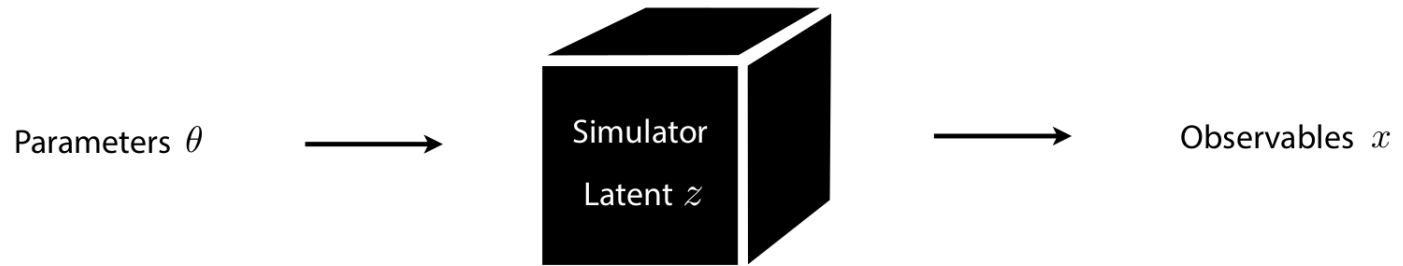


Highlights

PHYSTAT Statistics meets Machine Learning
September 12, 2024

Gilles Louppe
g.louppe@uliege.be

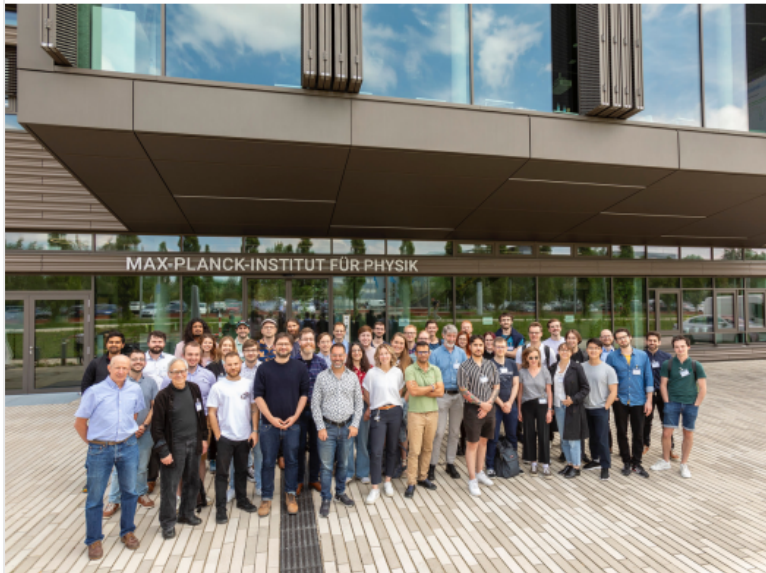


- Prediction:
- Well-motivated mechanistic, causal model
 - Simulator can generate samples $x \sim p(x|\theta)$

- Inference:
- Interactions between low-level components lead to challenging inverse problems
 - Likelihood $p(x|\theta) = \int dz p(x, z|\theta)$ is intractable

Takeaway 0: Simulation-based inference is taking off!

Simulation-Based Inference in PHY-STAT



Workshop in Munich this summer

11:45	Simulation-based machine learning for gravitational-wave analysis	45m
12:10	Anomaly aware machine learning for dark matter direct detection at the DARWIN experiment	25m
14:25	lsbc: linear simulation based inference	25m
11:15	pop-cosmos: an interpretable generative model for the galaxy population over cosmic time	30m
17:35	Feldman-Cousins' ML Cousin	25m
11:00	Cosmology and machine learning	45m
18:09	Advanced techniques for Simulation Based Inference in collider physics	1m
18:10	SBI for wide field weak lensing	1m
18:13	Accounting for Selection Effects in Supernova Cosmology with Simulation-Based Inference and Hierarchical Bayesian Modelling	1m
11:00	Simulation-based Inference (SBI)	45m

