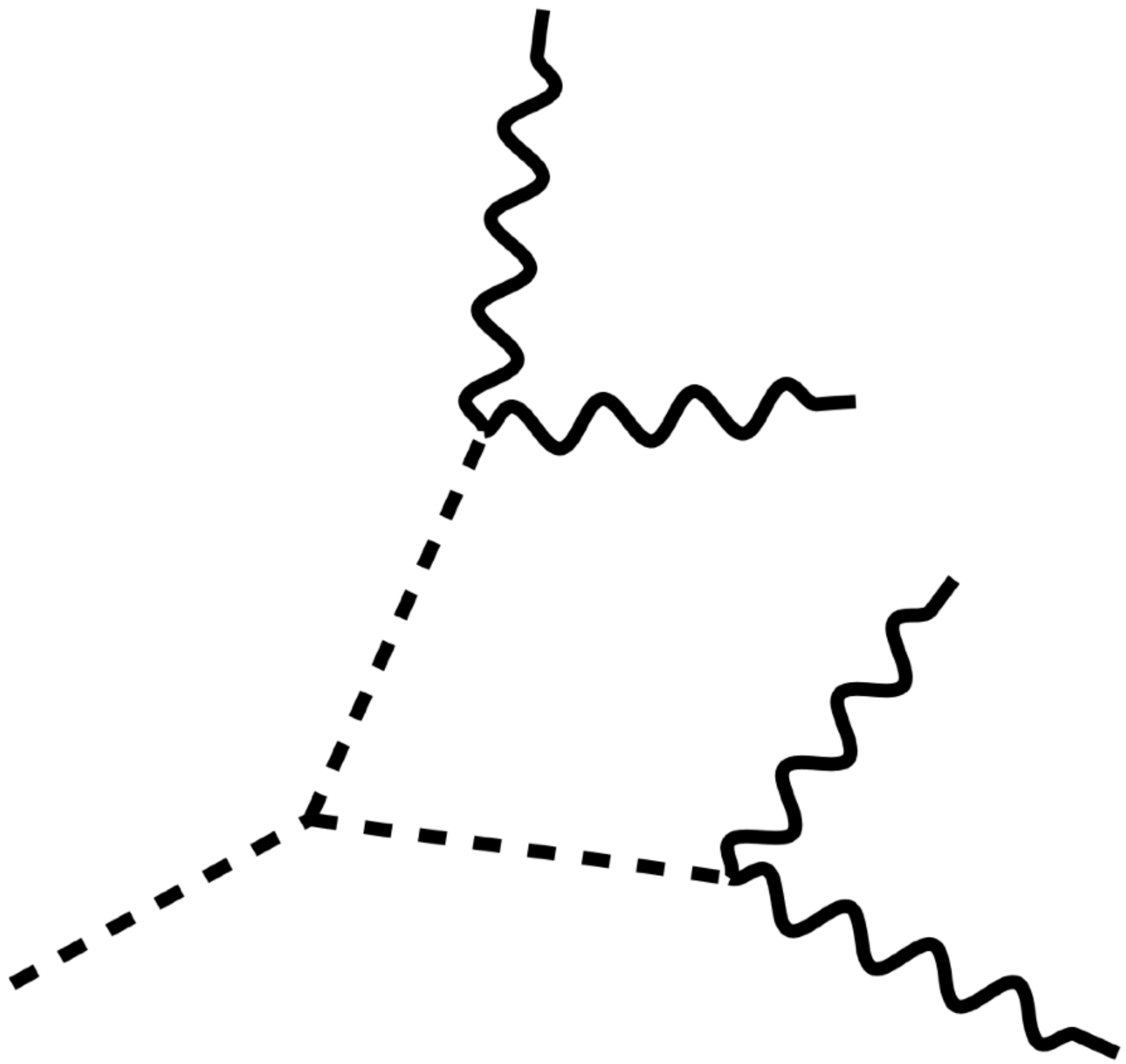


$h(125)\rightarrow aa\rightarrow \gamma\gamma\gamma\gamma$

TOYOKO ORIMOTO
ANDREA MASSIRONI
TANVI WAMORKAR

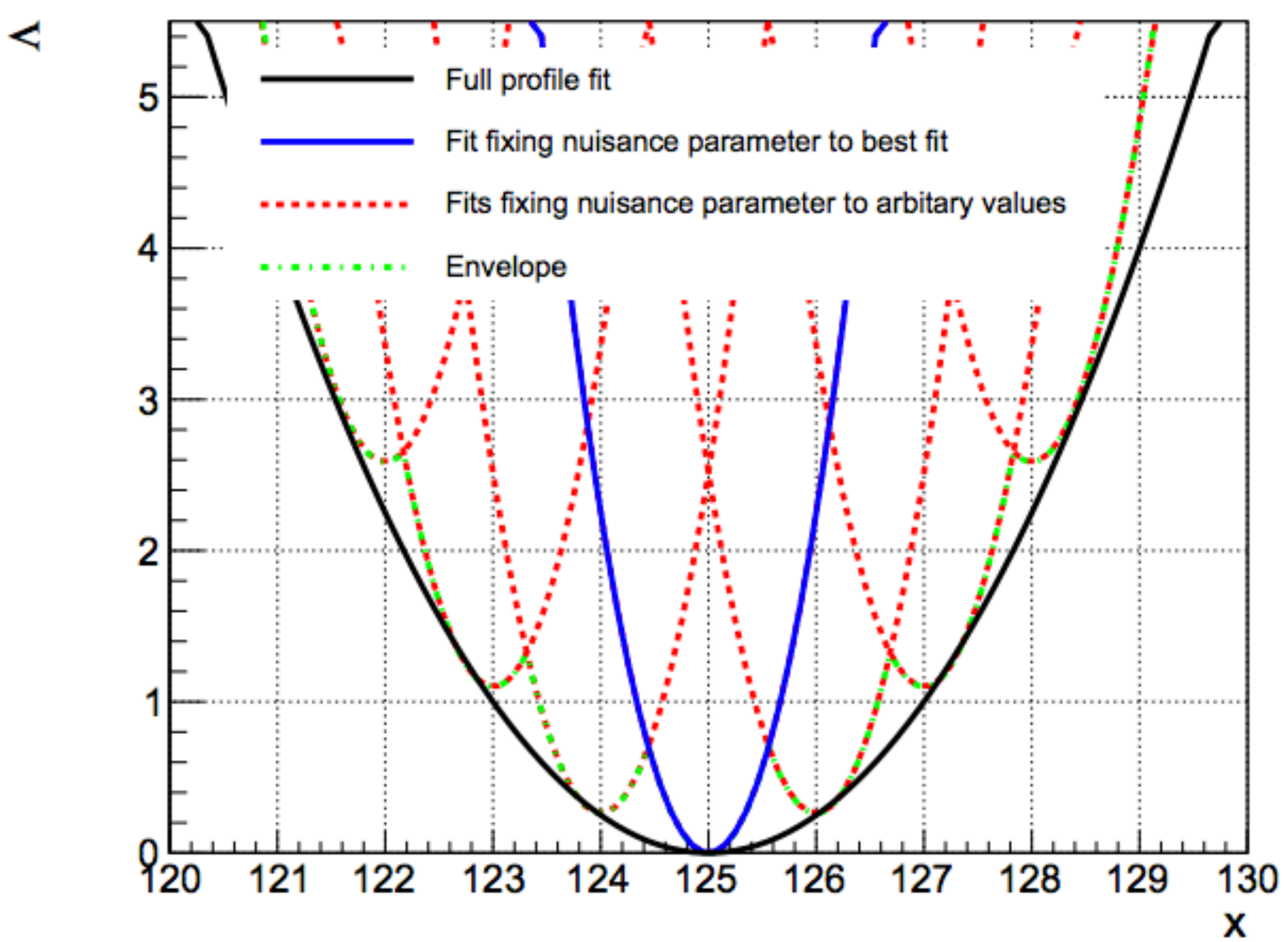
30th November 2017
Northeastern Meeting

- Presentation in the Hgg group
  https://indico.cern.ch/event/668514/contributions/2770370/attachments/1549402/2433498/Hgg_Oct30_2017.pdf

- Today's presentation
  - Focus on Background modeling
  - Present preliminary results of fits of various PDF's to data

- h(125)→aa→$\gamma\gamma\gamma\gamma$
  - Unbinned parametric shape analysis
  - As in the "standard H→$\gamma\gamma$" + "Low mass (<110 GeV) resonance search in the diphoton final state" analysis , the background estimation will be extracted directly from data
  - Background will be modeled by fitting analytic functions to the observed tetraphoton mass distribution $100 < M_{\gamma\gamma\gamma\gamma} < 160$ GeV
  - Method being used is the Discrete Profiling or "Envelope" method
- ENVELOPE METHOD
  - Designed to determine the systematic uncertainty associated with choosing a particular analytic function to fit to to the background $M_{\gamma\gamma\gamma\gamma}$ distribution
  - The choice of background function is treated as a discrete parameter in the likelihood fit to produce the final results
  - Resulting systematic uncertainty is calculated in a similar way to systematic uncertainties arising due to other measurements
  - A complete set of candidate function families should be considered

