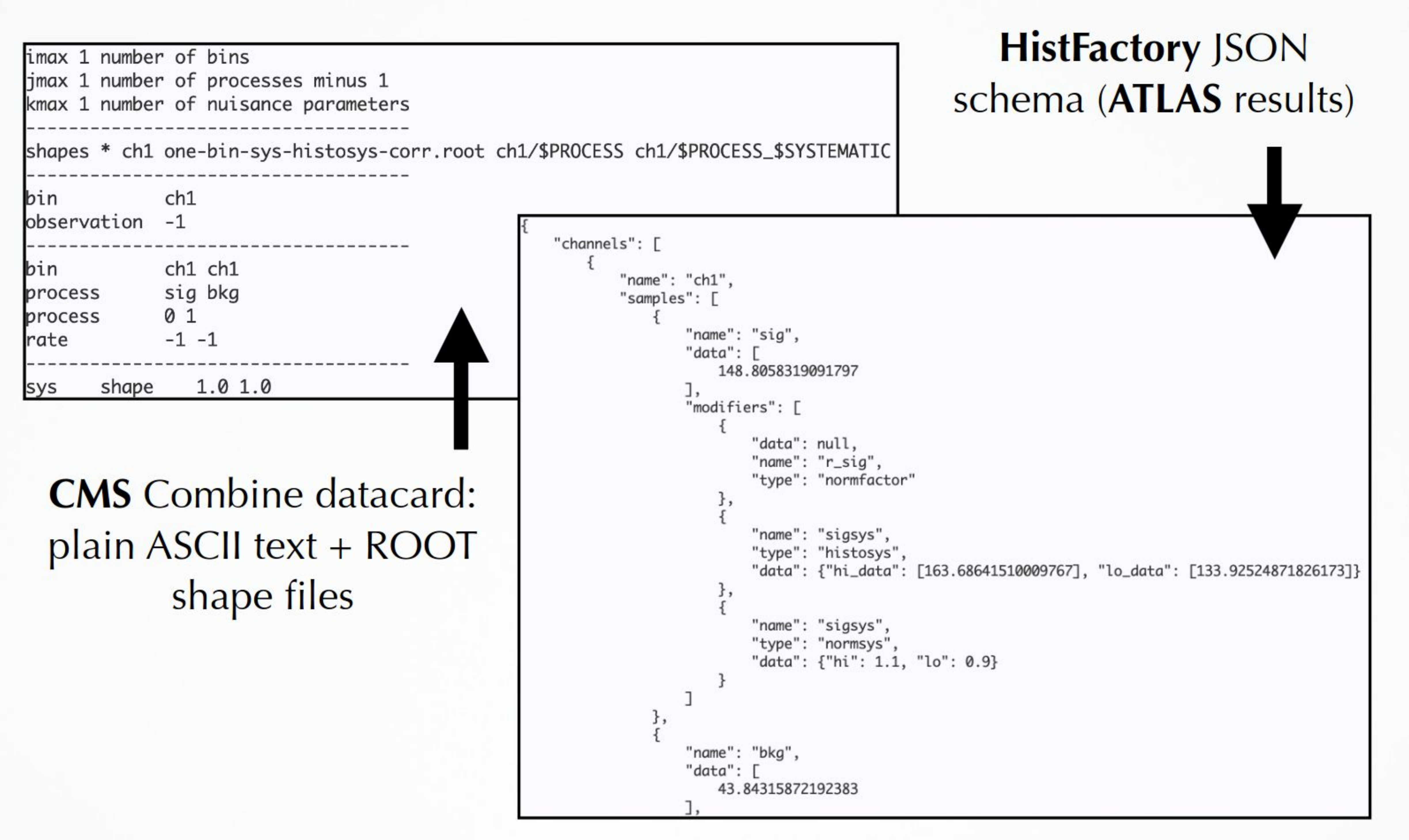
Additional features: under-the-hood improvements

- Number of other performance improvements (without going into detail)
- Standard RooFit on tutorial workspace is ~5x slower than combine MultiDimFit
- Similar to speed-ups showed yday with new RooFit features
- Should sit down and discuss pushing some of these things upstream

Benchmarking

Benchmarking combine

- Kirill has worked on comparing combine to pyhf (with input conversion tool combine2pyhf)



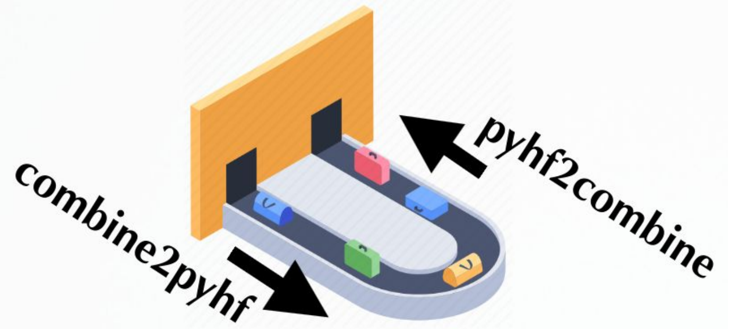
```
imax 1 number of bins
jmax 1 number of processes minus 1
kmax 1 number of nuisance parameters
------------------------------------
shapes * ch1 one-bin-sys-histosys-corr.root ch1/$PROCESS ch1/$PROCESS_$SYSTEMATIC
------------------------------------
bin          ch1
observation  -1
------------------------------------
bin          ch1 ch1
process      sig bkg
process      0 1
rate         -1 -1
------------------------------------
sys     shape     1.0 1.0
```

