



University of  
Massachusetts  
Amherst

# Parametrized Optimal Observable Approach to Simulation-Based Inference

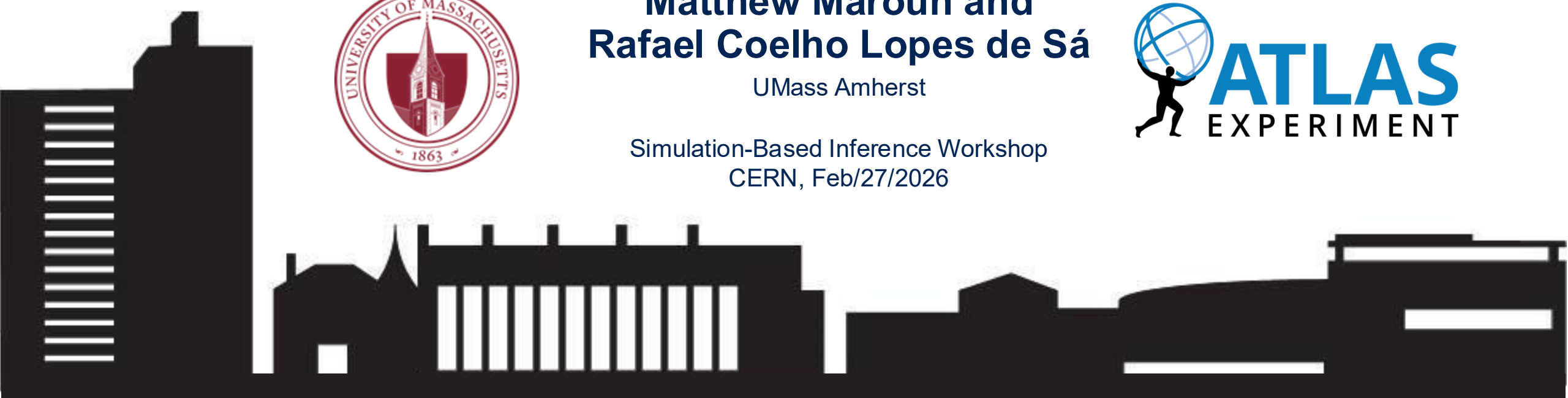
---



**Matthew Maroun and  
Rafael Coelho Lopes de Sá**

UMass Amherst

Simulation-Based Inference Workshop  
CERN, Feb/27/2026



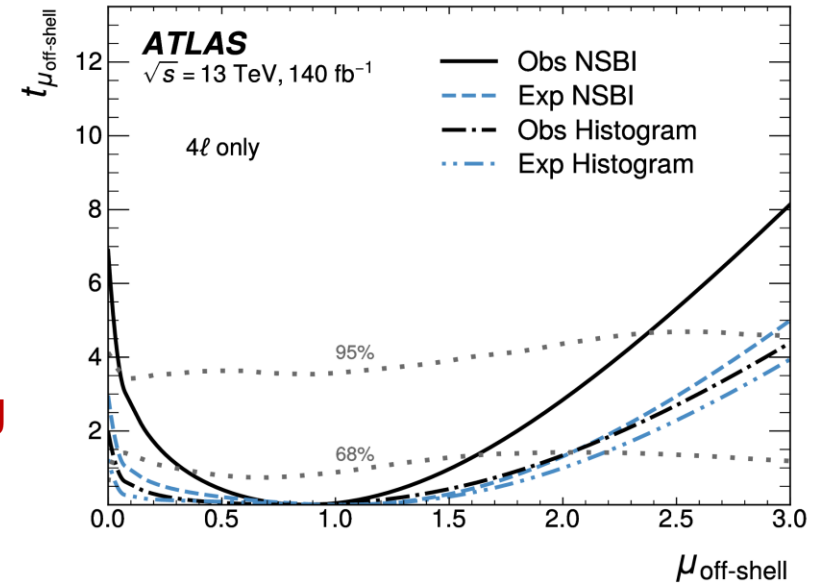
# Introduction



ATLAS presented the first full NSBI analysis with the analysis of the off-shell Higgs boson production in the  $H^* \rightarrow ZZ \rightarrow 4\ell$  final state [[Rep. Prog. Phys. 88 \(2025\) 057803](#)].

The paper presents a full SBI analysis, but several challenges were not addressed in the timeline of the publication:

- The analysis was done with JAX and not implemented in RooFit, making it a challenge to combine with other analyses in ATLAS.
- Since it is an unbinned analysis, it was challenging to define an Asimov dataset. We used a MC sample with a very large number of events for the paper. However, such large datasets are not practical in serialized workspaces used in combinations.
- An explicit Neyman Construction was done, but it relied on computationally expensive bootstrapped samples. Not very practical.



