LEARNING LIKELIHOODS

AI@INFN Workshop
Bologna

03 May 2022

Riccardo Torre

INFN Genova

Istituto Nazionale di Fisica Nucleare

Based on 1911:03305 [hep-ph], 2 master theses, and work in progress

The DNNLikelihood: enhancing likelihood distribution with Deep Learning

Andrea Coccaro[a], Maurizio Pierini[b], Luca Silvestrini[b,c], and Riccardo Torre[a,b]

[a] *INFN, Sezione di Genova, Via Dodecaneso 33, I-16146 Genova, Italy*

[b] *CERN, 1211 Geneva 23, Switzerland*

[c] *INFN, Sezione di Roma, P.le A. Moro, 2, I-00185 Roma, Italy*

The Likelihood Function

Bayes Theorem:

$$\mathcal{P}(\mathrm{data}|\mathrm{pars})\mathcal{P}(\mathrm{pars}) = \mathcal{P}(\mathrm{pars}|\mathrm{data})\mathcal{P}(\mathrm{data})$$

Likelihood function — Prior probability — Posterior probability — Bayesian evidence

Frequentist inference	Bayesian inference
e.g. Maximum Likelihood Estimation (MLE)	e.g. Maximum A Posteriori (MAP)

The Likelihood Function (LF) is the central object in statistical inference!

Distributing likelihoods: existing proposals

Likelihoods produced by modern experiments (in HEP, but not only) are becoming more and more complicated to treat, sample, maximize, distribute, analyze, and combine

Information loss is generally inevitable in performing analyses and presenting results

Different approaches correspond to different levels of information loss

Examples are:

1. Present just 1D-2D plots and numbers with errors

2. Differential measurements with uncertainties and correlations

3. Simplified Likelihood: parametrize the likelihood in terms of “combined” nuisance parameters using Gaussian approximation up to 3rd moment

4. HistFactory framework (ATLAS): this is going towards publishing all information that allows to reconstruct the full likelihood (only binned) and is supported by a Python framework (Pyhf)

5. …?

Distributing likelihoods: our approach

Our approach: encode the full likelihood with all the dependence on elementary nuisance parameters into a DNN function. This allows for:

Supervised learning

$$\vec{x} = (\vec{\mu}, \vec{\delta}) \longrightarrow y = \mathcal{L}(\vec{\mu}, \vec{\delta})$$

DNN Interpolator

$$\mathcal{L}_{\mathrm{DNN}}(\vec{\mu}, \vec{\delta})$$

1. Encoding also unbinned likelihoods
2. Very fast sampling (up to several order of magnitude faster than traditional methods)
3. Re-sampling with custom priors to study the impact of different hypotheses on nuisance parameters (systematic uncertainties)
4. Efficient combination of different likelihoods (when correlations are known)
5. Interpretation of results within different statistical approaches (Frequentist vs Bayesian)
6. Simple framework-independent distribution through the ONNX format, which ensures portability to any software environment (Python, R, Matlab, Mathematica, etc..)

A toy example

Toy experiment already considered in the literature

Buckley, Citron, Fichet, Kraml, Waltenberger, Wardle, 1809.05548 [hep-ph]