CMS Combine datacard: plain ASCII text + ROOT shape files

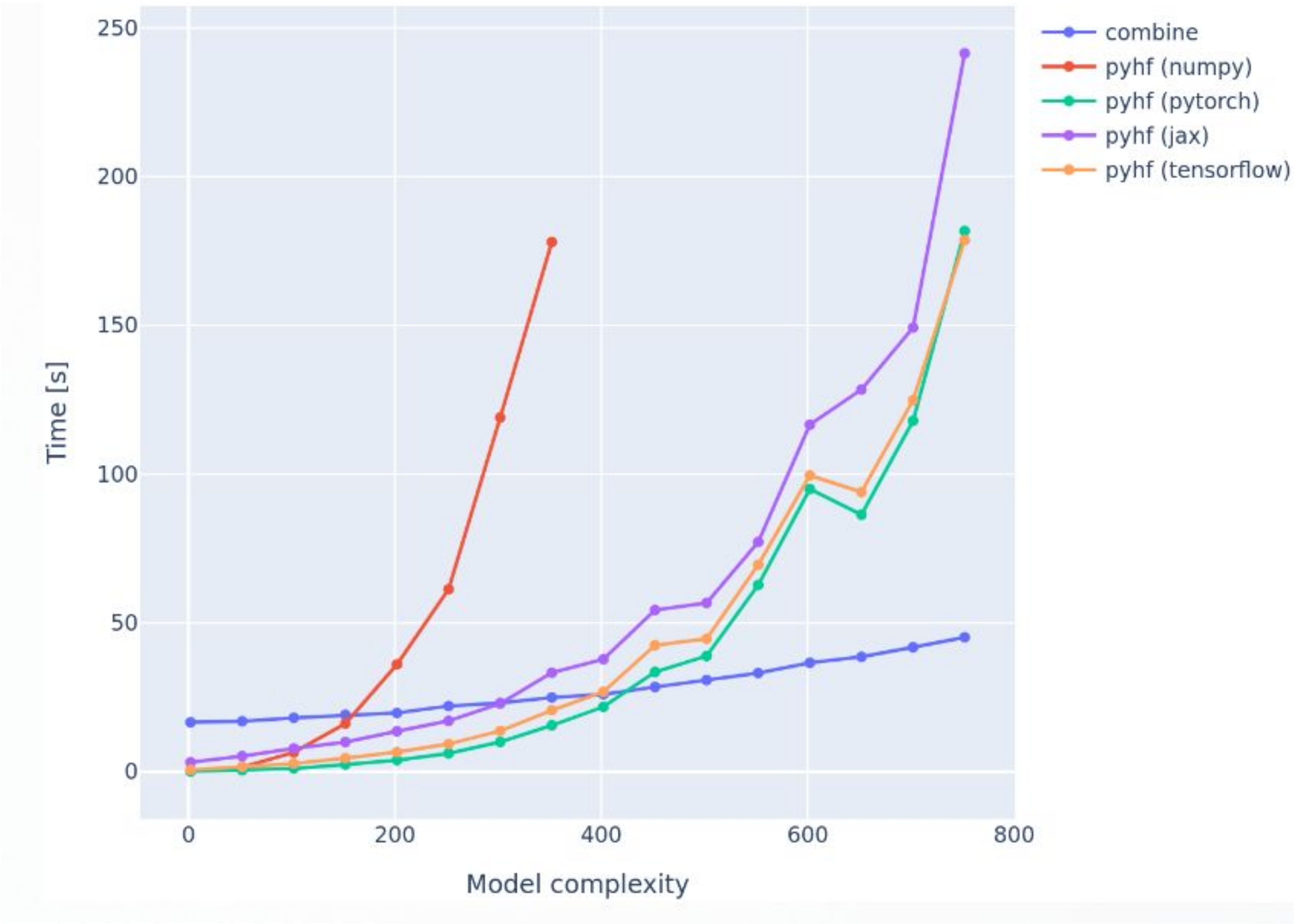
HistFactory JSON schema (**ATLAS** results)

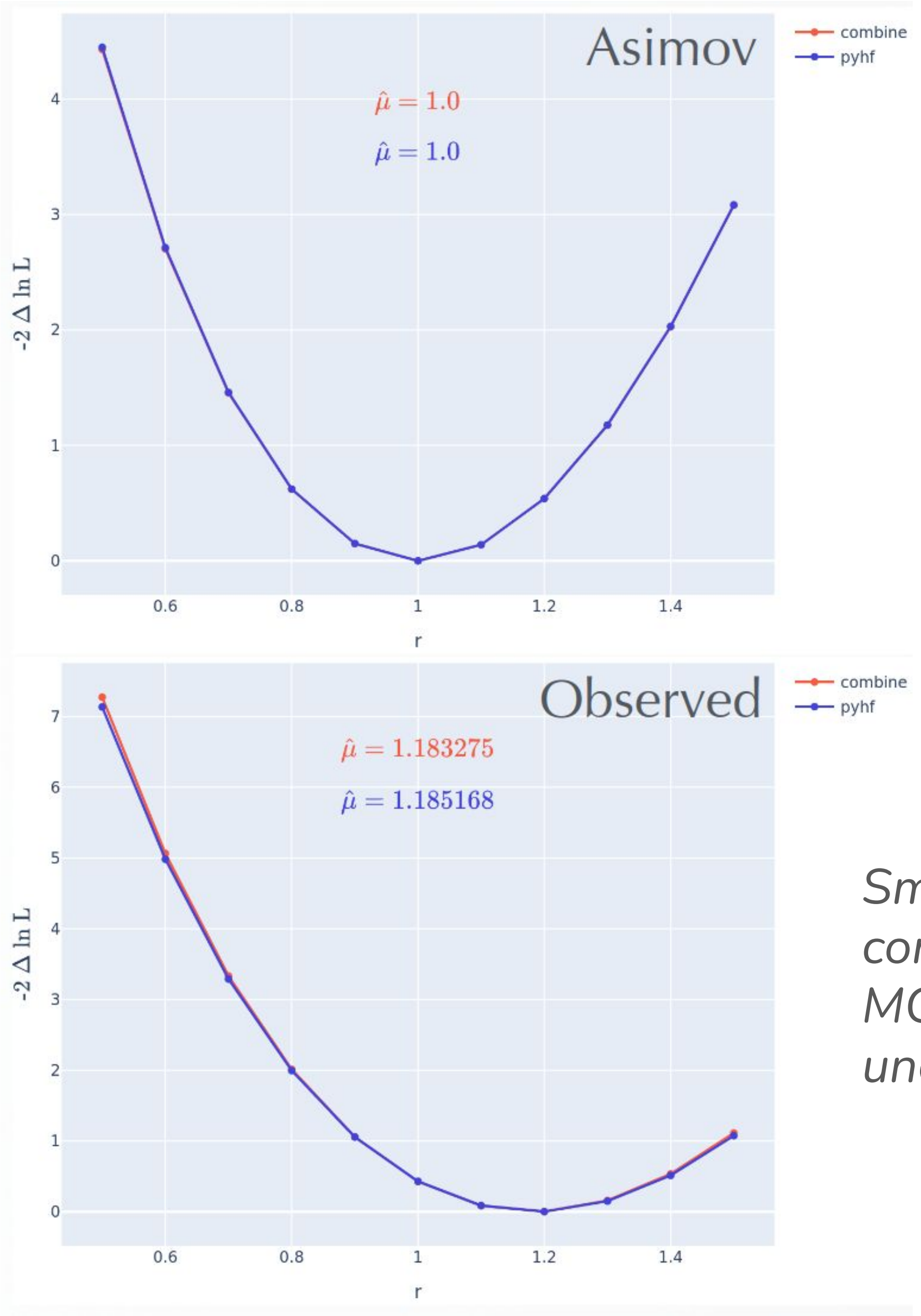
```
{
    "channels": [
        {
            "name": "ch1",
            "samples": [
                {
                    "name": "sig",
                    "data": [
                        148.8058319091797
                    ],
                    "modifiers": [
                        {
                            "data": null,
                            "name": "r_sig",
                            "type": "normfactor"
                        },
                        {
                            "name": "sigsys",
                            "type": "histosys",
                            "data": {"hi_data": [163.68641510009767], "lo_data": [133.92524871826173]}
                        },
                        {
                            "name": "sigsys",
                            "type": "normsys",
                            "data": {"hi": 1.1, "lo": 0.9}
                        }
                    ]
                },
                {
                    "name": "bkg",
                    "data": [
                        43.84315872192383
                    ],
```

Benchmarking combine



- Kirill has worked on comparing combine to pyhf (with input conversion tool combine2pyhf)
  - Successfully validated translation of inputs and physics results (likelihood scans, impacts etc)
  - Automated: would like to test more inputs!



