Global Fitting Update

Emrys Peets - Stanford University
Cameron Bravo - SLAC

10/17/2023

U.S. DEPARTMENT OF ENERGY
Office of Science

SLAC NATIONAL ACCELERATOR LABORATORY

Road Map

1. Quick Recap
2. List of Updates
3. Latest large statistics study results and promising function
4. Round 2 Fitting
5. Seed Finding Strategy
6. Summary + Discussion

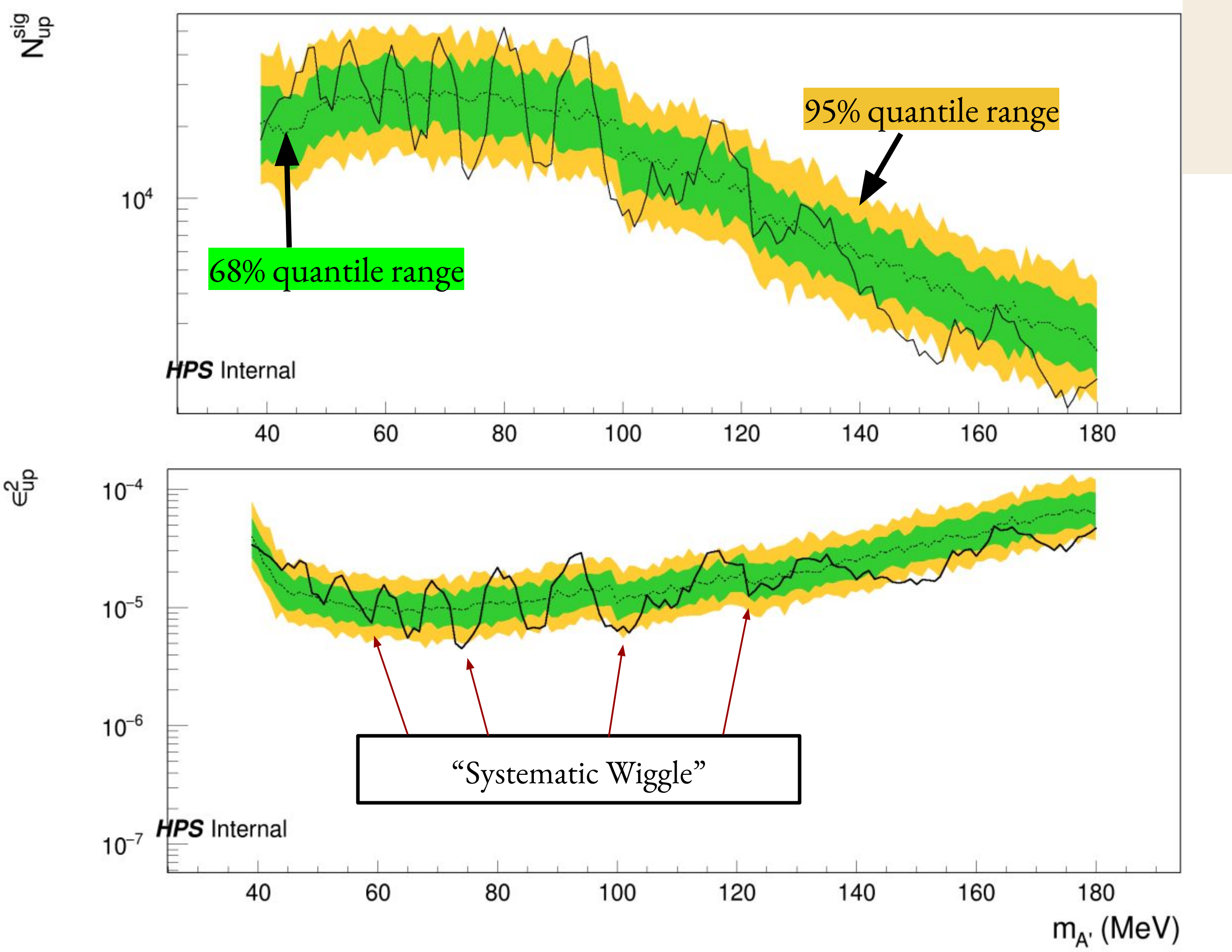
Important vocab to keep in mind

- IMD - Invariant Mass Distribution
- Window Range: Range by which a function is fit over some subset of the the IMD, generally of the form **[WinMin, WinMax]**
- WinMax: Maximum value for a given window range
- WinMin: minimum value for a given window range

2016 Limits

Signal yield and ε² upper limits. Includes statistical and systematic effects plotted over limit bands*.

*band from toy mc background models



Global Fit to the Invariant Mass Distribution (IMD)

May be able to take into account systematic features present in background shape.

Will be useful to freeze these features and determine reach.

Monotonically decreasing functions fitting the range above acceptance threshold

$$f_{dijet1}(x) = \frac{p_0(1-x)^{p_1}}{x^{p_2}} \qquad f_{dijet2}(x) = \frac{p_0(1-x)^{p_1}}{x^{p_2+p_3\log(x)}}$$

$$f_{dijet3}(x) = \frac{p_0(1-x)^{p_1}}{x^{p_2+p_3\log(x)+p_4\log^2(x)}} \qquad f_{ATLAS1}(x) = \frac{p_0(1-x^{1/3})^{p_1}}{x^{p_2}}$$

$$f_{ATLAS2}(x) = \frac{p_0(1-x^{1/3})^{p_1}}{x^{p_2+p_3\log^2(x)}} \qquad f_{UA2_1}(x) = p_0 x^{p_1} e^{p_2 x}$$

$$f_{UA2_2}(x) = p_0 x^{p_1} e^{p_2 x + p_3 x^2} \qquad f_{UA2_3}(x) = p_0 x^{p_1} e^{p_2 x + p_3 x^2 + p_4 x^3}$$

$$f_{cmsBH1}(x) = \frac{p_0(1+x)^{p_1}}{x^{p_2 \log x}} \qquad f_{cmsBH2}(x) = \frac{p_0(1+x)^{p_1}}{x^{p_3+p_2\log x}}$$

$$f_{ATLASBH1}(x) = p_0(1-x)^{p_1} x^{p_2\log(x)} \qquad f_{ATLASBH2}(x) = p_0(1-x)^{p_1}(1+x)^{p_2\log(x)}$$

$$f_{ATLASBH3}(x) = p_0(1-x)^{p_1} e^{p_2\log(x)} \qquad f_{ATLASBH4}(x) = p_0(1-x^{1/3})^{p_1} x^{p_2\log(x)}$$

$$f_{ATLASBH5}(x) = p_0(1-x)^{p_1} x^{p_2 x)} \qquad f_{ATLASBH6}(x) = p_0(1-x)^{p_1}(1+x)^{p_2 x}$$

Error function used:

$$\mathrm{Er}(x) = \frac{1}{2}\left(\mathrm{Erf}\left(\frac{(x-[q_0])}{[q_1]}\right)+1\right)$$

model acceptance threshold

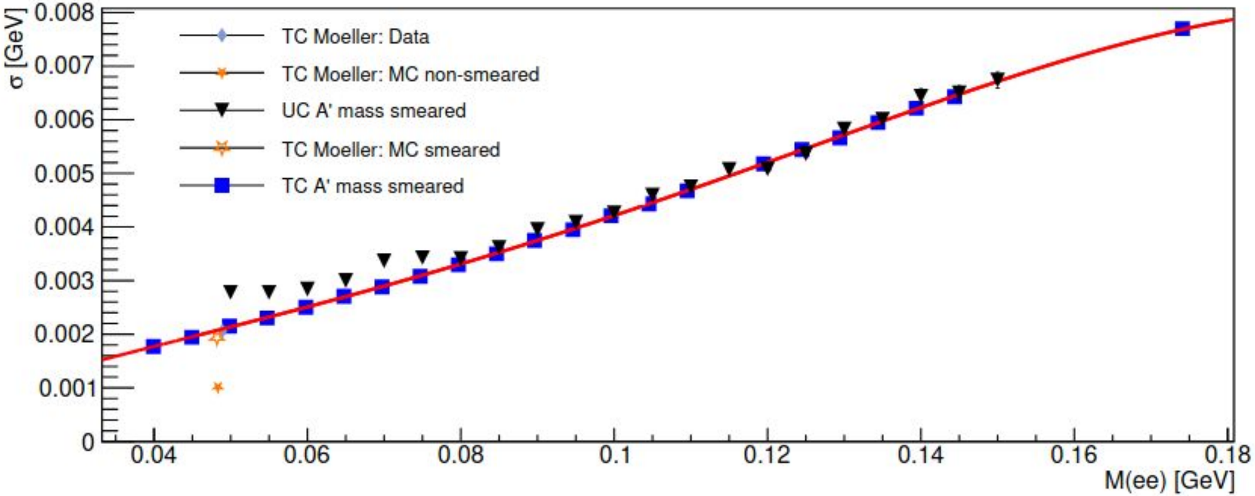
C. Bravo. *Thesis used to get list of monotonically decreasing functions*

Determining Optimal Window Range

Must consider mass resolution when cutting window ranges to prioritize good fits.

To fit a desired window range well, must consider 3σ below win_min and 3σ above win_max.

- want to fit [win_min, win_max]? → need good fit over range [win_min-3σ_{win_min}, win_max+3σ_{win_max}]



HPS Mass Resolutions as shown in 2016 Result

e.g. To have similar sensitivity to 2016 range of **[39, 179] MeV**

→ must fit [39 - 3$\sigma_{39\ MeV}$, 179 + 3$\sigma_{179\ MeV}$] ~~ **[34, 203] MeV**

Last Update (08/08/23)

125 Functions → 131 Functions

- add to function list by modifying previously used functions
- best pvalue 3e-3 → **3.7e-3** over range [38, 202] MeV

Incorporated Global Fit Toolkit into all current fit infrastructure.