- When fitting these functions to the $M_{\gamma\gamma\gamma\gamma}$ distribution, the value of 2NLL is minimized
- The number of floating parameters in each candidate function are taken into account
- The Discrete Profile method determines the envelope of the lowest values of 2NLL profiled as a function of the parameters of interest
- The envelope obtained through this method yields a broader curve than the 2NLL curve obtained from a single choice of function

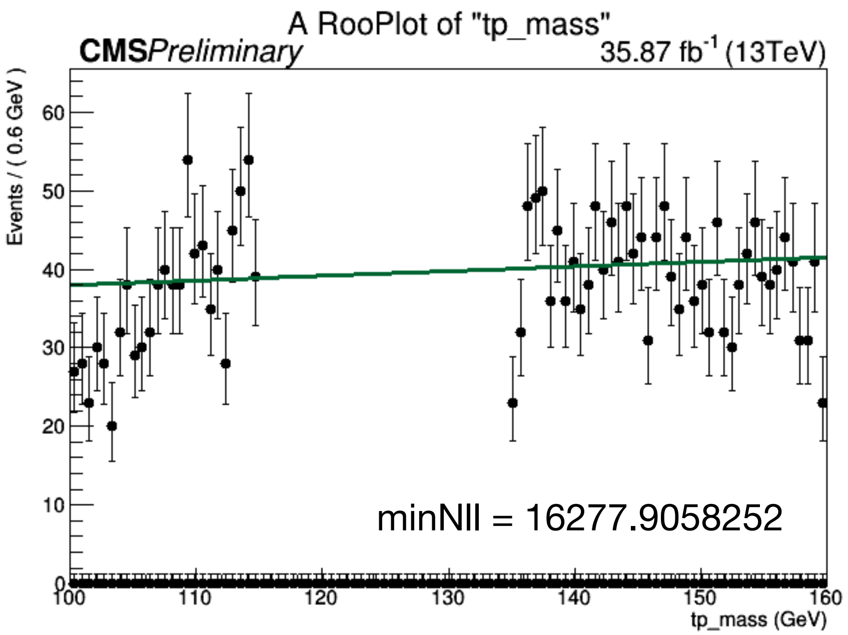


https://arxiv.org/pdf/1408.6865.pdf

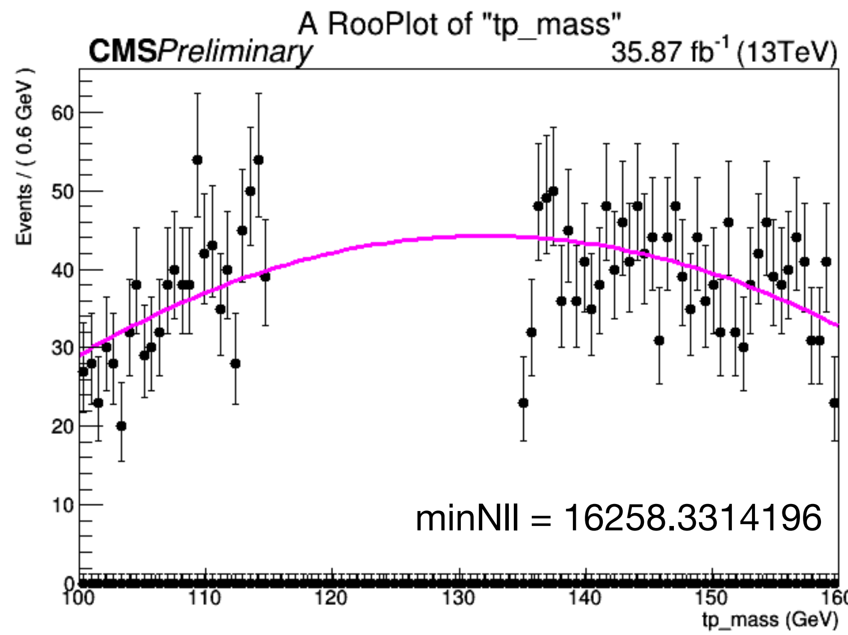
FAMILY OF BERNSTEIN POLYNOMIALS

$$f_N(x) = \sum_{i=0}^{N} p_i b_{(i,N)}, \text{ where } b_{(i,N)} := \binom{N}{i} x^i (1-x)^{N-i},$$