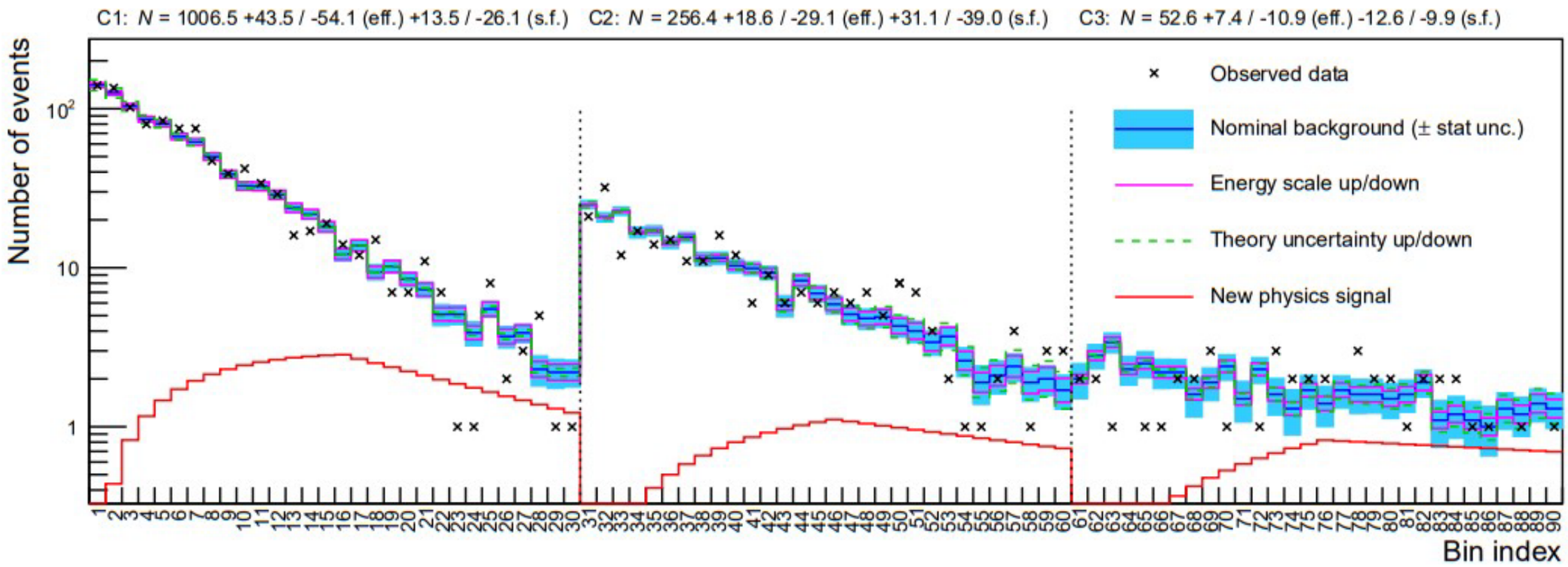


Figure 2. LHC-like search for new physics (mockup). The search is performed across three event categories, each divided into 30 bins to make a total of 90 search regions. The nominal expected contribution in each bin from the background and from the new physics signal is shown by the blue and red lines, respectively. The solid and dashed lines show the $\pm 1\sigma$ correlated variation in each bin expected