- save, plot and organize best fits for each function over all tested window ranges
- includes, residual, residual / sqrt(N), residual^2 / N, Pull plot

Higher statistics functionability: **Batches**

- **fit_merger.py** merge best fits for a given window range of all batches into a "best of" plot
- **fit_compiler.py** compile best fits of all functions together

"Rebinner" Tool

- Capable of rebinning any desired already fit function using terminal inputs
- can take into account any desired rebinning factor

Updates Since Last Update

Still using 131 Functions

I. Finished a 5 batch higher statistics study
    A. Surprisingly good fit – 2.3e-2 on range [38, 194] MeV
    B. motivated rebinning, next study and parameter selection changes
II. Developed Parameter Storer
    A. parses **best_fit_info.txt** file, selects parameters with fits above specified pval, stores in new param file
III. “Round 2” Fitting Procedure developed and tested
    A. Uses parameters from^ and tries >1000 fits for each window range
    B. Found 5ish functions with good fits with WinMin ~ 50 MeV
IV. New Display Plot
    A. fixes window minimum + varies window max (tested on round 2 results)
V. Increased Variance Study
    A. purpose is more efficient filtering of functions / conserve computing power
    B. 8 hour study
    C. 11 Functions found with pval $>10^{-2}$ over range [50,198] MeV

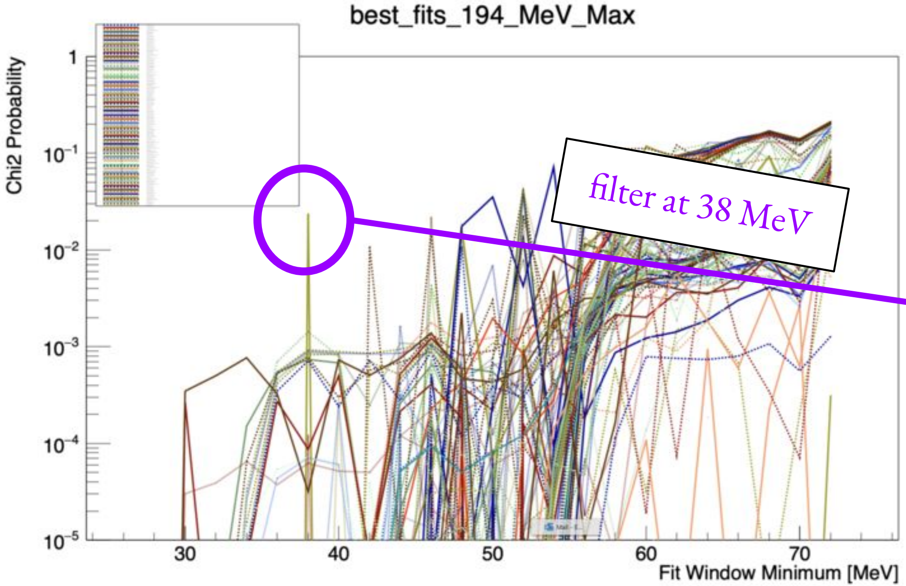
Workflow of Finding/Using Good Seeds

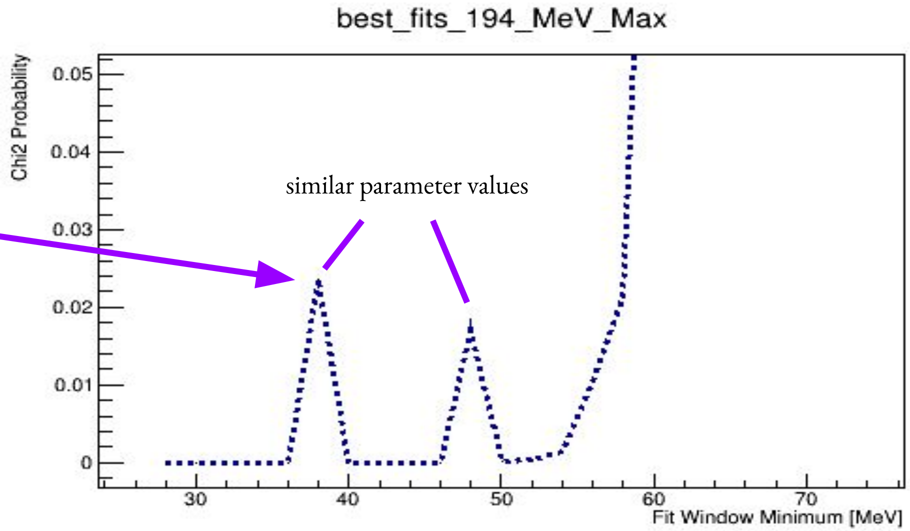
Selected Result of 5 Batch, 1250 iteration study:

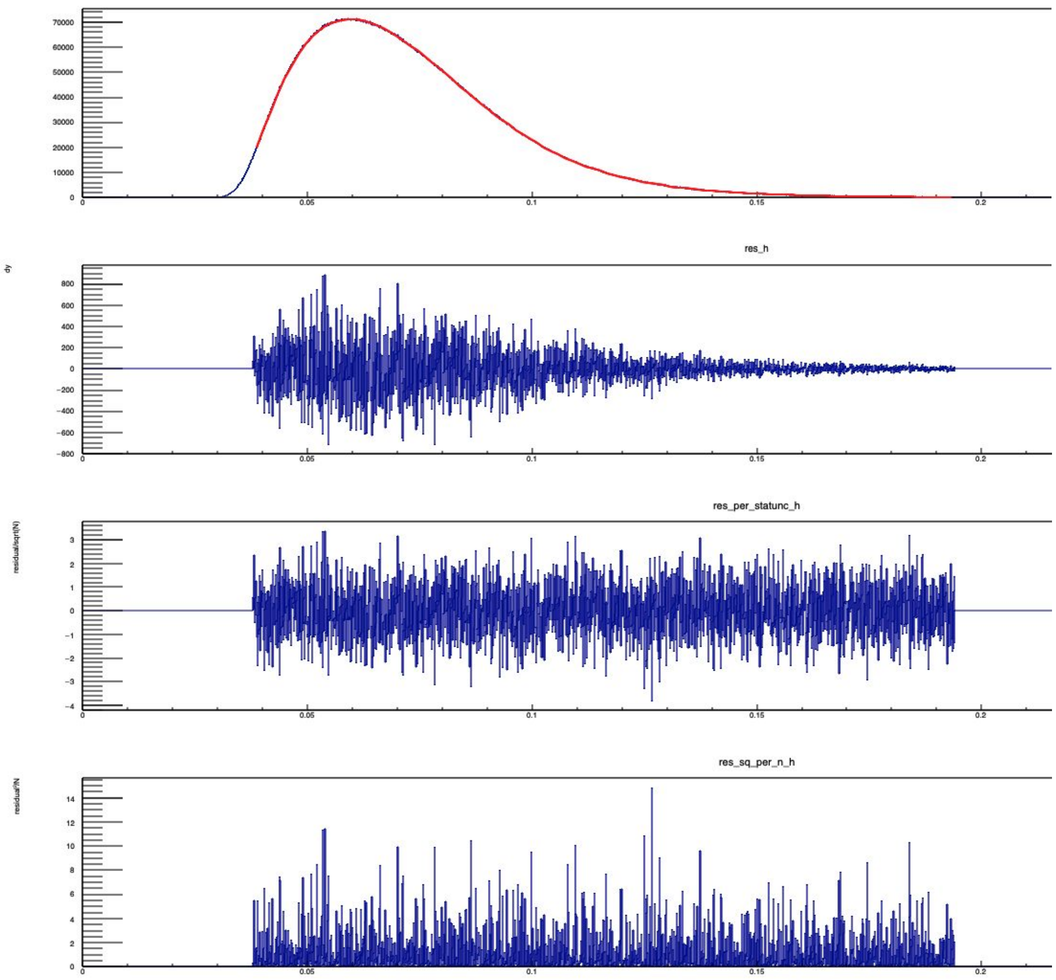
- winmin range: [28, 72] MeV - 2 MeV Steps
- winmax range: [178, 210] MeV - 4 MeV Steps

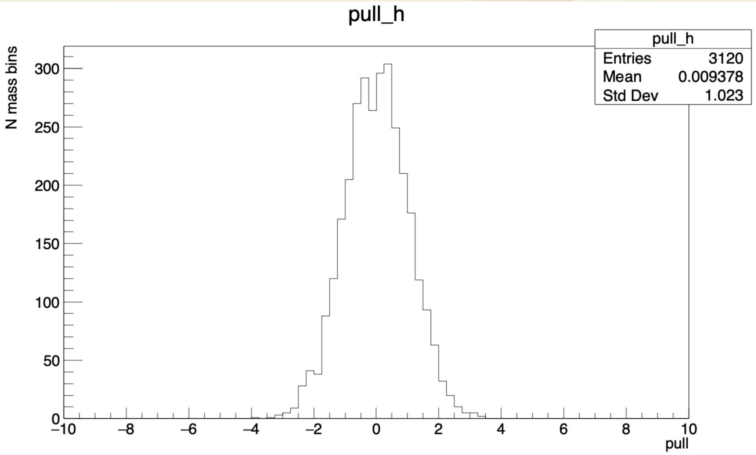
Promising Function:

- las3_plus_las6 (10 parameters)
- pvalue of **2.3e-2** on window range [38,194] MeV
- fit information, displays, rebinned info located here