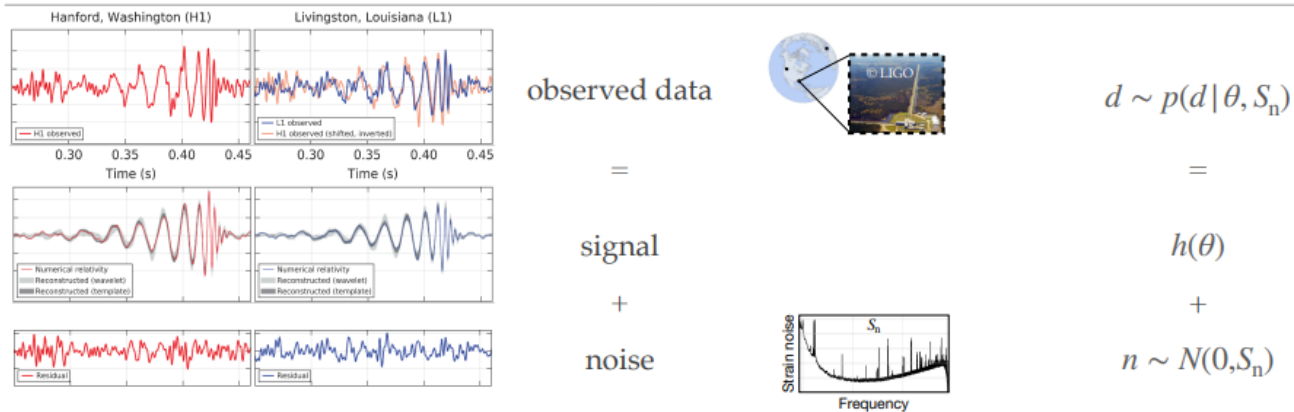
This workshop alone !

Takeaway 1: SBI is also valuable when likelihoods are tractable!

For example, with deterministic simulators and additive noise,

$$p(x|\theta) = \mathcal{N}(x; f(\theta), \Sigma(\theta)).$$

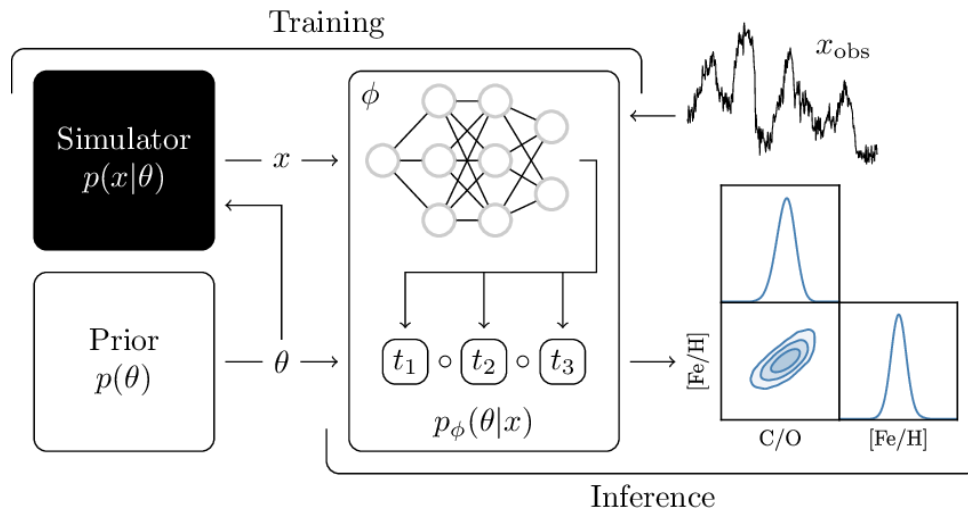
GW forward model



Parameters of GW model:

- Binary parameters $\theta \in \mathbb{R}^{15}$ — Parameters of interest
- Detector noise spectrum $S_n \in \mathbb{R}^k$ — Detector property, from external data

$$\Rightarrow p(\theta | d, S_n)$$



To account for measurement noise and make the simulation model similar to instrumental data, we consider a Gaussian noise model with a standard deviation σ . The spectra $f(\theta)$ generated by `petitRADTRANS` are randomly perturbed with additive noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$, where $\epsilon \in \mathbb{R}^{379}$ is a vector of random noise instances in each wavelength bin. Here we assume the same noise variance in each wavelength bin for the sake of simplicity, but more complex noise models (including noise covariance) could be used in our simulator. **The final simulator output is given by $x = f(\theta) + \epsilon$.**

Takeaway 2: Unfolding = Aggregated amortized posteriors.

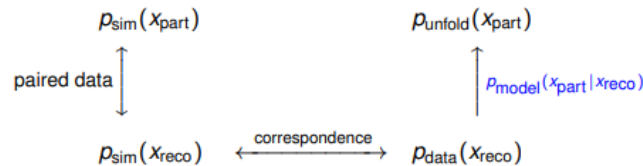
$$p(x_{\text{part}}) = \int p(x_{\text{reco}})p(x_{\text{part}} | x_{\text{reco}}) dx_{\text{reco}}$$

ML-Unfolding
 Tilman Plehn
 Generative AI
 Unfolding
 Top decays
 Bayesian NNs
 Calibration

Unfolding by generation

Targeting conditional probability [Butter, TP, Winterhalter,...]

