# Model misspecification meets ML: a HEP perspective

**Alexander Held**[1]

[1] University of Wisconsin-Madison

*PHYSTAT - Statistics meets ML*
https://indico.cern.ch/event/1407421/
Sep 12, 2024

This work was supported by the U.S. National Science Foundation (NSF) under Cooperative Agreements OAC-1836650 and PHY-2323298.

# The big picture

*take-away message:*

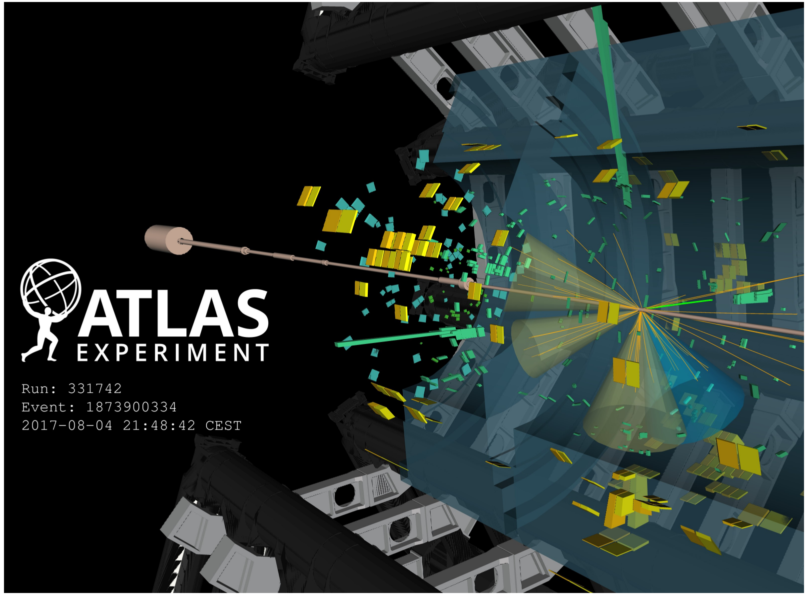
Histogram-based density estimation is a popular and effective technique in HEP.

# Big picture: turning collisions into publications

- **What we want:** statements about physical parameters $\theta$, given data $x$ collected by an experiment
  - connection: the likelihood $L_x(\theta) = p(x \mid \theta)$ — key ingredient for all subsequent statistical inference

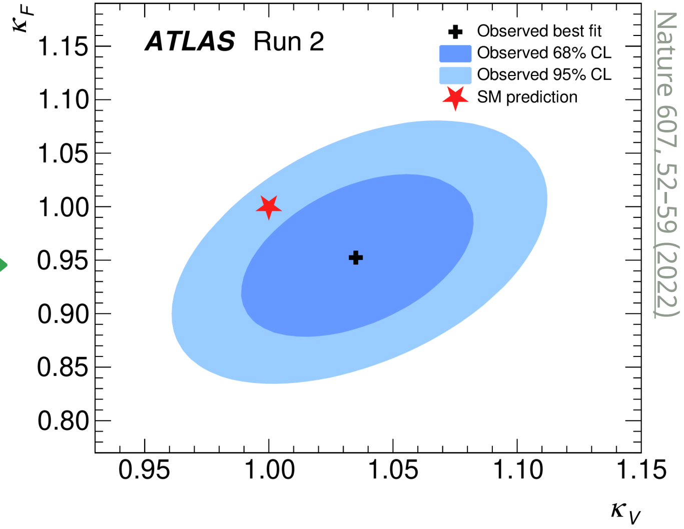
**observations $x$**

Phys. Lett. B 784 (2018) 173




$p(x \mid \theta)$

**statements about parameters $\theta$**



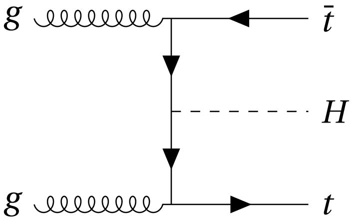

Nature 607, 52–59 (2022)

# An intractable likelihood function

- We **need** $p(x \mid \theta)$ — unfortunately this very high-dimensional **integral** is ***intractable***, **cannot evaluate** this

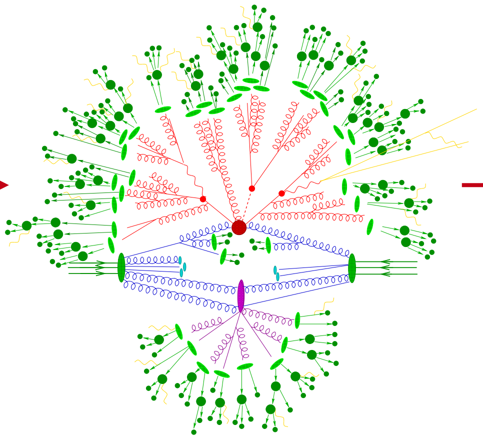
$$p\left(x \mid \theta\right) = \int \mathrm{d}z_D \mathrm{d}z_S \mathrm{d}z_P \; p\left(x \mid z_D\right) p\left(z_D \mid z_S\right) p\left(z_S \mid z_P\right) p\left(z_P \mid \theta\right)$$

**parton level** $z_P$



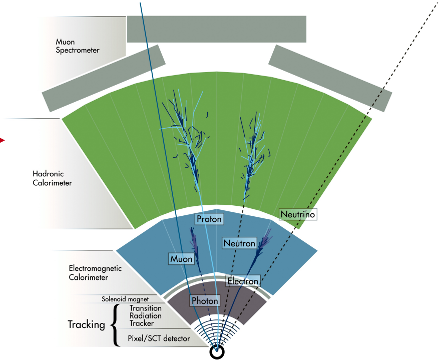

The dependence on parameters $\theta$ is here.

**parton shower** $z_S$



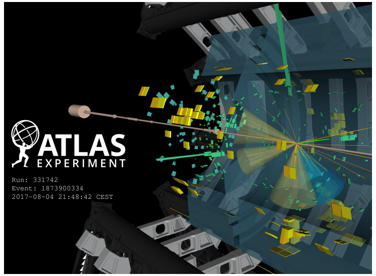
JHEP 0902 (2009) 007

**detector interaction** $z_D$




CERN-EX-1301009

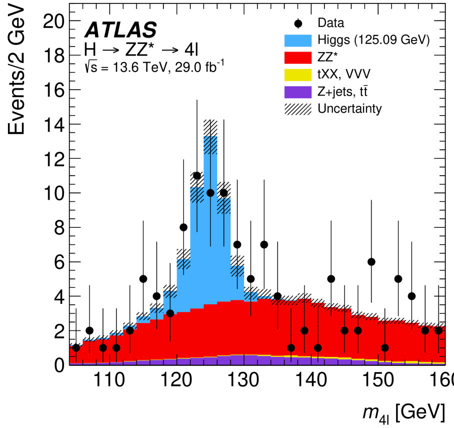
**observables** $x$




Phys. Lett. B 784 (2018) 173

# Density estimation & summary statistics

- There is one thing we *can* do: **simulate samples** $x_i \sim p(x \mid \theta)$
  - use MC samples to estimate the density $p(x \mid \theta)$, e.g. by filling histograms with the samples $x_i$
- Histograms are hit by the **curse of dimensionality**
  - number of samples $x_i$ needed scales exponentially with dimension of observation
- We use **summary statistics** to reduce dimensionality of our measurements
  - operate on objects like jets instead of detector channel responses
  - use physicists & machine learning to efficiently compress information
- **Challenge:** finding the right low-dimensional summary statistic — crucial for sensitivity