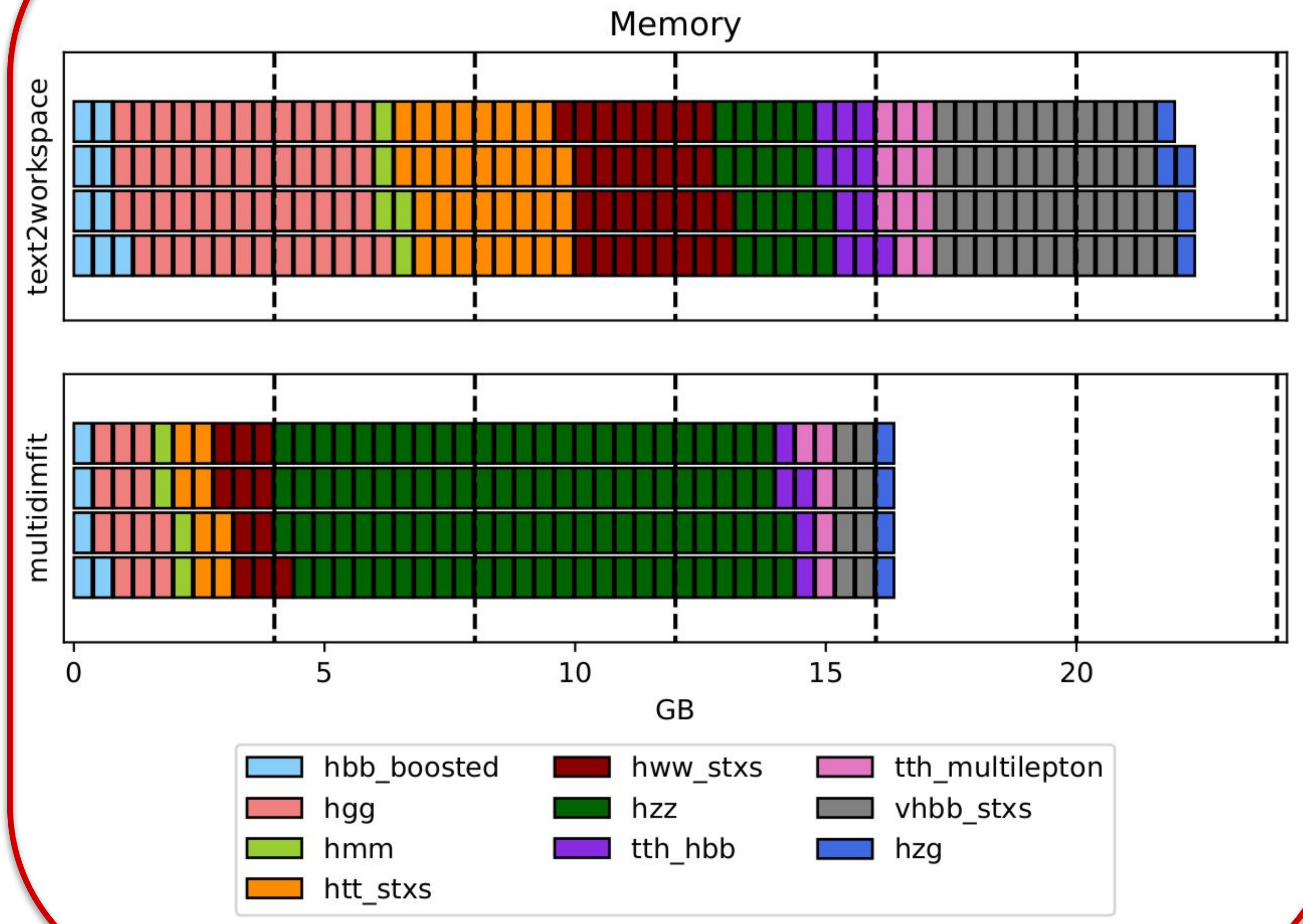


Small differences connected to treatment of MC statistical uncertainties

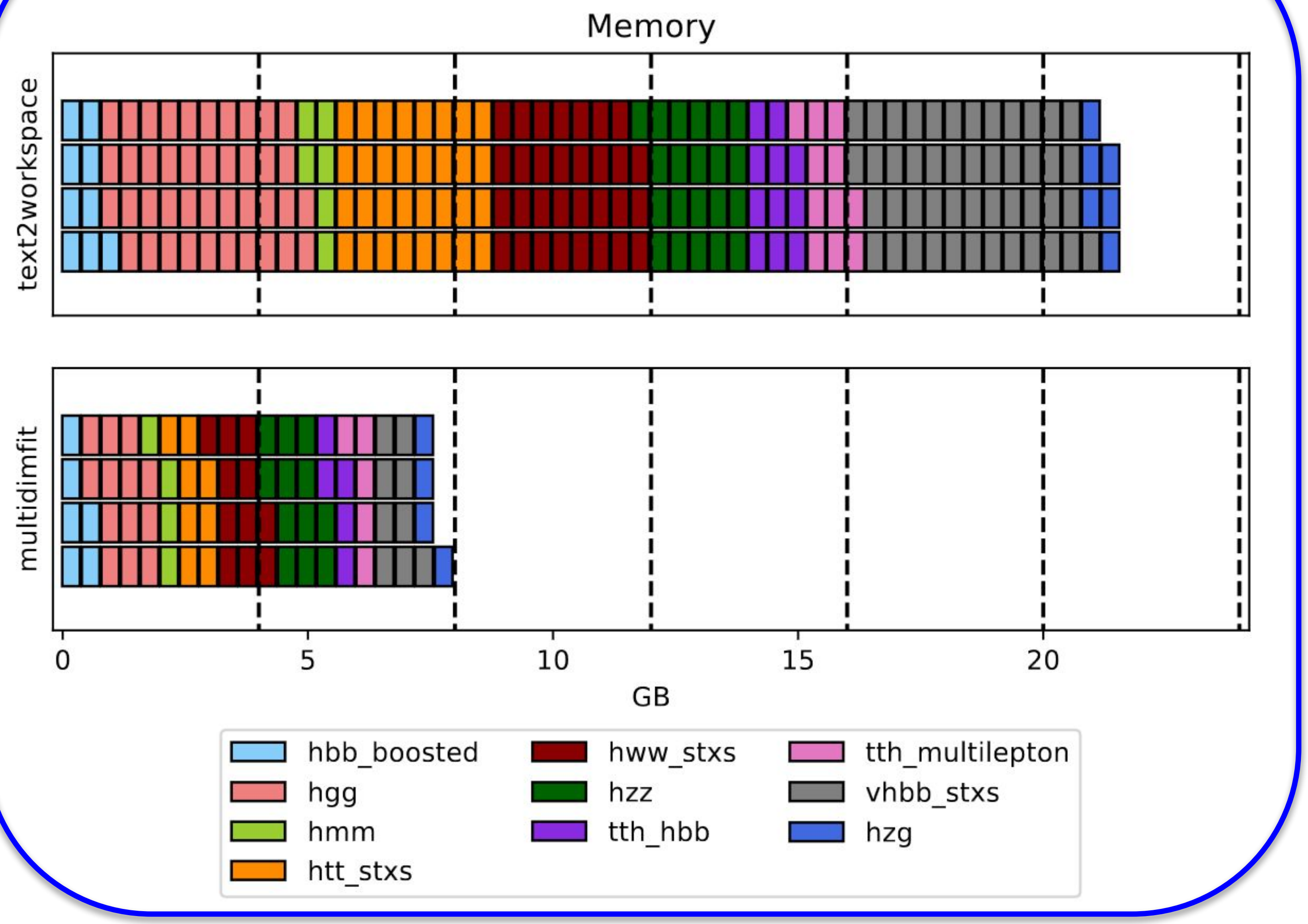
Updating RooFit (v6.22 → v6.26)