**This presentation shows our current efforts to address these challenges.**



# Implementation of NSBI in RooFit

The NSBI analysis of the off-shell Higgs boson production used a hybrid approach to estimate density ratios where:

- The dependency with respect to the POI, background normalization, and NP was parametrized
- The dependency with event kinematics is non-parametric.

The model used can be summarized as:

$$\frac{p_X(x|\mu, \alpha)}{p_{\text{ref}}(x)} = \frac{1}{v(\mu, \alpha)} \sum_X \left\{ f_X(\mu) v_X \frac{p_X(x)}{p_{\text{ref}}(x)} \left[ \prod_m G_{X,m}(\alpha_m) \right] \left[ \prod_m g_{X,m}(x_i, \alpha_m) \right] \right\}$$

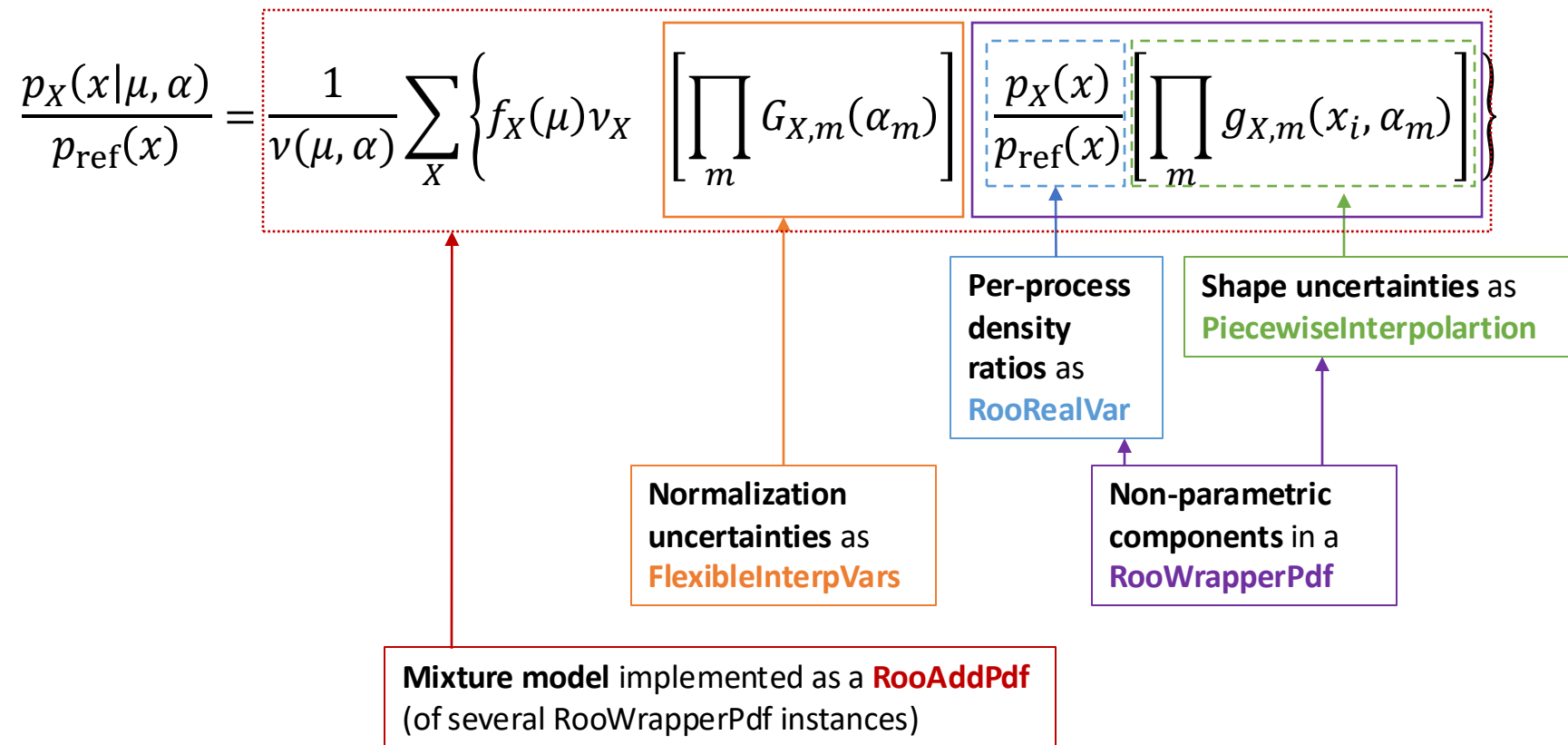
Two approaches have been attempted to introduce this formula in RooFit

1. Use the event kinematics  $x$  as observable. See example of implementation [here](#).
  - Memory efficient, but CPU-intensive since thousands of NN have to be evaluated at each point of the likelihood.
  - It may be a good application for MPI programming in large HPC systems.
2. Use the non-parametric density ratios ( $p_X/p_{\text{ref}}(x)$  and  $g_{X,m}(x)$ ) as observables
  - CPU-efficient, but it can be memory demanding.

