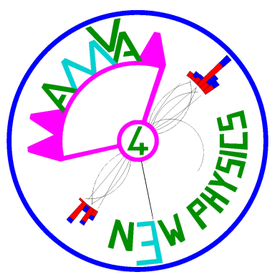
DIRECT LEARNING OF SYSTEMATICS-AWARE SUMMARY STATISTICS

Pablo de Castro (@pablodecm) and Tommaso Dorigo (@dorigo)

11th April 2018 @ 2nd IML Workshop (CERN)



AMVA4NewPhysics has received funding from European Union's Horizon 2020 Programme under Grant Agreement number 675440

THE HEPML' THREE-BODY PROBLEM

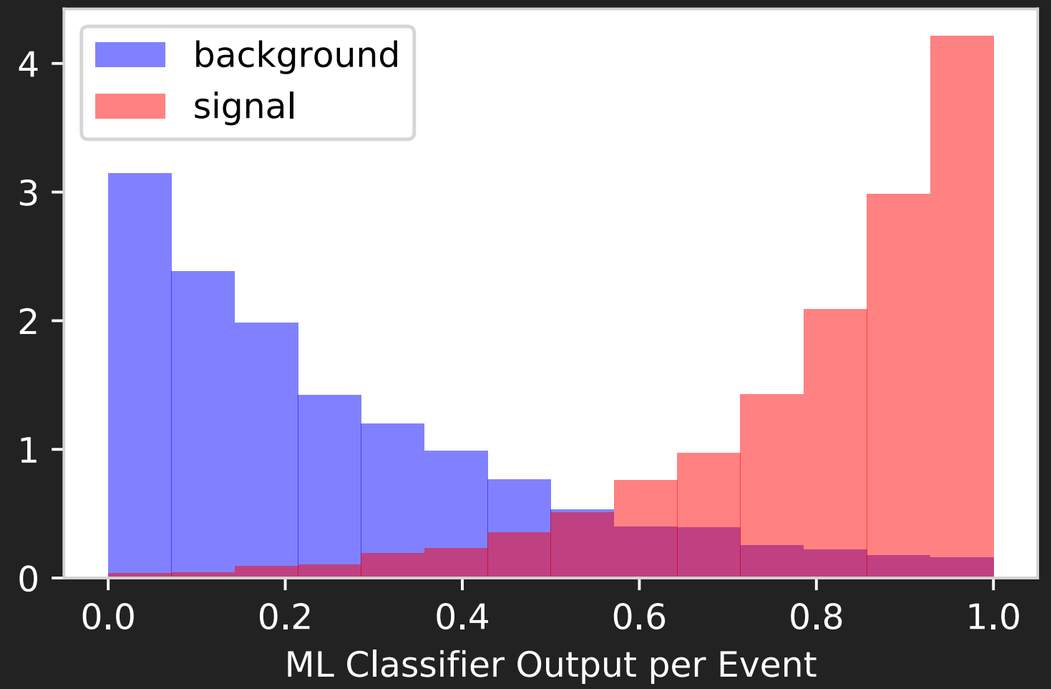
The three main analysis components only share processed data, each step is carried out independently, without taking into account details about the remaining other two

However, the ultimate aim of analyses is **powerful statistical inference** (i.e. interval estimation or hyphotesis testing) based on the observed data

IS THIS THE BEST WE CAN DO?

MACHINE LEARNING WITHIN LHC ANALYSES

Most data analysis problems tacked with machine learning are cast as one the two canonical supervised learning tasks: classification or regression



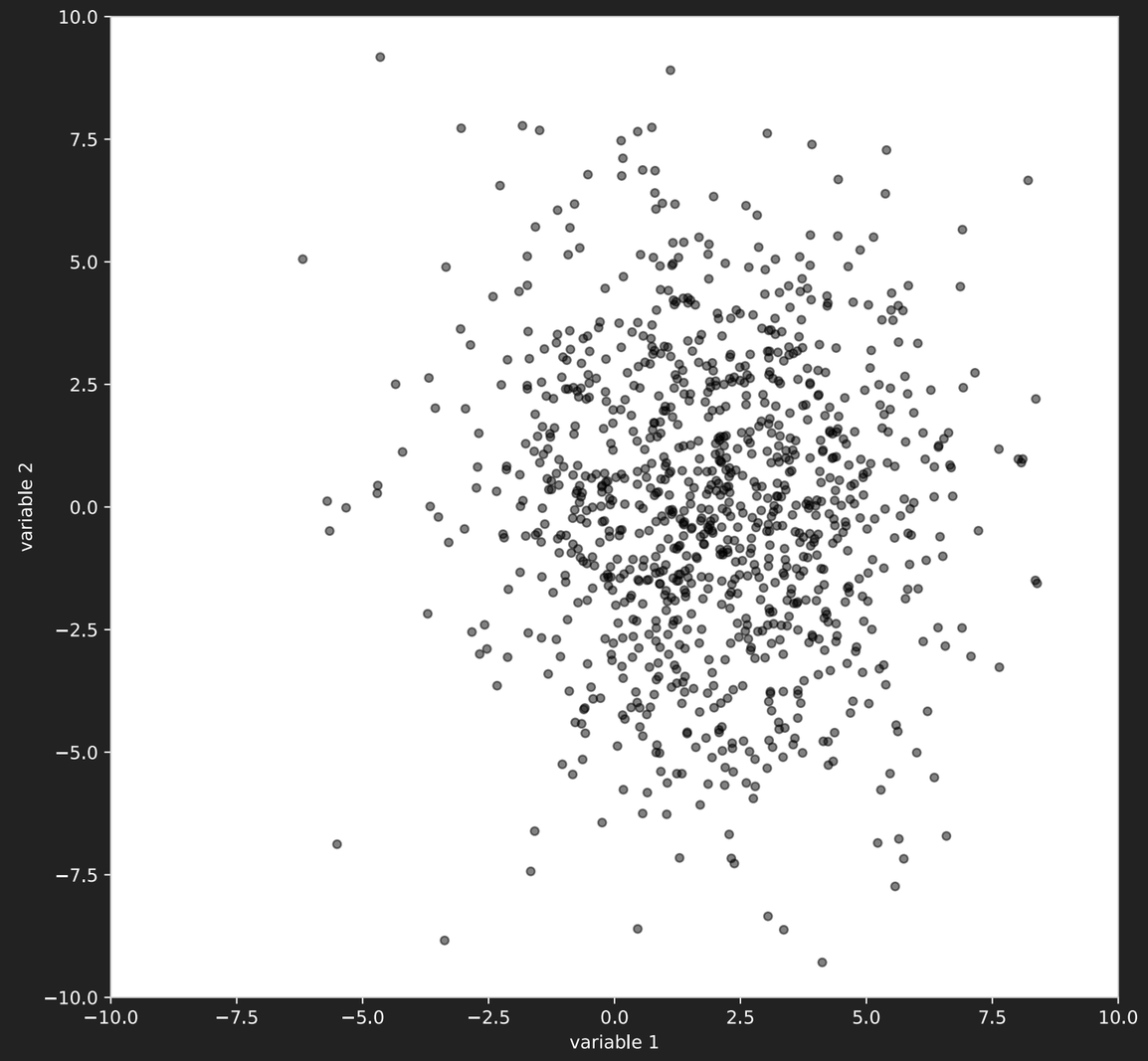
Event-by-Event Signal vs Background

- Higgs Kaggle challenge
- Every other LHC analysis

IS IT REALLY A CLASSIFICATION PROBLEM?