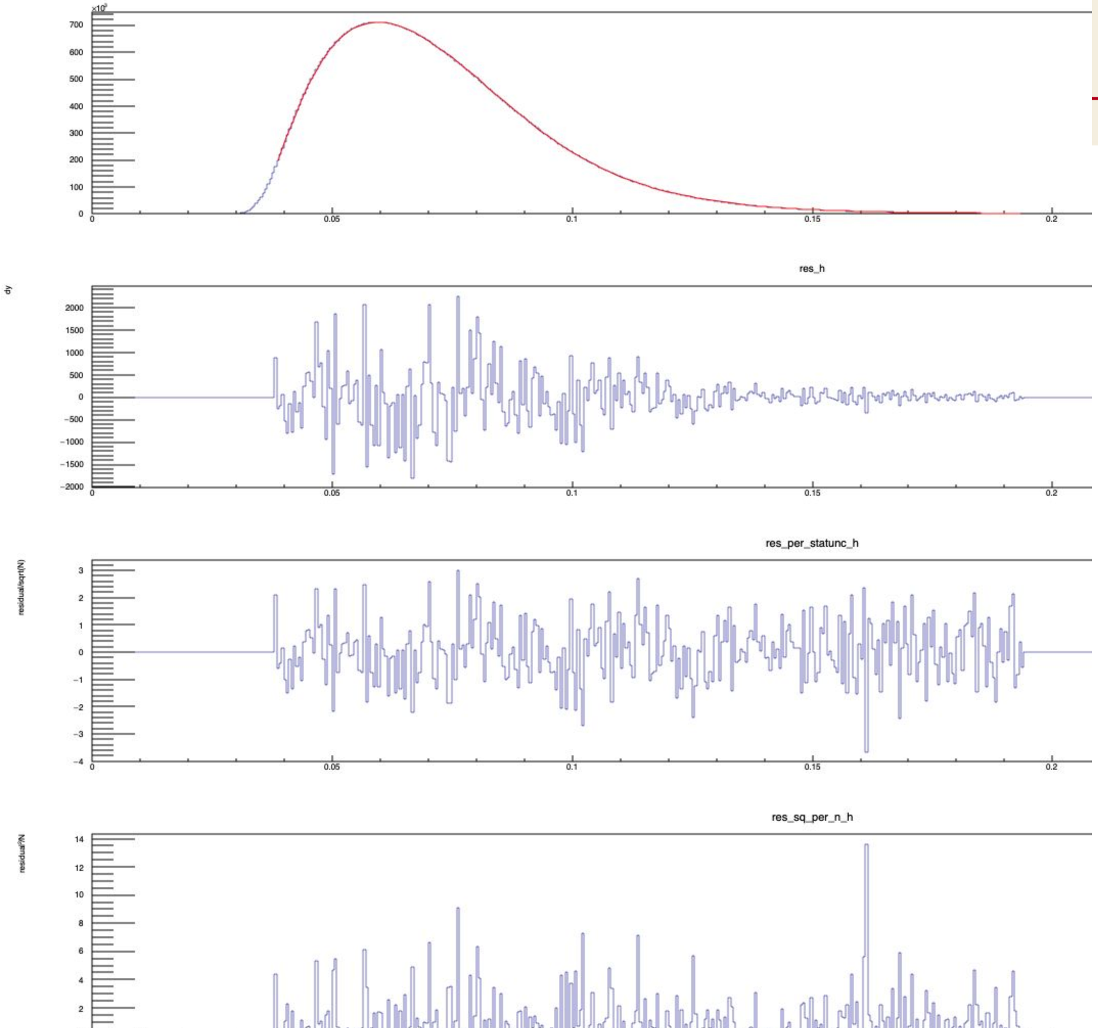


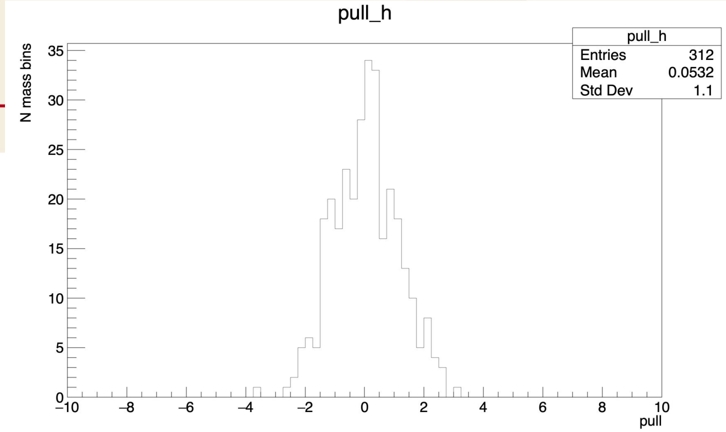




No rebinning: **PValue of 2.3 e-2**

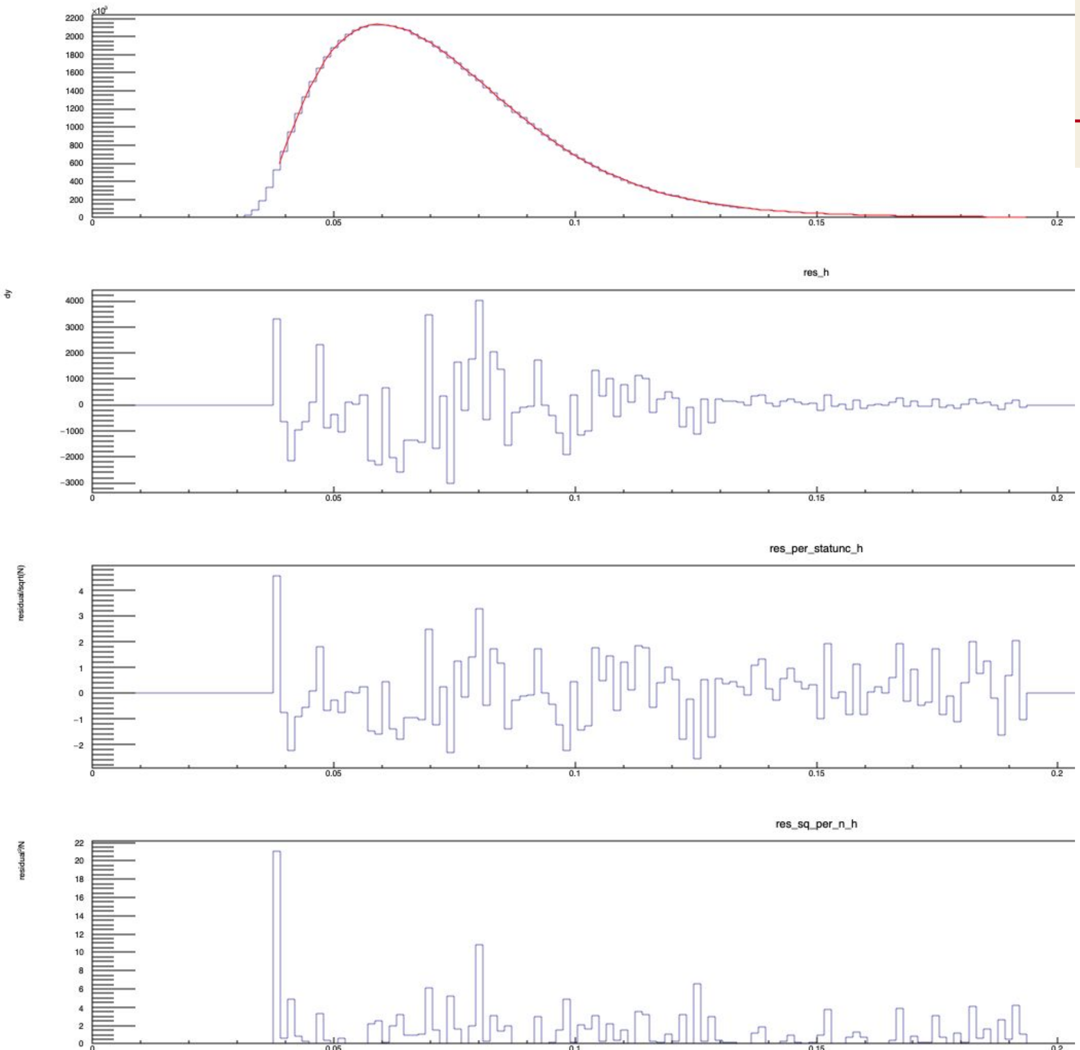
- window range [38,194] MeV

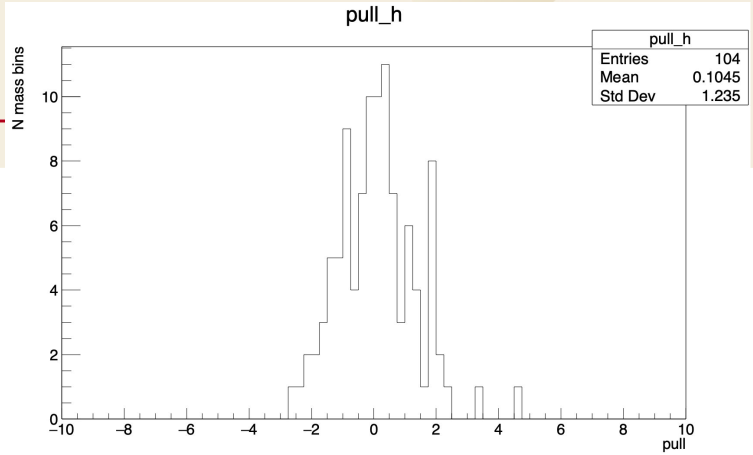




Rebinning factor 10: **PValue of 2.3 e-2**

- window range [38,194] MeV



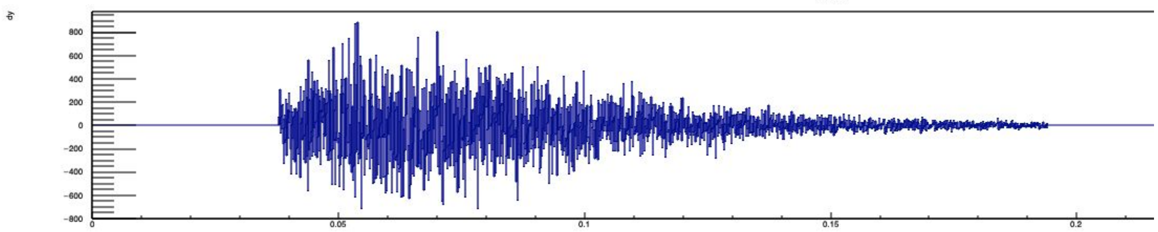


Rebinning factor 30: **PValue of 2.3 e-2**

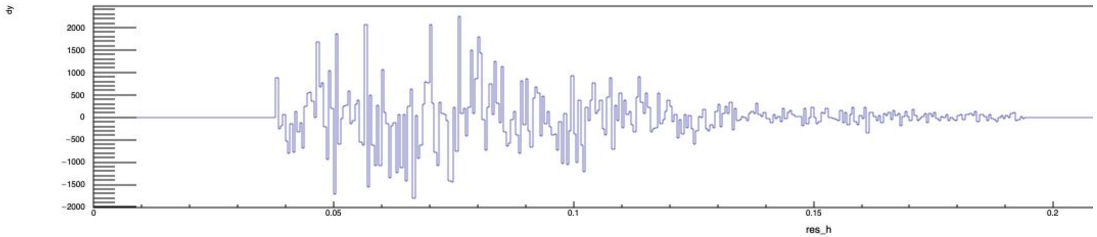
- window range [38,194] MeV

Residual Comparison

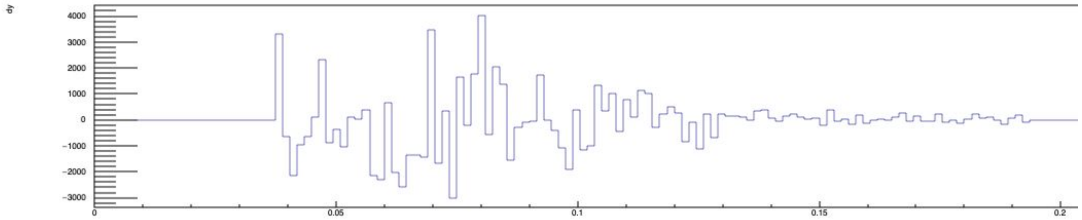
rebin factor: 0



rebin factor: 10

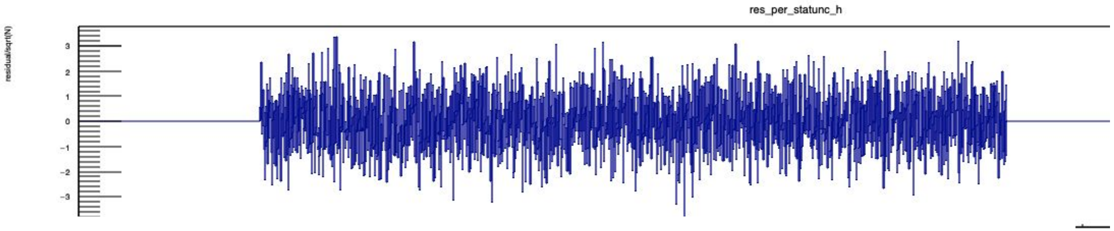


rebin factor: 30