Since we don't have many GPUs available for statistical inference (we are still using lxplus for several of these fits), we decided to take approach (2).

# Implementation of NSBI in RooFit

- The following classes were used to implement the model in (native) RooFit
- The advantage of using native RooFit classes is to benefit from recent improvements in the RooFit backend which greatly improve fitting time



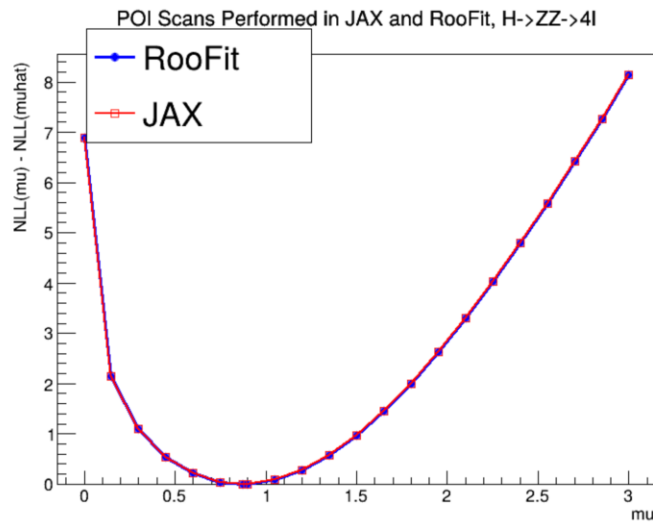


# Validation

This is an example of the RooDataSet for the off-shell production analysis using approach (2).

- NN-based observed are pre-calculated. Large number of observables, but no need to perform NN inference on the fly.
- Challenges with memory consumption and with maximum size for serialization in workspaces (improved a lot in recent versions of ROOT)