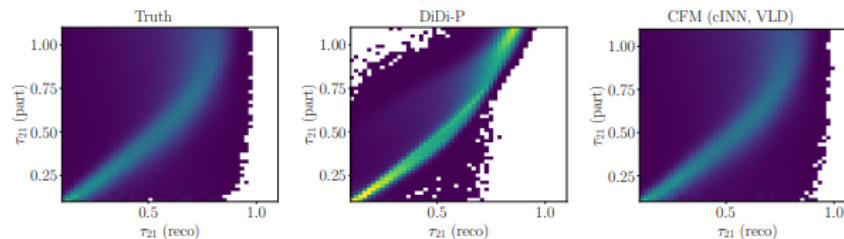
- just like forward ML-generation
- learn inverse conditional probability from $(x_{\text{part}}, x_{\text{reco}})$



Improvements crucial [Xavier Villadamiago's poster]

- make networks more precise → TraCFM
- remove training prior [Backes, Butter, Dunford, Malaescu]

→ Success through generative progress

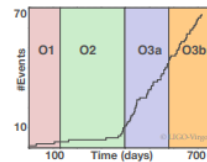


Takeaway 3: Requirements for accuracy and reliability are strict.

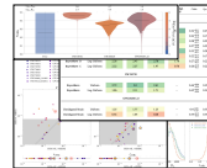
Motivation

Why GWs need ML

Increasing event rate



Large-scale analyses



Follow-up searches



Why ML needs GWs

Strict requirements for **accuracy**, **reliability** and **interpretability**

Complexity of GW data

⇒ GW data analysis pushes existing ML past its limits

Maximilian Dax

2

Challenges for NSBI:

- Robustness: Design and validation
- Uncertainties: Quantifying and propagating systematics
- Neyman Construction: Throwing toys in high-dimensions

Takeaway 4: Use diagnostics* to assess the quality of the approximation.

Diagnostics with classifiers

This paper also introduced two diagnostics

- classifier tests with data reweighted

Approximating Likelihood Ratios with Calibrated Discriminative Classifiers

Kyle Cranmer¹, Juan Pavez², and Gilles Louppe³
¹New York University
²Federico Santa Maria University

3.5 Diagnostics

The second diagnostic procedure leverages the connection of this technique to direct density ratio estimation and its application to covariate shift and importance sampling. The idea is simple: we test the relationship $p(\mathbf{x}|\theta_0) = p(\mathbf{x}|\theta_1)r(s(\mathbf{x}|\theta_0, \theta_1))$ with the approximate ratio $\hat{r}(s(\mathbf{x}|\theta_0, \theta_1))$ and samples drawn from the generative model. More specifically, we can train a classifier to distinguish between unweighted samples from $p(\mathbf{x}|\theta_0)$ and samples from $p(\mathbf{x}|\theta_1)$ weighted by $\hat{r}(s(\mathbf{x}|\theta_0, \theta_1))$. If the classifier can distinguish between the distributions, then $\hat{r}(s(\mathbf{x}|\theta_0, \theta_1))$ is not a good approximation of $r(s(\mathbf{x}|\theta_0, \theta_1))$. In contrast, if the classifier is unable to distinguish between the two distributions, then either $\hat{r}(s(\mathbf{x}|\theta_0, \theta_1))$ is a good approximation or the discriminator is not effective. The two situations can be disentangled to some degree by training another classifier to distinguish between an unweighted distribution of samples from $p(\mathbf{x}|\theta_1)$.

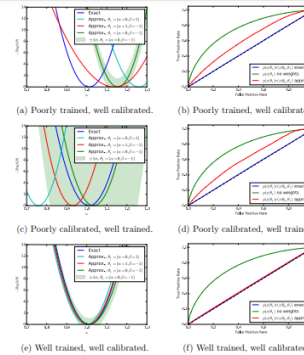


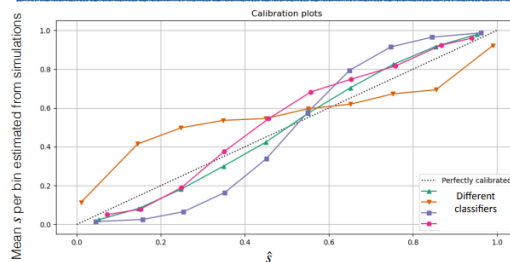
Figure 5: Results from the diagnostics described in Sec. 3.5. The rows correspond to the quality of the training and calibration of the classifier. The left plots probe the sensitivity to θ_1 , while the right plots show the ROC curve for a classifier trained to discriminate samples from $p(\mathbf{x}|\theta_0)$ and samples from $p(\mathbf{x}|\theta_1)$ weighted as indicated in the legend.