DOES NOT SEEM LIKE CLASSIFICATION AT FIRST SIGHT

Let us consider some i.i.d. observed data from our experiment $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$



NO CLASS LABELS

Common problem → hyphotesis testing

Testing an alternate hyphotesis H_1 (e.g. theory that predicts new particle) against the null hyphotesis H_0 (e.g. Standard Model without that particle)

HOW TO APPROACH THIS PROBLEM?

SOLVED IN 1933: NEYMAN-PEARSON LEMMA

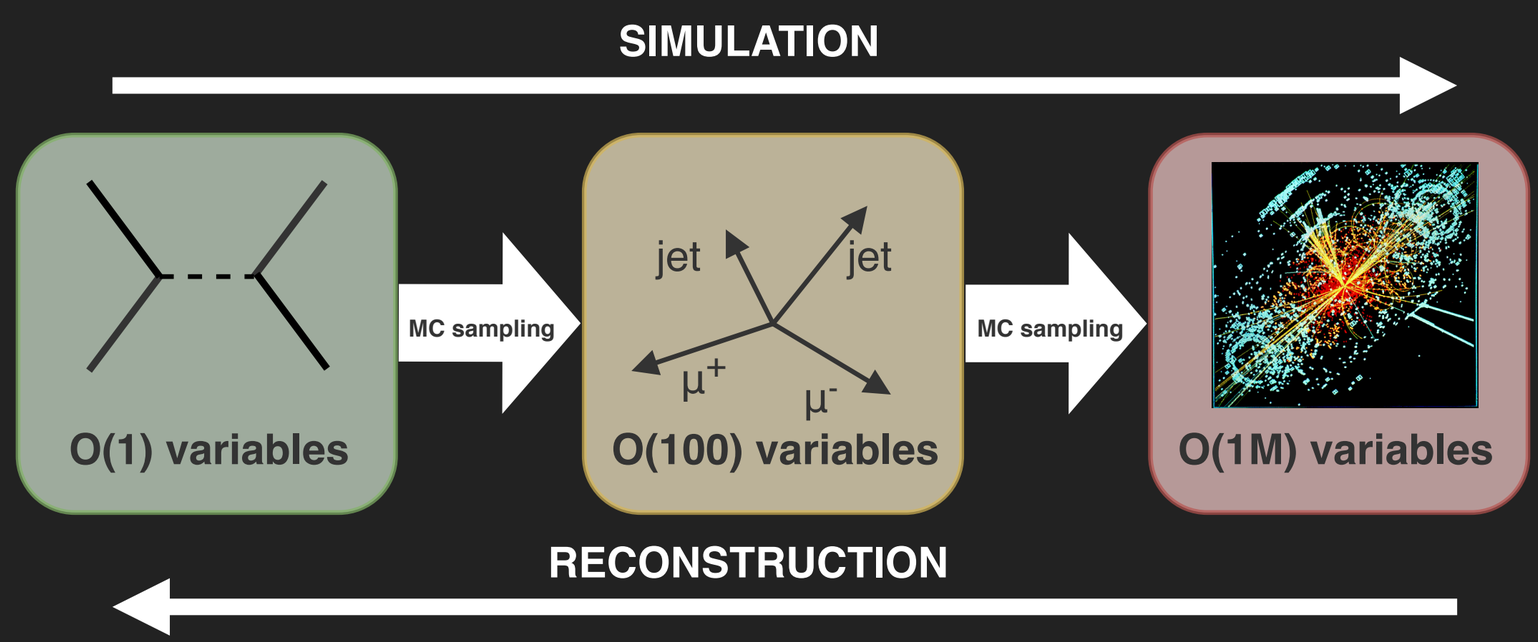
$$\Lambda(\mathcal{D}; H_0, H_1) = \prod_{\mathbf{x} \in \mathcal{D}} \frac{p(\mathbf{x}|H_0)}{p(\mathbf{x}|H_1)}$$

The **likelihood ratio** is the most powerful test (lowest Type II error) at a fixed significance level α between two simple hypotheses (i.e. that completely specify the distribution, **do not have additional parameters**)

MAIN COMPLICATION → $\mathbf{p(x|H_0)}$ ARE $\mathbf{p(x|H_1)}$ ARE INTRACTABLE

$\mathbf{p(x|model)}$ IS NOT KNOWN AT LHC EXPERIMENTS

Samples under different hypotheses can be simulated via complex physics-based MC programs but $p(\mathbf{x})$ cannot be directly evaluated → LIKELIHOOD-FREE INFERENCE



good approximations of $p(\mathbf{x})$ are unachievable due to curse of dimensionality

DIMENSIONALITY REDUCTION $\mathbf{R^n \rightarrow R^{O(1)}}$

KEEPING AS MUCH USEFUL INFORMATION FOR INFERENCE AS POSSIBLE

THE MIXTURE TRICK

Alternate hyphotesis H_1 is commonly a mixture model of a background component $p_\mathrm{b}(\mathbf{x}) = p(\mathbf{x}|H_0)$ and a small fraction μ of signal component $p_\mathrm{s}(\mathbf{x})$

$$\Lambda^{-1} \sim \frac{p(\mathbf{x}|H_1)}{p(\mathbf{x}|H_0)} = \frac{(1-\mu)\cdot p_\mathrm{b}(\mathbf{x}) + \mu \cdot p_\mathrm{s}(\mathbf{x})}{p_\mathrm{b}(\mathbf{x})}$$

$$\Lambda^{-1} \sim 1 - \mu \cdot \left(\frac{p_\mathrm{s}(\mathbf{x})}{p_\mathrm{b}(\mathbf{x})} - 1 \right)$$

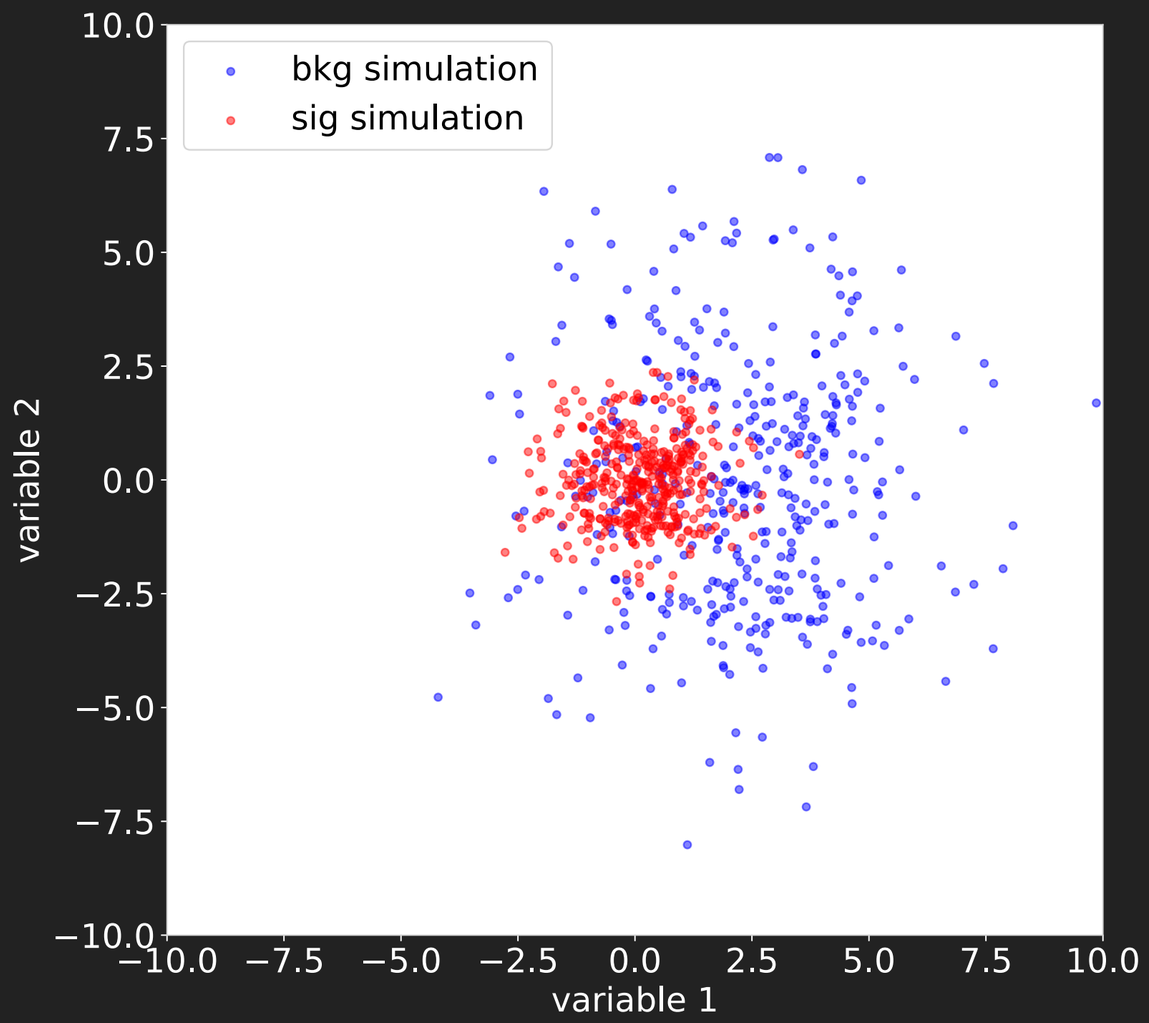
Therefore Λ is monotonically linearly decreasing with $p_\mathrm{s}(\mathbf{x})/p_\mathrm{b}(\mathbf{x})$