CMS Higgs combination

ROOT 6.22

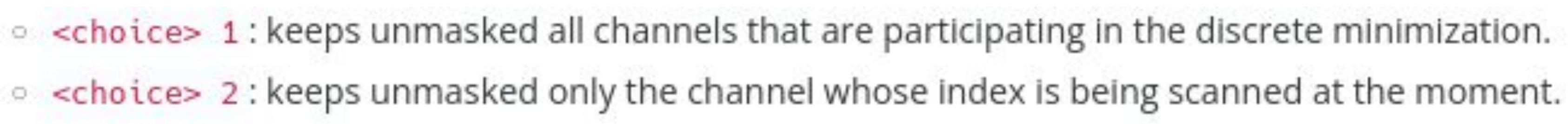


ROOT 6.26



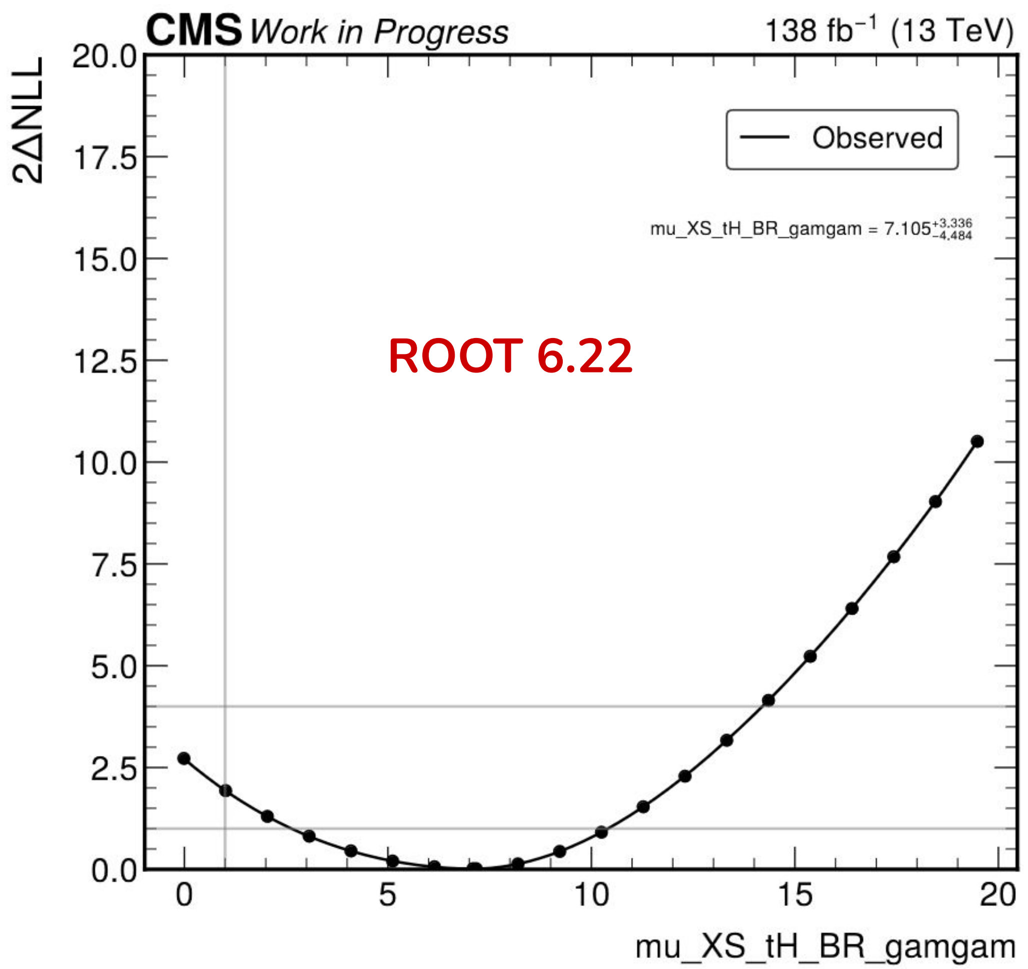
- Fixed major memory leak in H→ZZ (multidimfit), performance similar for other channels
- text2workspace is heavy but manageable