due to an experimental and theoretical uncertainty while the blue shaded band shows the uncorrelated uncertainty in each bin due to limited MC simulation. The "observed" number of events in data in each bin is indicated by the black points.

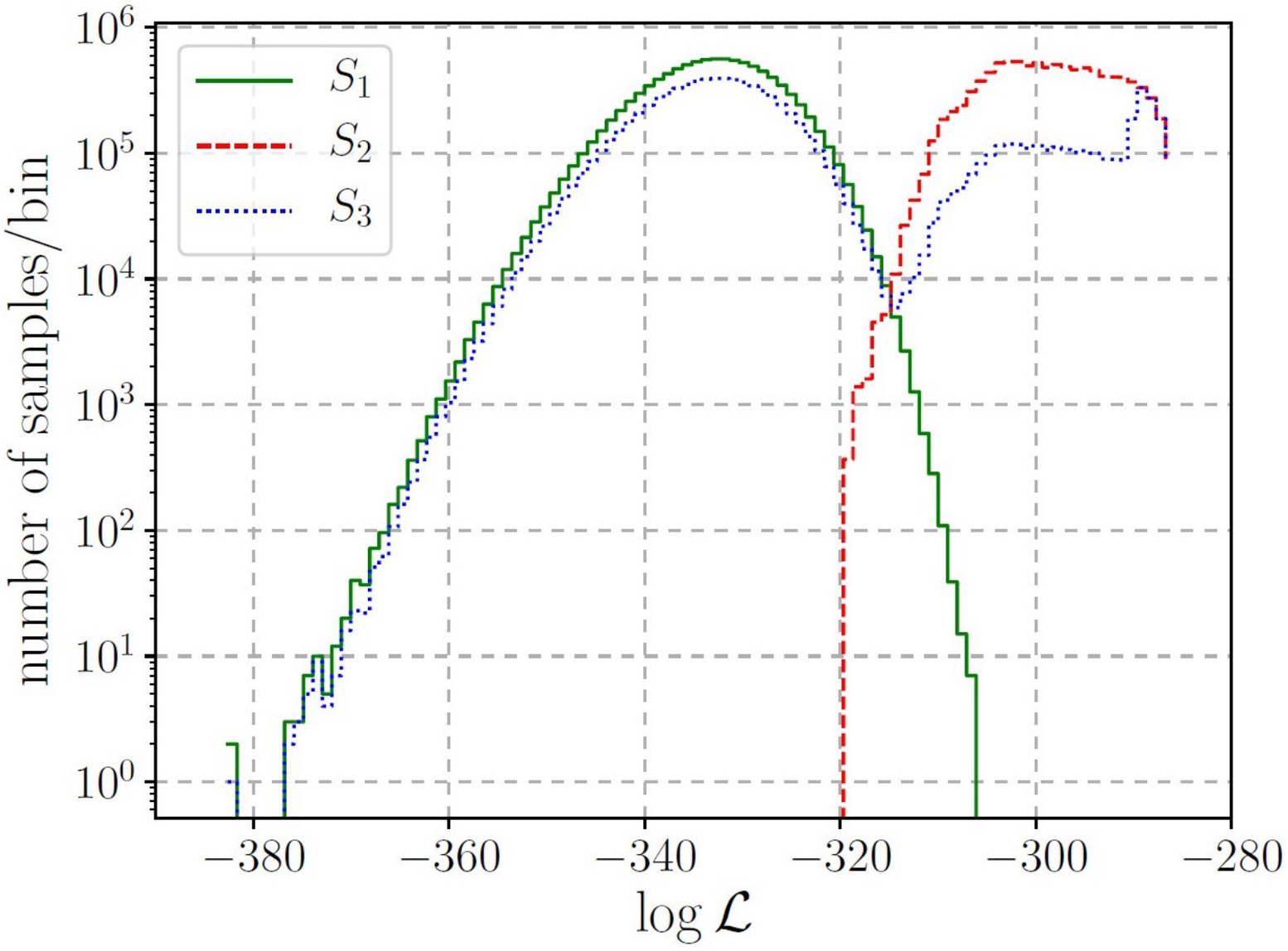
- One physical parameter (signal strength μ)
- 94 nuisance parameters (90 fully uncorrelated, two fully correlated, two normalizations)
- Non Gaussian Likelihood