NEW TASK: APPROXIMATING $\mathbf{p_s(x)/p_b(x)}$

However, the generating distributions of background $p_{\mathrm{b}}(\mathbf{x})$ or signal $p_{\mathrm{s}}(\mathbf{x})$ are still not known

Only forward-simulated samples $\mathcal{S} = \{\mathbf{x}_0^{\mathrm{s}}, \ldots, \mathbf{x}_S^{\mathrm{s}}\}$ & $\mathcal{B} = \{\mathbf{x}_0^{\mathrm{b}}, \ldots, \mathbf{x}_B^{\mathrm{b}}\}$ are available

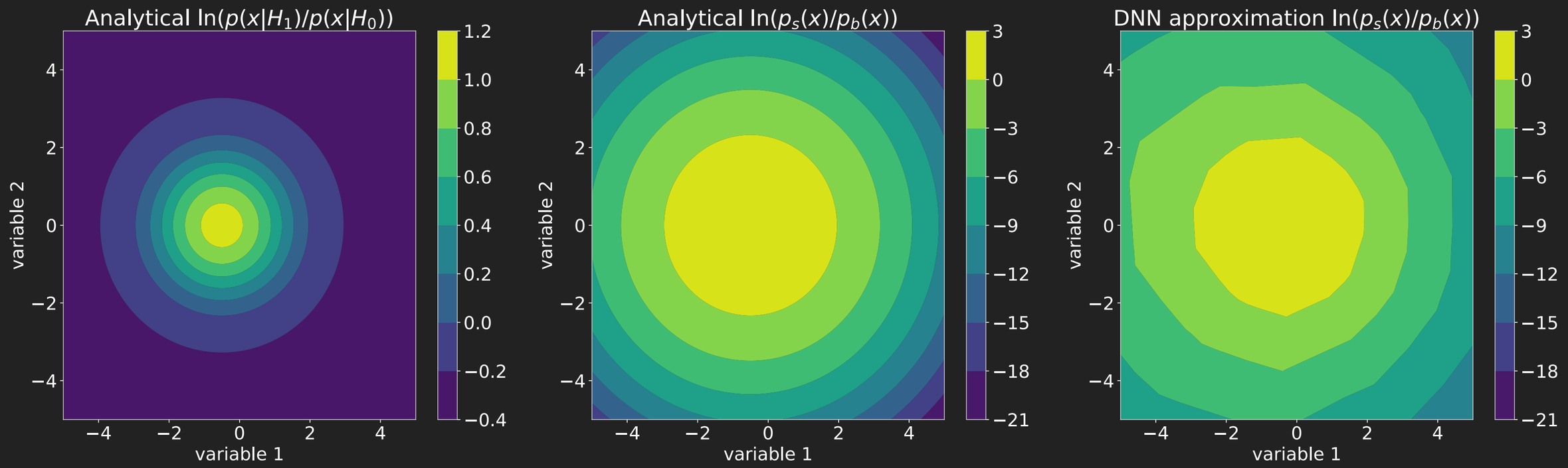
This synthetic dataset (2D Gaussian mixture) will be used extensively in this talk to exemplify different techniques



AMENABLE BY ML CLASSIFICATION → SIGNAL VS BACKGROUND

CLASSIFICATION AS SURROGATE TASK

Most machine learning classification algorithms approximate the likelihood ratio $p_{\mathrm{s}}(\mathbf{x})/p_{\mathrm{b}}(\mathbf{x})$
(e.g. a deep neural network minimizing cross entropy loss $-\sum_i k_i \log y_i$)

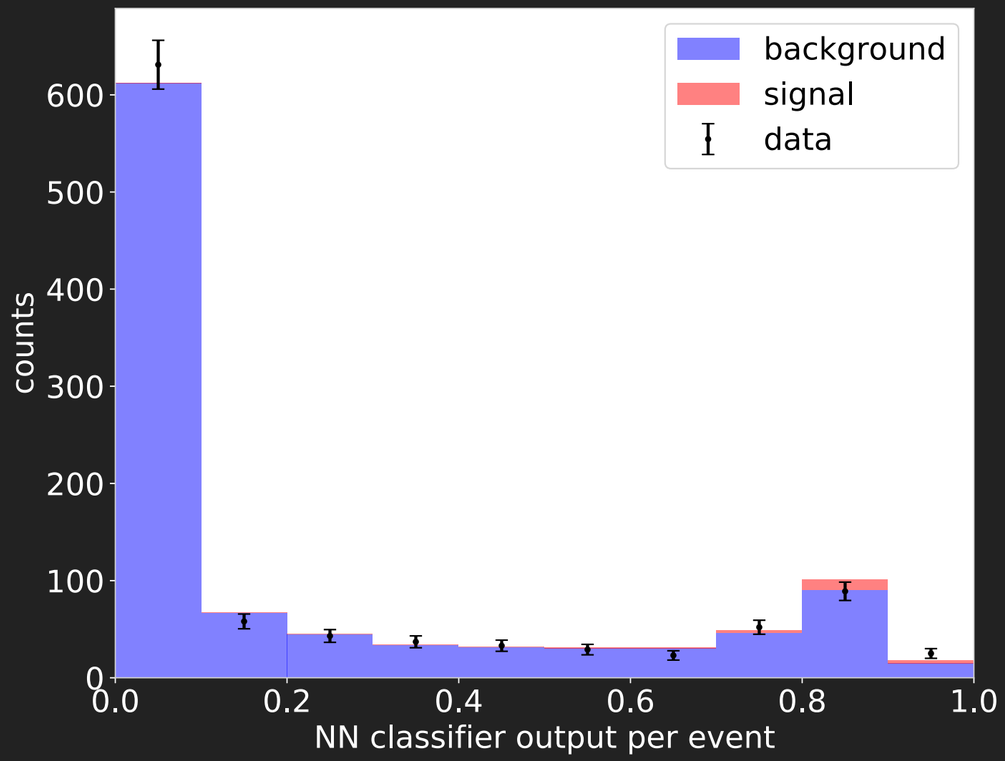


A direct exploitation of this for inference is carried out in "Approximating Likelihood Ratios with Calibrated Discriminative Classifiers" by K. Cranmer et al.