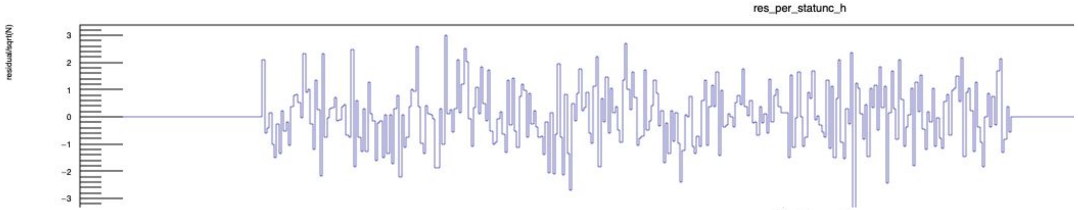


Residual/N^(1/2) Comparison

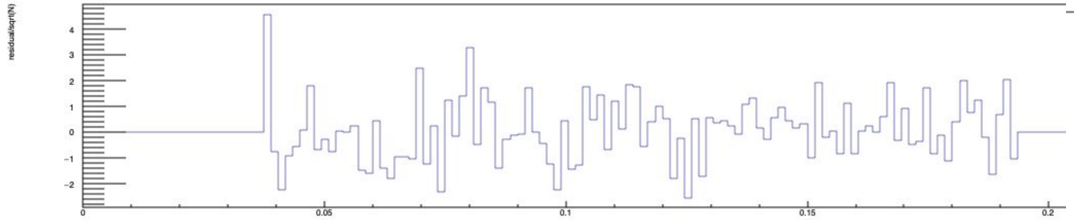
rebin factor: 0



rebin factor: 10

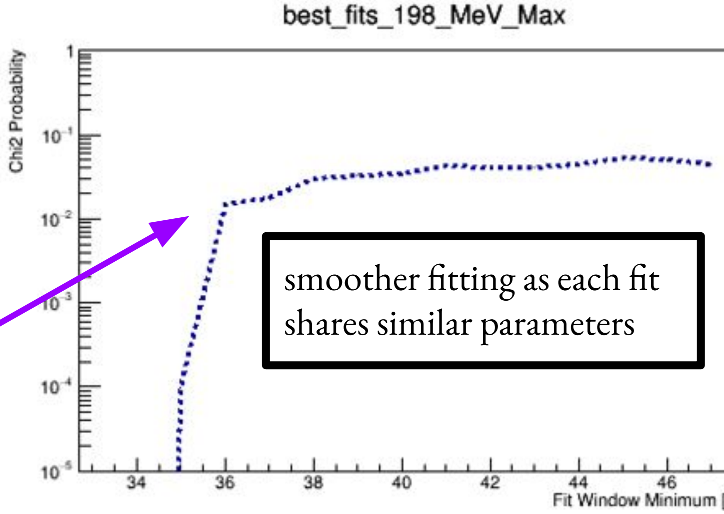


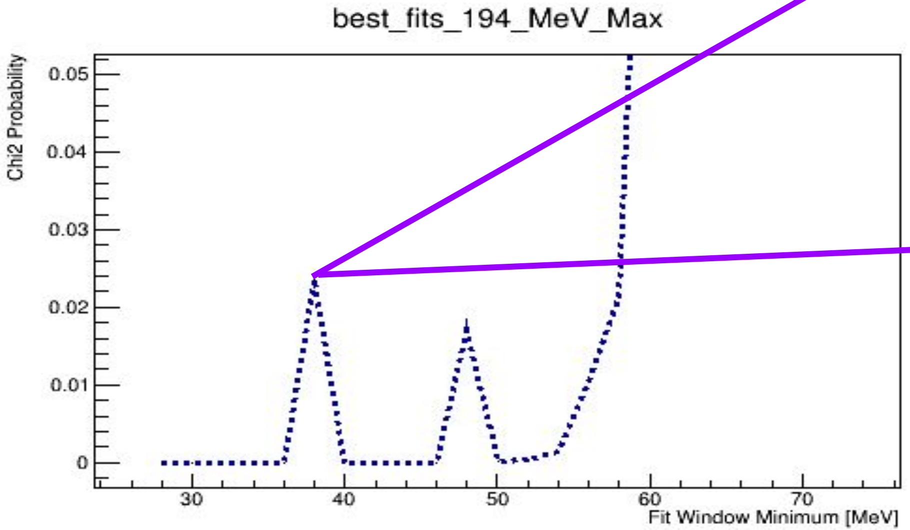
rebin factor: 30

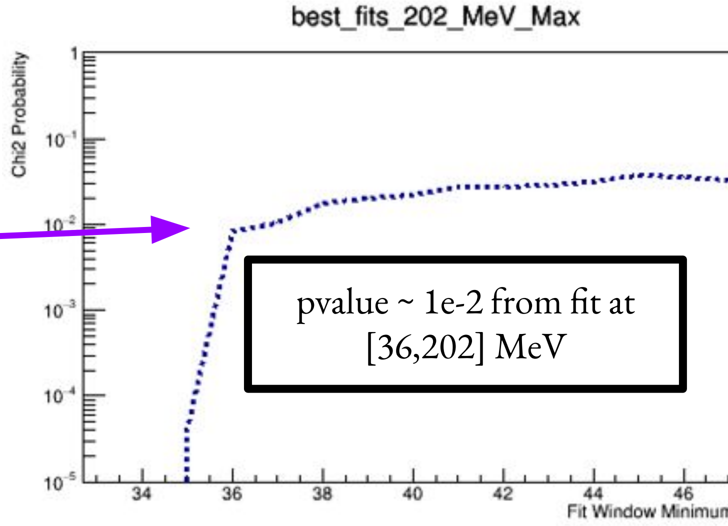


Round 2 Fitting [n = 1]

Tested method of using parameters found for las3_plus_l
as the exact seeds used in fitting each window range for a f
hundred iterations.







Collecting Parameters

Motivated by spikiness of las3_plus_las6 fits. Now parsing stored best fit parameters for all tested functions over an arbitrary number of ranges.