Sampling

Supervised learning problem (interpolation) where high precision is needed

If we want to allow for both Bayesian and Frequentist inference, we need to know the LF well in very different regions (where prior volume is large and close to local maxima of the LF, respectively)

We sampled with the emcee3 MCMC (ensemble sampling method) Python package (checking convergence with several different techniques)

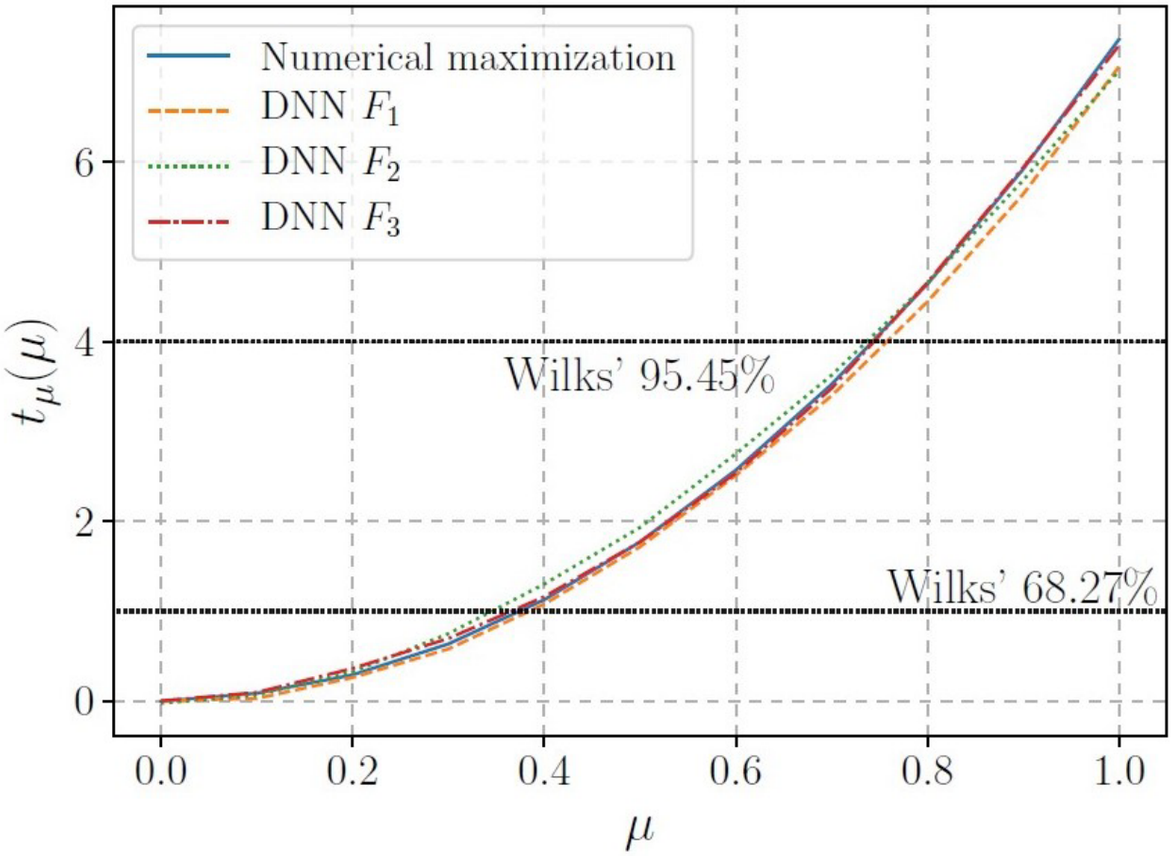


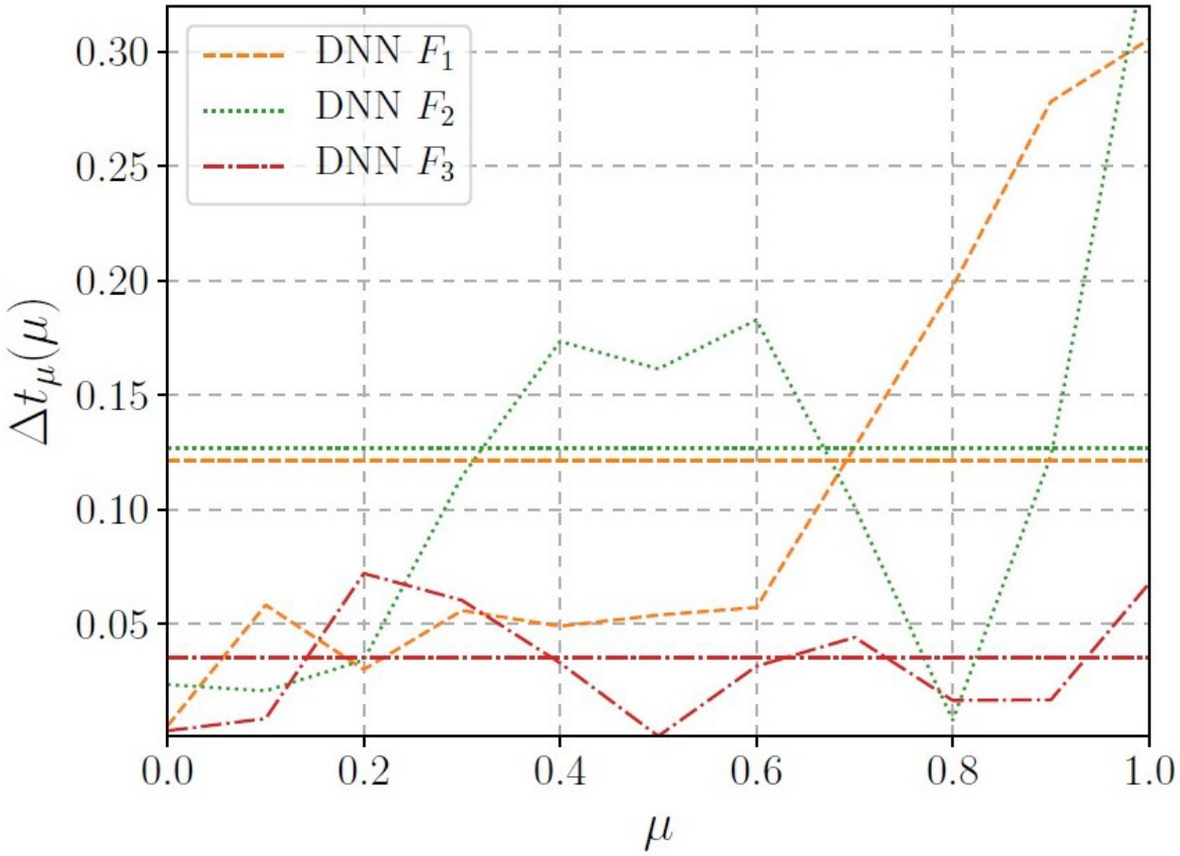
Inference with DNNL: Frequentist

For frequentist inference we construct the test statistics

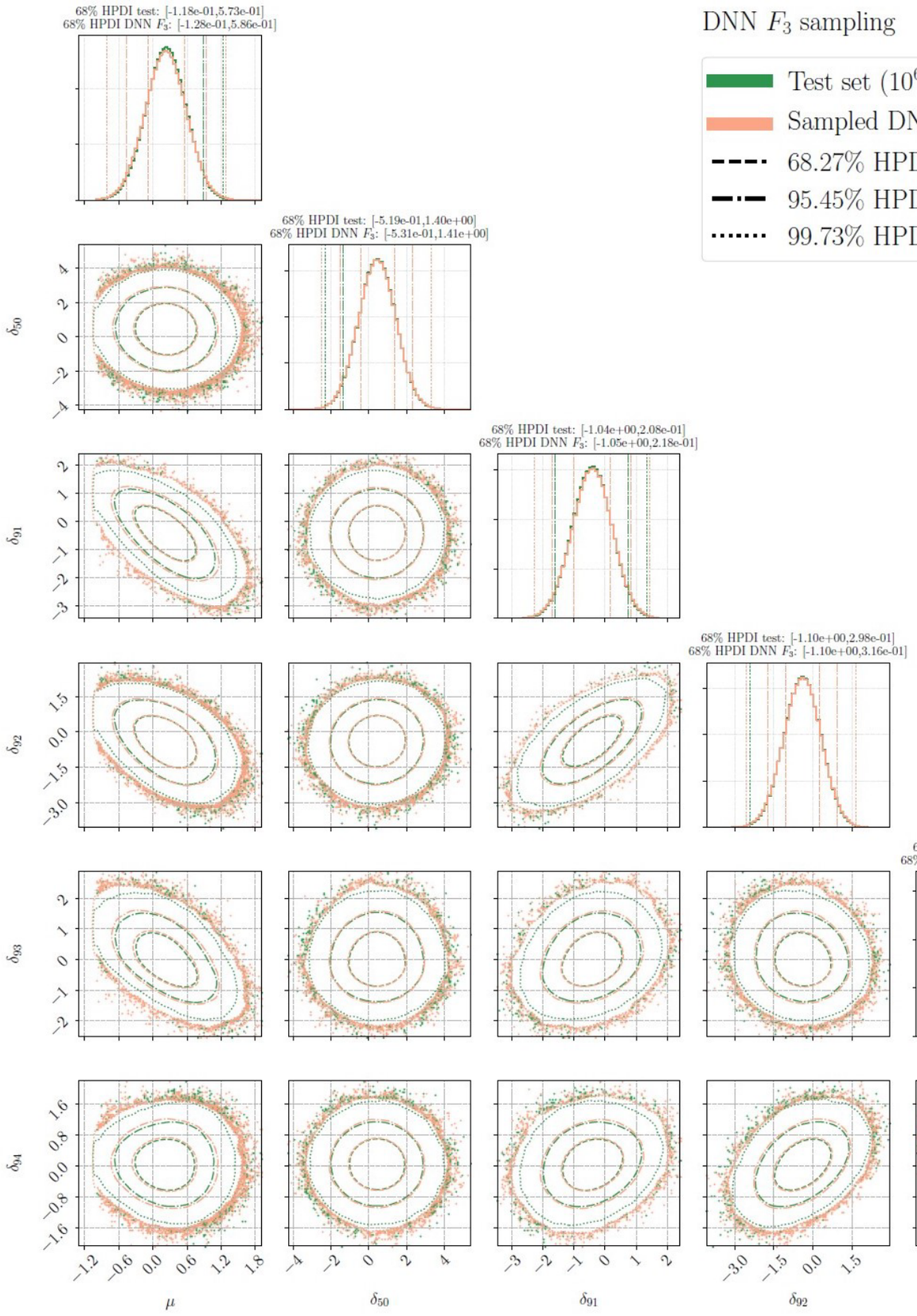
$$t_\mu(\mu) = -2\log\frac{\mathcal{L}_{\text{prof}}(\mu)}{\mathcal{L}_{\max}}$$

The DNNLikelihood allows to reproduce with great precision the results of Frequentist inference