CLASSIFIER-BASED INFERENCE

A trained ML classifer $d(\mathbf{x})$ is an uncalibrated approximation of $p_\mathrm{s}(\mathbf{x})/p_\mathrm{b}(\mathbf{x})$

How can it be used for statistical inference from observed data $\mathcal{D}$?



1-D → **cut or histogram to build a Poisson counts non-parametric likelihood**

$$\mathcal{L}(\mu) = \prod_{\mathrm{c}\in\mathrm{bins}}^{\mathrm{C}} \mathrm{Pois}(n_\mathrm{c}\,|\mu \cdot s_\mathrm{c} + b_\mathrm{c})$$

which can be used for further inference, such as measuring μ given observed $\mathcal{D}$

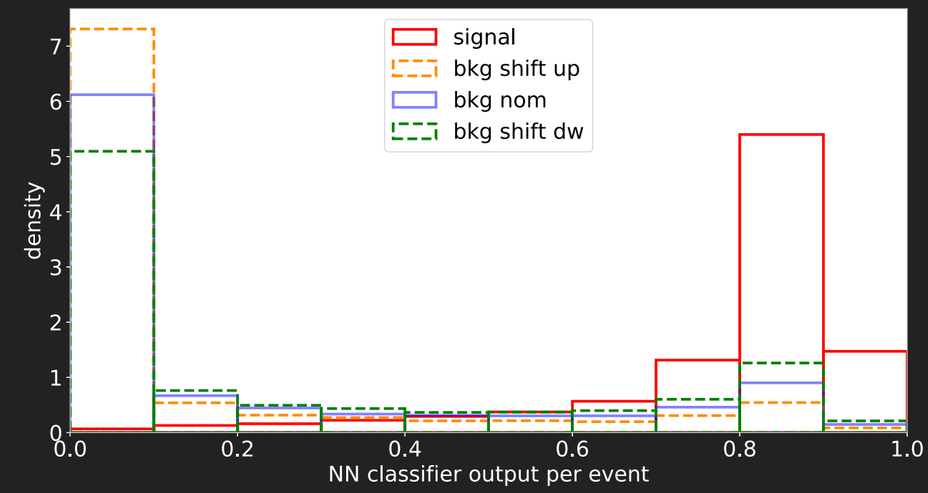
Choice of binning can be tricky, for cut and count "Consistent optimization of AMS by logistic loss minimization" by W. Cranmer is interesting.

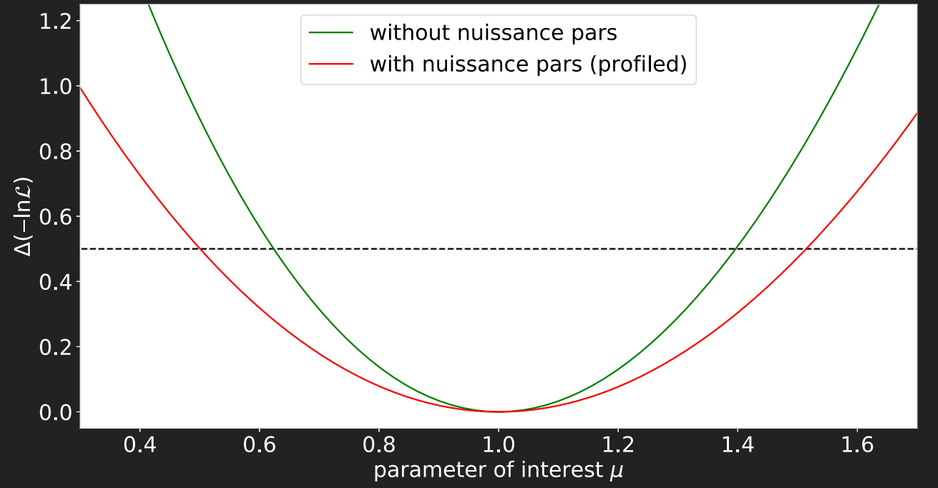
MODELLING UNCERTAINTIES DEGRADE INFERENCE

Simulations are imperfect, mainly due to the limited information of the system being modelled

Lack of knowledge for inference accounted by additional unknown parameters (nuisance parameters ν)

Causes a degradation of classifier-based inference, leading to larger measurement uncertainties



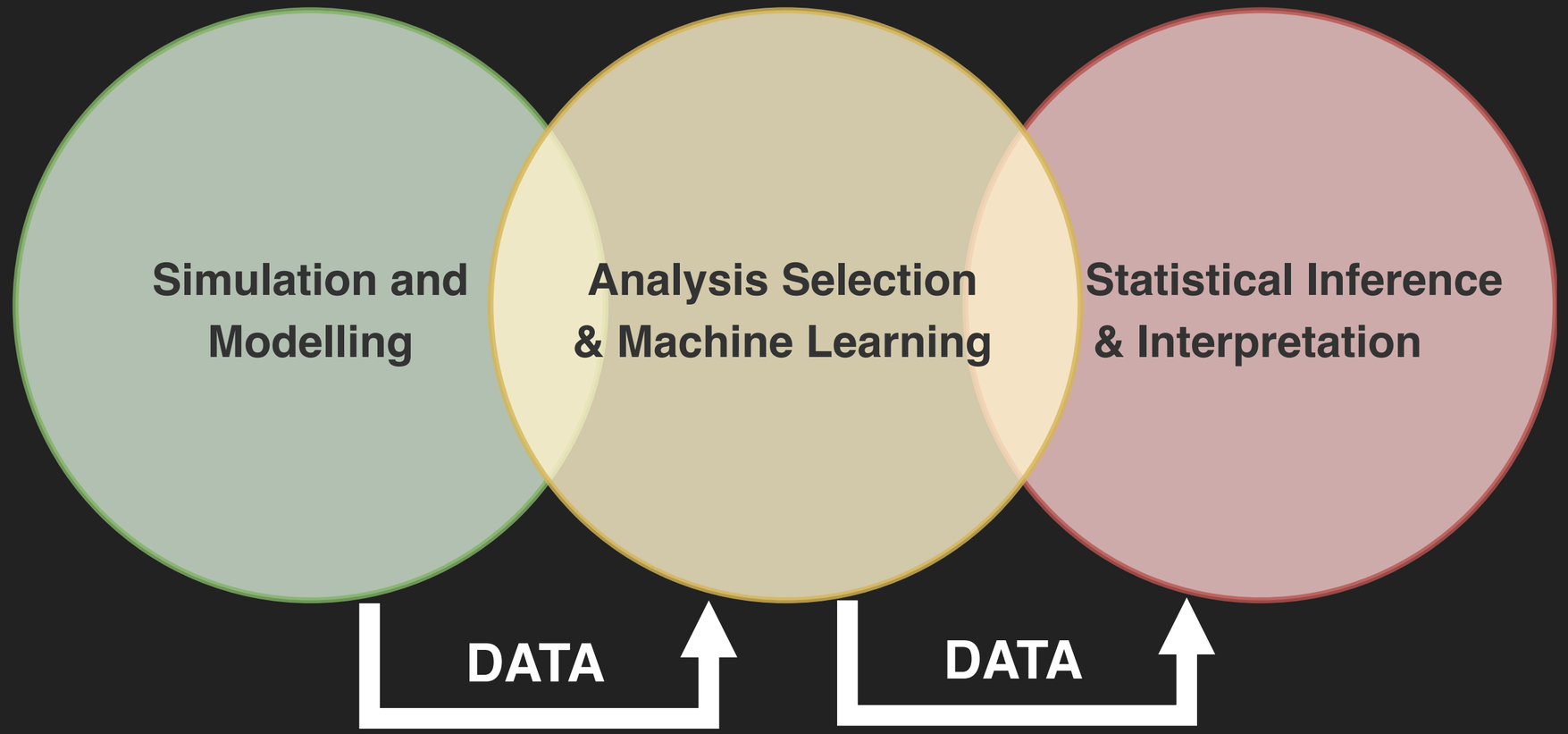


UPPER LIMIT OF ML USEFULNESS IN LHC ANALYSES

Classifiers can be made pivotal as described in "Learning to Pivogn" by G. Louppe et al. A review/benchmarks on how to deal with systematics when using machine learning can be found in Adversarial learning to eliminate systematic errors: a case study in High Energy Physics by Victor Estrade et al NIPS2017.