Extensively validated against published JAX version.



```
DataStore testSetSR (testSetSR)
2507) var_JET_fJvtEfficiency_ratio_qqZZ_0_down = 1.00221008972249 L(-INF - +INF) "var_JET_fJvtEffi
2508) var_JET_fJvtEfficiency_ratio_qqZZ_0_up = 1.002933156978657 L(-INF - +INF) "var_JET_fJvtEffi
2509) var_JET_fJvtEfficiency_ratio_qqZZ_1_down = 1.002933156978657 L(-INF - +INF) "var_JET_fJvtEffi
2510) var_JET_fJvtEfficiency_ratio_qqZZ_1_down = 0.997066843021343 L(-INF - +INF) "var_JET_fJvtEffi
2511) var_JET_fJvtEfficiency_ratio_qqZZ_2_up = 1.002933156978657 L(-INF - +INF) "var_JET_fJvtEffi
2512) var_JET_fJvtEfficiency_ratio_qqZZ_2_down = 0.997066843021343 L(-INF - +INF) "var_JET_fJvtEffi
2513) var_JET_fJvtEfficiency_ratio_ttV_up = 1.000000000000000 L(-INF - +INF) "var_JET_fJvtEffi
2514) var_JET_fJvtEfficiency_ratio_ttV_down = 1.000000000000000 L(-INF - +INF) "var_JET_fJvtEffi
2515) var_JET_fJvtEfficiency_ratio_EWB_up = 0.985908820483145 L(-INF - +INF) "var_JET_fJvtEffi
2516) var_JET_fJvtEfficiency_ratio_EWB_down = 1.014091179516855 L(-INF - +INF) "var_JET_fJvtEffi
2517) var_JET_fJvtEfficiency_ratio_EWSBI_up = 0.990719382911146 L(-INF - +INF) "var_JET_fJvtEffi
2518) var_JET_fJvtEfficiency_ratio_EWSBI_down = 1.009280617088854 L(-INF - +INF) "var_JET_fJvtEffi
2519) var_JET_fJvtEfficiency_ratio_EWSBI10_up = 0.987658713152590 L(-INF - +INF) "var_JET_fJvtEffi
2520) var_JET_fJvtEfficiency_ratio_EWSBI10_down = 1.012341286847410 L(-INF - +INF) "var_JET_fJvtEffi
2521) var_PRW_DATASF_ratio_SBI_up = 0.990188224313772 L(-INF - +INF) "var_PRW_DATASF_ra
2522) var_PRW_DATASF_ratio_SBI_down = 1.009811775686228 L(-INF - +INF) "var_PRW_DATASF_ra
2523) var_PRW_DATASF_ratio_B_up = 1.000000000000000 L(-INF - +INF) "var_PRW_DATASF_ra
2524) var_PRW_DATASF_ratio_B_down = 1.000000000000000 L(-INF - +INF) "var_PRW_DATASF_ra
2525) var_PRW_DATASF_ratio_S_up = 0.993613714849656 L(-INF - +INF) "var_PRW_DATASF_ra
2526) var_PRW_DATASF_ratio_S_down = 1.006386285150344 L(-INF - +INF) "var_PRW_DATASF_ra
2527) var_PRW_DATASF_ratio_qqZZ_0_up = 0.994857368142509 L(-INF - +INF) "var_PRW_DATASF_ra
2528) var_PRW_DATASF_ratio_qqZZ_0_down = 1.005142631857491 L(-INF - +INF) "var_PRW_DATASF_ra
2529) var_PRW_DATASF_ratio_qqZZ_1_up = 0.994857368142509 L(-INF - +INF) "var_PRW_DATASF_ra
2530) var_PRW_DATASF_ratio_qqZZ_1_down = 1.005142631857491 L(-INF - +INF) "var_PRW_DATASF_ra
2531) var_PRW_DATASF_ratio_qqZZ_2_up = 0.994857368142509 L(-INF - +INF) "var_PRW_DATASF_ra
2532) var_PRW_DATASF_ratio_qqZZ_2_down = 1.005142631857491 L(-INF - +INF) "var_PRW_DATASF_ra
2533) var_PRW_DATASF_ratio_ttV_up = 1.000000000000000 L(-INF - +INF) "var_PRW_DATASF_ra
2534) var_PRW_DATASF_ratio_ttV_down = 1.000000000000000 L(-INF - +INF) "var_PRW_DATASF_ra
2535) var_PRW_DATASF_ratio_EWB_up = 0.995988698990961 L(-INF - +INF) "var_PRW_DATASF_ra
2536) var_PRW_DATASF_ratio_EWB_down = 1.004011301009039 L(-INF - +INF) "var_PRW_DATASF_ra
2537) var_PRW_DATASF_ratio_EWSBI_up = 0.991782482431858 L(-INF - +INF) "var_PRW_DATASF_ra
2538) var_PRW_DATASF_ratio_EWSBI_down = 1.008217517568142 L(-INF - +INF) "var_PRW_DATASF_ra
2539) var_PRW_DATASF_ratio_EWSBI10_up = 0.990995255181680 L(-INF - +INF) "var_PRW_DATASF_ra
2540) var_PRW_DATASF_ratio_EWSBI10_down = 1.009004744818320 L(-INF - +INF) "var_PRW_DATASF_ra
2541) r_SBI = 10.184329986572266 L(-INF - +INF) "r_SBI"
2542) r_B = 10.448518753051758 L(-INF - +INF) "r_B"
2543) r_S = 1.214688420295715 L(-INF - +INF) "r_S"
2544) r_qqZZ_0 = 19.536570646318705 L(-INF - +INF) "r_qqZZ_0"
2545) r_qqZZ_1 = 0.000000000000000 L(-INF - +INF) "r_qqZZ_1"
2546) r_qqZZ_2 = 0.000000000000000 L(-INF - +INF) "r_qqZZ_2"
2547) r_ttV = 1.495809316635132 L(-INF - +INF) "r_ttV"
2548) r_EWB = 0.491356581449509 L(-INF - +INF) "r_EWB"
2549) r_EWSBI = 1.151288032531738 L(-INF - +INF) "r_EWSBI"
2550) r_EWSBI10 = 0.888289511203766 L(-INF - +INF) "r_EWSBI10"
Dataset variable "total_weight" is interpreted as the event weight
```

Per-event, per process, per-uncertainty systematic variations.

Per-event, per-process density ratios

r\_SBI = 10.184329986572266 L(-INF - +INF) "r\_SBI"  
r\_B = 10.448518753051758 L(-INF - +INF) "r\_B"  
r\_S = 1.214688420295715 L(-INF - +INF) "r\_S"  
r\_qqZZ\_0 = 19.536570646318705 L(-INF - +INF) "r\_qqZZ\_0"  
r\_qqZZ\_1 = 0.000000000000000 L(-INF - +INF) "r\_qqZZ\_1"  
r\_qqZZ\_2 = 0.000000000000000 L(-INF - +INF) "r\_qqZZ\_2"  
r\_ttV = 1.495809316635132 L(-INF - +INF) "r\_ttV"  
r\_EWB = 0.491356581449509 L(-INF - +INF) "r\_EWB"  
r\_EWSBI = 1.151288032531738 L(-INF - +INF) "r\_EWSBI"  
r\_EWSBI10 = 0.888289511203766 L(-INF - +INF) "r\_EWSBI10"