K.C., G. Louppe, J. Pavez: Approximating Likelihood Ratios with Calibrated Discriminative Classifiers [arXiv:1506.02169]

See also: 2016 talk on reweighting at the DS@HEP

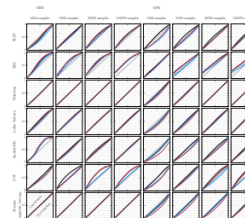
43

Calibration Curves



Recall, classifier was trained to estimate:

$$s(x_i) = \frac{p(x_i | \theta_0)}{p(x_i | \theta_0) + p(x_i | \theta_1)}$$



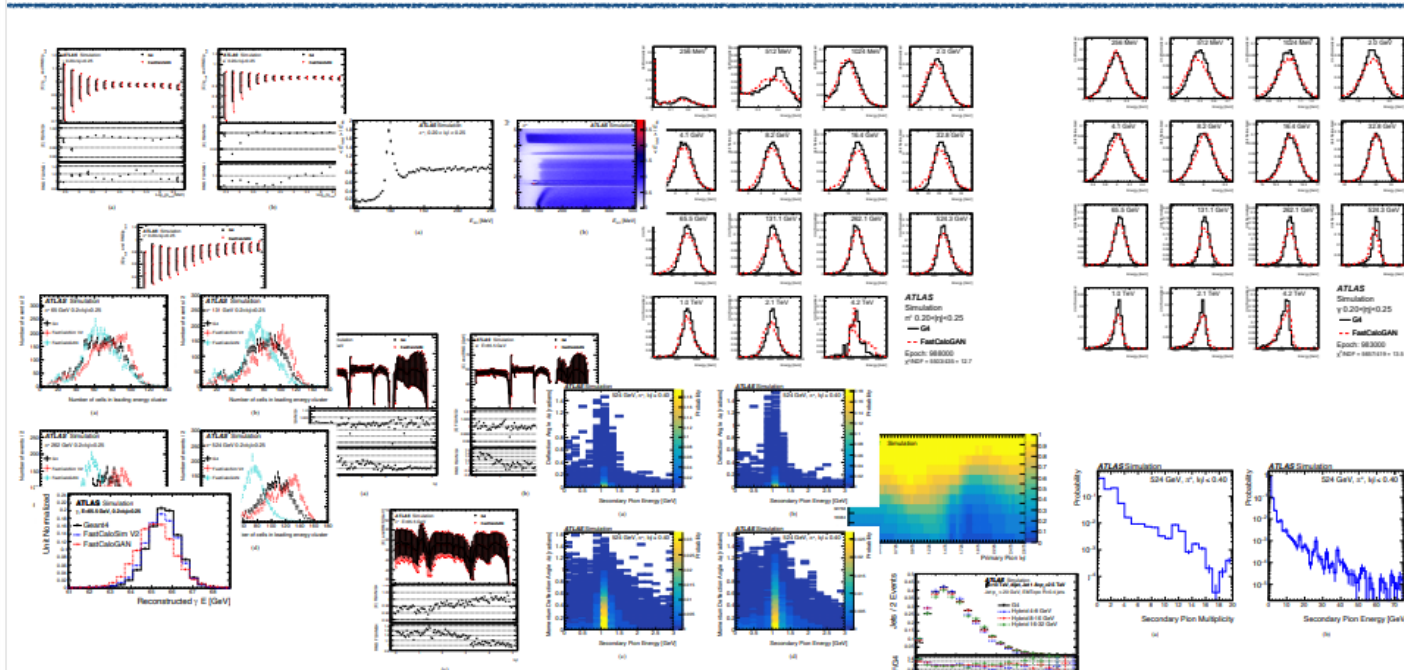
Similar tests possibly for many NSBI methods, see Hermans et al

26

*None of those diagnostics provide sufficient guarantees.

Credits: Kyle Cranmer, Aishik Ghosh.

Evaluating Fast Calo Simulators



Takeaway 4b: ... and look at (too) many plots!

What if diagnostics fail?

Takeaway 5: More data, more parameters, more compute.

i.e., $S_n^{(i)} \sim p(S_n)$, $n^{(i)} \sim p(S_n^{(i)})$. We construct training sets based on 5×10^6 sets of intrinsic parameters, but by sampling extrinsic parameters and noise realizations during training, the effective size of the training set is infinite in these dimensions.

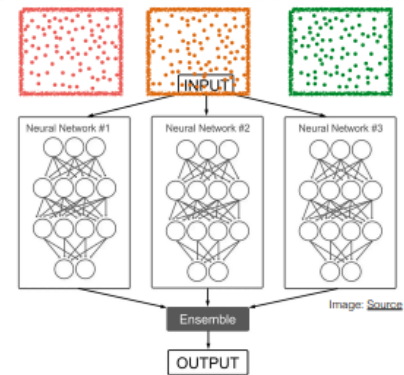
We also increase the total number of coupling transforms to 30 from 15 in [36]. The flow therefore consists of 300 hidden layers. In total, the embedding network and the flow combined have $1.31 \cdot 10^8$ learnable parameters for $n_{\text{detectors}} = 2$ and $1.42 \cdot 10^8$ for $n_{\text{detectors}} = 3$.