RooFit struggles

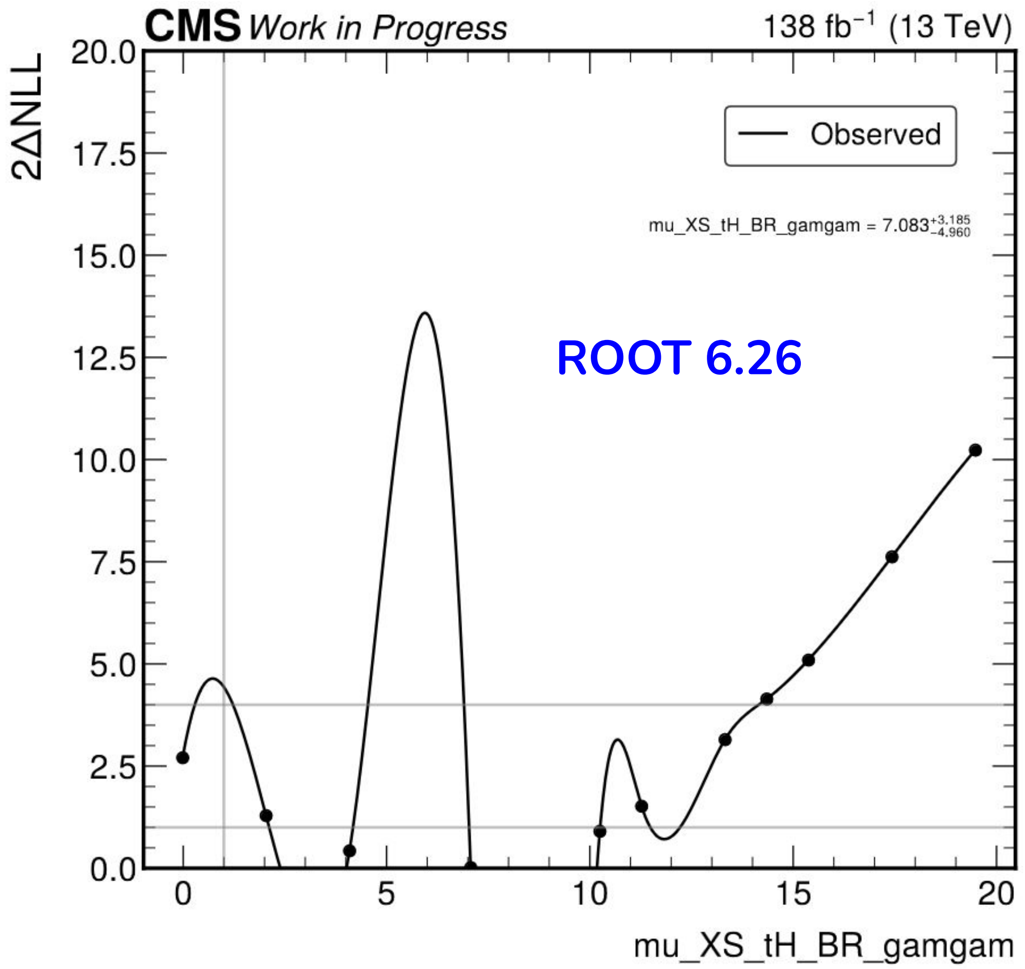


- `<choice> 1` : keeps unmasked all channels that are participating in the discrete minimization.
- `<choice> 2` : keeps unmasked only the channel whose index is being scanned at the moment.

- A few problems when moving from ROOT v6.22 → v6.26



Exact same inputs

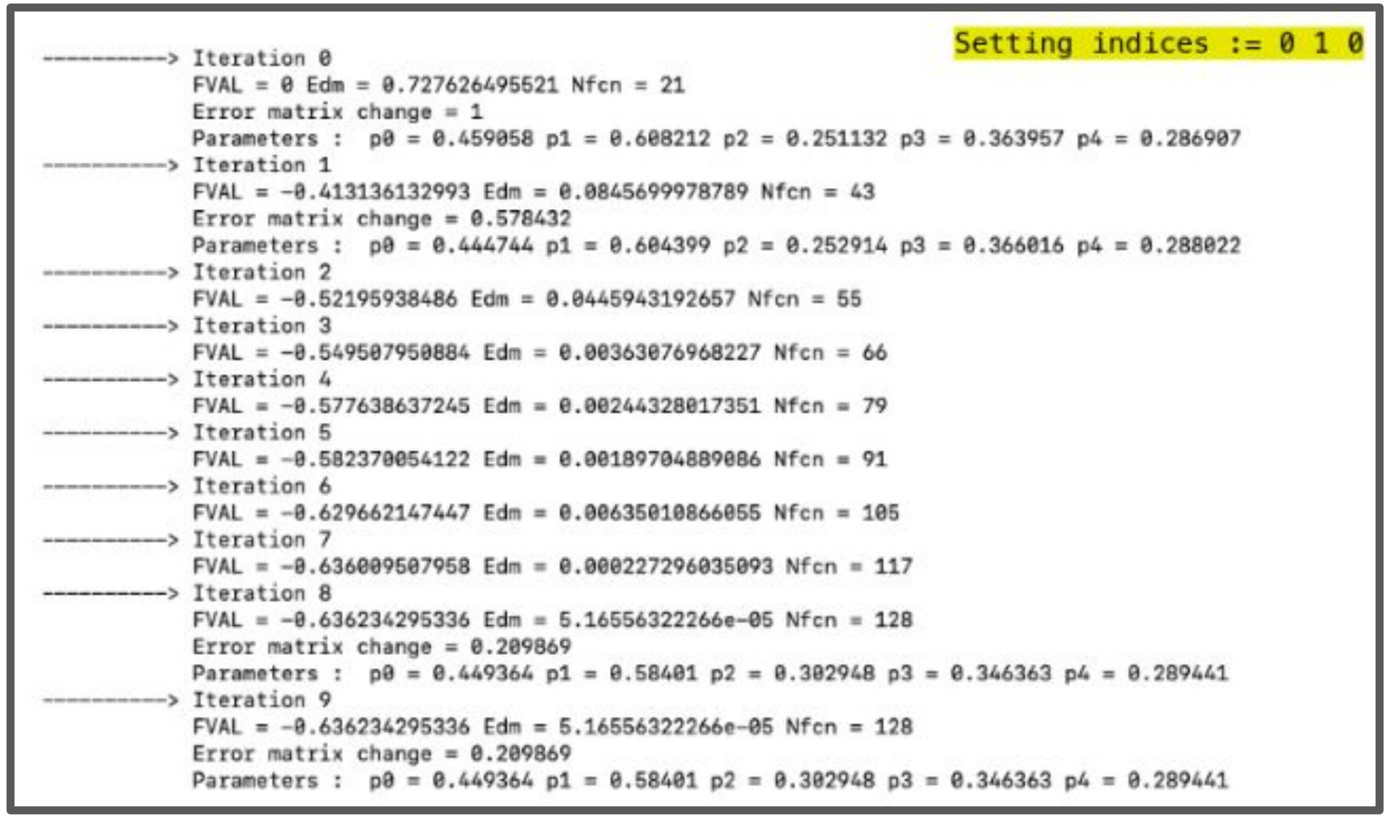


- Isolated issue to channels with discrete-profiling e.g. H→$\gamma\gamma$
- Option in combine: `--X-rtd MINIMIZER_multiMin_maskChannels=2` to mask channels which are not needed from NLL
- Compare logs…