- Useful in finding good starting seeds for next set of fits.
- Stores results in **/functions/good_parameters/search_[search_range_n]_[threshold_n]/[function_name].txt**

Terminal Input Example: python3 collect_params.py -n 28 74 2 -x 178 214 4 -S 32 50 45 65 -Q 2 -F 1e-2 2e-2 -B 5

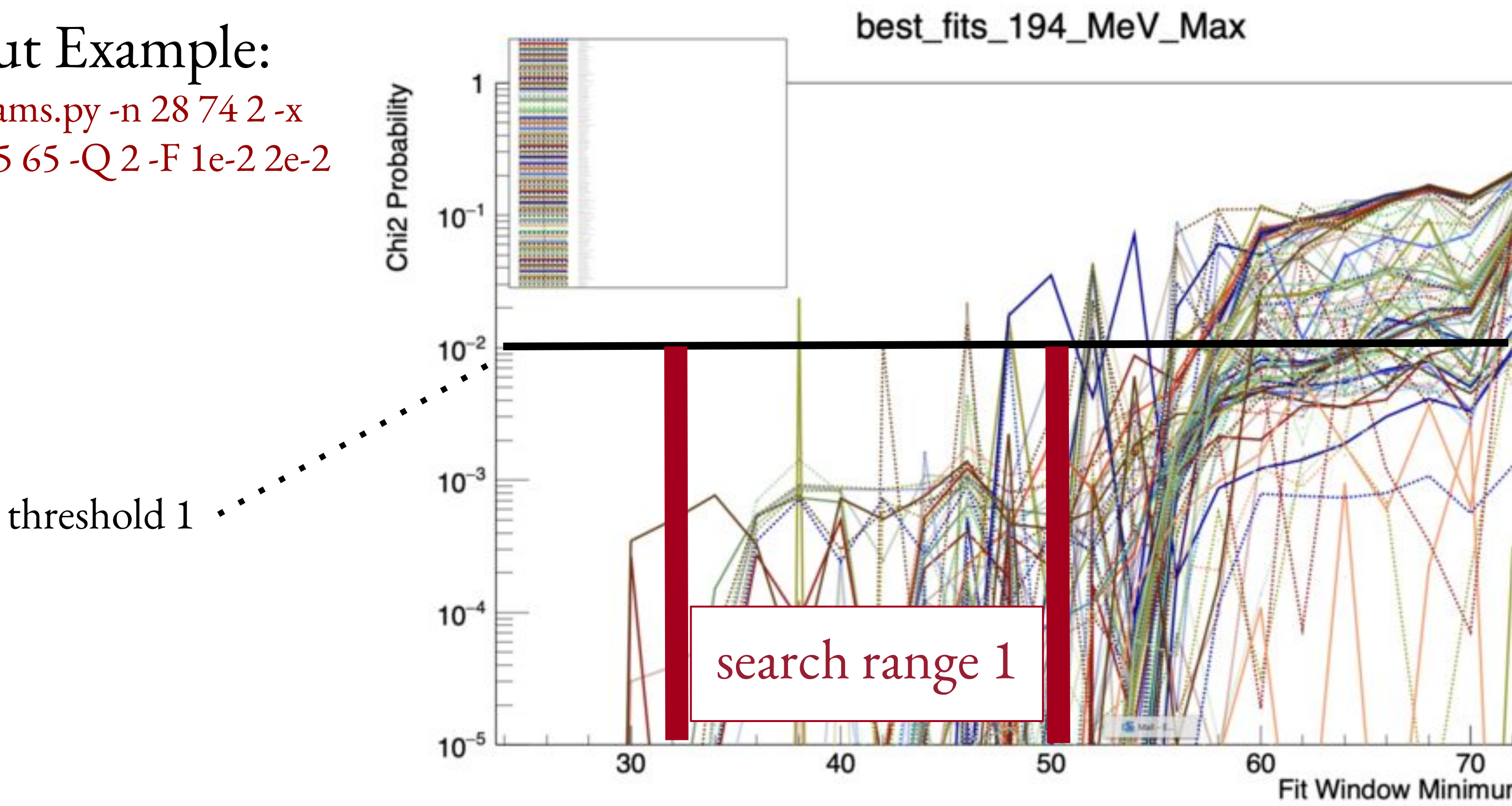
-s [winmin_r1_min , winmin_r1_max] [win_min_r2_min win_min_r2_max]
-Q [number of ranges]
-F [Threshold for range 1] [Threshold for range 2]
-B [n many batches to parse]

Output Example : /functions/good_parameters/search_4565_2e-2/ua23_mod_11.txt

Collecting Parameters [Visual]

Terminal Input Example:

python3 collect_params.py -n 28 74 2 -x 178 214 4 -S 32 50 45 65 -Q 2 -F 1e-2 2e-2 -B 5

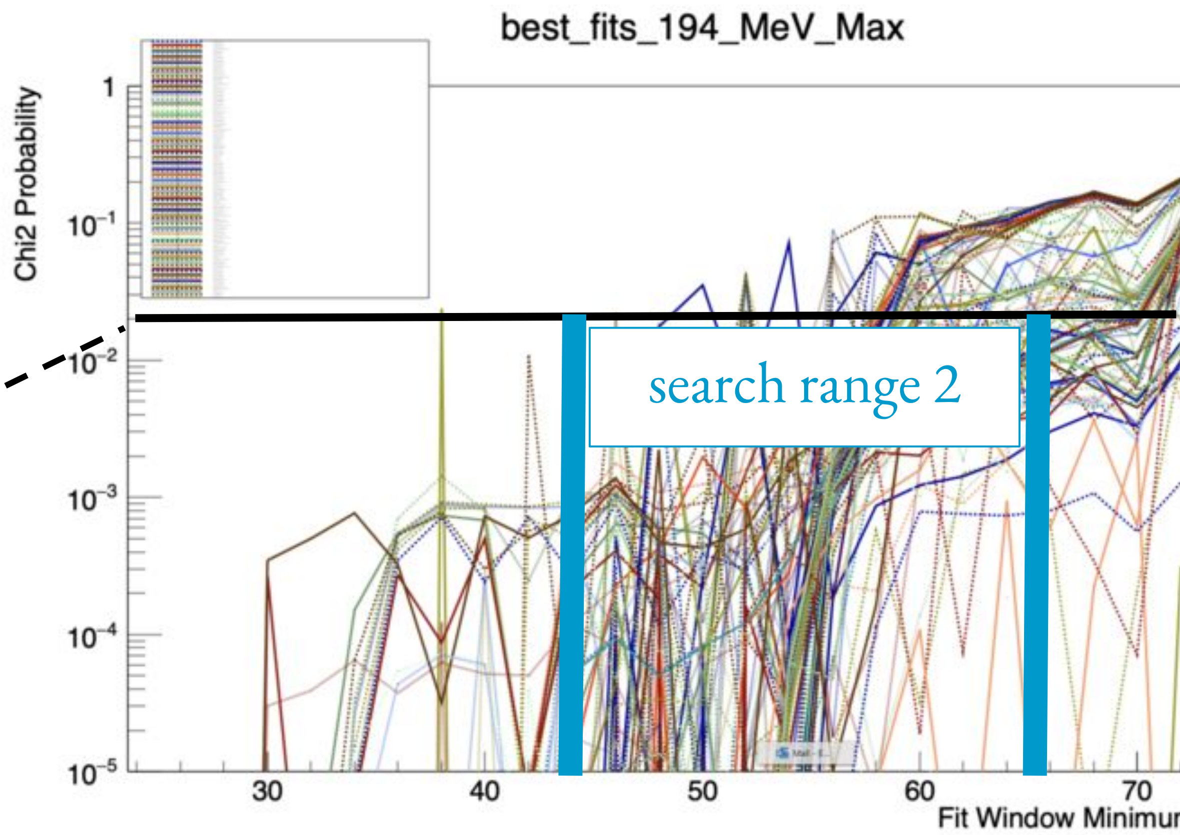


Collecting Parameters [Visual]

Terminal Input Example:

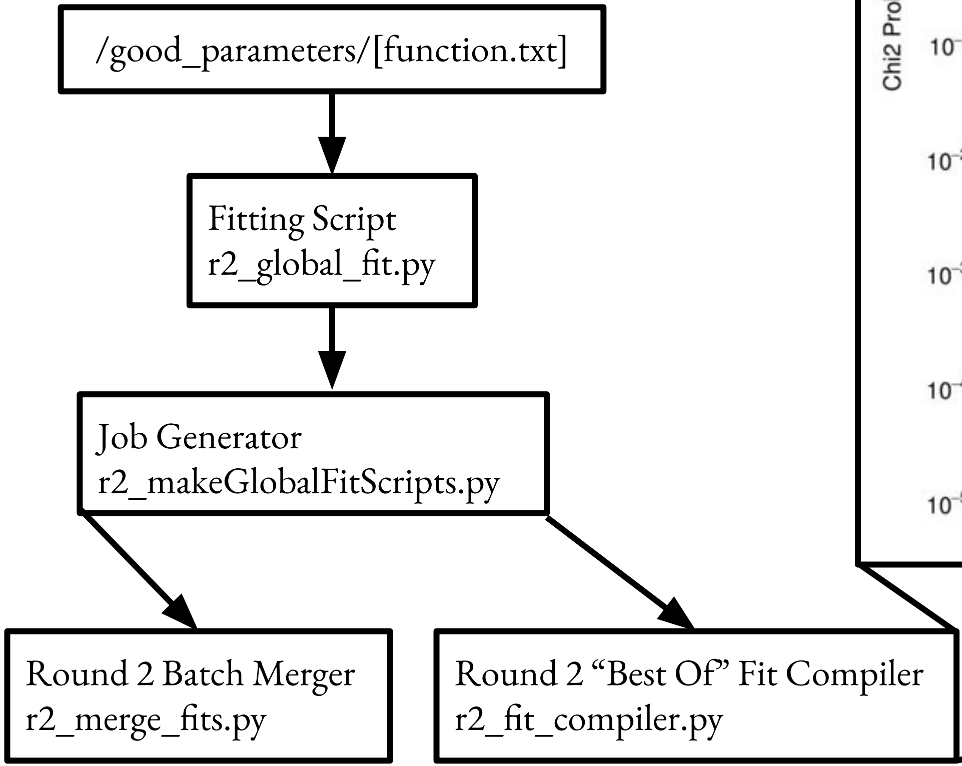
python3 collect_params.py -n 28 74 2 -x 178 214 4 -S 32 50 45 65 -Q 2 -F 1e-2 2e-2 -B 5

