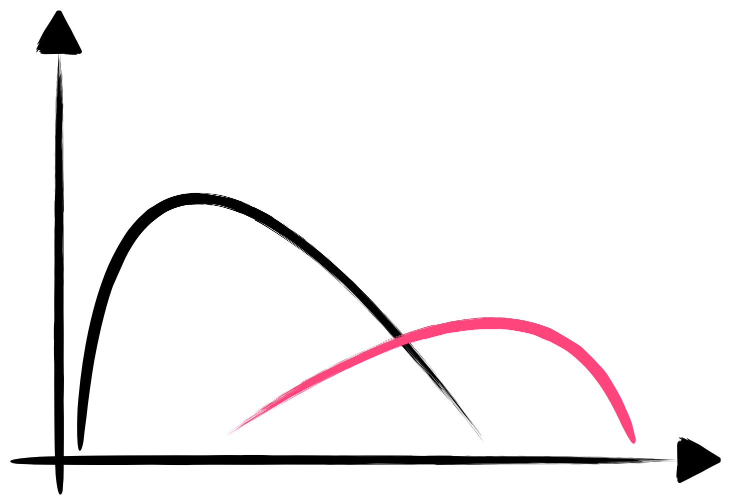
Tooling Issues for Binned nSBI Analyses with pyhf - And How to Overcome Them

Lukas Heinrich, Malin Horstmann
SBI Blueprint at CERN - 27.02.2026

In the context of this talk - What is nSBI?

Neural surrogates for the likelihood ratio r and Neyman-Pearson:

Neural classifiers * converge on the best test statistic $r = \frac{p_s}{p_B} = \frac{s}{1-s}$



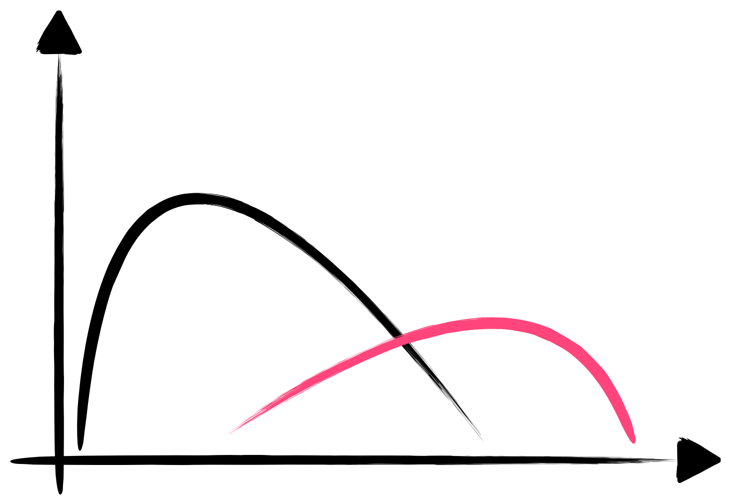
https://arxiv.org/abs/2412.01600

* hopefully

In the context of this talk - What is nSBI?