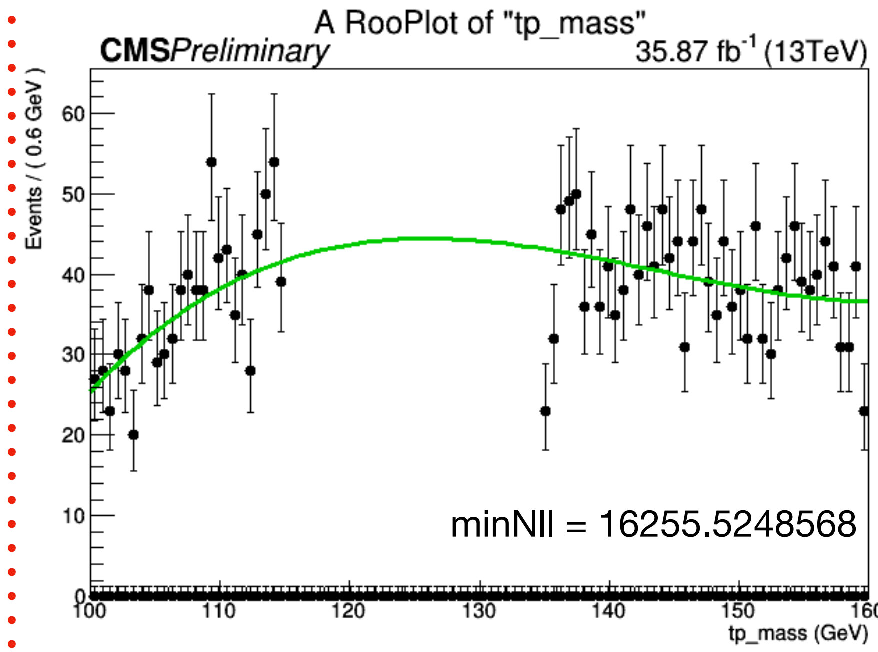
Order 1



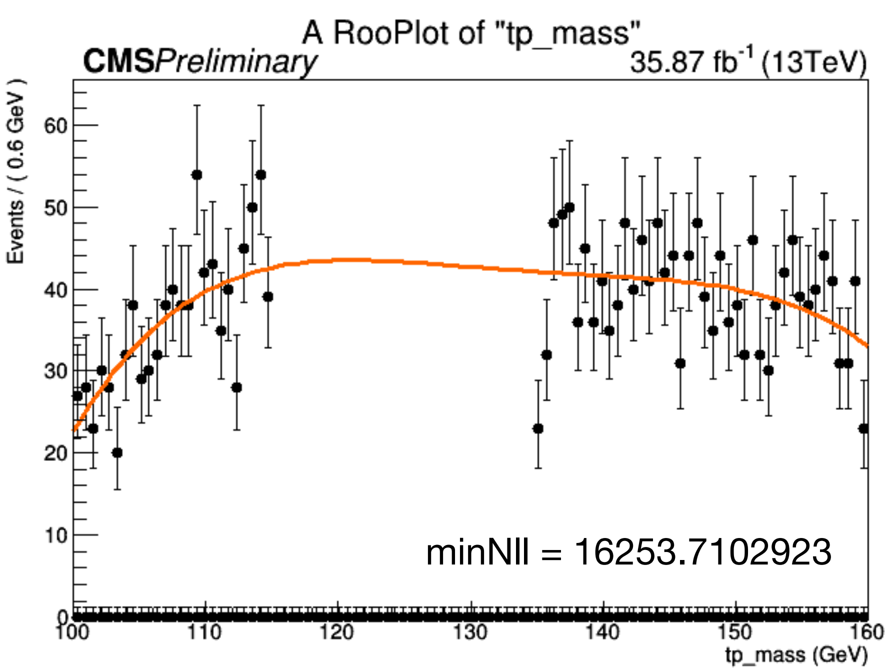
Order 2



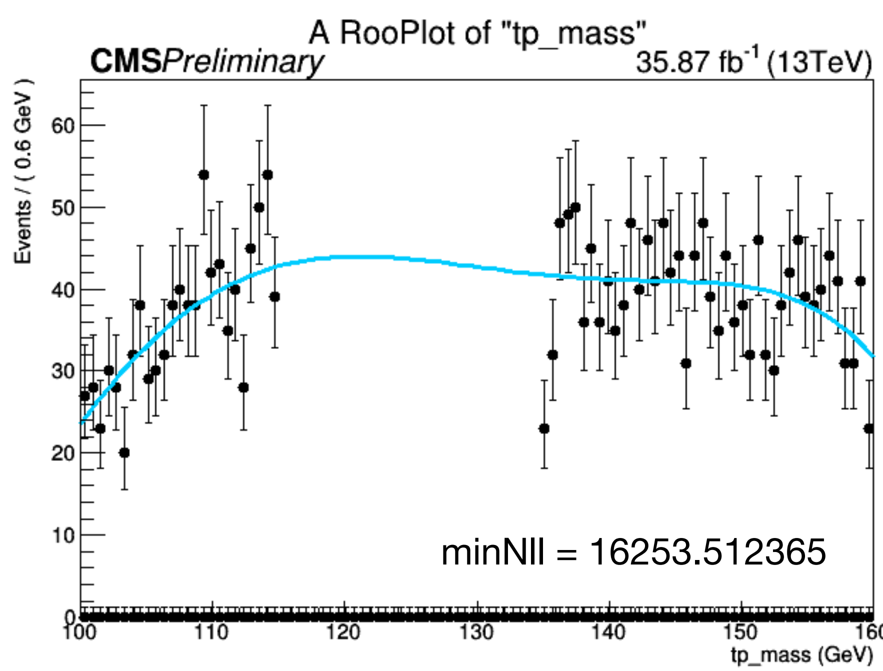
Order 3



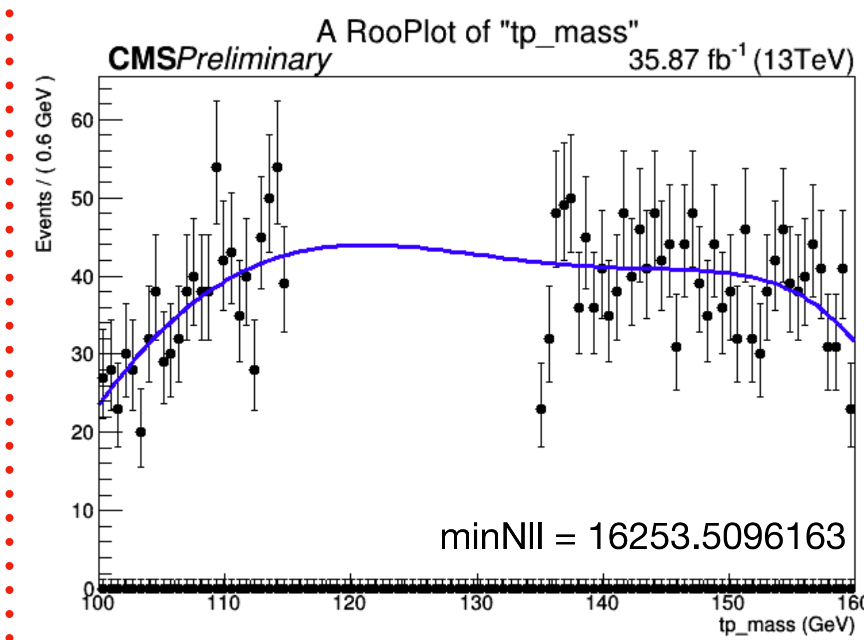
Order 4



Order 5

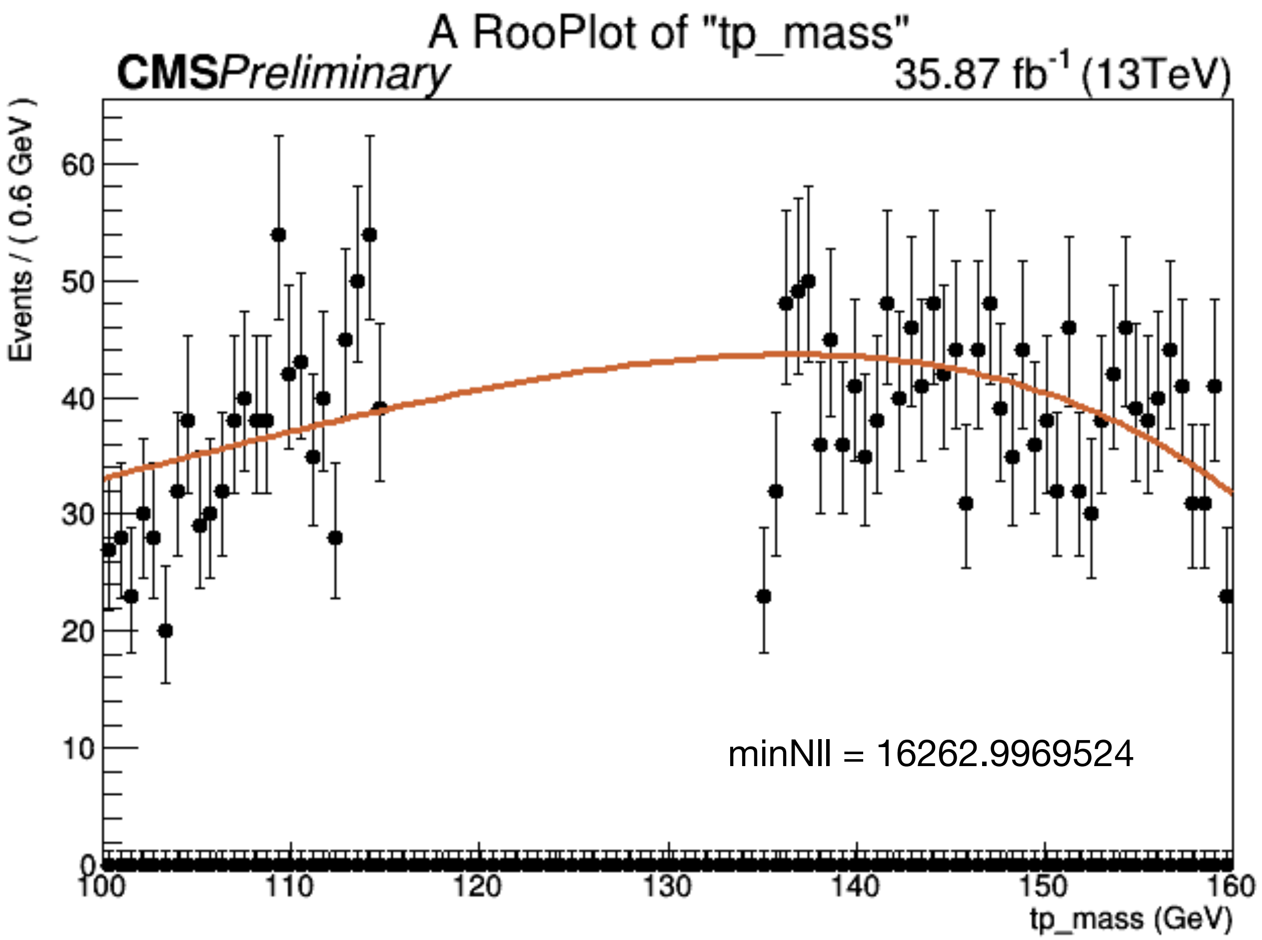