for validation to check for overfitting. Since the training and validation loss are in close agreement in Fig. 5 we conclude that overfitting is minimal. Training 450 epochs with a batch size of 4096 takes roughly 10 days on a single NVIDIA A100 GPU.³

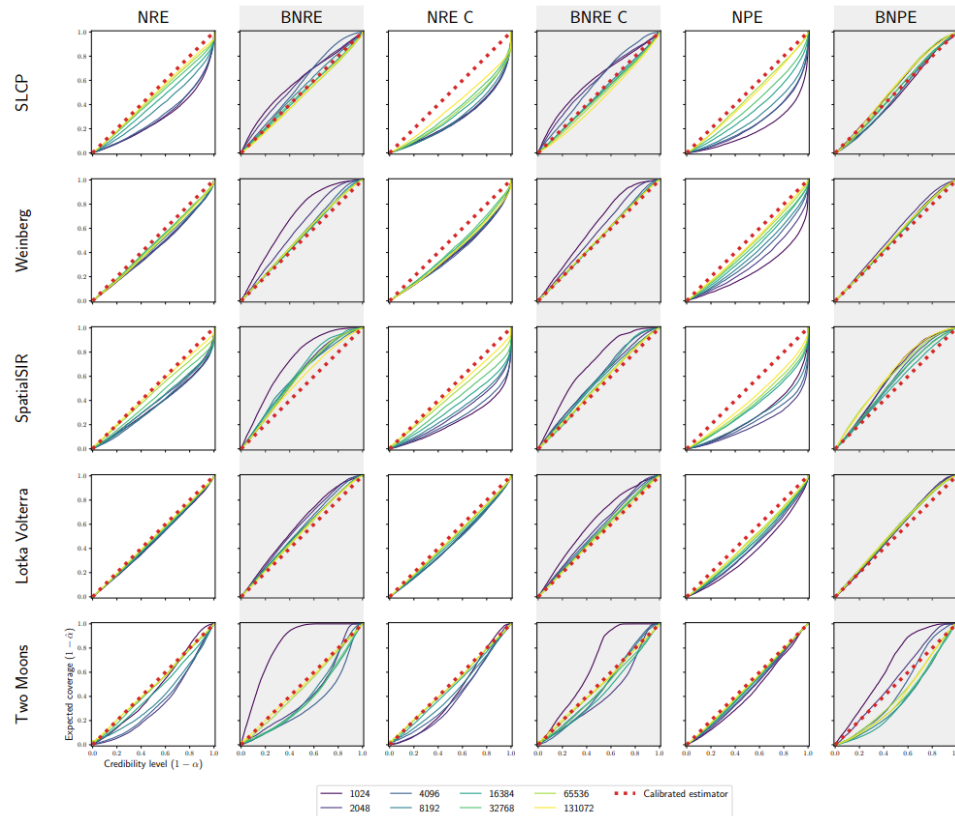
Takeaway 6: Regularize the inference network.

Quantifying uncertainty on estimated likelihood..

- Train an ensemble of networks, each on a bootstrapped version of the training dataset
 - Or Bayesian networks ? [Delaunoy et al, [arXiv2408.15136](https://arxiv.org/abs/2408.15136)]
- The spread in their prediction provides the uncertainty due to limited training statistics, and random network initialisation
- Ensemble average used as final prediction, so what's the uncertainty on that ?
 - Too expensive to train thousands of ensembles
 - Create bootstrapped ensembles ?
 - Each network trained on bootstrapped training dataset ?



Ensembles and Bayesian model averaging (BNMs) smooth out the approximation.



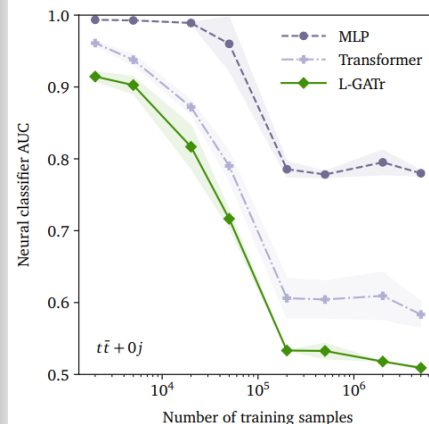
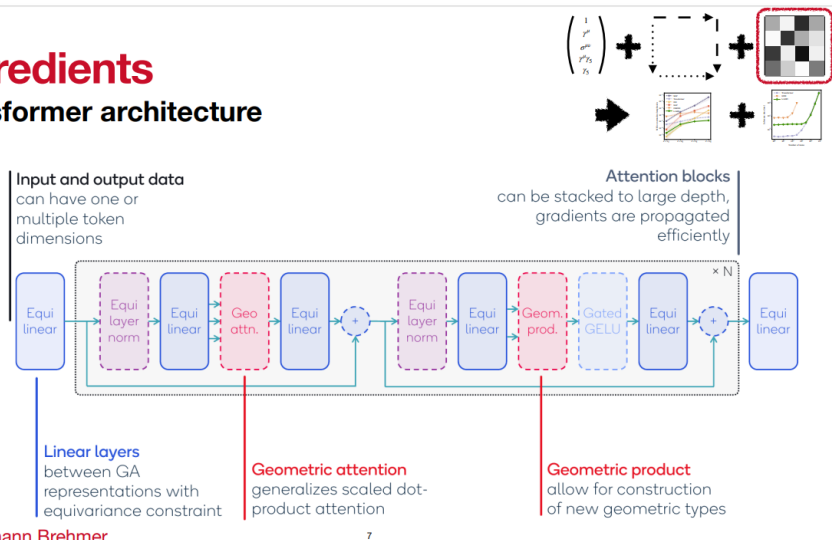
Approximations can also be regularized to be conservative (BNRE, BNPE).