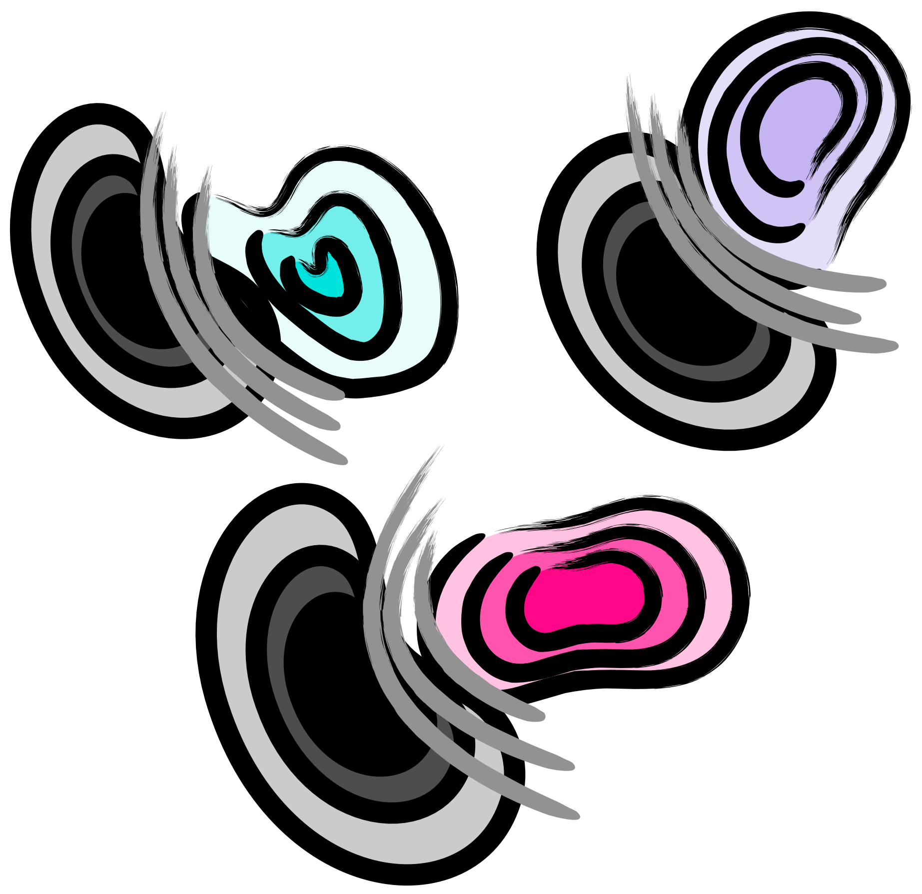
Neural surrogates for the likelihood ratio r and Neyman-Pearson:

Neural classifiers * converge on the best test statistic $r = \frac{p_s}{p_B} = \frac{s}{1-s}$



What happens if you have more than one signal p_s?

Parametrised observables help sensitivity across regions of signal space



- Binned fits over the optimal observable s
- Binned fits over the sufficient statistic $T(x)$