Eur. Phys. J. C 84 (2024) 78

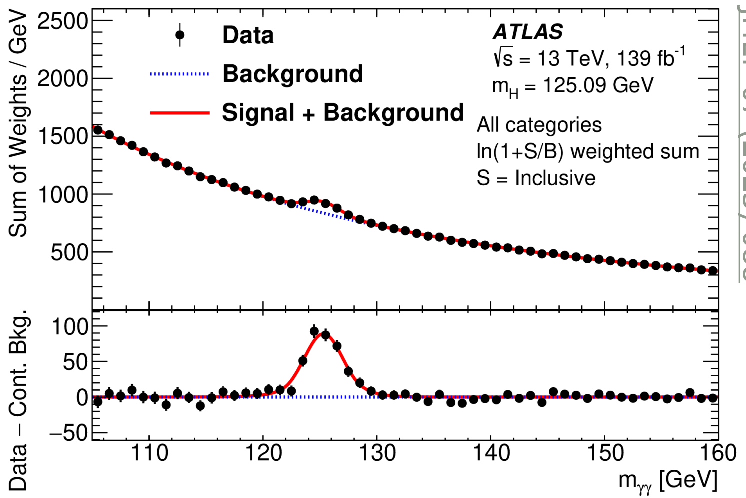
# Model building in practice: the `HistFactory` example

*take-away message:*

We are used to building statistical models with a lot of structure.
This makes them easier to develop, debug & use.

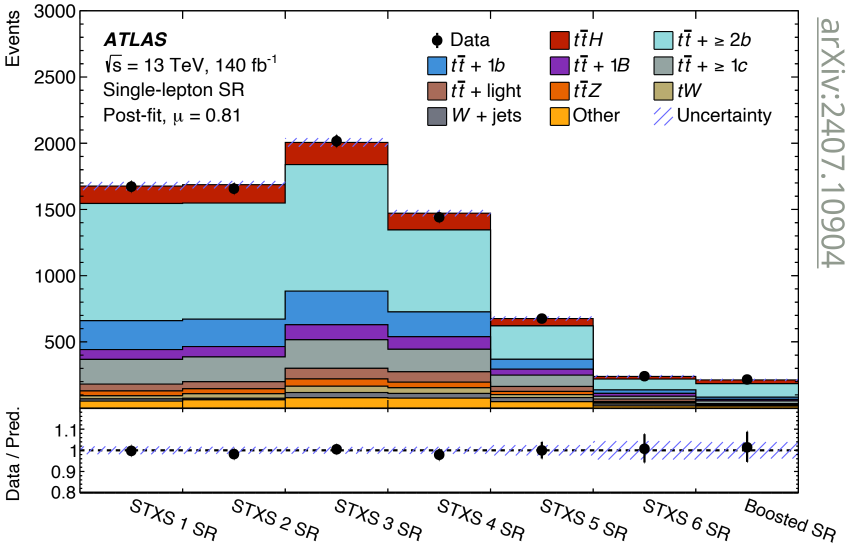
# Different styles of measurements

**analytic functions, sometimes unbinned**




JHEP 07 (2023) 088

**simulation-based template histograms, binned**

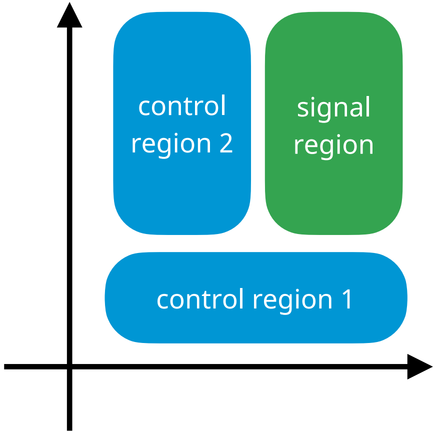



arXiv:2407.10904

- **Template histogram** approach is **more common**, will focus on this here
  - also in practice have cases without (or with only a partial) good simulation-based model

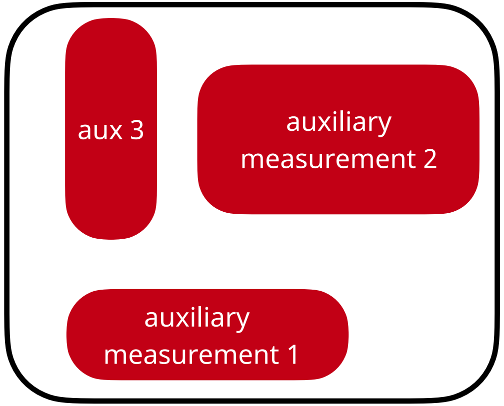
# A measurement: primary and auxiliary observables

*primary observables* $\vec{x}$




**data in our analysis**

*auxiliary observables* $\vec{a}$




**calibration measurements + theory**
(assumed to be statistically independent)

- Our models are a **combination of primary and auxiliary measurements** $p_{primary}(\vec{x} \mid \vec{\nu}\,) \cdot p_{aux}(\vec{a}\,)$
  - auxiliary: both experimental (e.g. detector calibration) and theory (e.g. changes in simulation)

# The HistFactory model: overview

- **HistFactory** is a statistical model for **binned template fits** (CERN-OPEN-2012-016 )
  - prescription for constructing probability density functions (pdfs) from small set of building blocks
  - covers a wide range of use cases (and can be extended if needed)
  - here: primary observables are $\vec{n}$, auxiliary observables are $\vec{a}$

***primary term*** ***auxiliary term***

unconstrained parameters, e.g. POI

observed data

prediction (summed over samples)

constraint term (e.g. Gaussian)

$$p(\vec{n}, \vec{a} \mid \vec{k}, \vec{\theta}) = \prod_i \mathrm{Pois}(n_i \mid \nu_i(\vec{k}, \vec{\theta})) \cdot \prod_j c_j(a_j \mid \theta_j)$$

auxiliary data, e.g. from calibration measurement

constrained nuisance parameters

product over all bins

# The model prediction: $\nu_i(\vec{k}, \vec{\theta})$

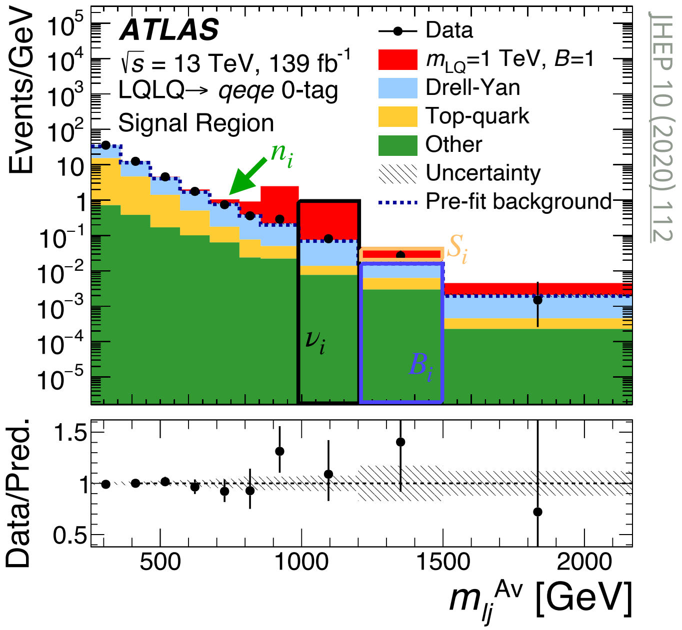
- The **prediction** in each bin is a **sum of all contributing samples,** e.g. $\nu_i = \mu \cdot S_i(\vec{\theta}) + B_i(\vec{\theta})$
  - template histograms are obtained from our simulator chain
  - samples correspond to different kinds of collision processes
  - nuisance parameters $\vec{\theta}$ affect the model prediction