CAN WE PUT IT ALL TOGETHER?

Embedding some of the knowledge about modelling and statistical inference such as systematic uncertainties in the dimensionality-reduction step with machine learning



GLUE → AUTODIFF GRAPH FRAMEWORKS

CHEAP AND EXACT DERIVATIVES ARE A BIG DEAL!

Autodiff Frameworks (e.g. TensorFlow)

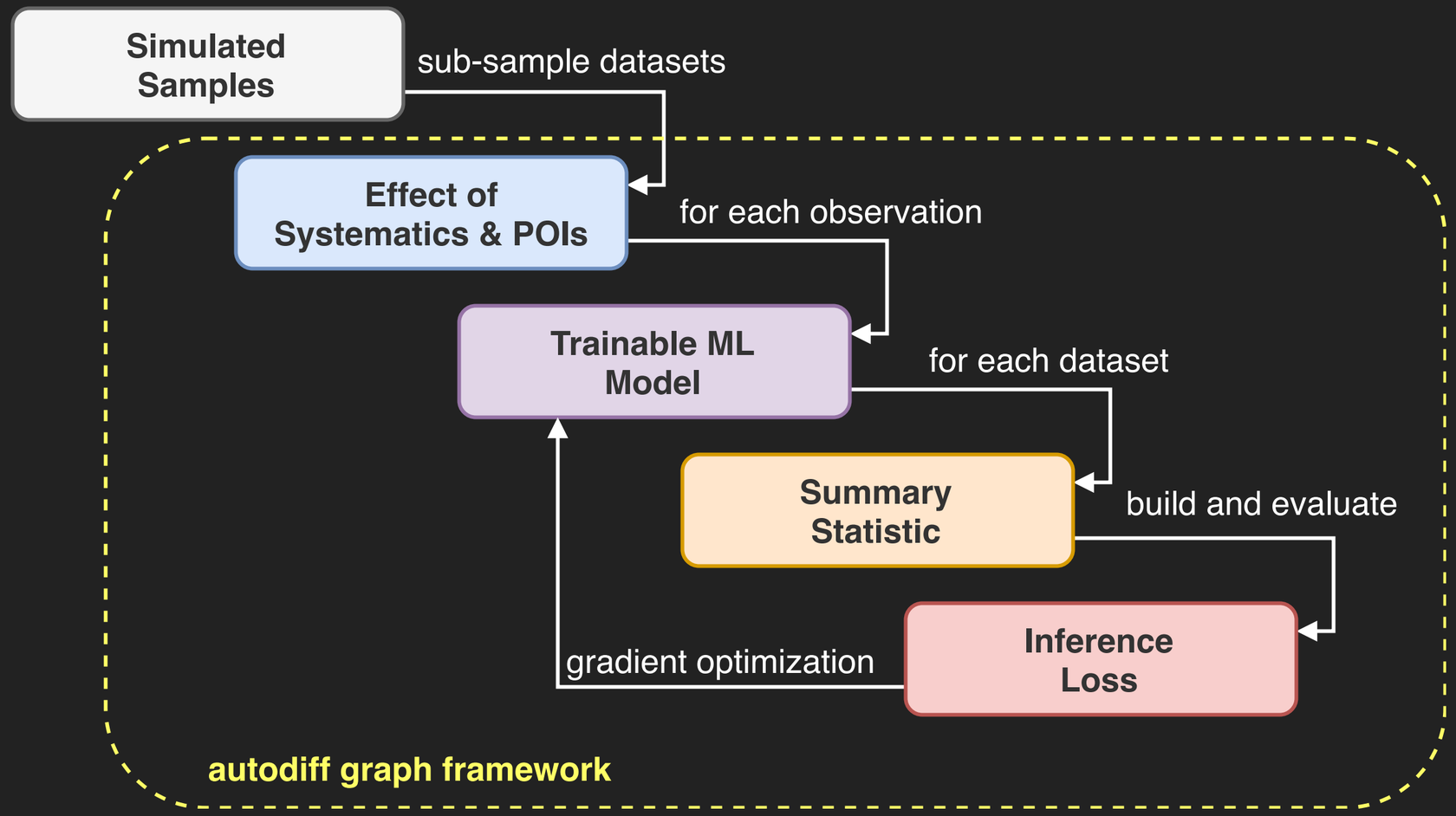
- Highly parallel (CPU/GPU) or distributed
- Support higher order-gradients
  - Hessian very useful for inference
- **Statistical libraries available**
  - TensorFlow Distributions
  - Edward → probabilistic modelling
- **DNNs and Stochastic optimization**

```
import tensorflow as tf
ds = tf.contrib.distributions

bkg = ds.MultivariateNormalFullCovariance(loc=[2.,
        covariance_matrix=[[5.,0.],[0.,9.]],
        name="bkg")
sig = ds.MultivariateNormalDiag(loc=[0., 0.],
        scale_diag=[1., 1.],
        name="sig")

mu = tf.placeholder(shape=(),
        dtype=tf.float32, name="mu")
mix = ds.Mixture(cat=ds.Categorical(probs=[1.-mu, m
        components = [bkg, sig], name="mix")
```

Code specifying 2D Gaussian mixture synthetic model used all along the talk

Autodiff graph frameworks are also useful to speed up common inference tasks such as likelihood fits, toy generation or limit setting. Non machine learning uses within HEP include TensorFlowAnalysis or pyhf.