https://indico.cern.ch/event/1600677/contributions/6932984/

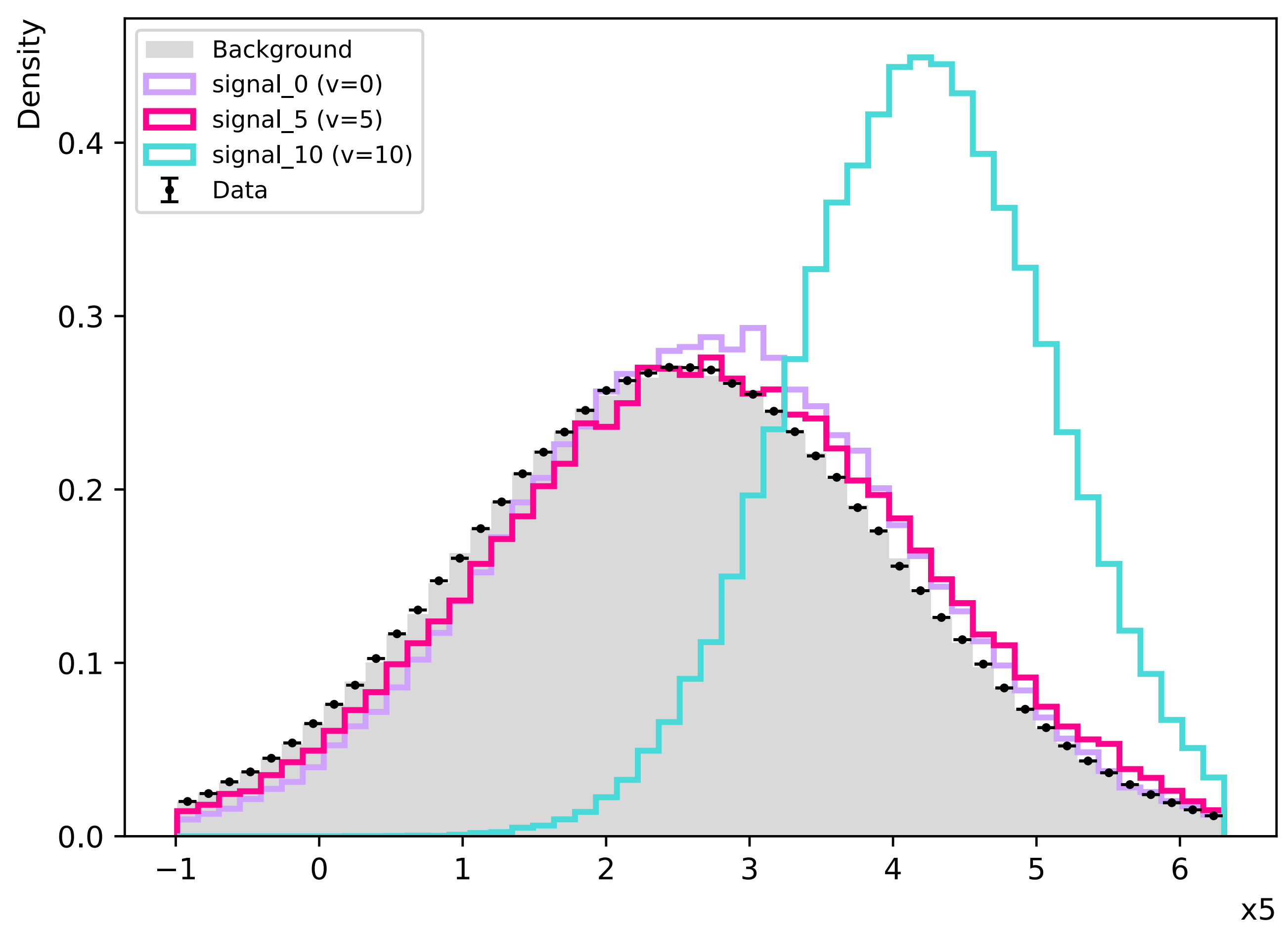
* hopefully

Our toy problem - Inspired by ggF HH

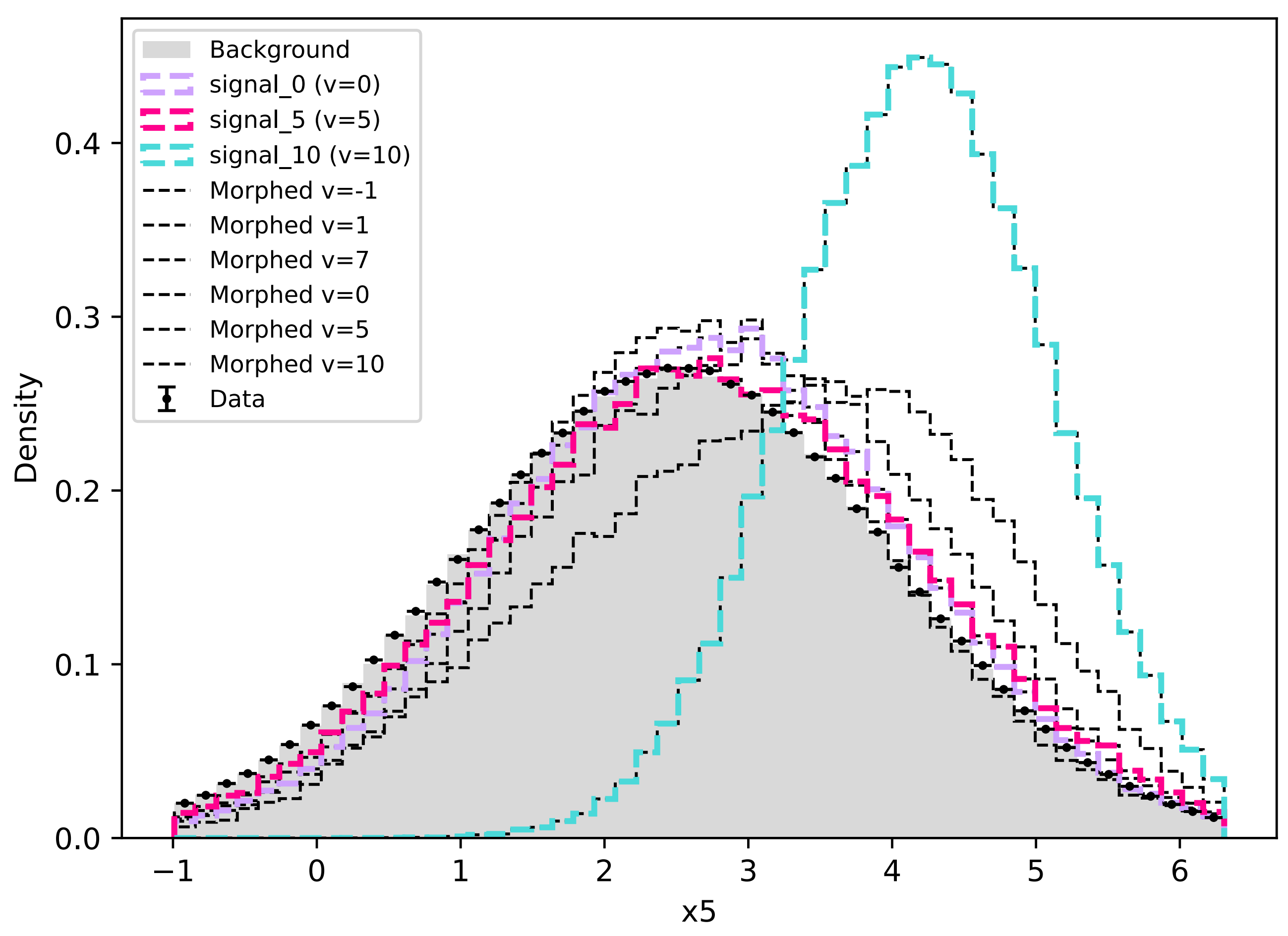
We want to separate background from a wide region of signal phase space.

Our signal phase space v can be built from the following Lagrange morphing model:

$$p(x \mid v) = \sum_{i \in [0,\, 5,\, 10]} L_i(v)\, p(x \mid v_i)$$



$\rightarrow$



Our toy problem - Inspired by ggF HH

We want to separate background from a wide region of signal phase space.

How far can we get with pyhf in building optimal fits?

The optimal observable

If fits at discrete values of v * are sufficient, the optimal observable $s(x \mid v)$ can be constructed with

$$r(x \mid v) = \frac{p(x \mid v)}{p_B(x)} = \sum_{i \in [0,\, 5,\, 10]} L_i(v) \, \frac{r(x \mid v_i)}{p_B(x)} \quad \text{and} \quad s(x \mid v) = \frac{r(x \mid v)}{1 + r(x \mid v)}$$