# The Asimov problem

The approach used to implement NSBI in RooFit significantly worsens the challenges with the Asimov dataset

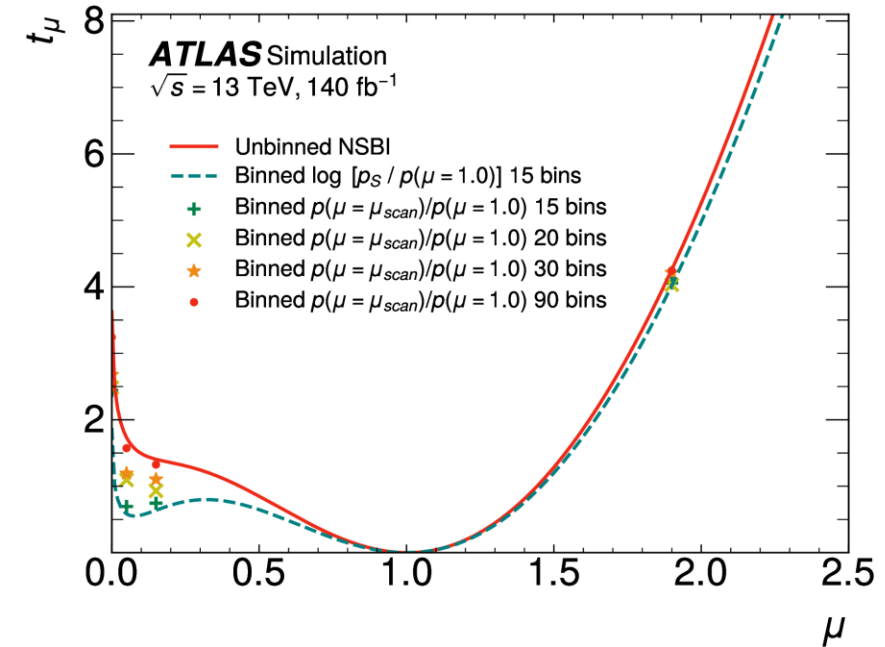
- Large MC samples RooDataSet with a very large number of observables have a large memory footprint.

Together with the off-shell analysis paper, the ATLAS collaboration published another paper describing several of the NSBI methods we developed for the results [[Rep. Prog. Phys. 88 \(2025\) 067801](#)].

In this paper we showed that the unbinned NSBI analysis can be well approximated by a binned **hypothesis  $\mu'$ -dependent** observable

$$O(x|\mu') = \frac{1}{v(\mu')} \sum_x f_X(\mu') v_X \frac{p_X(x)}{p_{\text{ref}}(x)}$$

Traditionally, several ATLAS and CMS analysis call a fixed observable “optimal”. However, these observables are only “optimal” around some hypothesis  $\mu'$ .



A (much) better “optimal” observable is obtained by using a different observable for each hypothesis.

**Which provides a simple binned Asimov approximation to the full unbinned NSBI analysis.**



# The Parametrized OO approach

Implementing hypothesis-dependent observables in RooFit is challenging (this is not a concept envisioned by the authors).

A practical implementation changing our point of view (the parametrized OO approach):

Instead of thinking about a hypothesis-dependent observable, think about a single binned observable where the yields depend on both hypothesis  $\mu'$  and POI  $\mu$

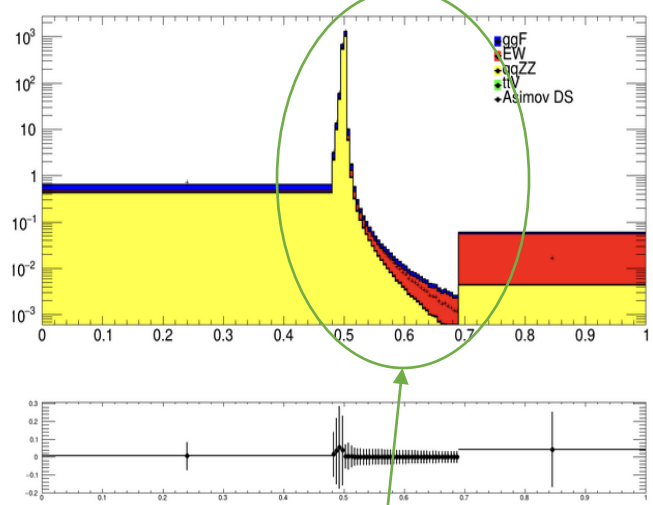
One problem is that distributions  $OO(x|\mu')$  can be very different for different values of  $\mu'$ .