Not enough!

Takeaway 7: Hardcode domain knowledge in the inference network.

Ingredients

Transformer architecture



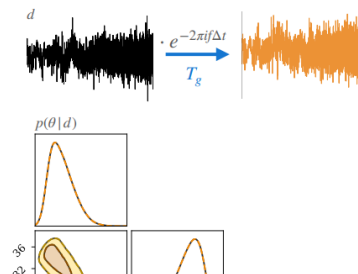
NPE with symmetries: Group-equivariant NPE

- Equivariance (covariance) under time shift

$$p(\theta | d) = p(g\theta | T_g d) |\det J_g| \quad \forall g \in G$$

- NPE learns such symmetries from simulation data
 \Rightarrow requires network and training capacity
 \Rightarrow can we instead **enforce such symmetries?**

- Group-equivariant NPE (GNPE)

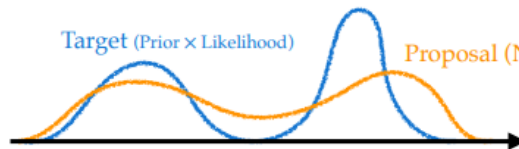


Takeaway 8: Importance sampling is a cheat code for asymptotically exact inference with imperfect inference networks.

Dax+ (PRL 2023)

Importance-sampled NPE (NPE-IS)

- If likelihood is tractable, can reweight NPE results



$$\theta_i \sim q(\theta | d)$$

$$w_i = \frac{p(\theta_i)p(d|\theta_i)}{q(\theta_i|d)}$$

⇒ asymptotically exact

- **Effective number of samples** as performance metric

$$n_{\text{eff}} = (\sum_i w_i)^2 / \sum_i (w_i^2) \quad \epsilon = n_{\text{eff}} / n \in (0, 1]$$

⇒ **verification** without for ground truth posterior

- Estimate of **Bayesian evidence**

$$p(d) = \frac{1}{n} \sum_i w_i \quad \sigma_{\log p(d)} = \sqrt{(1 - \epsilon) / (n \cdot \epsilon)}$$

⇒ unbiased & precise estimate of **evidence**

If $p(x_i | \theta_i)$ is intractable, the correction factor w_i can be estimated by a classifier.

- Classifier-based diagnostics provide diagnostics, but also a way to correct the approximation.
- If repeated, then one obtains a **sequential** NRE algorithm.

Takeaway 9: Nuisance parameters are a nuisance.

Curse of dimensionality for nuisance parameters

The traditional binned-template analysis approach uses a fixed interpolation / “template morphing” strategy

- Dependence on the parameters of interest are usually very well motivated
- makes assumptions about factorization of systematics that might not be true
- ... either way, fixed parametric form makes it VERY sample efficient

In contrast, parametrized NN is physics-agnostic and the interpolation is non-parametric

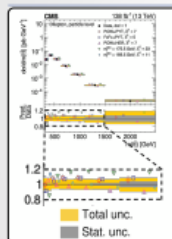
- Flexible, but requires many samples for a high-dimensional nuisance parameter space
- Curse of dimensionality

Is there a way to apply similar assumptions as template-based morphing strategy in neural SBI context?