RooFit struggles

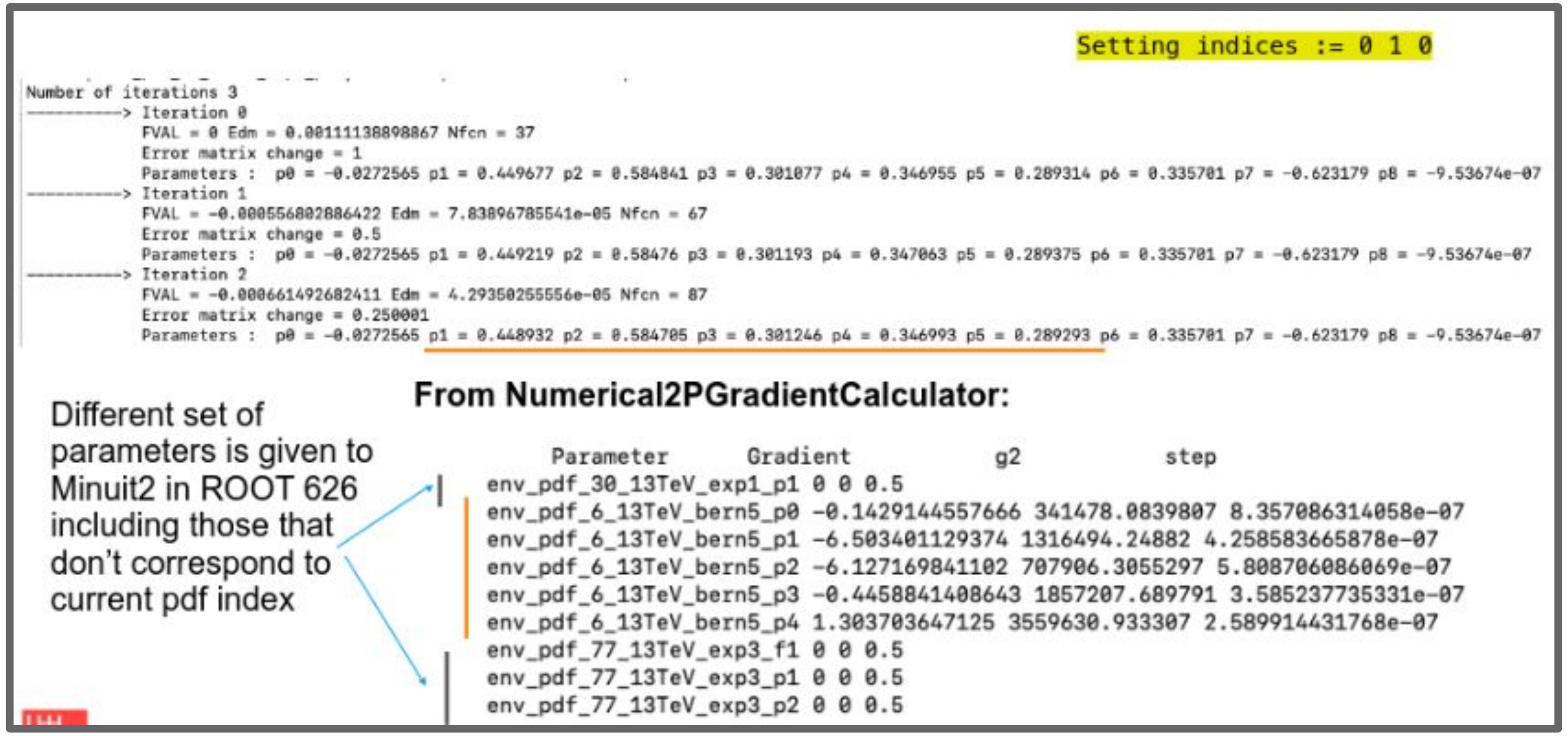
- Pdf iterations in discrete-profiling give different sets of parameters to minuit
  - ROOT 6.22: only gives parameters corresponding to updated index
  - ROOT 6.26: also parameters from other pdfs even if they are “masked” → Zero gradient problem!

ROOT 6.22



```
                                                              Setting indices := 0 1 0
-----------> Iteration 0
             FVAL = 0 Edm = 0.727626495521 Nfcn = 21
             Error matrix change = 1
             Parameters :  p0 = 0.459058 p1 = 0.608212 p2 = 0.251132 p3 = 0.363957 p4 = 0.286907
-----------> Iteration 1
             FVAL = -0.413136132993 Edm = 0.0845699978789 Nfcn = 43
             Error matrix change = 0.578432
             Parameters :  p0 = 0.444744 p1 = 0.604399 p2 = 0.252914 p3 = 0.366016 p4 = 0.288022
-----------> Iteration 2
             FVAL = -0.52195938486 Edm = 0.0445943192657 Nfcn = 55
-----------> Iteration 3
             FVAL = -0.549507950884 Edm = 0.00363076968227 Nfcn = 66
-----------> Iteration 4
             FVAL = -0.577638637245 Edm = 0.00244328017351 Nfcn = 79
-----------> Iteration 5
             FVAL = -0.582370054122 Edm = 0.00189704889086 Nfcn = 91
-----------> Iteration 6
             FVAL = -0.629662147447 Edm = 0.00635010866055 Nfcn = 105
-----------> Iteration 7
             FVAL = -0.636009507958 Edm = 0.000227296035093 Nfcn = 117
-----------> Iteration 8
             FVAL = -0.636234295336 Edm = 5.16556322266e-05 Nfcn = 128
             Error matrix change = 0.209869
             Parameters :  p0 = 0.449364 p1 = 0.58401 p2 = 0.302948 p3 = 0.346363 p4 = 0.289441
-----------> Iteration 9
             FVAL = -0.636234295336 Edm = 5.16556322266e-05 Nfcn = 128
             Error matrix change = 0.209869
             Parameters :  p0 = 0.449364 p1 = 0.58401 p2 = 0.302948 p3 = 0.346363 p4 = 0.289441
```

ROOT 6.26



```
                                                              Setting indices := 0 1 0
Number of iterations 3
-----------> Iteration 0
             FVAL = 0 Edm = 0.00111138898867 Nfcn = 37
             Error matrix change = 1
             Parameters :  p0 = -0.0272565 p1 = 0.449677 p2 = 0.584841 p3 = 0.301077 p4 = 0.346955 p5 = 0.289314 p6 = 0.335701 p7 = -0.623179 p8 = -9.53674e-07
-----------> Iteration 1
             FVAL = -0.000556802886422 Edm = 7.83896785541e-05 Nfcn = 67
             Error matrix change = 0.5
             Parameters :  p0 = -0.0272565 p1 = 0.449219 p2 = 0.58476 p3 = 0.301193 p4 = 0.347063 p5 = 0.289375 p6 = 0.335701 p7 = -0.623179 p8 = -9.53674e-07
-----------> Iteration 2
             FVAL = -0.000661492682411 Edm = 4.29350255556e-05 Nfcn = 87
             Error matrix change = 0.250001
             Parameters :  p0 = -0.0272565 p1 = 0.448932 p2 = 0.584705 p3 = 0.301246 p4 = 0.346993 p5 = 0.289293 p6 = 0.335701 p7 = -0.623179 p8 = -9.53674e-07
```