Using a “score” variable which has a compact image  $0 \leq s(x|\mu') \leq 1$ , greatly improves the situation:

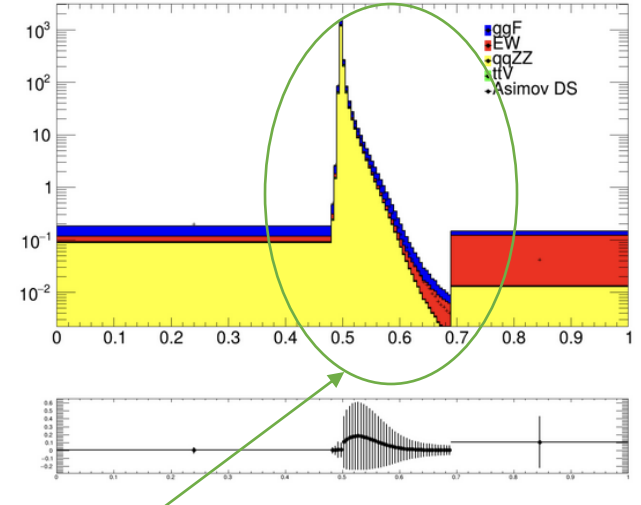
$$s(x|\mu') = \frac{OO(x|\mu')}{OO(x|\mu') + OO(x|SM)}$$

# The (binned) score distributions

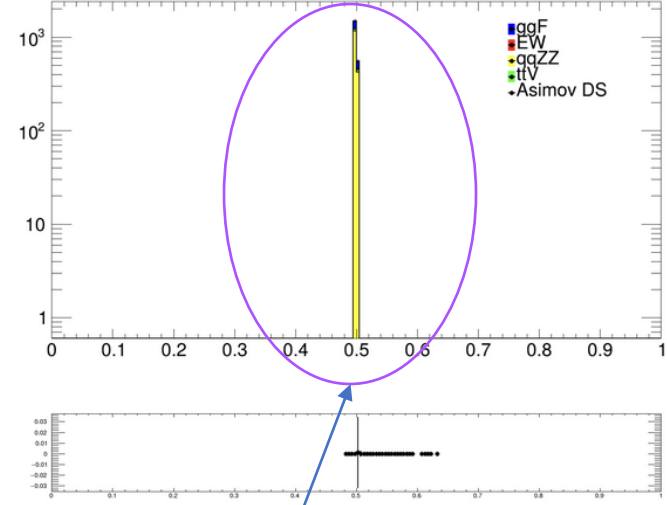
Binned NSBI Asimov,  $\mu' = 0.35$



Binned NSBI Asimov,  $\mu' = 2.30$



Binned NSBI Asimov,  $\mu' = 1.05$



After the score transformation, the different optimal observables for different hypotheses can be binned with the same bins without losing statistical power.

Near the SM value, the observable becomes degenerate (cut-and-count at  $s = 0.5$ ).

Once the binning is uniform, we can stop thinking about hypothesis-dependent observables and think about hypothesis-dependent yield (do not confuse with POI!) in each bin. And this is easy to implement in RooFit.