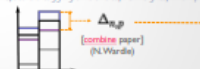
Recent developments shown at PhyStat SBI

R. Schöfbeck & Jay Sandesara both showed work in that direction at PhyStat SBI

BACK TO REALITY! R. Schöfbeck @ PhyStat SBI (SMC-TCP-2019-02)



- Systematics dominate in many/most applications
- Binned analyses? Use additive model with exponentials

$$\text{prediction}(\theta, v) = \sum_{p=1}^N R_{p,0}(\theta) \exp(v^T \Delta_{p,0,1} + v^T \Delta_{p,0,2}) \sigma_{p,0}(\text{SM})$$
- How to find the parameters Δ ?
 - "Vary simulation" \leftrightarrow Generate synthetic datasets
 - shift JEC, scale b-tagging efficiencies, PS weights, bRamp
 - 
 - [combine paper] (N. Wardle)
- Decades of experience with modeling choices

LEARNING PARAMETERIZATIONS R. Schöfbeck @ PhyStat SBI

- "Likelihood ratio trick"

$$L_{\text{ratio}}[f] = -(\log f(x))_{x \sim \text{SM}} - (\log(1 - f(x)))_{x \sim \bar{\text{SM}}}$$

$$f_{\text{SM}}^*(x) = \frac{1}{1 + \frac{\log(f(x))}{\log(1-f(x))}}$$
- Parametric ansatz:

$$f(x) = \frac{1}{1 + \exp(k_{\text{SM}}(x))}$$

$$k_{\text{SM}}(x; \Delta_1, \Delta_2, \dots) = v^T \Delta_1(x) + v^T \Delta_2(x) + \dots$$

Implement coefficient functions $\Delta_i(x)$ as NNs or trees: $\Delta_{1,2,\dots}(x) = \sum_{j=1}^n \mathbb{1}_{j(x)} \Delta_{j,1,2,\dots}$
- Sufficiently many synthetic data sets

$$L[\hat{\Delta}] = \sum_{\text{SM}} L_{\text{ratio}}[f_{\text{SM}}] \rightarrow \exp(g^*(x)) \approx \frac{d\sigma(x; v)}{d\sigma(x; \text{SM})}$$

- Basic idea:** use same interpolation point wise in x , replace histogram bin with function of x .

where for each component $d\sigma_{i,j}(x; \theta, v)$, we can obtain event samples from Eq. 23, Eq. 30, or Eq. 31. Next, we factorize the systematic effects and the POI dependence. The SM joint corresponds to $\theta := v := 0$ and for each $d\sigma_{i,j}(x; \theta, v)$ we have

$$\frac{d\sigma_{i,j}(x; \theta, v)}{d\sigma_{i,j}(x; 0, 0)} = \frac{d\sigma_{i,j}(x; \theta, v)}{d\sigma_{i,j}(x; \theta, 0)} \frac{d\sigma_{i,j}(x; \theta, 0)}{d\sigma_{i,j}(x; 0, 0)} \approx \frac{d\sigma_{i,j}(x; \theta, 0)}{d\sigma_{i,j}(x; 0, 0)} \frac{d\sigma_{i,j}(x; \theta, 0)}{d\sigma_{i,j}(x; \text{SM})} \frac{d\sigma_{i,j}(x; 0, 0)}{d\sigma_{i,j}(x; \text{SM})} \quad (58)$$

The approximation is valid if the relative SMEFT effects are independent of the relative systematic effects, i.e.,

$$\frac{d\sigma_{i,j}(x; \theta, v)}{d\sigma_{i,j}(x; 0, v)} \approx \frac{d\sigma_{i,j}(x; \theta, 0)}{d\sigma_{i,j}(x; 0, 0)} \quad (59)$$

The factor

$$R_{i,j}(x; \theta) \approx \frac{d\sigma_{i,j}(x; \theta, 0)}{d\sigma_{i,j}(x; \text{SM})} \quad (60)$$

approximates the SMEFT variations and is a polynomial in θ . It can be obtained from one of the techniques in Refs. [9–11]. Systematic effects are parameterized by

$$S_{i,j}(x; v) \approx \frac{d\sigma_{i,j}(x; v, 0)}{d\sigma_{i,j}(x; \text{SM})} \quad (61)$$

leads to good accuracy. We estimate $\hat{S}_{i,j}(x; v)$ with Eq. 53 for each process. Following the same steps, we can furthermore factorize $S_{i,j}(x; v)$ into mutually uncorrelated groups of systematic uncertainties and train each factor individually. For example, uncorrelated one-parameter systematic uncertainties as quadratic approximations reduce the surrogate to

$$S_{i,j}(x; v) = \prod_{k=1}^K \exp(v_k \Delta_{k,1,1}(x) + v_k^2 \Delta_{k,1,2}(x)) \quad (62)$$

with 2K one-body functions $\Delta_{k,1,1}(x)$ and $\Delta_{k,1,2}(x)$ for each p . In most cases, first or second-order polynomials provide excellent approximations although this is not a limitation of our methodology.