observed data

prediction (summed over samples)

$$p(\vec{n}, \vec{a} \mid \vec{k}, \vec{\theta}) = \prod_i \mathrm{Pois}(n_i \mid \nu_i(\vec{k}, \vec{\theta})) \cdot \prod_j c_j(a_j \mid \theta_j)$$

unconstrained parameters, e.g. POI



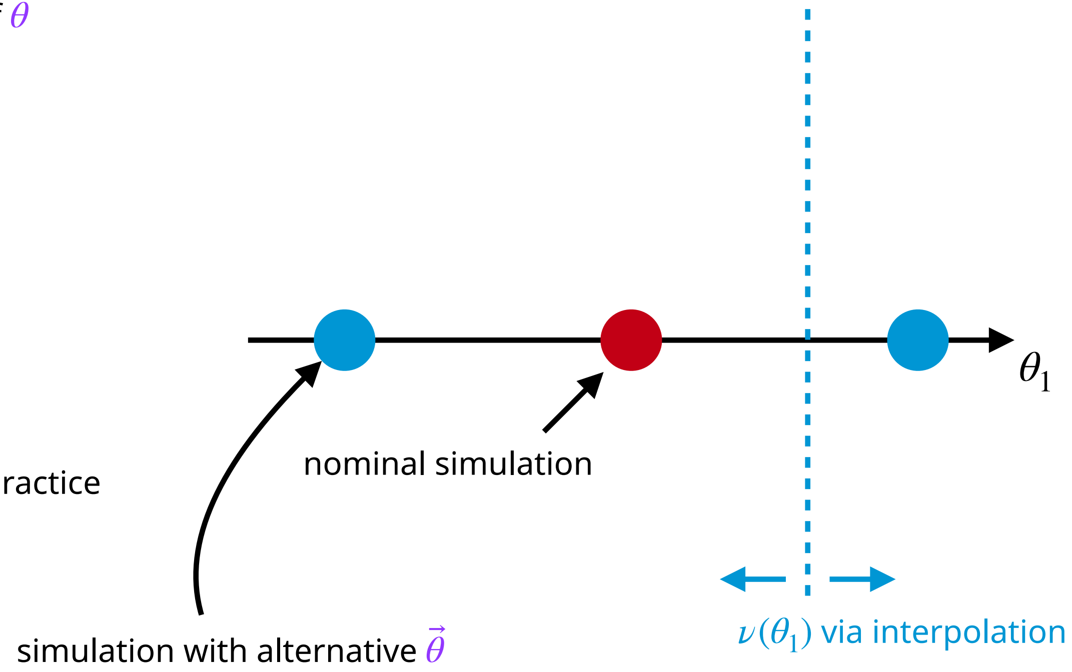
# Systematic variations

- Need to model $\nu(\vec{k}, \vec{\theta})$ for any value of nuisance parameters $\vec{\theta}$ encoding systematic uncertainties

- **Ideal case:** just run simulator for any value of $\vec{\theta}$
  - not computationally feasible in practice

- **Instead:** pick some values & **interpolate**
  - in practice we use on-axis variations
  - variations typically are “one at a time”

- Lots of **assumptions** here that we rely on in practice
  - where to simulate
  - interpolation choice
  - effects factorize



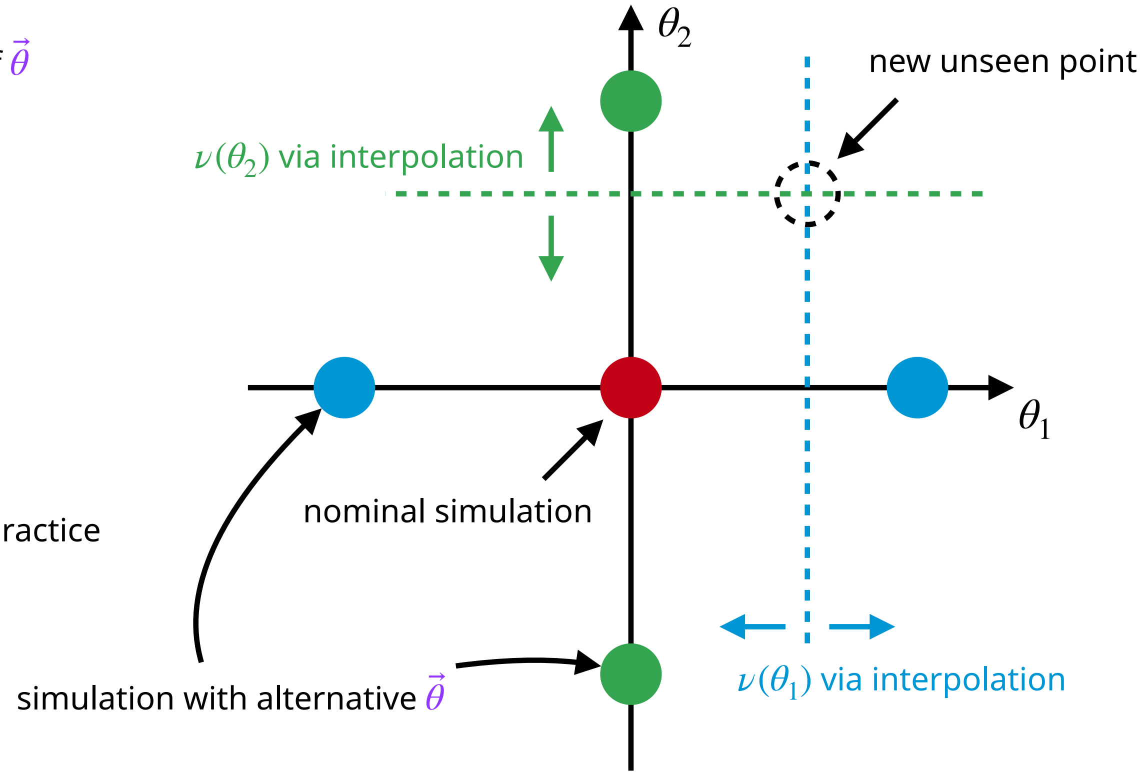
# Systematic variations

- Need to model $\nu(\vec{k}, \vec{\theta})$ for any value of nuisance parameters $\vec{\theta}$ encoding systematic uncertainties

- **Ideal case:** just run simulator for any value of $\vec{\theta}$
  - not computationally feasible in practice

- **Instead:** pick some values & **interpolate**
  - in practice we use on-axis variations
  - variations typically are "one at a time"

- Lots of **assumptions** here that we rely on in practice
  - where to simulate
  - interpolation choice
  - effects factorize