END-TO-END DIFFERENTIABILITY FOR LHC ANALYSES



Within this general framework, several approaches are possible, focus here is

DIRECT LEARNING OF SYSTEMATICS-AWARE SUMMARY STATISTICS

MODELLING THE EFFECT OF SYSTEMATIC UNCERTAINTIES

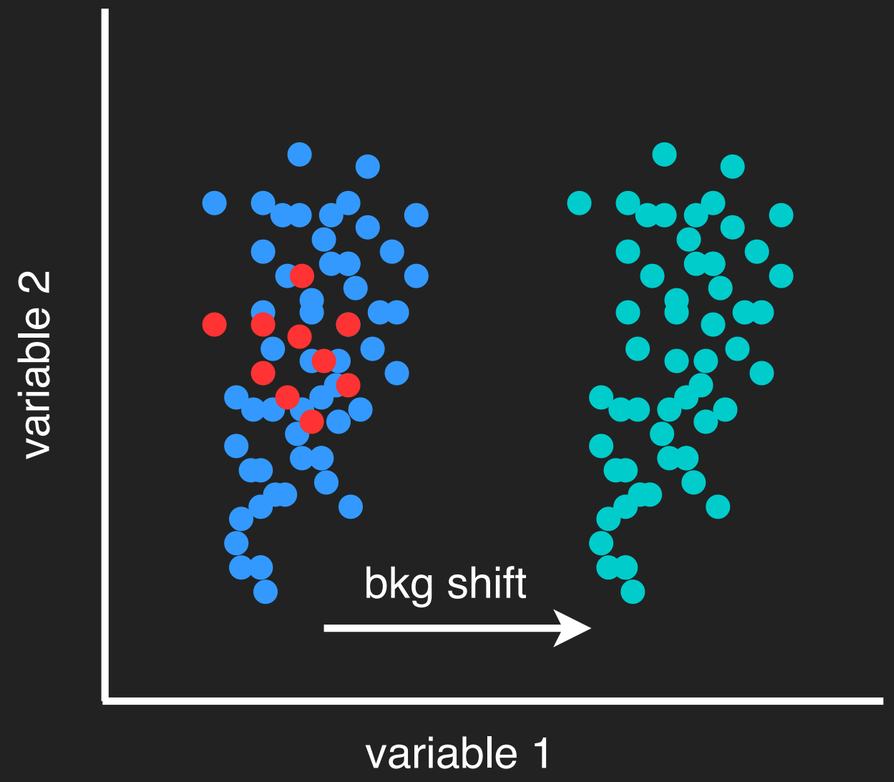
Differentiable approximation of the effect of parameters of interest θ and nuisance parameters ν over the training datasets

A (potentially non-linear) function that depends on the details of problem and transforms event features and/or weights

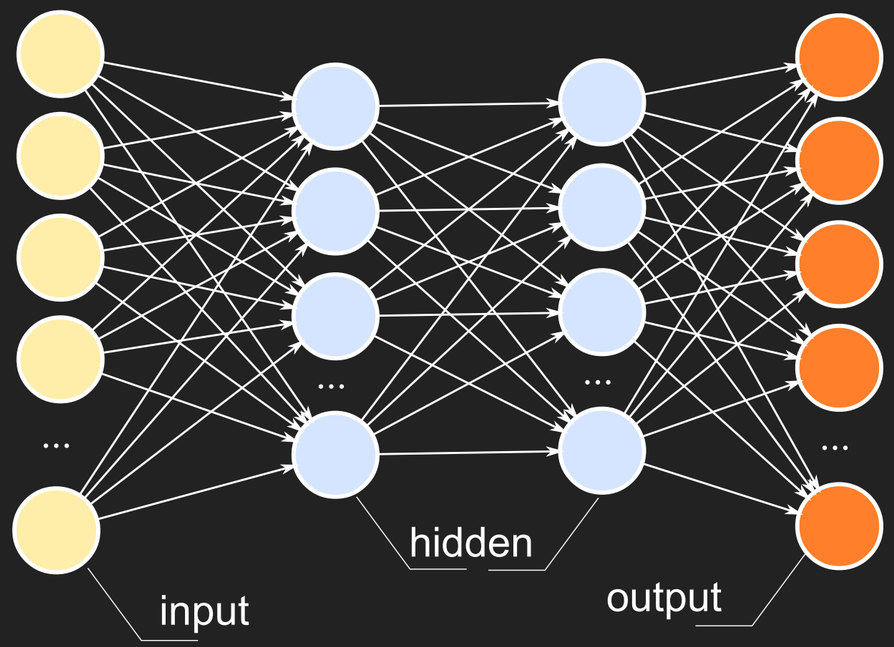
- jet/tau/muon energy scale
- PDF/QCD uncertainties

Could also depend on simulation/latent variables, such the event category (S/B)

Simple example: shift for background observations (i.e. $\mathbf{x}' = \mathbf{x} + \nu \cdot \mathbf{s}$)



TRAINABLE PARAMETRIZED MODEL

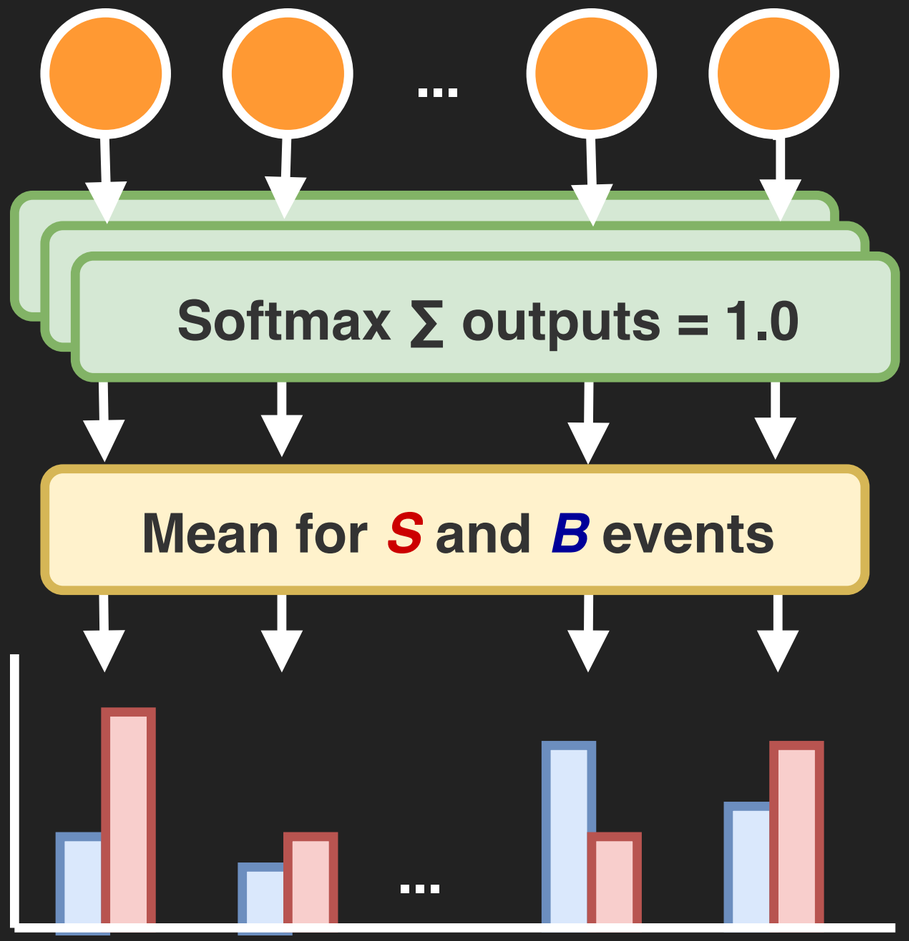


Any (deep) neural network will do

Could in principle re-use the same tweaks, techniques and architectures as for standard supervised Deep Learning

A small two-hidden layer MLP (10 units each, ReLU activation, glorot_normal initd) has been used for initial attempts

NEURAL NETWORK OUTPUT → SUMMARY STATISTIC



We can approximate a histogram-like summary statistic from the NN output adding applying softmax for each event and summing over each dataset

$$\mathcal{L}(\theta,\nu) = \prod_{c \in \text{bins}}^{C} \text{Pois}(n_c | \alpha_s \cdot s_c + \alpha_b \cdot b_c)$$

The likelihood depends both on the neural network parameters and the statistical model parameters