<https://arxiv.org/abs/2406.19076>

Factorizing the SBI analysis

Inspired by the "Mixture Models" trick defined in CARL paper

The known analytical form is used to factorize out the POI μ dependence. We also factorize out the NP α -dependence again to avoid the training and validation of parameterized NNs - especially difficult with the O(100) NNs in a typical ATLAS analysis.

$$\frac{P(x_i | \mu, \alpha)}{P_{\text{ref}}(x_i)} = \frac{1}{\sum_c g_c(\alpha) \cdot f_c(\mu) \cdot \nu_c} \left[f_c(\mu) \cdot w_c(x_i | \alpha) \cdot \nu_c \cdot \frac{P_c(x_i | \alpha)}{P_{\text{ref}}(x_i)} \right]$$

Usual parameterized event yields, like in Histogram

$$y_i(\mu, \alpha)$$

with $g_c(x) = \nu_c(\alpha)/\nu_c$, estimated using analytic interpolation techniques from inputs $\nu_c(1)$, $\nu_c(-1)$ and the nominal yield $\nu_c(0) = \nu_c$

Per-event Density Ratio

Per-event parameterized ratios:

$$w_c(x | \alpha) = \prod_k \frac{P_c(x | \alpha_k)}{P_{\text{ref}}(x_k)}$$

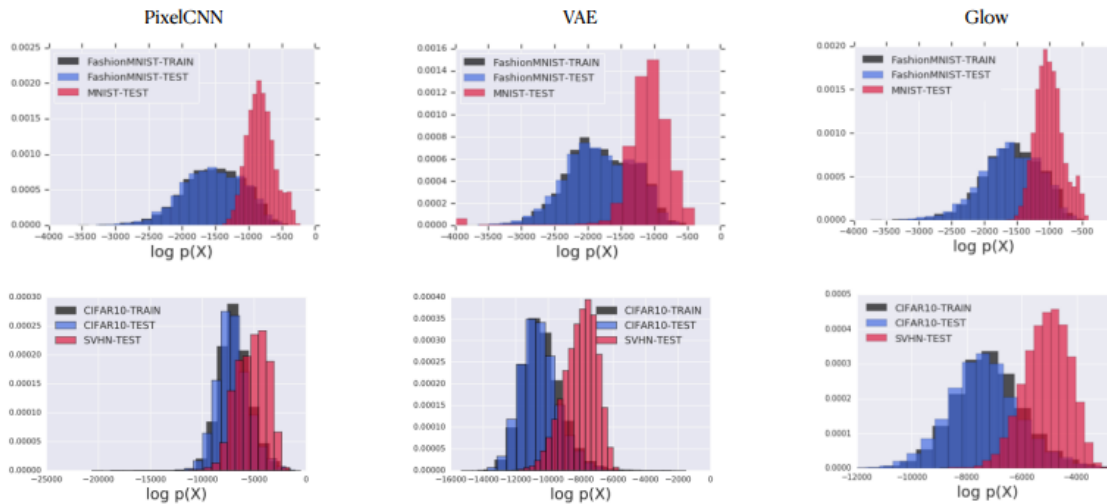
Same strategy as Histogram - but instead of histograms, we do per-event interpolation from

$$\frac{P_c(x | \alpha_k = \pm 1)}{P_{\text{ref}}(x)}$$

Per-event Density Ratio

Jay Sandesara @ PhyStat SBI

Takeaway 10: Neural density estimators do not know what they do not know.



Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, Balaji Lakshminarayan. "Do Deep Generative Models Know What They Don't Know?" *ICLR 2019*.

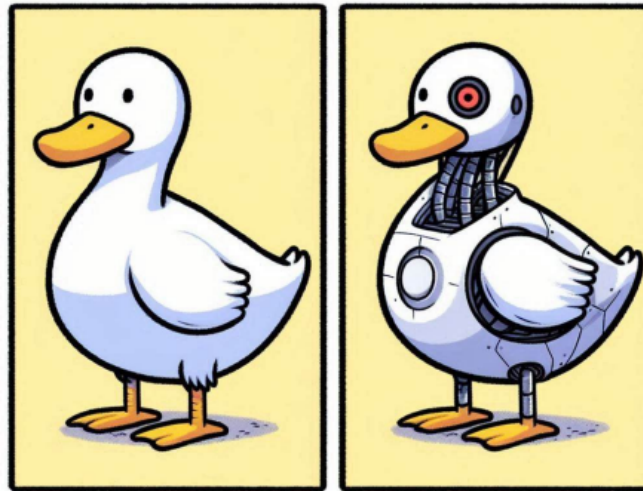
Modeling ducks

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]

Simulators are imperfect, but good enough?



Wait a minute... If the simulator is misspecified,
then data data may be OOD for the inference network.

My takeaways

1. Simulation-based inference is also valuable when likelihoods are tractable!
2. Unfolding = Aggregated amortized posteriors.
3. Requirements for accuracy and reliability are strict.
4. Use diagnostics to assess the quality of the approximation.
5. More data, more parameters, more compute.
6. Regularize the inference network.
7. Hardcode domain knowledge in the inference network.
8. Importance sampling is a cheat code for asymptotically exact inference with imperfect inference networks.
9. Nuisance parameters are a nuisance.
10. Neural density estimators do not know what they do not know.