# A change in perspective

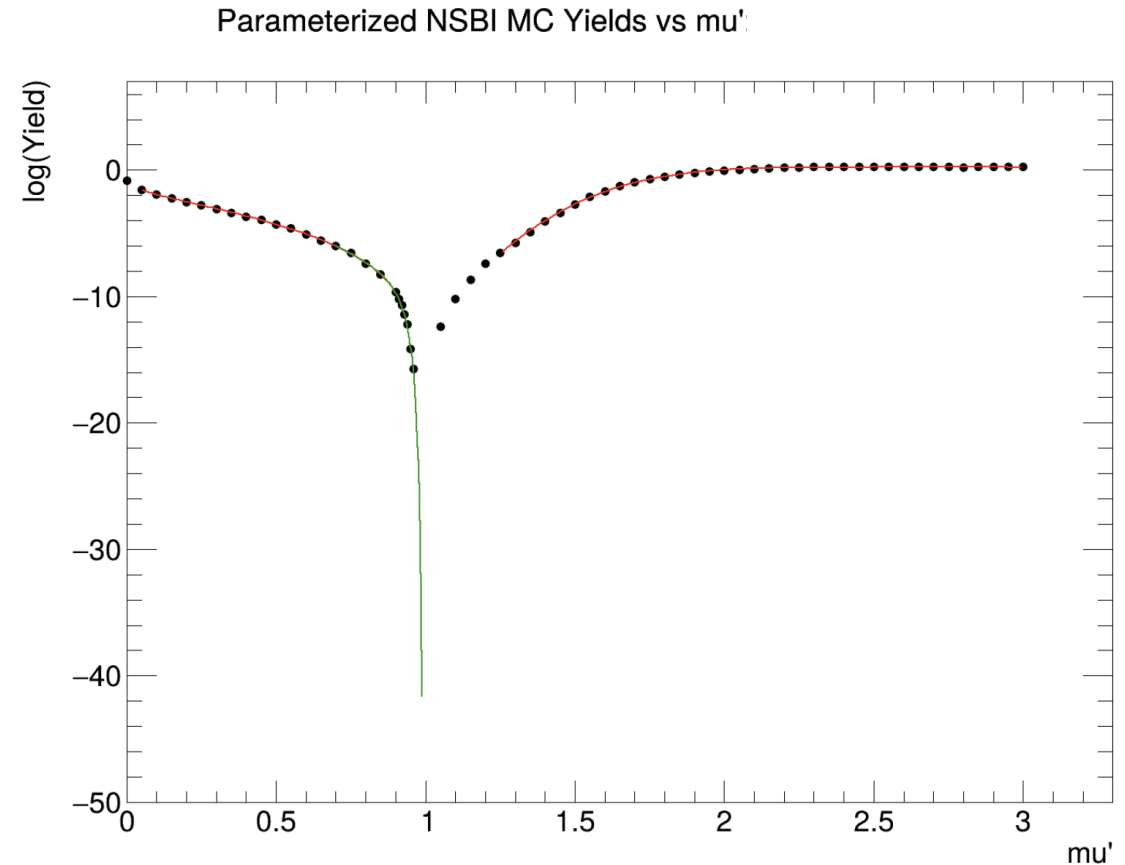
The yield in each bin can then be parametrized as a function of the hypothesis  $\mu'$  using analytical functions.

Special care is needed to avoid negative yields, but other than that, several different methods can be used:

- Polynomial functions (possibly a set, in different domains)
- Bernstein functions
- Spline interpolation
- etc...

A parametrized description of the yield as a function of the hypothesis  $\mu'$  in each bin can be achieved with very high precision.

The dependency with the POI  $\mu$  is still implemented via a mixture model (as in any other analysis).



Note that this is only feasible in low-dimensions. For high-dimensional models (STXS, several floating background normalizations, ...) some power will be lost. You cannot beat true NSBI.



# The problem with the RooDataSet

While parameterizing the yields of a binned PDF as a function of the hypothesis  $\mu'$  is not a problem in RooFit (details in the next few slides), RooFit does not have the concept of a parametrized RooDataSet.

In order to implement the Parametrized OO approach, a workaround is needed.

In a traditional binned PDF, all bins are equal. We break this hypothesis to encode the dependency of the data with the hypothesis. **Yes, we know this is ugly (it is a workaround!).** The PDF should know nothing about the data.

We define a parametrized “bin probability”  $p(\mu, \mu')$  so that the extended PDF reproduces

$$p(\mu, \mu')^{N_{\text{data}}(\mu'_{\text{ref}})} e^{-N_{\text{MC}}(\mu, \mu')} N_{\text{MC}}(\mu, \mu')^{N_{\text{data}}(\mu'_{\text{ref}})} = \frac{e^{-N_{\text{MC}}(\mu, \mu')} N_{\text{MC}}(\mu, \mu')^{N_{\text{data}}(\mu')}}{e^{-\hat{N}_{\text{MC}}(\mu')} \hat{N}_{\text{MC}}(\mu')^{N_{\text{data}}(\mu')}}$$

We would love to have a better solution

The presence of the MLE  $\hat{N}_{\text{MC}}(\mu')$  and of the data  $N_{\text{data}}(\mu')$  shows that  $p(\mu, \mu')$  can only be defined *a-posteriori*.

For (true, not data-like) Asimov datasets, that is not a limitation, since  $\hat{N}_{\text{MC}}(\mu') = N_{\text{data}}(\mu') = N_{\text{MC}}(\mu = \text{SM}, \mu')$  and it can be defined *a-priori*. One can also take  $N_{\text{data}}(\mu'_{\text{ref}}) = \text{const}$  for an even simpler construction.



# A short comment on systematic uncertainties and Parametrized OO

In principle, the up- and down-variations used to model systematic uncertainties would also need to be parametrized as a function of the hypothesis  $\mu'$ .