Order 6



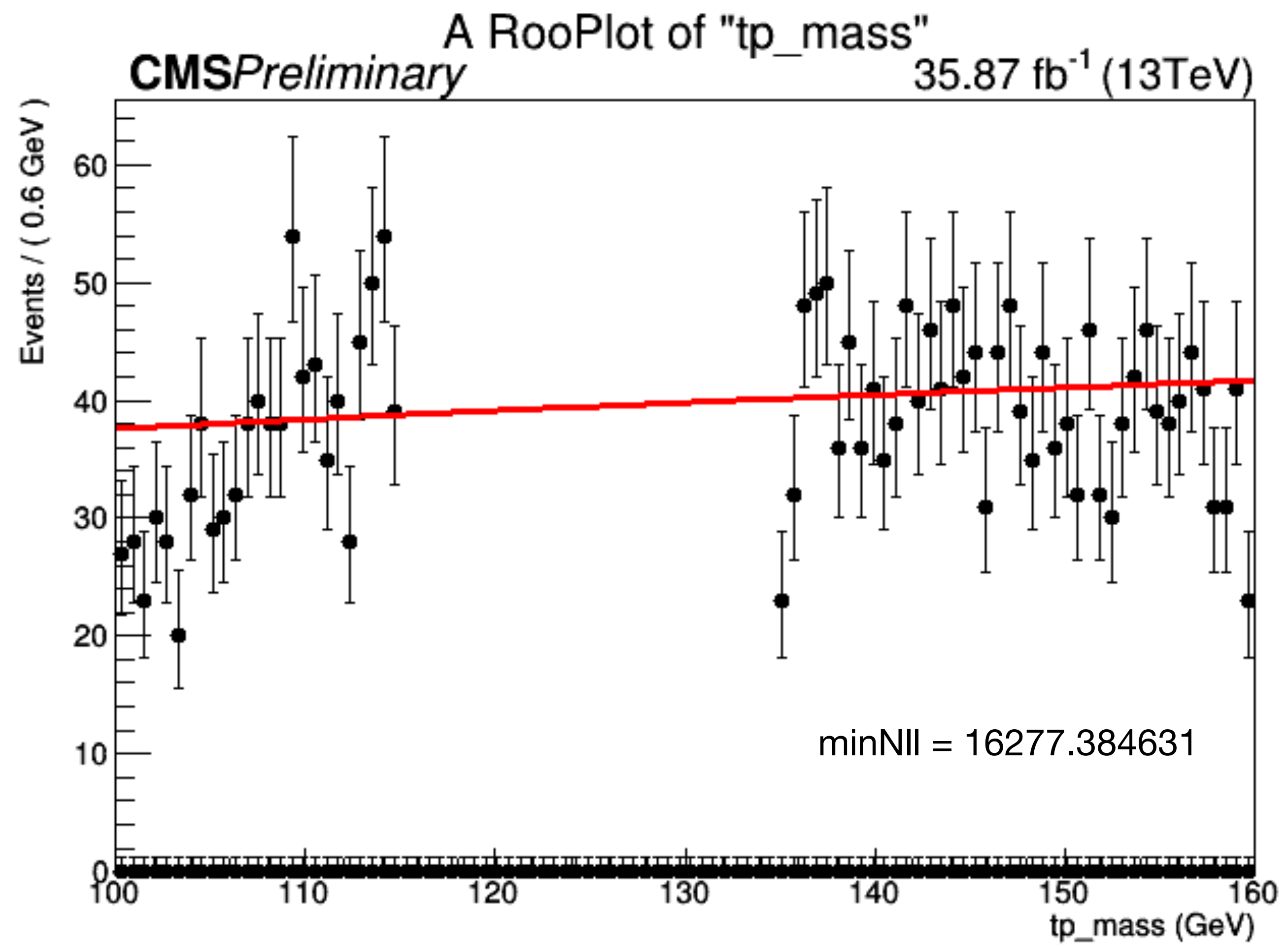
SUM OF 2 EXPONENTIALS

$$f_N(x) = \sum_{i=1}^{N} p_{2i} e^{p_{2i+1} x},$$



POWER LAW FUNCTION

$$f_N(x) = \sum_{i=1}^{N} p_{2i} x^{-p_{2i+1}},$$



TO DO LIST

- Save in workspace a multipdf containing all the fitted pdf's **DONE**
  - Add Laurent series as background pdf **TO BE DONE**

- Save in workspace the signal model
  - A simple model based on sum of gaussian functions **DONE**
  - Need to take a look at how the fit to the signal is performing **TO BE DONE**

- Bias Study
  - Combine (provides a command line interface to RooFit/RooStats) can choose between different background PDF's through multiPDF
  - Each PDF can be accessed through its pdf index
  - Generate toys using one function and fit with another **TO BE DONE**