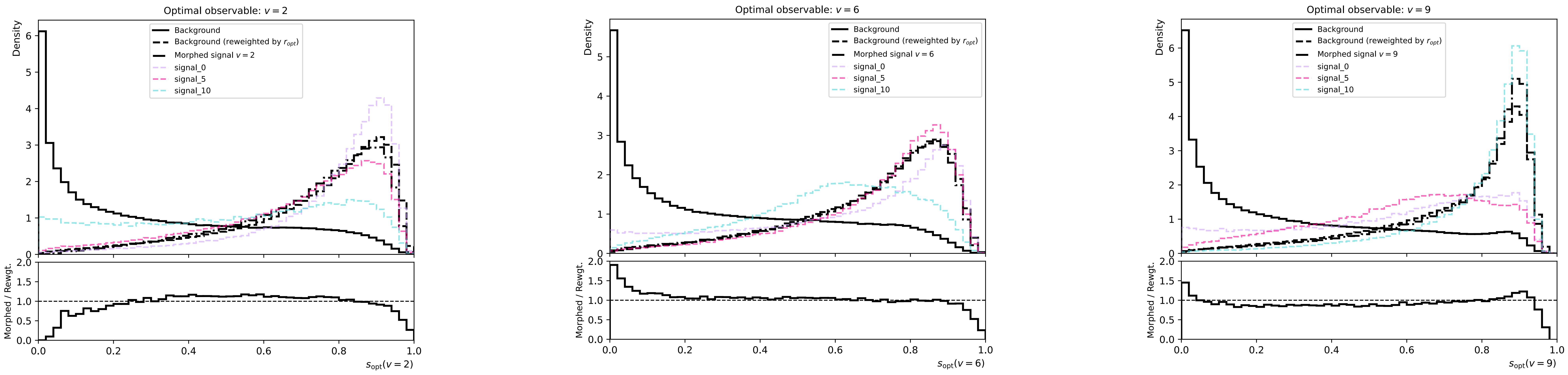
https://arxiv.org/abs/1907.10621

* I.e. setting limits on each signal strength μ_v

The optimal observable

If fits at discrete values of ν * are sufficient, the optimal observable $s(x|\nu)$ can be constructed with

$$r(x|\nu) = \frac{p(x|\nu)}{p_B(x)} = \sum_{i \in [0,\, 5,\, 10]} L_i(\nu)\, \frac{r(x|\nu_i)}{p_B(x)} \quad \text{and} \quad s(x|\nu) = \frac{r(x|\nu)}{1 + r(x|\nu)}$$



Binned fits over $s(x|\nu)$ should yield optimal performance for each ν

* I.e. setting limits on each signal strength μ_ν

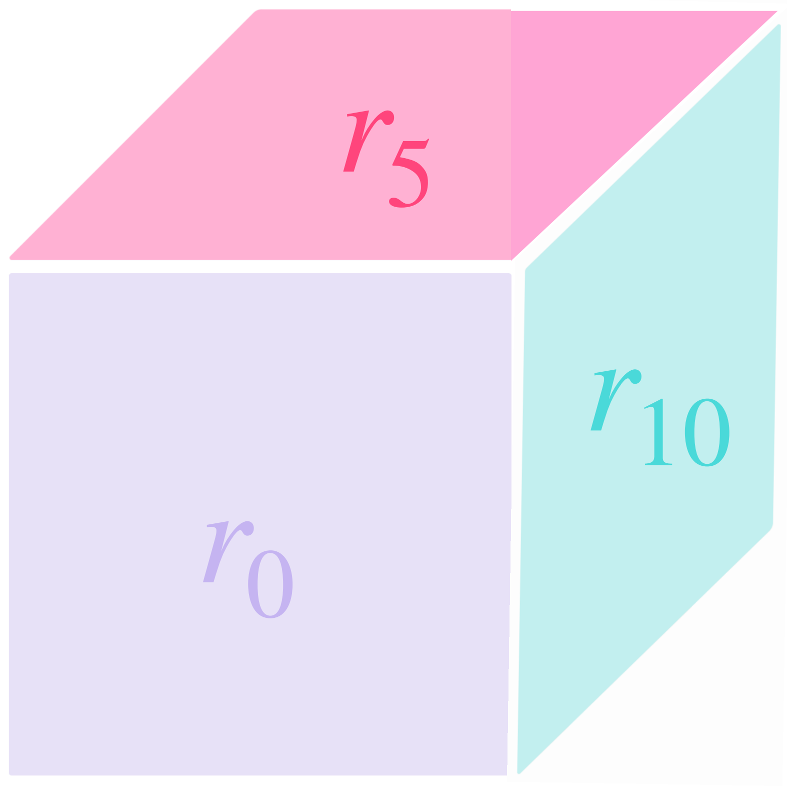
The sufficient test statistic

Fisher-Neyman factorisation:

A test statistic $T(x)$ is sufficient if $f(x\,|\,v) = h(x)\; g\left(v,\, T(x)\right)$

$T(x)$ **for our toy likelihood:**

$$L(x\,|\,v) \approx p_B(x) + \mu_v\; p(x\,|\,v) = p_B(x)\left[1 + \mu_v \sum_{i\,\in\,[0,\,5,\,10]} L_i(v)\; r(x\,|\,v)\right]$$



Thanks to **Jay** for introducing me to the concept of sufficiency!