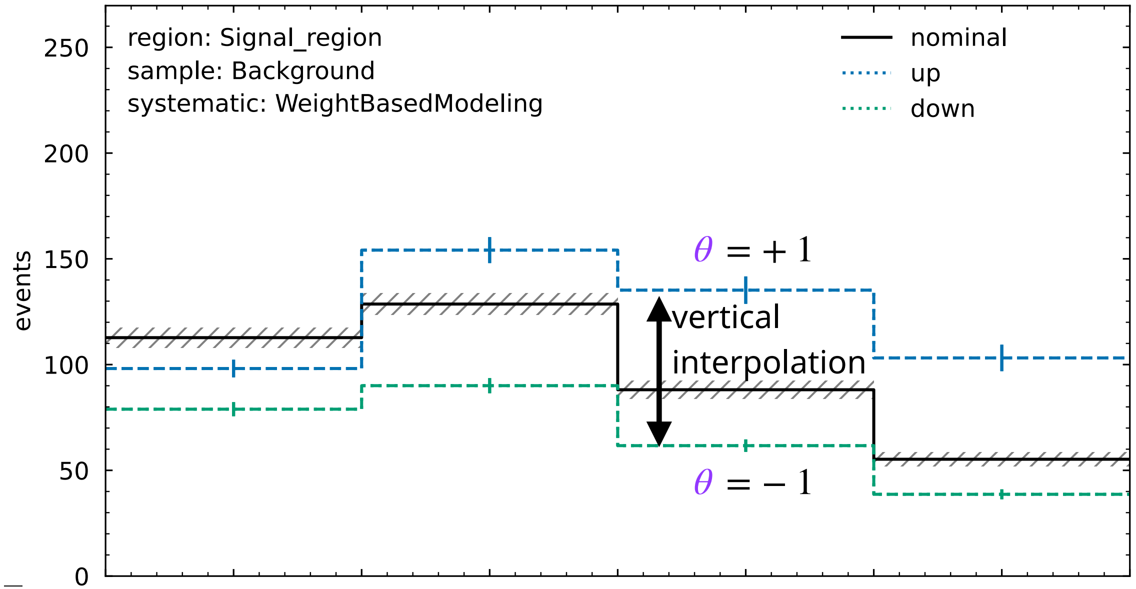


# Interpolating between points

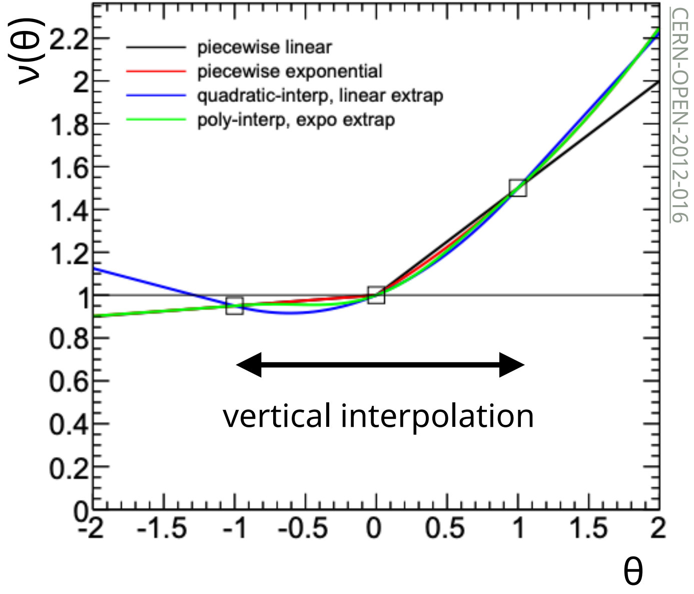
interpolation approach is technically relatively simple
➡ limit risk of surprises
➡ "warm fuzzy feeling" (Jesse Thaler's talk)

- Use model prediction $\nu_i(\vec{k}, \vec{\theta})$ for three points $\theta$, **interpolate to generalize**
  - interpolation is typically "vertical", other approaches exist (but more specialized)
  - note: information about statistical uncertainties in varied templates is lost here (arXiv:1809.05778)

**toy example: distributions for $\theta = -1, 0, +1$**




**interpolation in one bin**

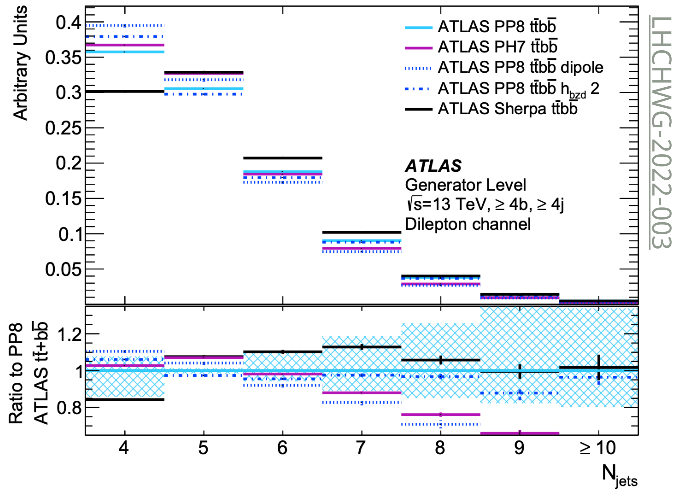


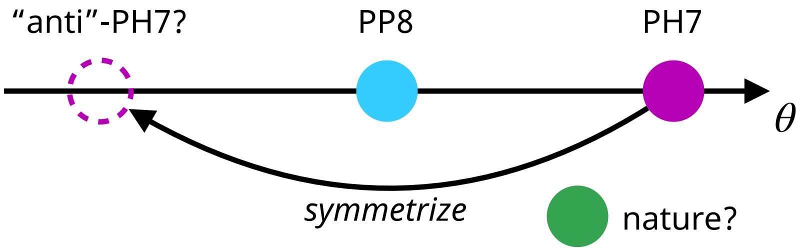
# Complication: two-point systematics

two-point systematics are inherently problematic and deserve special attention

- Sometimes have cases where **variations in simulator chain are discrete**
  - e.g. choice of one simulator vs alternative
- Typical treatment: **interpolate to treat as continuous, symmetrize**
  - lots of assumptions here, but need to make a choice to profile
- Especially **tricky to deal with** when these play a large role
  - concerns about overly constraining uncertainty of nuisance parameter
  - best-fit model prediction may lie away from both choices