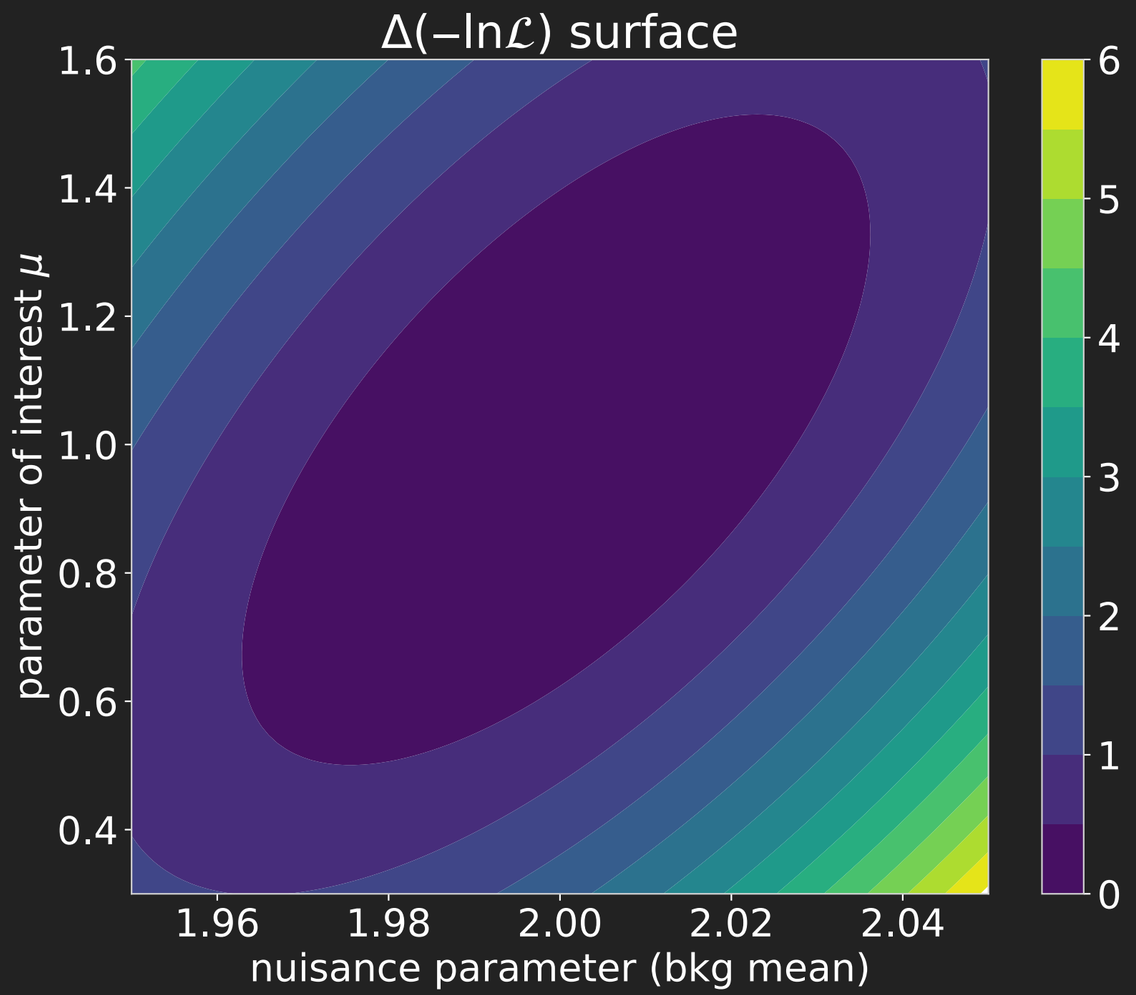
INFERENCE-MOTIVATED LOSS FUNCTION

If we expand the negative log-likelihood around minimum (e.g. Asimov $n_c = \alpha_s \cdot s_c + \alpha_b \cdot b_c$), due to Cramér-Rao bound:

$$\text{covariance} \geq \mathbf{H}^{-1}(-\ln \mathcal{L})$$

which can be computed via autodiff. Can use as loss function directly the variance bound on the parameters of interest

$$\text{loss} \approx \text{Var}(\mu) \quad \text{(expected)}$$



WIP: SYNTHETIC EXAMPLE IMPLEMENTATION

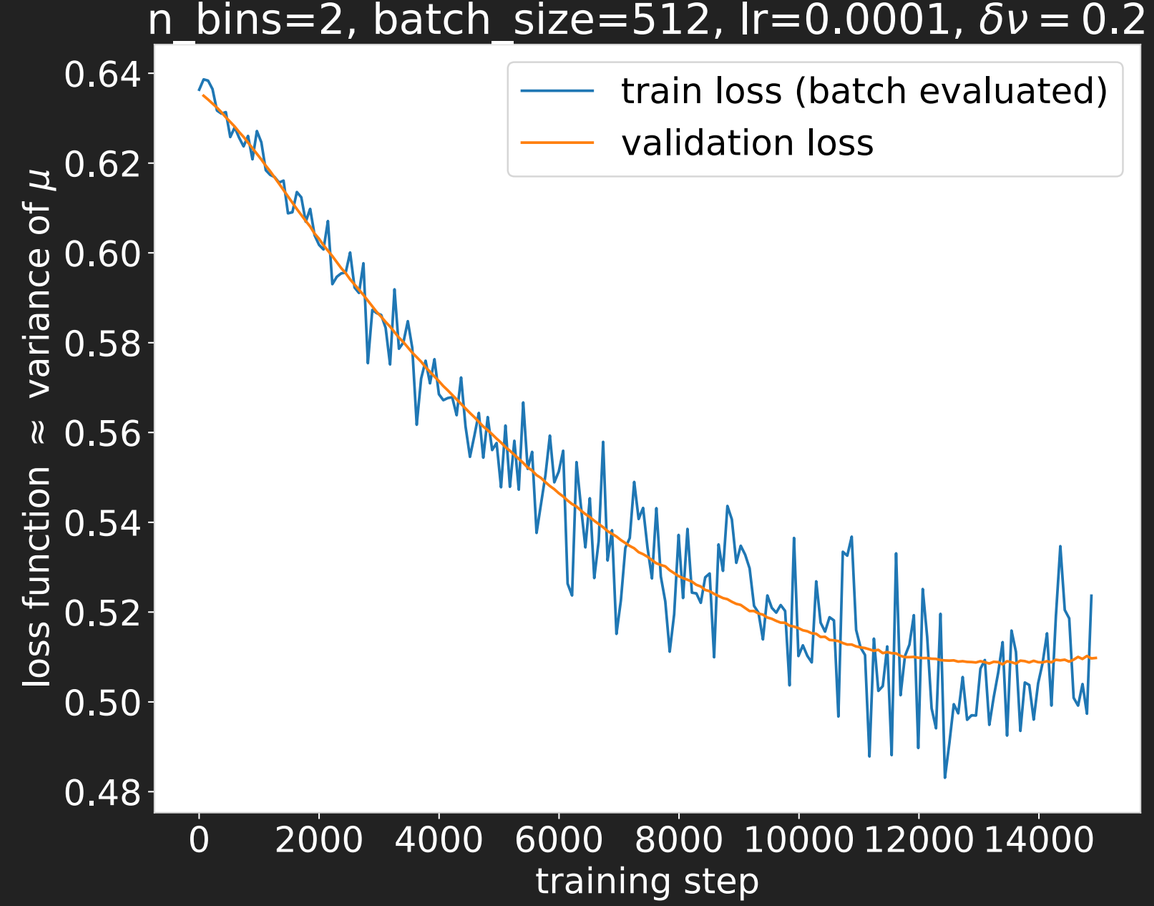
Applied on 2D Gaussian two-component mixture toy dataset (same as previous slides), with unknown background mean in one of the coordinates → **nuisance parameter**