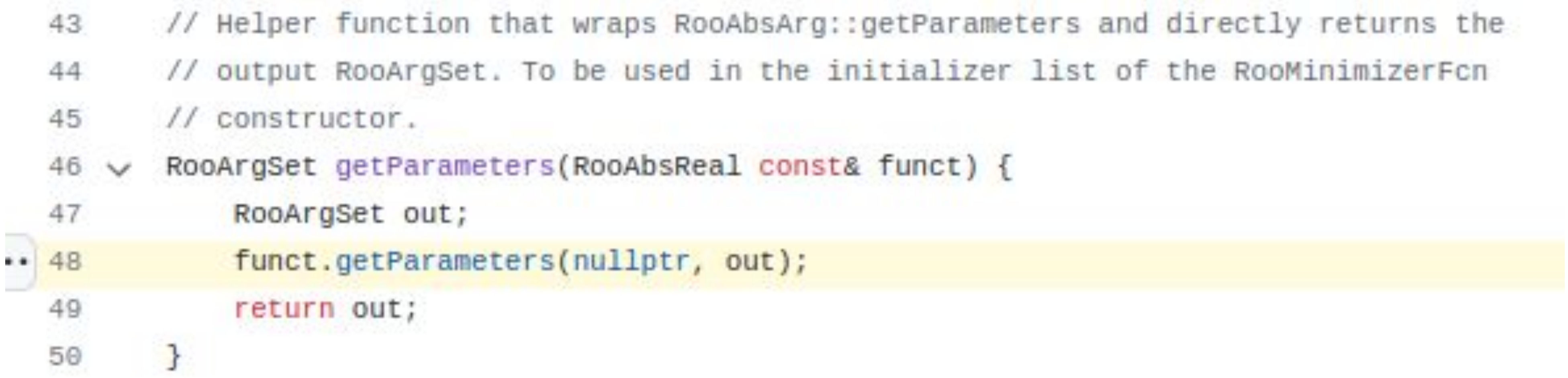
From Numerical2PGradientCalculator:

```
         Parameter        Gradient              g2              step
env_pdf_30_13TeV_exp1_p1 0 0 0.5
env_pdf_6_13TeV_bern5_p0 -0.1429144557666 341478.0839807 8.357086314058e-07
env_pdf_6_13TeV_bern5_p1 -6.503401129374 1316494.24882 4.258583665878e-07
env_pdf_6_13TeV_bern5_p2 -6.127169841102 707906.3055297 5.808706086069e-07
env_pdf_6_13TeV_bern5_p3 -0.4458841408643 1857207.689791 3.585237735331e-07
env_pdf_6_13TeV_bern5_p4 1.303703647125 3559630.933307 2.589914431768e-07
env_pdf_77_13TeV_exp3_f1 0 0 0.5
env_pdf_77_13TeV_exp3_p1 0 0 0.5
env_pdf_77_13TeV_exp3_p2 0 0 0.5
```

Different set of parameters is given to Minuit2 in ROOT 626 including those that don’t correspond to current pdf index

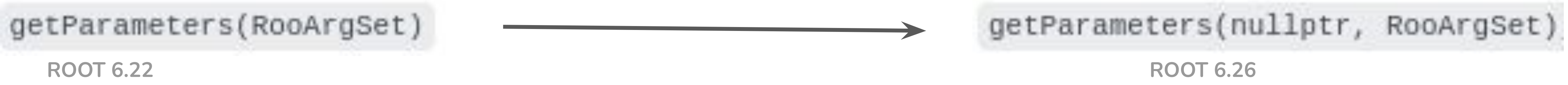
RooFit struggles

- We found the culprit: added method in ROOT 6.26 which overrides getParameters call



```
// Helper function that wraps RooAbsArg::getParameters and directly returns the
// output RooArgSet. To be used in the initializer list of the RooMinimizerFcn
// constructor.
RooArgSet getParameters(RooAbsReal const& funct) {
    RooArgSet out;
    funct.getParameters(nullptr, out);
    return out;
}
```

- No longer calling the method from CachingSimNLL in combine because arguments have changed



`getParameters(RooArgSet)` → `getParameters(nullptr, RooArgSet)`

ROOT 6.22

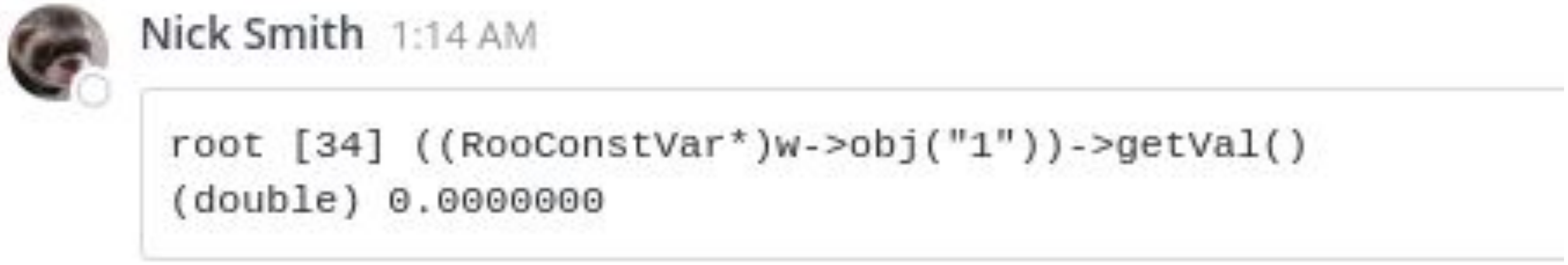
ROOT 6.26

- Was no longer applying the dedicated channel masking (defined in combine)

- Fix: update CachingSimNLL:getParameters with arguments list to match call in RooMinimizerFcn (PR)

- Lesson: upstream changes can lead to unexpected/hidden behaviour, can we prevent this in future?