*modeling choices for main background of ttH(bb)*



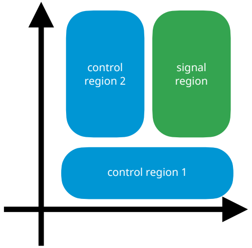
# The `HistFactory` model: structure

structure helps with tooling and with debugging

- **`HistFactory`** models are **highly structured**

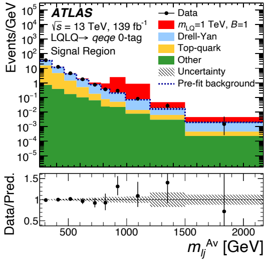
**channels**

subsets of data



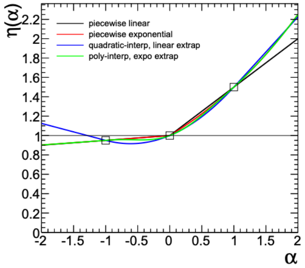

**samples**

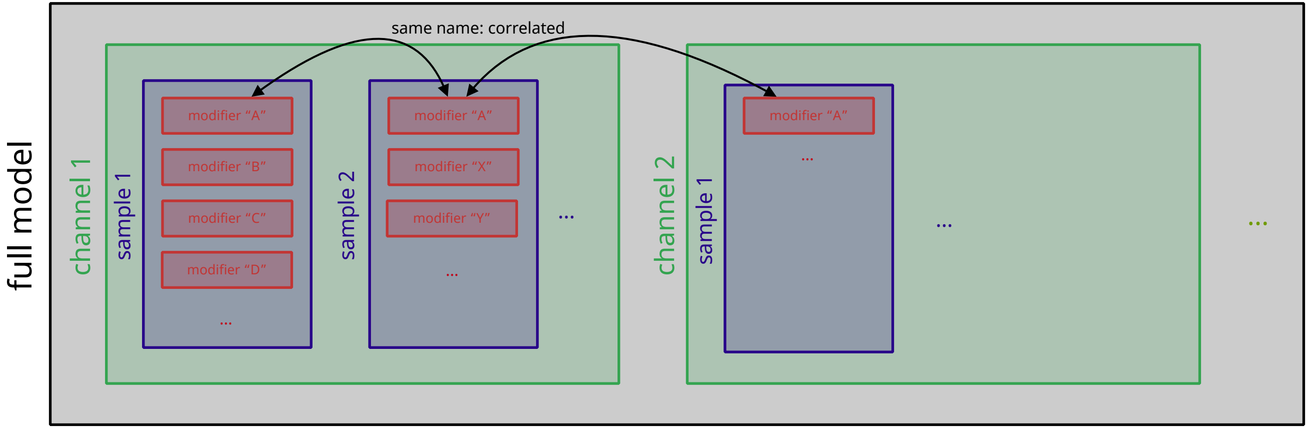
different contributions to a channel




**modifiers**

acting on the samples



# Physics analysis design & ML / AI

*take-away message:*
Analysis design is an iterative process, often guided by mismodeling concerns.
ML unlocks many capabilities but can require special consideration.

Despite the connotations of machine learning and artificial intelligence as a mysterious and radical departure from traditional approaches, we stress that machine learning has a mathematical formulation that is closely tied to statistics, the calculus of variations, approximation theory, and optimal control theory.

[PDG ML review by Cranmer, Seljak, Terao]

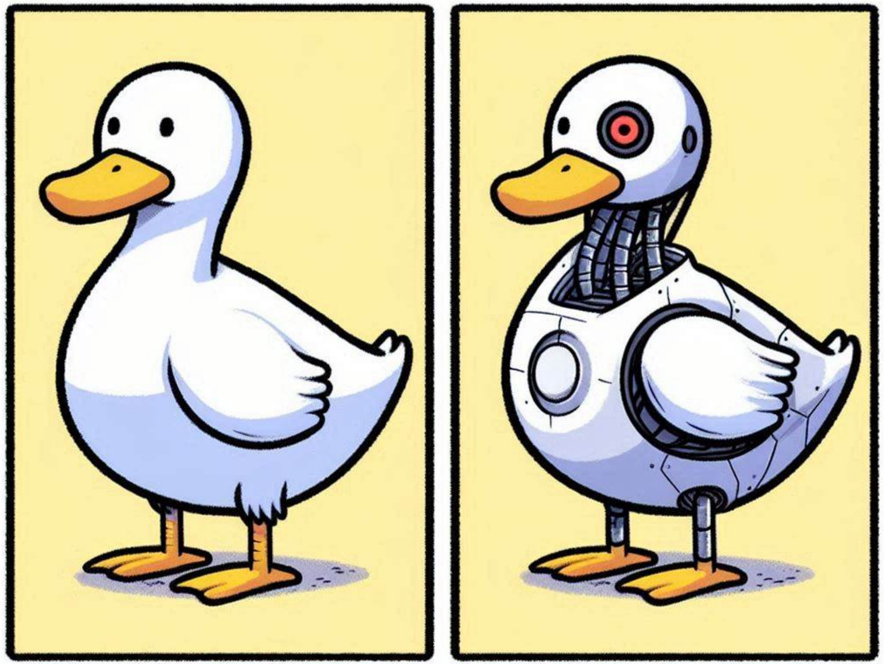
# Modeling ducks

What is "good enough"?

- We know our simulators are imperfect: just need them to be **good enough** for our specific needs

*If it looks like a duck, swims like a duck, and quacks like a duck, then it probably is a duck.*

*[If it looks like data, it's a sufficiently good simulator?]*

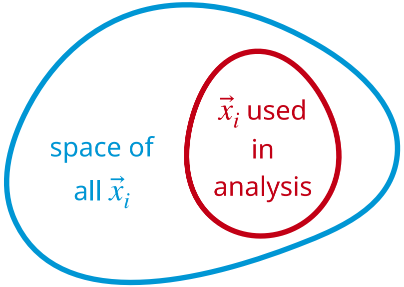


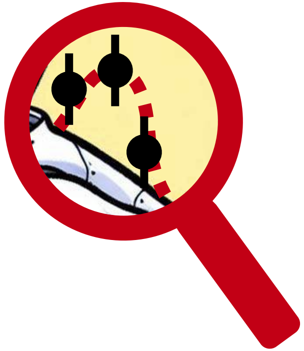
[DALL·E 3 take on the topic]

# Model misspecification & analysis design

- We have a lot of **great simulators** — which we also sometimes **push to their limits**
  - may not always trust samples from simulators to model the full joint distribution $\vec{x}_i \sim p(\vec{x} \mid \theta)$

- **In practice**
  - restrict to subset of $\vec{x}$ space / select only specific events
  - use specific and few summary statistics
    - ensure good modeling, often by visual inspection*
  - many detailed design choices that vary by analysis

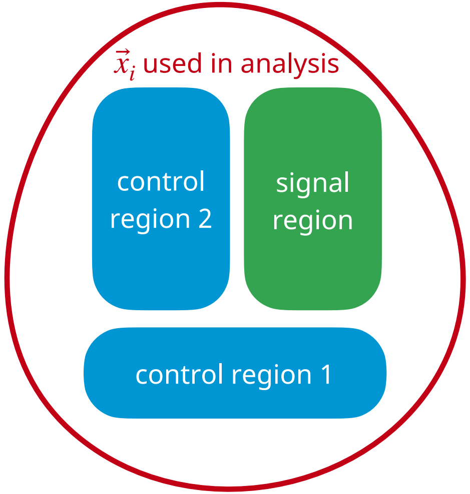


* not uncommon to regularly look at 100s of 1d histogram stacks

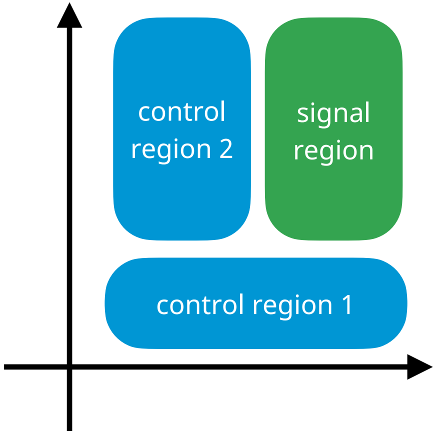
# An iterative process

- Designing an analysis is an **iterative process** with **interconnected decisions** to be made
    - which subset of $\vec{x}$ space / events do I use
    - which summary statistics / kinematic observables do I use
    - which uncertainty model is suitable
    - conscious choice how to design signal / control regions
        - blind analysis, validation of observables