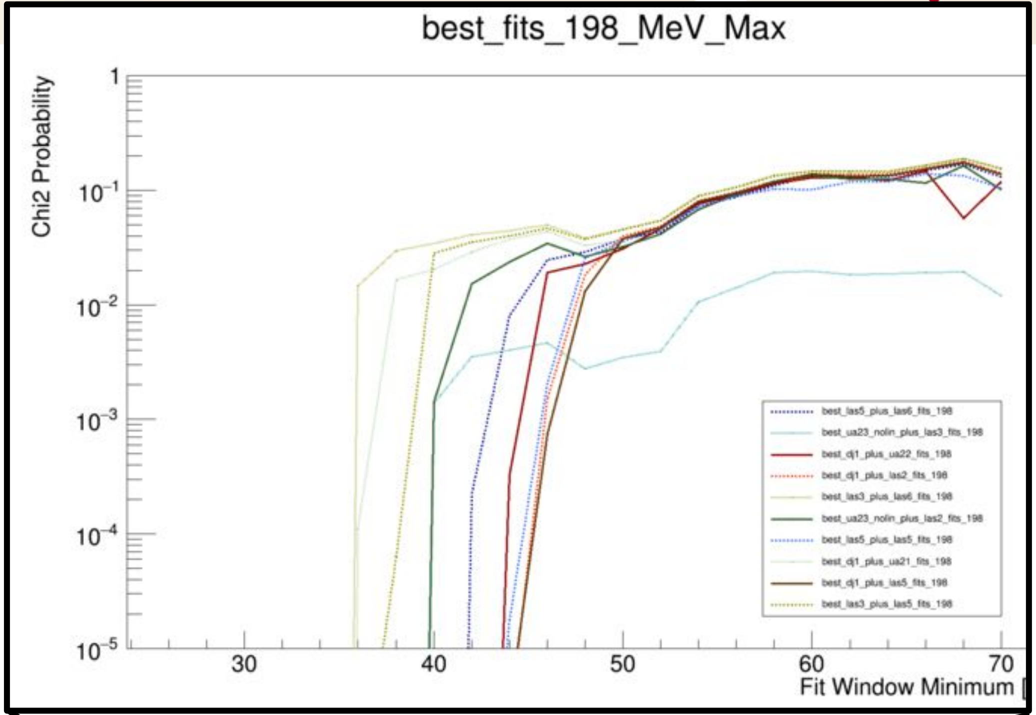
threshold 2



Round 2 Fitting [generalized]

New work flow for round 2 fitting.





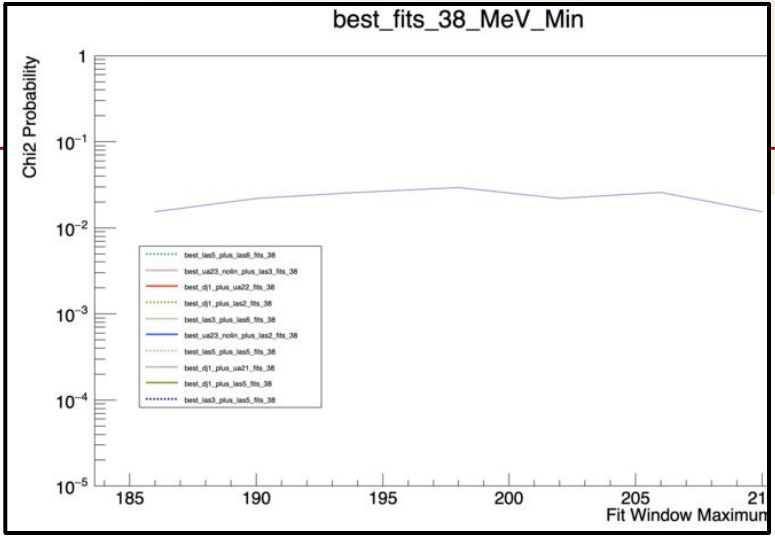
Round 2 Fitting [New Display Tool]

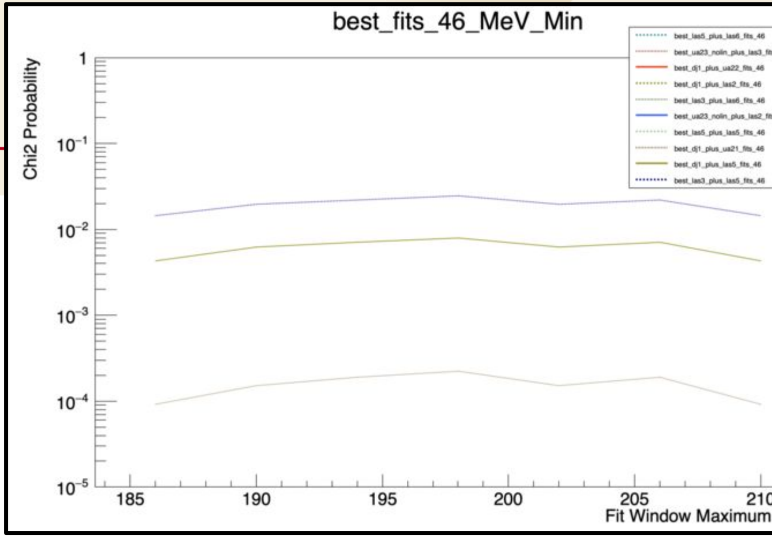
- To begin to determine where the maximum window range may break down in fitting, we want to see the trend in fitting across ranges with the window minimum remained fixed.
- **r2_wmax_fit_compiler.py** written to parse all fits of a particular search range

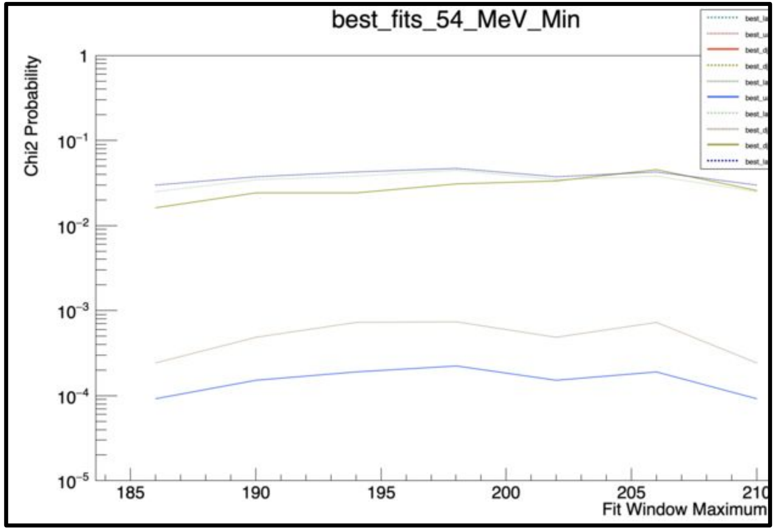
Terminal Input

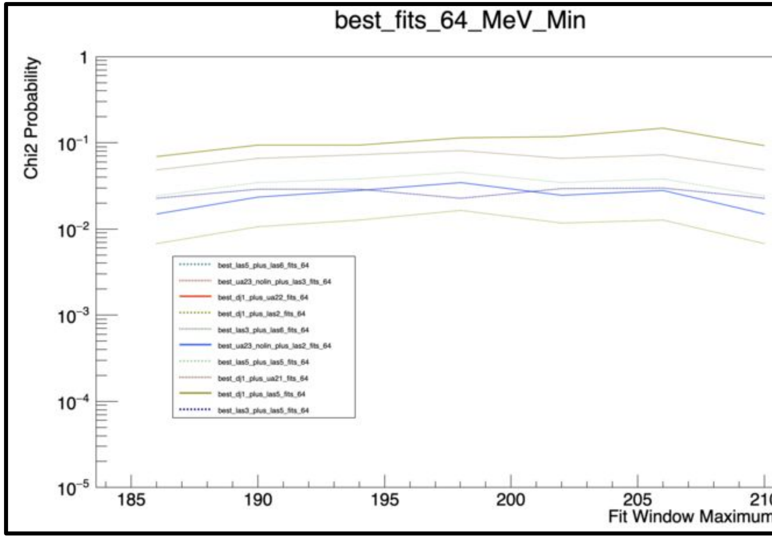
python3 r2_wmax_fit_compiler.py -n 28 72 2 -x 186 214 4 -B 5 -i /sdf/group/hps/users/epeets/run/resonance_fitting/functions/good_parameters/search_3250_1e-2/ -S 32 50 -F 1e-2

Varying WinMax









Challenge in Fitting

Goal Study

- ~15-20 batches of 50-70 iterations each
- increase variance to 5 % per iteration, and then add a factor of 100-200% for each batch
- win min = [35,65] in 1 MeV steps
- win max = [190, 300] in 1 MeV steps
- pull parameters from above study and use round 2 fitting infrastructure on low (35-45), middle (45-55), and high (55-65) win min parameter search ranges

Wanted to conduct a large study with increased variance.