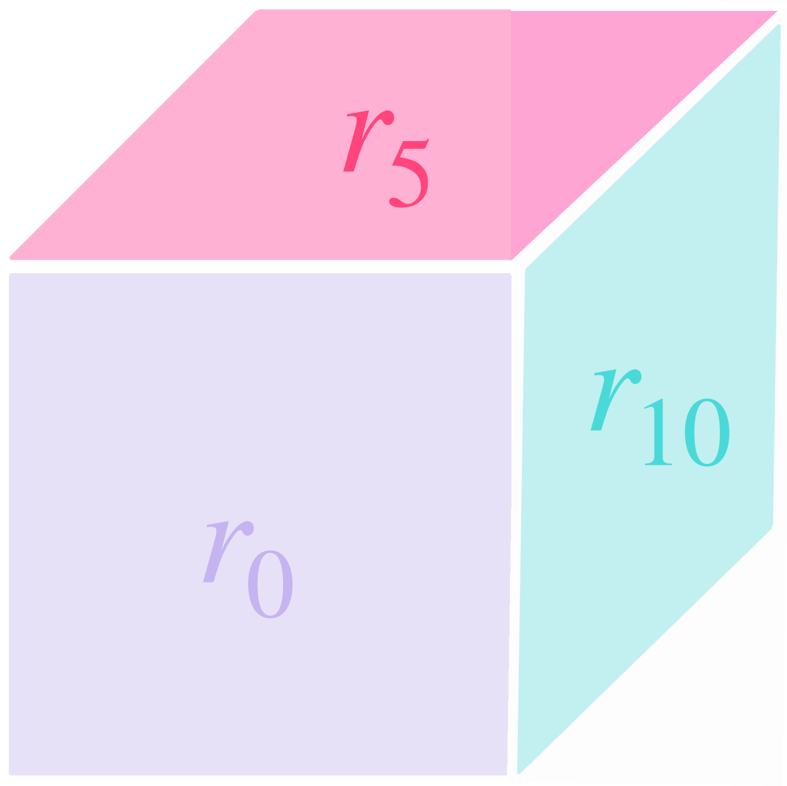
The sufficient test statistic

Fisher-Neyman factorisation:

A test statistic $T(x)$ is sufficient if $f(x\,|\,\nu) = h(x)\; g\left(\nu,\, T(x)\right)$

$T(x)$ **for our toy likelihood:**

$$L(x\,|\,\nu) \approx p_B(x) + \mu_\nu\; p(x\,|\,\nu) = p_B(x)\left[1 + \mu_\nu \sum_{i\,\in\,[0,\,5,\,10]} L_i(\nu)\; r(x\,|\,\nu)\right]$$



How do use this in a fit?

1) Just build a 3D histogram over $[r_0,\, r_5,\, r_{10}]$, unroll it and use this as template for all ν

2) Store $[r_0,\, r_5,\, r_{10}]$ and morph the optimal observable on-the-fly during fitting

Thanks to **Jay** for introducing me to the concept of sufficiency!

1) Sufficient statistic

Why this is nice to try:

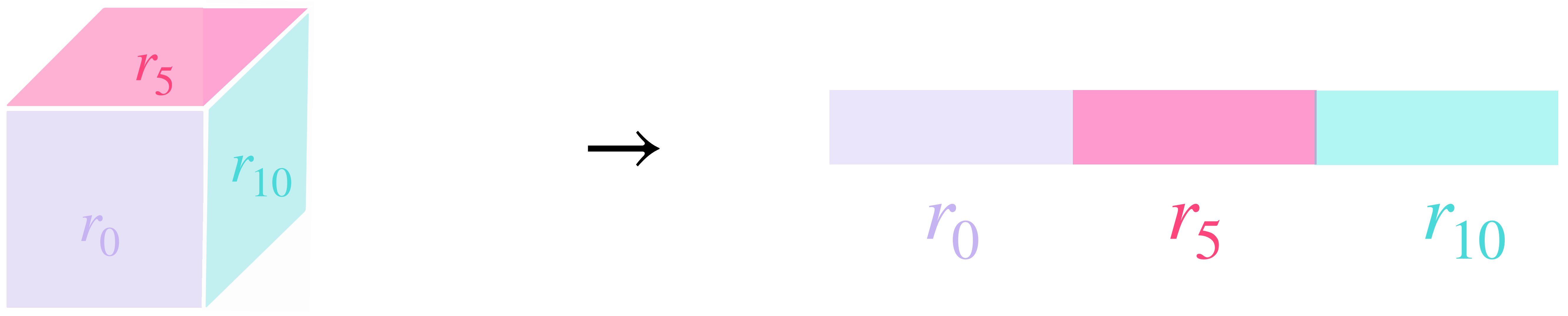
Binned observables based on classifiers are just a small * step from a classic analysis

One observable for all values ν

Issues:

* No pyhf modifier available that can morph the signal templates on the fly
→ preliminary Lagrange template-morphing modifier is available here

Loss of sensitivity from binning and using the 3D histogram



Thanks to **Jay** for introducing me to the concept of sufficiency!

2) Sufficient statistic

Why this is nice to try:

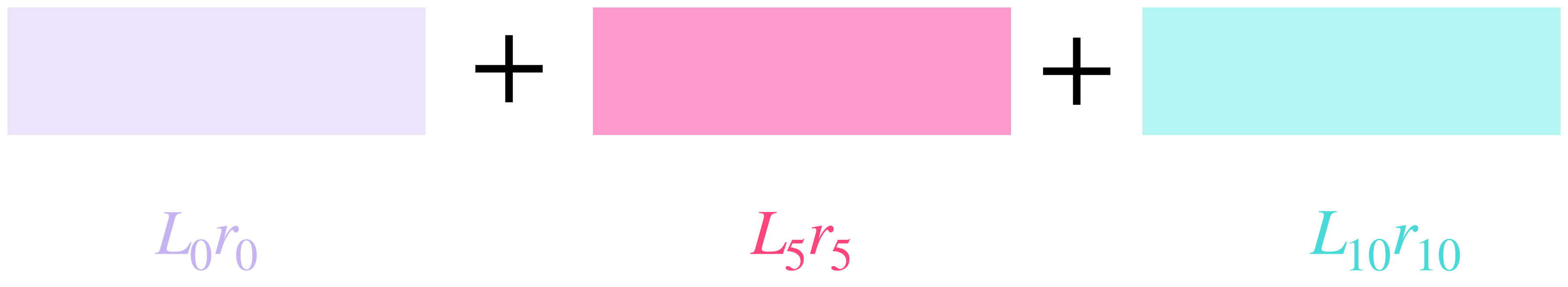
Defaults to the optimal observable, but allows for fitting ν

Issues:

The signal templates need 2 types of morphing - Morphing the signal events and the observable → Not yet implemented

We also need to store data under $r_0,\ r_5, r_{10}$ and implement data morphing → Not yet implemented

How do we ensure healthy binning for the data and background statistics if we morph to new observables on the fly?



Thanks to **Jay** for introducing me to the concept of sufficiency!

In conclusion

Good sensitivity needs converged classifiers

Discrete optimal-observable fits:

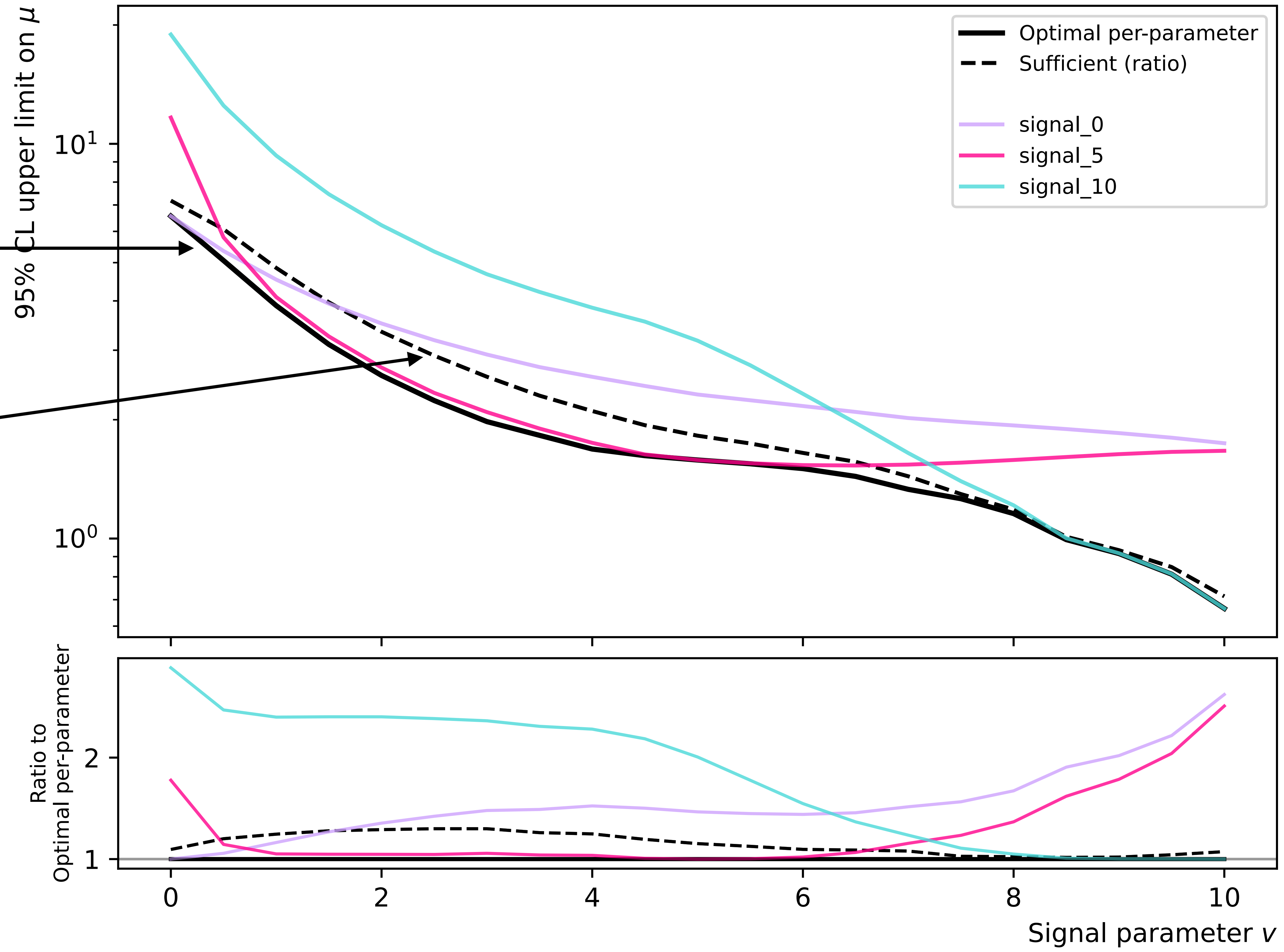
- Optimal observable for every ν
- Optimal binning for every ν
- No new fitting tools needed
- No continuous fits over ν

1) Sufficient statistic

- Continuous fits over ν
- Experimental tooling available in pyhf
- Sensitivity loss from the simple 3D histogram
- Finding a good 3D binning is hard

2) Sufficient statistic

- Continuous fits over ν
- Optimal observable for every ν
- Morning events and observable for signal and data morphing not implemented
- Binning for morphed observable?




ATLAS
EXPERIMENT
Run: 362619
Event: 524614423
2018-10-03 08:06:34 CEST
THANKS!