# Examples requiring further model updates

- **"Constraining" nuisance parameters**: primary observables allow better measuring of nuisance parameters
  - general concern: may underestimate uncertainties due to (local?) model misspecification
  - try to locate & understand source of effect
    - traditional setup: usually analysis split up into "regions" / "channels"
    - neural SBI & other ML methods: may want to consider similar splits
  - typical operation: replace single nuisance parameter by multiple parameters
    - may imply another round of training for SBI setups



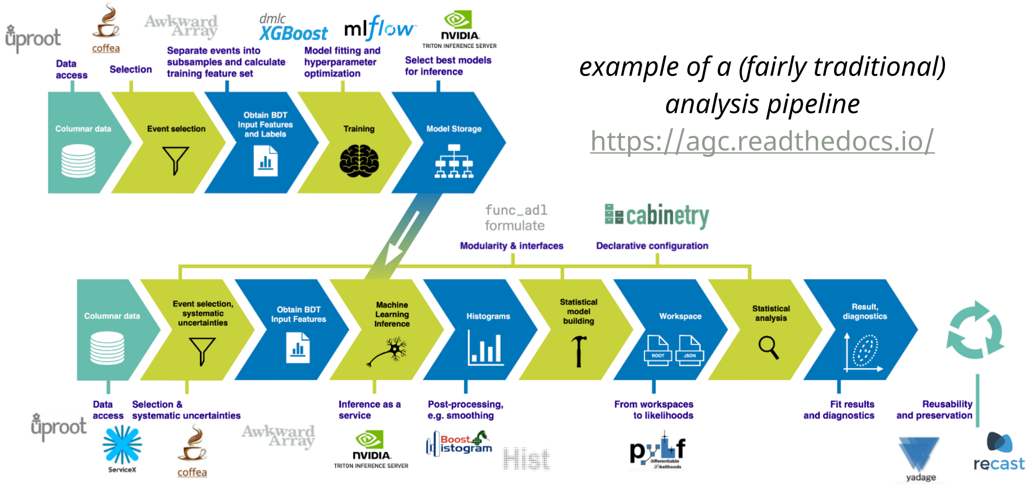

Special consideration is given to the correlation of modelling uncertainties across different $p_{\mathrm{T}}^{H}$ bins, in order to provide the fit with enough flexibility to cover background mismodelling without biasing the signal extraction. The $t\bar{t} + \geq 1b$ NLO matching uncertainty is shown to depend on $p_{\mathrm{T}}^{H}$ and is therefore decorrelated across $p_{\mathrm{T}}$ bins in the SRs.

JHEP 06 (2022) 97

# Analysis pipeline and tooling

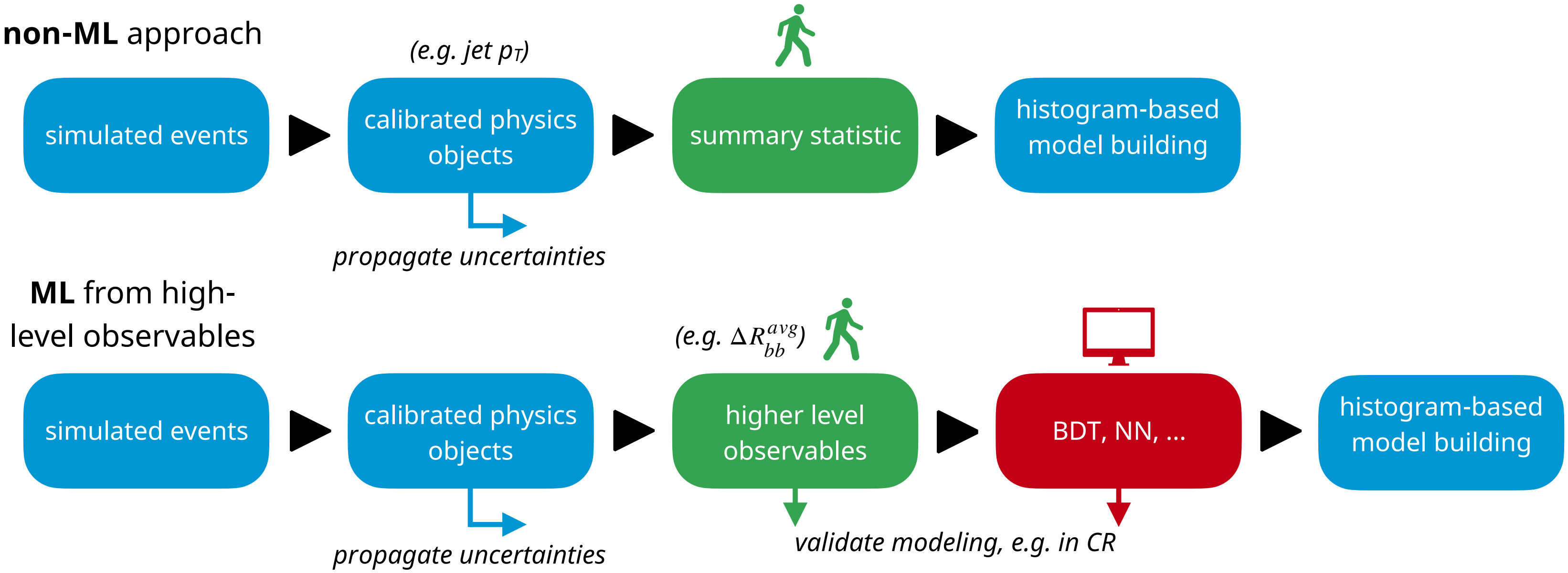
- **Fast turnaround** to develop analysis and adjust when changes are needed is important to speed up publication
  - is a new & expensive ML model training needed?
  - do multiple people need to coordinate workflow steps?

- **Good tooling should not be an afterthought**: it is crucial to help make your great ML ideas accessible