Loss is non-decomposable, because it is dataset-based instead of event-based

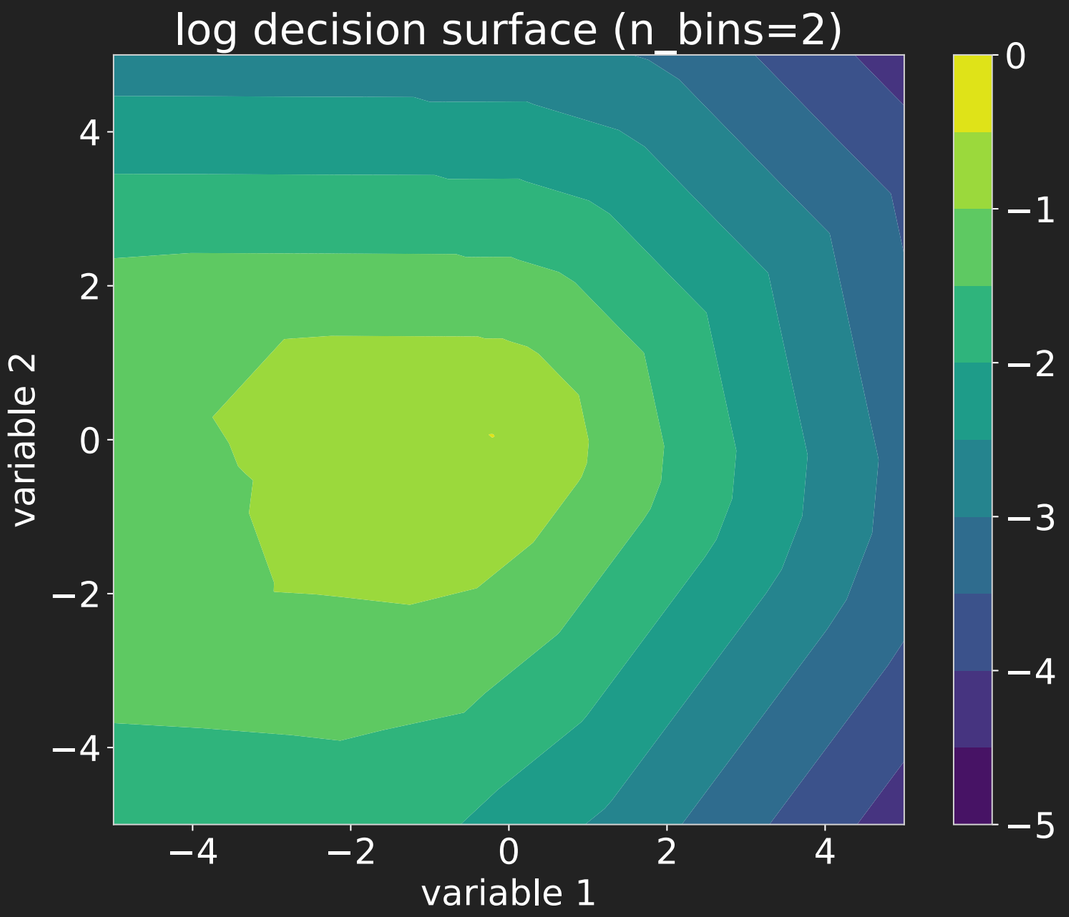
Number of simulated events sub-sampled for each mini-batch is especially relevant to regulate gradient variance

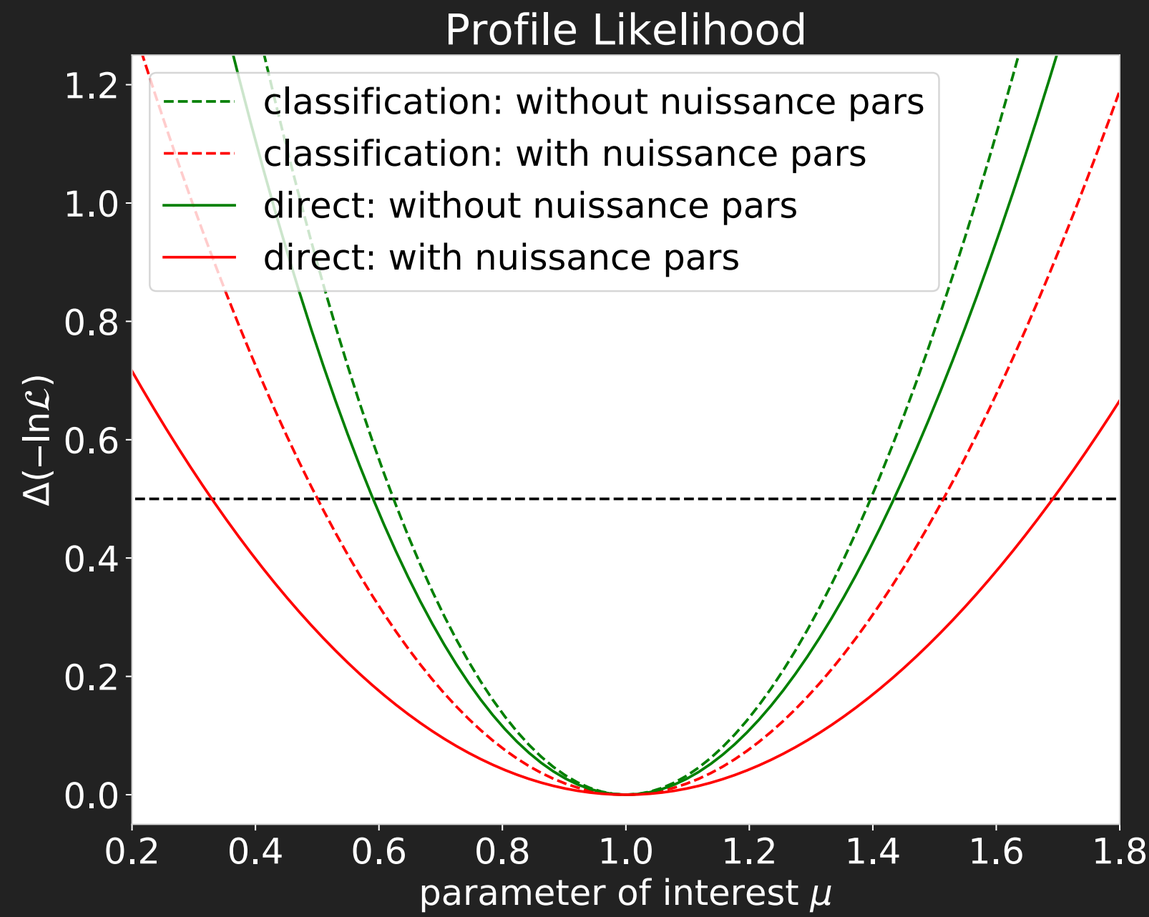
Learning rate is also a critical hyper-parameter, optimization easily diverges

Can also accommodate auxiliary arbitrary likelihood constraints on nuisance pars



WIP: COMPARISON WITH CLASSIFICATION-BASED APPROACH





SOFTER DECISION SURFACE

MORE TWEAKING IS NEEDED

Early results are encouraging, but more studies on more complex problems are needed to evaluate the adequacy of this approach and benchmark against classifer-based inference

CONCLUSIONS AND PROSPECTS

Presented a machine learning approach that directly optimises an inference-guided non-decomposable loss accounting for the effect of model uncertainties

Flexibility of current autodiff frameworks allows the inclusion of nuisance parameters effect (via derivatives) over the training batches

The application of this approach to a realistic systematics-dominated benchmark (e.g. systematic-extended Higgs benchmark dataset) could shed some light on its real-world usefulness

Working on a (more formal) write-up of this technique to be released together with example code (custom TensorFlow Estimator)

THANK YOU
FOR YOUR ATTENTION