- Would have taken far too much computing time
- Shifted immediate priorities towards filtering out low performing functions after increasing variance on a single fitting window.
- Would have had ~2000 jobs taking 50 hours minimum each….
- Turns out variance idea listed would have been terrible anyway!

Challenge in Fitting

SLAC

Goal Study

- ~15-20 batches of 50-70 iterations ea [illegible]
- increase variance to 5 % per ite [illegible] actor of 100-200% for each batch
- win min = [35,65] in 1 M [illegible]
- win max = [190, 2 [illegible]
- pull param [illegible] y and use round 2 fitting infrastructure on low (35-4 [illegible] 5), and high (55-65) win min parameter search ranges

BAD IDEA

Wanted to conduct a large study with increased variance.

- Would have taken far too much computing time
- Shifted immediate priorities towards filtering out low performing functions after increasing variance on a single fitting window.
- Would have had ~2000 jobs taking 50 hours minimum each....
- Turns out variance idea listed would have been terrible anyway!

Starting to Select Good Functions

Focusing on searching for better seeds.

- Fitting a single window range for all functions vastly saves computing time
- Window range study to follow on reduced list of functions

Changes to Fitting/Variance:

- Use initial parameters as gaussian mean and select new parameters randomly about a gaussian with width 1% **[width = .01 * mean]**
- If fit found with better pvalue in less than a specified many fits, make that the new mean
- If fit is not found with better pvalue, pick parameters from a gaussian width of 2% [width = .01 * mean + 0.1*1*mean = **.02 * mean**]

  General form: **[width = .01 * mean + .01 * counter * mean]** -
  where counter is number of times fitting failed to find a better pvalue within a user input number of tries

New Save State Approach:

- Now fitting using a time based limit on fitting.
- Each time a better pvalue is found, write a file with best fit info
- Saves win_min, win_max, number of fits, chi2/ndof, pvalue, parameter values

Save State Workflow

python3 /sdf/group/hps/users/epeets/run/resonance_fitting/makeGlobalFitScripts_svst.py -d /sdf/group/hps/users/epeets/run/resonance_fitting/sh/ -m 50 51 1 -x 198 199 1 -F /sdf/group/hps/users/epeets/run/resonance_fitting/functions/ -B 1 -T 8 -v 1 -f 100

-T = total fitting time [hr]
-f = number of fits before increasing variance

Generating Jobs
makeGlobalFitScripts_svst.py

Fitting
global_fit_svst.py

Compiling / Organizing
- sorter_table.py for table
- `compileEmrysFiles.py` for 1D Histogram

8 Hour Study - [50, 198] MeV

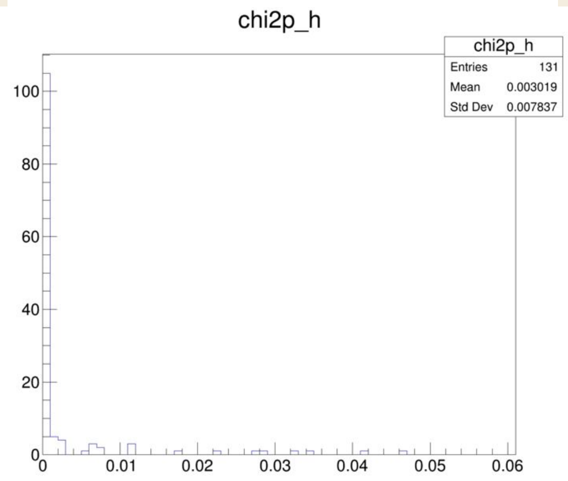
- ended up with 11 functions with pvalue >10^-2
    - **before only had 3**
- Much higher statistics for many functions
- Would like to try 0.5% variance from selected parameters before filtering

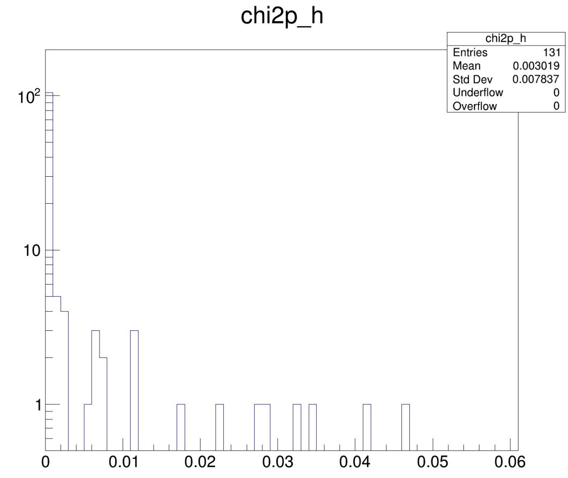
top performing 20 functions

function_table_50MeV_198MeV

Function Name	Number of Fits	Chi2/Ndf	PValue
las3_plus_las6	1702.0	1.0443	0.0460
ua23_nolin_plus_las1	1053.0	1.0457	0.0412
las2_plus_las6	10053.0	1.0480	0.0342
ua23_nolin_plus_las3	126.0	1.0487	0.0323
dj1_plus_dj1	16601.0	1.0501	0.0286
ua23_nolin_plus_las2	8124.0	1.0506	0.0275
las1_plus_ua23	2391.0	1.0527	0.0228
dj1_plus_las2	1110.0	1.0557	0.0174
dj1_plus_ua22	2704.0	1.0599	0.0117
las1_plus_ua21	2999.0	1.0600	0.0117
las3_plus_las3	953.0	1.0605	0.0111
dj1_plus_cms1	665.0	1.0640	0.0078
las1_plus_las5	4455.0	1.0645	0.0075
dj1_plus_las6	292.0	1.0654	0.0068
las1_plus_las1	2905.0	1.0659	0.0064
las1_plus_las6	1623.0	1.0661	0.0063
ua23_er_er_10_2	8557.0	1.0680	0.0052
ua23_er_er_8_4	8728.0	1.0734	0.0029
ua23_er_er_11_2	36878.0	1.0755	0.0023
ua23_er_er_1	3562.0	1.0759	0.0022

1D Pvalue Distribution





Summary / Next Steps / Discussion

1. 11 Functions found with pval $>10^{-2}$ over range [50,198] MeV
   a. Huge improvement over previous update of 0 isolated functions
   b. One more variance study needed, maybe 24 hours this time
2. Conduct optimal window range study with narrower list of functions
   a. top 10 functions (?)
3. Cam presented on reach effects found by fixing background fit parameters
   a. need to implement select functions with optimal window range

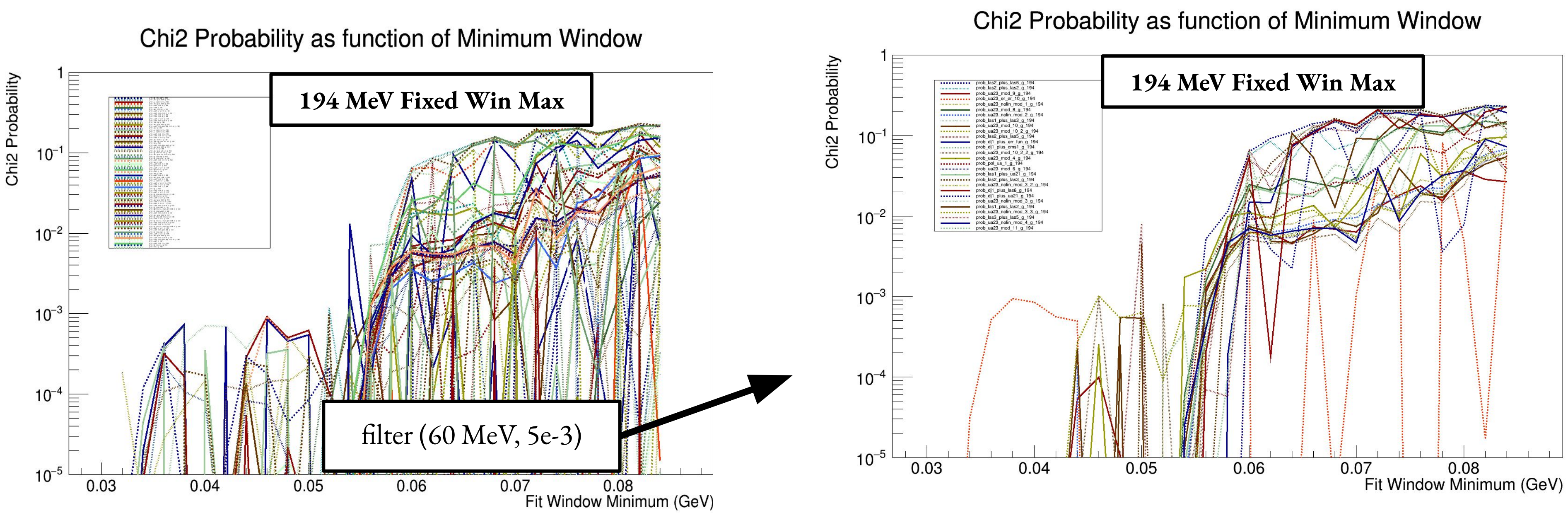
backup

- Function Filter
- Poor fit example
- Representative Good Fit
- Making global fitting scripts for each function