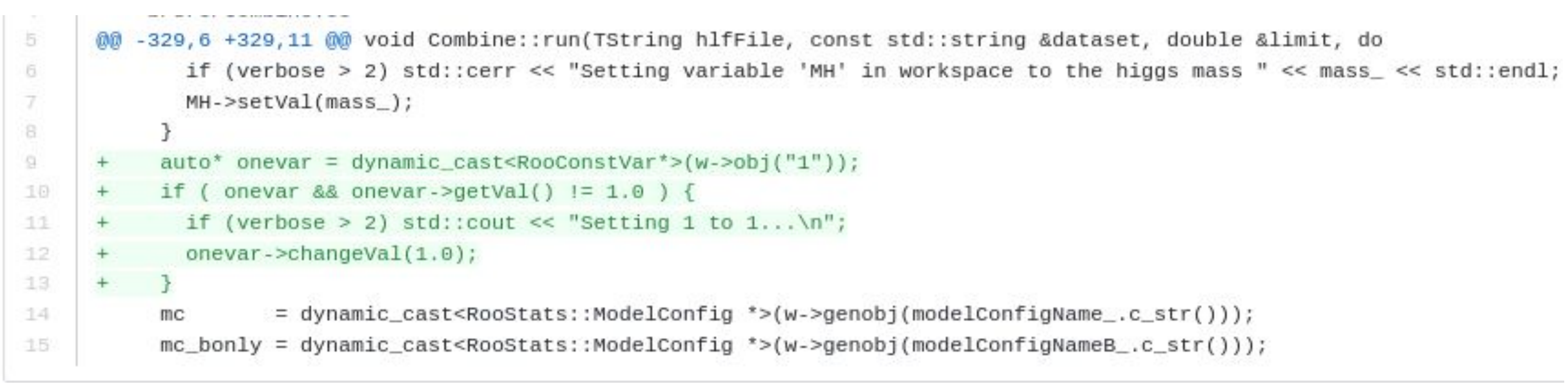
RooConstVar: “1” = 0



Nick Smith 1:14 AM

```
root [34] ((RooConstVar*)w->obj("1"))->getVal()
(double) 0.0000000
```

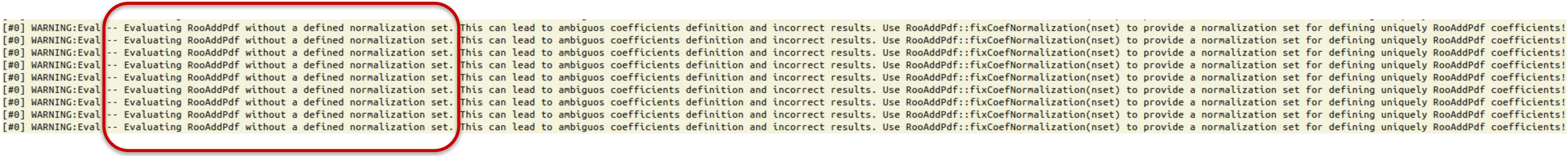
- RooConstVar are NOT backwards compatible (also saw problem in ATLAS workspaces with Gaussian constraint parameters)
    - Lead to zero gradients → No gradient = never adjusted in the whole fit
    - Issue present in ROOT 6.22 but not apparent in scans due to changes in PDF masking behaviour (previous slides)
    - Arises as RooConstVar’s used in H→$\gamma\gamma$ inputs were produced with ROOT v6.12
- Fix: swap all RooConstVars to RooRealVar (and set to const)
    - For now we added a patch to CH



```
@@ -329,6 +329,11 @@ void Combine::run(TString hlfFile, const std::string &dataset, double &limit, do
         if (verbose > 2) std::cerr << "Setting variable 'MH' in workspace to the higgs mass " << mass_ << std::endl;
         MH->setVal(mass_);
       }
+      auto* onevar = dynamic_cast<RooConstVar*>(w->obj("1"));
+      if ( onevar && onevar->getVal() != 1.0 ) {
+        if (verbose > 2) std::cout << "Setting 1 to 1...\n";
+        onevar->changeVal(1.0);
+      }
       mc       = dynamic_cast<RooStats::ModelConfig *>(w->genobj(modelConfigName_.c_str()));
       mc_bonly = dynamic_cast<RooStats::ModelConfig *>(w->genobj(modelConfigNameB_.c_str()));
```

Warning message in H→ZZ

- Switch to ROOT 6.26 lead to many (100k+) WARNINGS for H→ZZ inputs (unbinned)
  - Causes log files to be O(few Gb), issue when fetching outputs with HTCondor
  - For now we are sweeping issue under carpet by piping output to null
  - Prefer real fix!



```
[#0] WARNING:Eval -- Evaluating RooAddPdf without a defined normalization set. This can lead to ambiguos coefficients definition and incorrect results. Use RooAddPdf::fixCoefNormalization(nset) to provide a normalization set for defining uniquely RooAddPdf coefficients!
[#0] WARNING:Eval -- Evaluating RooAddPdf without a defined normalization set. This can lead to ambiguos coefficients definition and incorrect results. Use RooAddPdf::fixCoefNormalization(nset) to provide a normalization set for defining uniquely RooAddPdf coefficients!
[#0] WARNING:Eval -- Evaluating RooAddPdf without a defined normalization set. This can lead to ambiguos coefficients definition and incorrect results. Use RooAddPdf::fixCoefNormalization(nset) to provide a normalization set for defining uniquely RooAddPdf coefficients!
[#0] WARNING:Eval -- Evaluating RooAddPdf without a defined normalization set. This can lead to ambiguos coefficients definition and incorrect results. Use RooAddPdf::fixCoefNormalization(nset) to provide a normalization set for defining uniquely RooAddPdf coefficients!
[#0] WARNING:Eval -- Evaluating RooAddPdf without a defined normalization set. This can lead to ambiguos coefficients definition and incorrect results. Use RooAddPdf::fixCoefNormalization(nset) to provide a normalization set for defining uniquely RooAddPdf coefficients!
[#0] WARNING:Eval -- Evaluating RooAddPdf without a defined normalization set. This can lead to ambiguos coefficients definition and incorrect results. Use RooAddPdf::fixCoefNormalization(nset) to provide a normalization set for defining uniquely RooAddPdf coefficients!
[#0] WARNING:Eval -- Evaluating RooAddPdf without a defined normalization set. This can lead to ambiguos coefficients definition and incorrect results. Use RooAddPdf::fixCoefNormalization(nset) to provide a normalization set for defining uniquely RooAddPdf coefficients!
[#0] WARNING:Eval -- Evaluating RooAddPdf without a defined normalization set. This can lead to ambiguos coefficients definition and incorrect results. Use RooAddPdf::fixCoefNormalization(nset) to provide a normalization set for defining uniquely RooAddPdf coefficients!
[#0] WARNING:Eval -- Evaluating RooAddPdf without a defined normalization set. This can lead to ambiguos coefficients definition and incorrect results. Use RooAddPdf::fixCoefNormalization(nset) to provide a normalization set for defining uniquely RooAddPdf coefficients!
```

Future plans

- Provide v10 with ROOT v6.30 (fall-back to v6.26 if proving difficult)
- Additional standalone compilation within StatAnalysis package (cmake)
- Merge CombineHarvester functionalities into main combine tool
- API
- Alternative to input txt datacards e.g. HS3
- General optimisation of code
- Migration of additional classes/features into RooFit e.g. RooMultiPdf
- Automatic differentiation
- Vectorisation/GPU

As with everything else... slow progress due to lack of personpower!