WORKSPACE

```
RooWorkspace(w) w contents

variables
---------
(CMS_channel,bern1_p0,bern1_p1,bern2_p0,bern2_p1,bern2_p2,bern3_p0,bern3_p1,bern3_p2,bern3_p3,bern4_p0,bern4_p1,bern4_p2,bern4_p3,bern4_p4,bern5_p0,bern5_p1,bern5_p2,bern5_p3,bern5_p4,bern5_p5,bern6_p0,bern6_p1,bern6_p2,bern6_p3,bern6_p4,bern6_p5,bern6_p6,lambda1,lumi,lum
i_In,mass1,mass2,mass3,n_exp_bint4gamma-supercut_proc_bkg,ng1,ng2,pdf index,r,sigma1,sigma2,sigma3,tp_mass)

p.d.f.s
-------
RooBernstein::bkg_bern1_pdf[ x=tp_mass coefList=(bern1_p0,bern1_p1) ] = 0.0980284
RooBernstein::bkg_bern2_pdf[ x=tp_mass coefList=(bern2_p0,bern2_p1,bern2_p2) ] = 0.117682
RooBernstein::bkg_bern3_pdf[ x=tp_mass coefList=(bern3_p0,bern3_p1,bern3_p2,bern3_p3) ] = 0.115638
RooBernstein::bkg_bern4_pdf[ x=tp_mass coefList=(bern4_p0,bern4_p1,bern4_p2,bern4_p3,bern4_p4) ] = 0.110484
RooBernstein::bkg_bern5_pdf[ x=tp_mass coefList=(bern5_p0,bern5_p1,bern5_p2,bern5_p3,bern5_p4,bern5_p5) ] = 0.126505
RooBernstein::bkg_bern6_pdf[ x=tp_mass coefList=(bern6_p0,bern6_p1,bern6_p2,bern6_p3,bern6_p4,bern6_p5,bern6_p6) ] = 0.138954
RooExponential::bkg_exp1_pdf[ x=tp_mass c=lambda1 ] = 1.21767
SimpleGaussianConstraint::lumi_Pdf[ x=lumi mean=lumi_In sigma=1 ] = 1
RooSimultaneousOpt::model_b[ indexCat=CMS_channel t4gamma-supercut=pdf_bint4gamma-supercut_bonly extraConstraints=() channelMasks=() ] = 0.0980284
RooSimultaneousOpt::model_s[ indexCat=CMS_channel t4gamma-supercut=pdf_bint4gamma-supercut extraConstraints=() channelMasks=() ] = 0.00790766
RooProdPdf::nuisancePdf[ lumi_Pdf ] = 1
RooProdPdf::pdf_bint4gamma-supercut[ lumi_Pdf * pdf_bint4gamma-supercut_nuis * pdfbins_bint4gamma-supercut ] = 0.00790766
RooProdPdf::pdf_bint4gamma-supercut_bonly[ lumi_Pdf * pdf_bint4gamma-supercut_bonly_nuis * pdfbins_bint4gamma-supercut ] = 0.0980284
RooAddPdf::pdf_bint4gamma-supercut_bonly_nuis[ n_exp_bint4gamma-supercut_proc_bkg * shapeBkg_bkg_t4gamma-supercut ] = 0.0980284
RooAddPdf::pdf_bint4gamma-supercut_nuis[ n_exp_bint4gamma-supercut_proc_bkg * shapeBkg_bkg_t4gamma-supercut + n_exp_bint4gamma-supercut_proc_sig * shapeSig_sig_t4gamma-supercut ] = 0.00790766
RooProdPdf::pdfbins_bint4gamma-supercut[  ] = 1
RooMultiPdf::shapeBkg_bkg_t4gamma-supercut[ _pdfs=(bkg_bern1_pdf,bkg_bern2_pdf,bkg_bern3_pdf,bkg_bern4_pdf,bkg_bern5_pdf,bkg_bern6_pdf,bkg_exp1_pdf) _index=pdf index ] = 0.0980284
RooAddPdf::shapeSig_sig_t4gamma-supercut[ ng2 * sig_sum_gaus12_pdf + [%] * sig_gaus3_pdf ] = 2.6389e-10
RooGaussian::sig_gaus1_pdf[ x=tp_mass mean=mass1 sigma=sigma1 ] = 6.91087e-22
RooGaussian::sig_gaus2_pdf[ x=tp_mass mean=mass2 sigma=sigma2 ] = 3.93866e-10
RooGaussian::sig_gaus3_pdf[ x=tp_mass mean=mass3 sigma=sigma3 ] = 3.93866e-10
RooAddPdf::sig_sum_gaus12_pdf[ ng1 * sig_gaus1_pdf + [%] * sig_gaus2_pdf ] = 1.96933e-10

functions
--------
ProcessNormalization::n_exp_bint4gamma-supercut_proc_sig[ thetaList=(lumi) asymmThetaList=() otherFactorList=(r) ] = 71286

datasets
--------
RooDataSet::data_obs(CMS_channel,tp_mass)

named sets
----------
ModelConfig_GlobalObservables:(lumi_In)
ModelConfig_NuisParams:(lumi)
ModelConfig_Observables:(tp_mass,CMS_channel)
ModelConfig_POI:(r)
ModelConfig_bonly_GlobalObservables:(lumi_In)
ModelConfig_bonly_NuisParams:(lumi)
ModelConfig_bonly_Observables:(tp_mass,CMS_channel)
ModelConfig_bonly_POI:(r)
POI:(r)
globalObservables:(lumi_In)
nuisances:(lumi)
observables:(tp_mass,CMS_channel)

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