In practice we observe that, while the yields can vary widely, the relative variations are largely independent of  $\mu'$

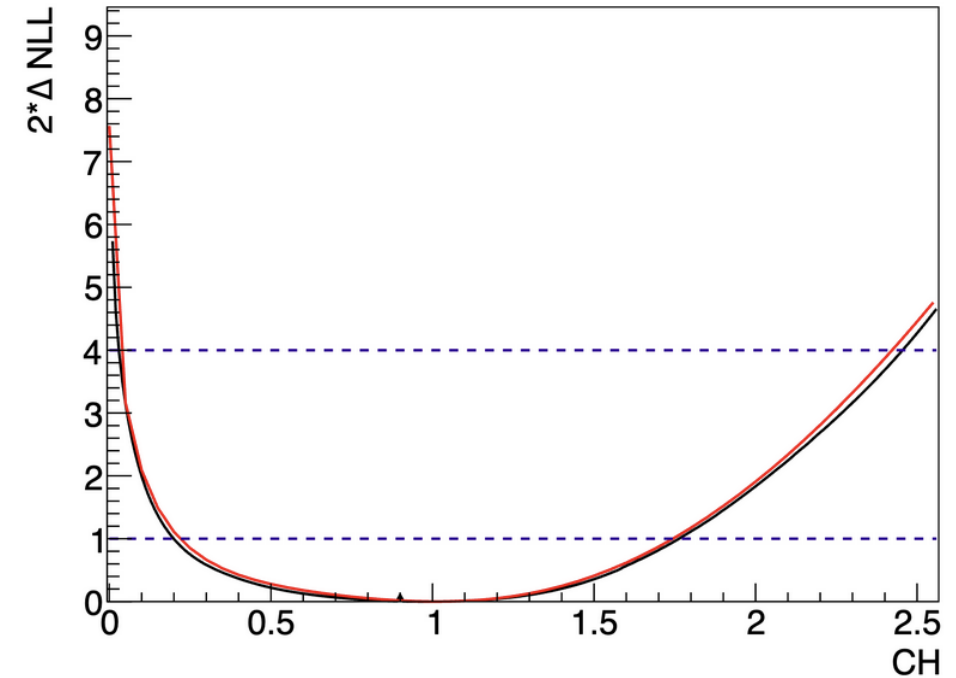
We found that using a constant relative systematic variation taken as **the envelope (in  $\mu'$ , for a reasonable range of interest of the analysis) of the relative variations** reproduce the pulls and constraints of the full NSBI result (at least for the off-shell analysis).

**If a fully automated framework for interpolation is developed, using parametrized systematics is not a problem, but we believe that choosing the envelope may be good compromise for most analysis without loss of sensitivity.**

# Implementation in RooFit and validation



1. Each bin of score histogram is described as a mixture model (RooAddPdf) of uniform PDFs (RooUniform).
  - Histograms models are formed by using RooSimultaneous
2. The probability of the RooUniform (data) is parametrized as a function of the hypothesis  $\mu'$  and of the POI  $\mu$  using RooFormulaVar.
  - The probability of the RooUniform is set by the range of the associated bin observable.
3. The coefficients of the RooAddPdf (MC yields) are parametrized as a function of the hypothesis  $\mu'$  and of the POI  $\mu$  using RooFormulaVar.
4. A fixed (reference) RooDataSet is used for all hypotheses  $\mu'$
5. Systematic variations are implemented as FlexibleInterpVars



Comparison between full NSBI Asimov (from the off-shell Higgs production analysis) and the parametrized OO Asimov construction presented here for the Higgs total width measurement.



# Advantages and Limitations

The method presented here has several advantages:

- Extremely lightweight binned approximation to full NSBI model with minimal loss of accuracy
- Construction of Asimov dataset without relying on large MC datasets. Memory usage under control.
- Neyman Construction can be performed without bootstrapping, using standard tools implemented in RooFit / RooStats.

But some caution is also needed:

- Parametrization only feasible in models with few POIs and background normalizations. Unbinned NSBI will be more powerful in high-dimensional models.
- Data and data-like Asimov workspaces must be generated *a posteriori*
- Combination with other models must be done with care, since the observables of the RooDataSet depend on non-fundamental RooFit objects

# Conclusion and Outlook



- We have presented a new parameterized optimal observable method which is a binned approximation to a full NSBI analysis.
- It addresses several challenges encountered in the ATLAS NSBI measurements of the off-shell Higgs production in the  $H \rightarrow ZZ \rightarrow 4\ell$  decay channel.
- It provides a practical, lightweight, and fast RooWorkspace implementation that can be used to perform large combinations (in our case, global Higgs coupling combinations), efficient Asimov construction, and practical Neyman Construction.
- The method is mature and we are considering an automatic workflow so that future analyses can benefit from the NSBI power without dealing with the challenges of an unbinned statistical inference.
- The method does not replace the full NSBI in all applications, but it has the potential of significantly simplify its usage in most cases.



Questions?