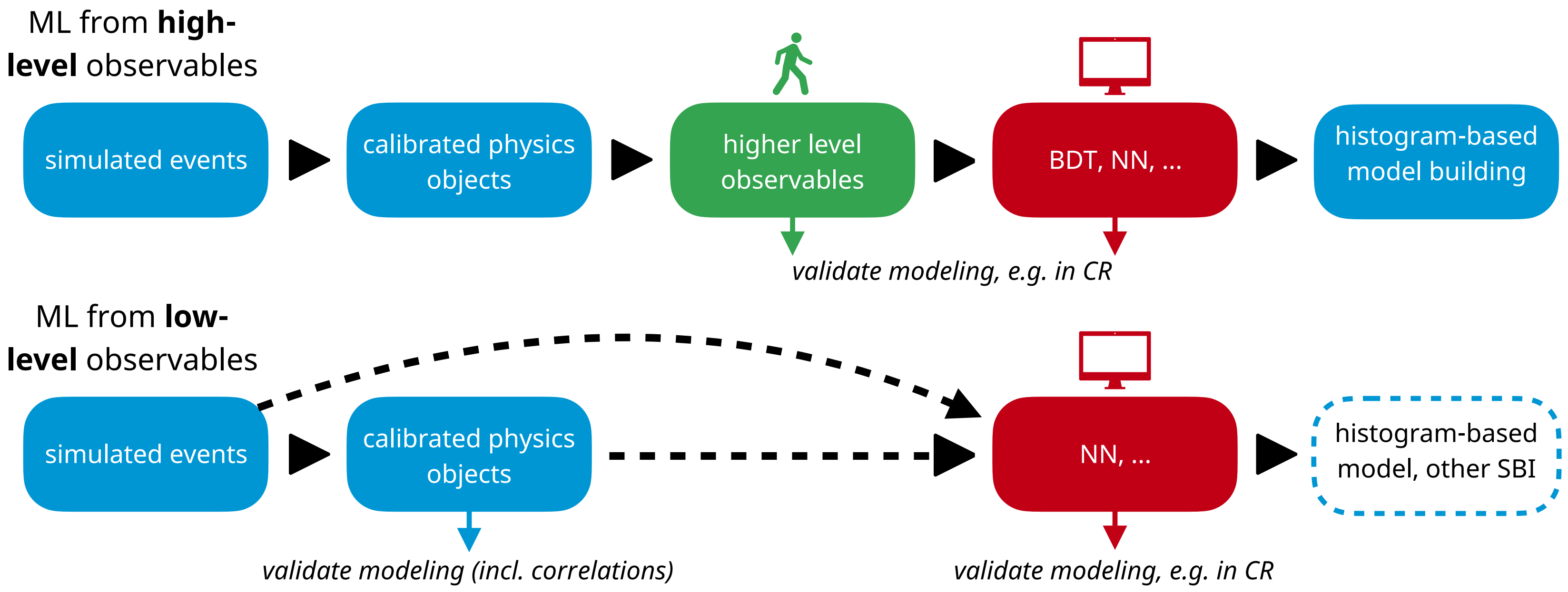

*example of a (fairly traditional) analysis pipeline*
https://agc.readthedocs.io/

# ML with high-level inputs




- In this picture the **ML step is "just a function"**, conceptually the same as a hand-crafted summary statistic
  - can propagate uncertainties through it and validate modeling of inputs

# ML with low-level inputs




- ML remains "just a function", but good **modeling** becomes **harder to validate** with **lower-level inputs**
  - does the simulator correctly capture correlations?
  - are we learning a bug in the simulator code? (→ desire for interpretability*)
  - are suitable calibration & uncertainties available for the inputs?

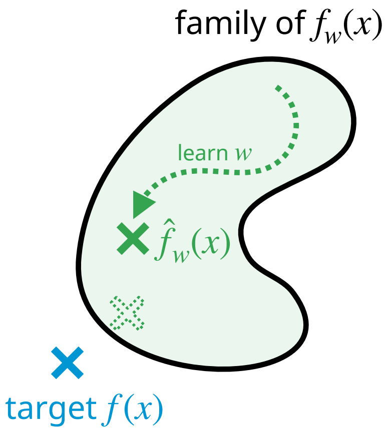
* not just to feel warm and fuzzy, but safe against bugs

# Systematics + ML: wrong vs suboptimal

- **Model misspecification** and (lack of) **systematic uncertainties** can make our results **wrong** and / or **suboptimal**

- **Avoiding wrong results**
  - incorporate and propagate all relevant sources of systematic uncertainty through chain
    - requires understanding which sources are relevant

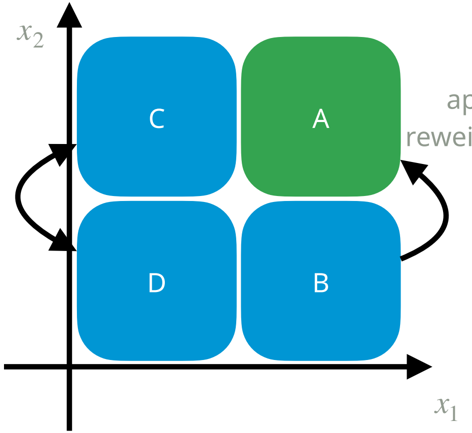
- **Striving towards optimal results**
  - possible limitations due to training dataset size, model capacity, domain shift
  - e.g. “are we using a good summary statistic?”
  - often ML training + systematic uncertainties are factorized, generally non-optimal
    - instead: e.g. data augmentation, parameterized models, … [e.g. Kyle Cranmer's talk yesterday]



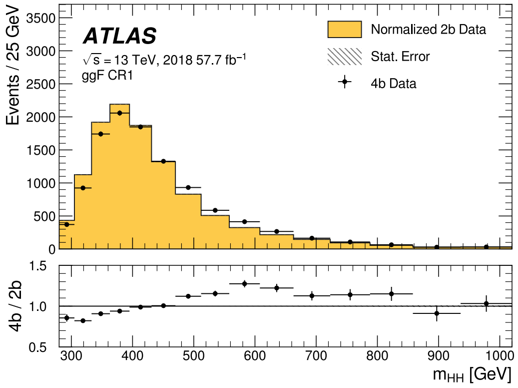
# Reweighting for background estimates

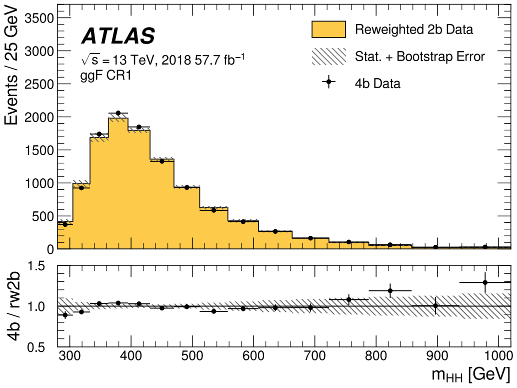
- Example from a di-Higgs analysis: **learn reweighting for background estimate**
  - need to propagate a statistical uncertainty here
  - deep ensembles with bootstrap to achieve this

- Similar idea to handle finite training statistics in Aishik Ghosh's talk yesterday



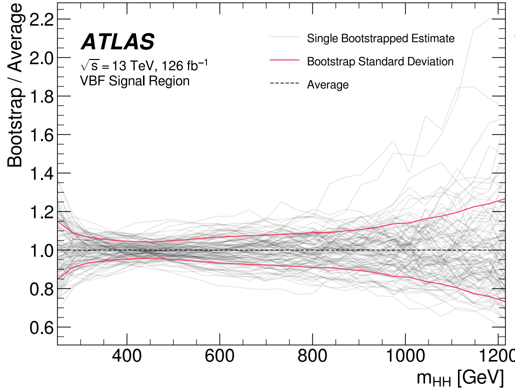

apply reweighting
(derived with independent observables)




Phys. Rev. D 108 (2023) 052003

*variation in prediction from bootstrap*




Phys. Rev. D 108 (2023) 052003

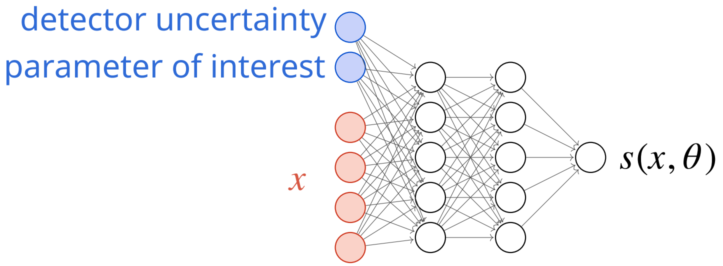
# SBI, differentiable physics analysis and beyond

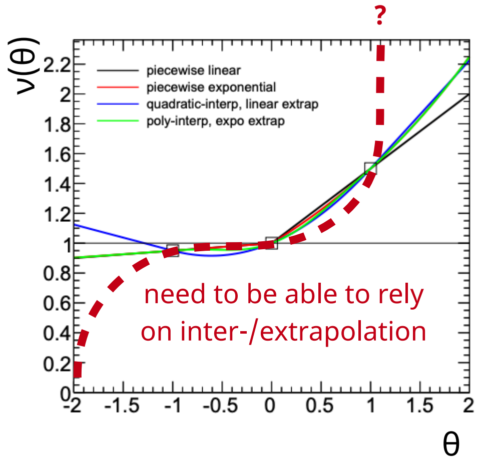
*take-away message:*

Some very interesting open questions left to answer!