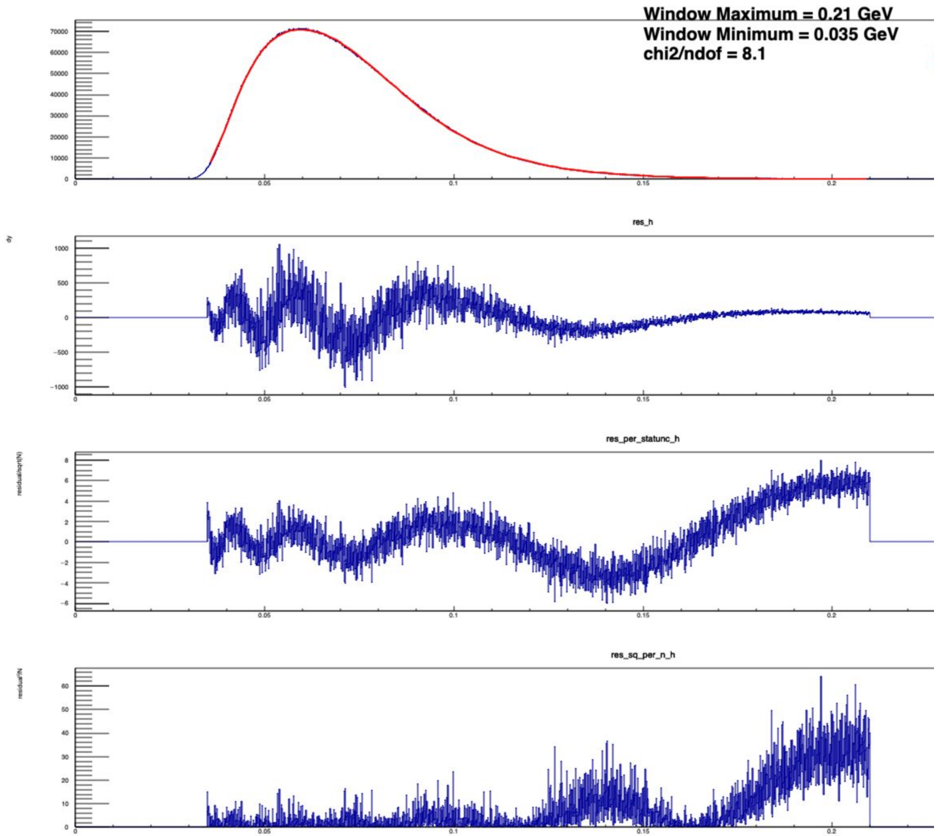
Filter Use

Use command line options to filter function results using a lower bound p-value threshold for a given window minimum.

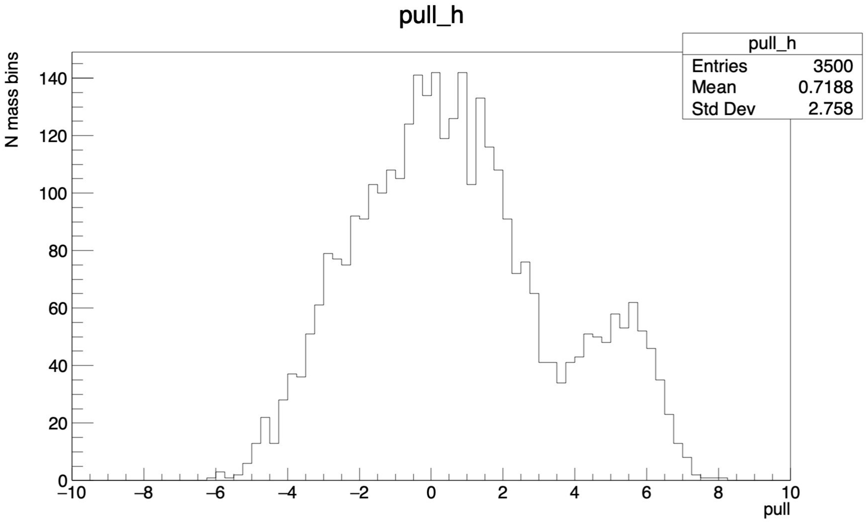
-F (specified window min) (specified pvalue threshold)



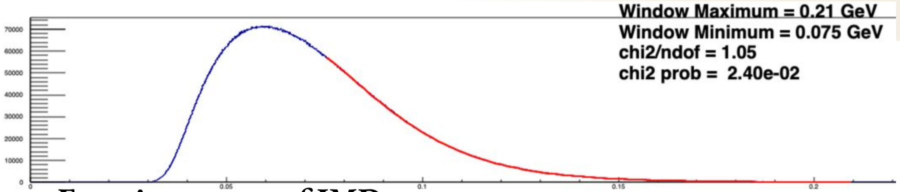
Poor Fit Example



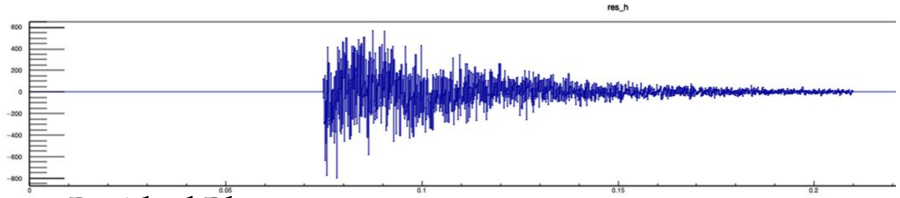
Using dijet1



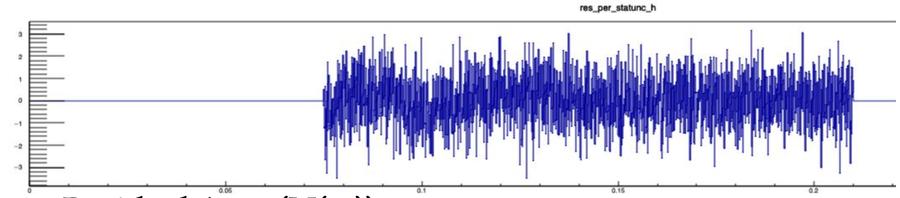
Representative “Good” Fit Using Global Fitting Tool



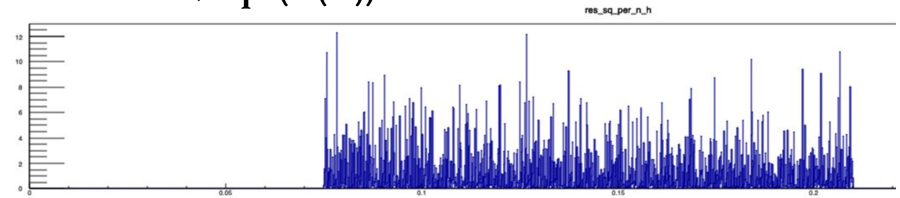
Function on top of IMD



Residual Plot

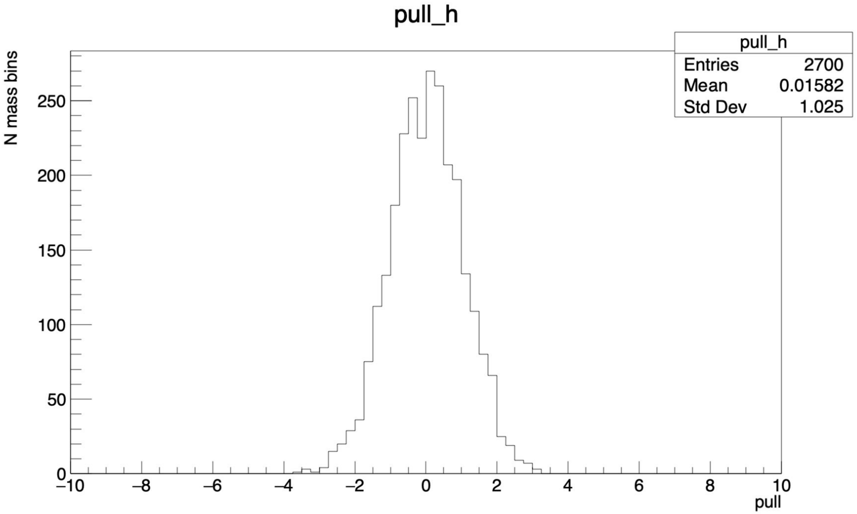


Residual / sqrt(N(m))



Residual^2 / (N(m))

- UA23 Function
- Fit Range: 75 MeV - 210 MeV
- Good ⇒ pvalue >10^{-2}



Making global fitting scripts for every function

Terminal Input

python3 /sdf/group/hps/users/epeets/run/resonance_fitting/makeGlobalFitScripts.py -d /sdf/group/hps/users/epeets/run/resonance_fitting/sh/ -m 28 40 1 -x 40 72 2 -F /sdf/group/hps/users/epeets/run/resonance_fitting/functions/

(WinMin,WinMax)

Expected Output

- resonance_fitting/sh/subJob_28_70.sh (to sbatch each function)
- resonance_fitting/sh/sh_28_70/[function.sh]

emryspeets — epeets@sdf-login04:~/HPS/run/resonance_fitting/sh/sh_28_70...

```
#!/usr/bin/scl enable devtoolset-8 -- /bin/bash
#SBATCH --ntasks=1
#SBATCH --time=24:00:00
#SBATCH --mem=2000M
#SBATCH --partition=shared
#SBATCH --job-name=fitB
#SBATCH --output=/scratch/epeets/log/cms1_28_40.txt
python3 /sdf/group/hps/users/epeets/run/resonance_fitting/global_fit_3.py -i /sdf/gro
up/hps/users/epeets/run/resonance_fitting/functions/cms1.txt -P /sdf/group/hps/users/
epeets/run/resonance_fitting/parameters/cms1.txt -m 28 40 1 -x 40 72 2 -R 0 -Q 1000 -
d /sdf/group/hps/users/epeets/run/resonance_fitting/functions/cms1_out/ -o cms1.root
~
[01] cms1.sh                                                                1,01 All
```

Automated fitting terminal input