# Systematic uncertainties & SBI

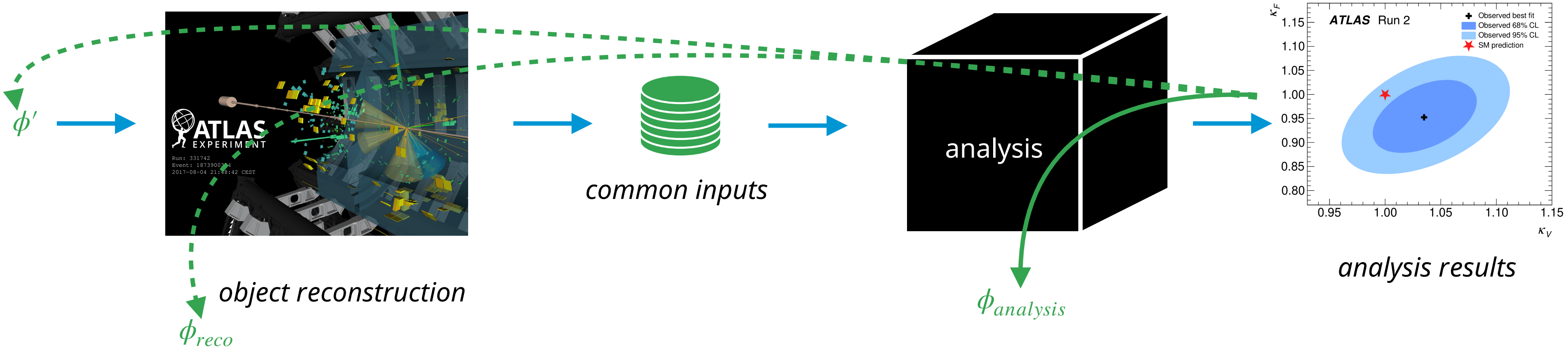
- Propagating effects of **systematic uncertainties through neural SBI setups** can be challenging
  - room for new ideas
- **Fully parameterize all effects**
  - parameterize O(100) effects of variations, learn full dependency
  - any guarantees for interpolation / extrapolation behavior?
  - how to capture & address potential statistical fluctuations? regularization?
- Need to **carefully validate** that **parameterization** works well
  - e.g. classifier: nominal events reweighted with $r(x \mid \theta)$ vs simulated variation



# Differentiable programming for physics analysis

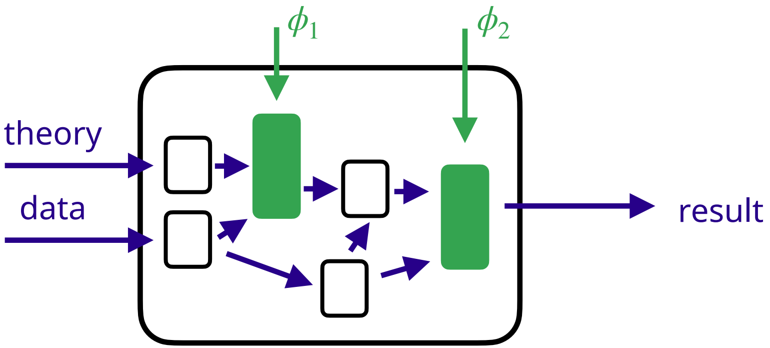
- A differentiable analysis pipeline would allow **optimizing physics analysis parameters $\phi$ via gradient descent**
  - what is the right loss function? can we do this in a manner that is robust to mismodeling?



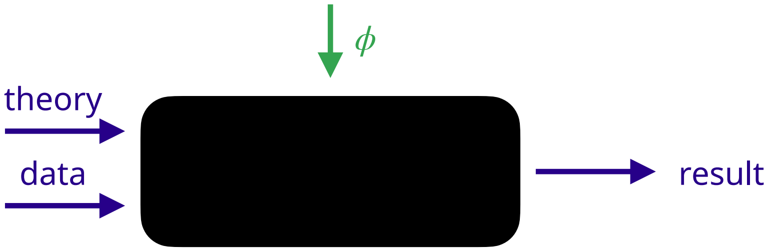

- Exploration of **differentiation of parts of this pipeline** has been ongoing for a while
  - see e.g. Artur Monsch's talk yesterday, INFERNO, neos

# The future?

- Increasingly many possible directions for how to do **physics analysis with ML in the future**
  - consider: how well do we understand relevant modeling & uncertainties, how and where can we validate that
  - lots of promise in newer approaches like neural SBI, but also some challenges to overcome




*physics + targeted ML*
*(+ differentiable programming pipeline?)*



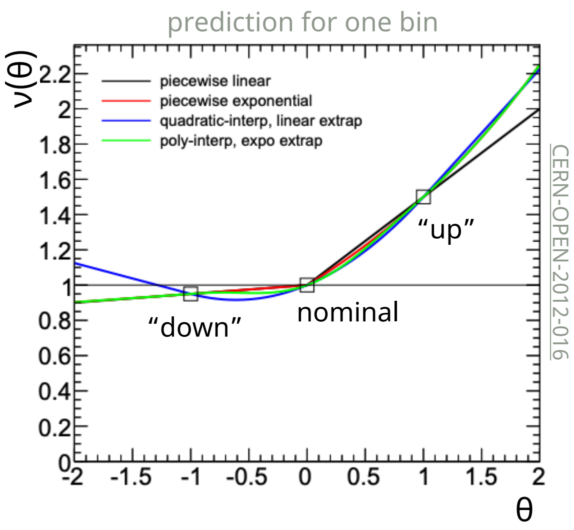

*"maximalist"*
*everything is one big network*

inspired by: L. Heinrich, ACAT 2024

# Backup

# Systematic uncertainties with HistFactory

- Common **systematic uncertainties** specified with **two template histograms**
  - "up variation": model prediction for $\theta = +1$
  - "down variation": model prediction for $\theta = -1$
  - interpolation & extrapolation provides model predictions $\nu$ for *any* $\vec{\theta}$




- **Gaussian constraint terms** used to model auxiliary measurements (in most cases)
  - centered around nuisance parameter (NP) $\theta_j$
  - normalized width ($\sigma = 1$) and mean (auxiliary data $a_j = 0$)
  - penalty for pulling NP away from best-fit auxiliary measurement value

$$p(\vec{n}, \vec{a} \mid \vec{k}, \vec{\theta}) = \prod_i \mathrm{Pois}(n_i \mid \nu_i(\vec{k}, \vec{\theta})) \cdot \prod_j c_j(a_j \mid \theta_j)$$