Inference with DNNL: Bayesian



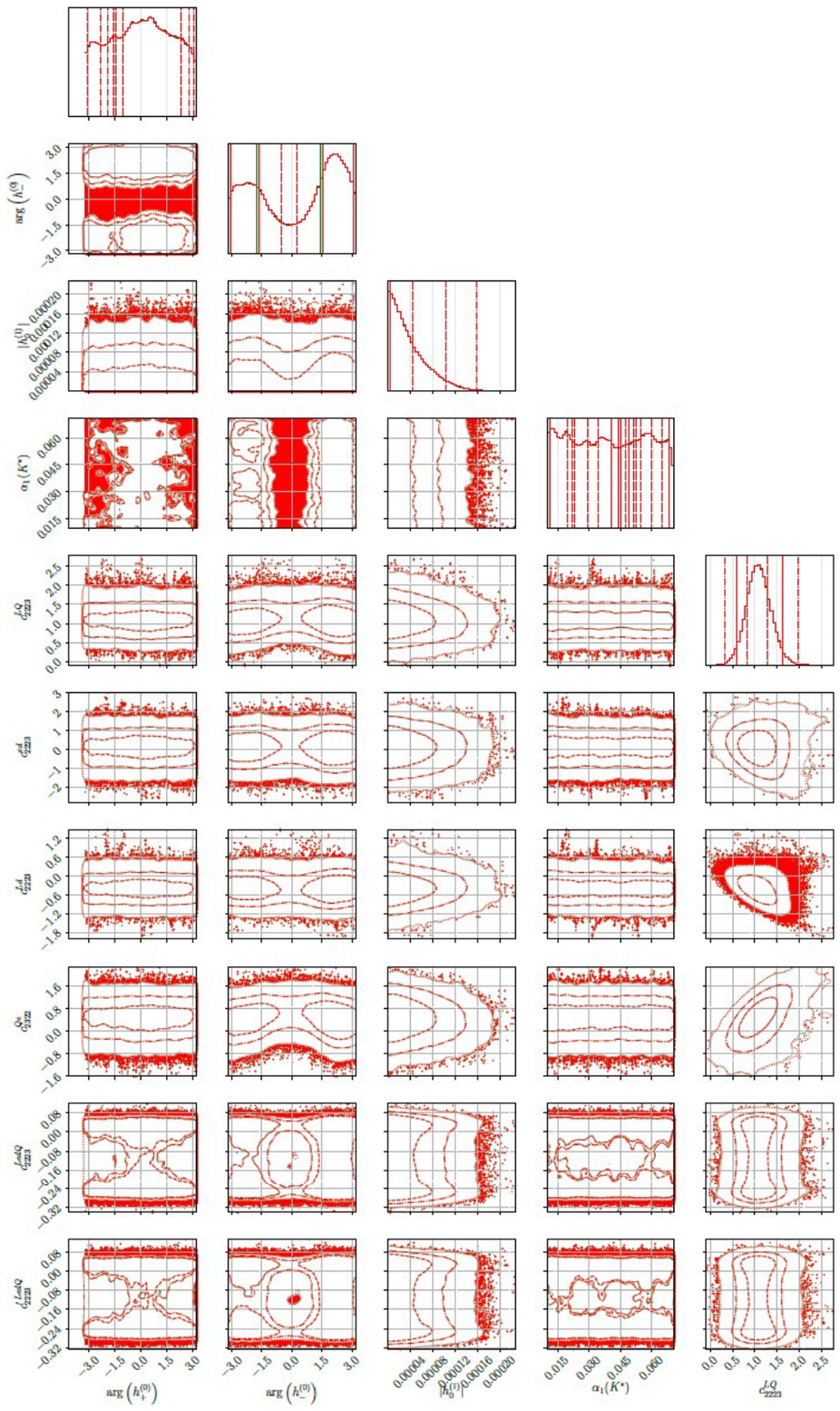
Bayesian inference is based on (marginal) posterior probability distributions

We quote results as marginalized Highest Posterior Density Intervals (HPDI) in 1D and 2D

HPDI	$\mu > -1$	$\mu > 0$	F_3
68.27%	$[-0.12, 0.58]$	0.48	0.48
95.45%	$[-0.47, 0.92]$	0.86	0.88
99.73%	$[-0.82, 1.26]$	1.22	1.28

Real-life example 1

Master Thesis, Enzo Canonero, 2021



Likelihood for EFT interpretation of flavor observables in B mesons neutral current decays (available and sampled through HEPFit)

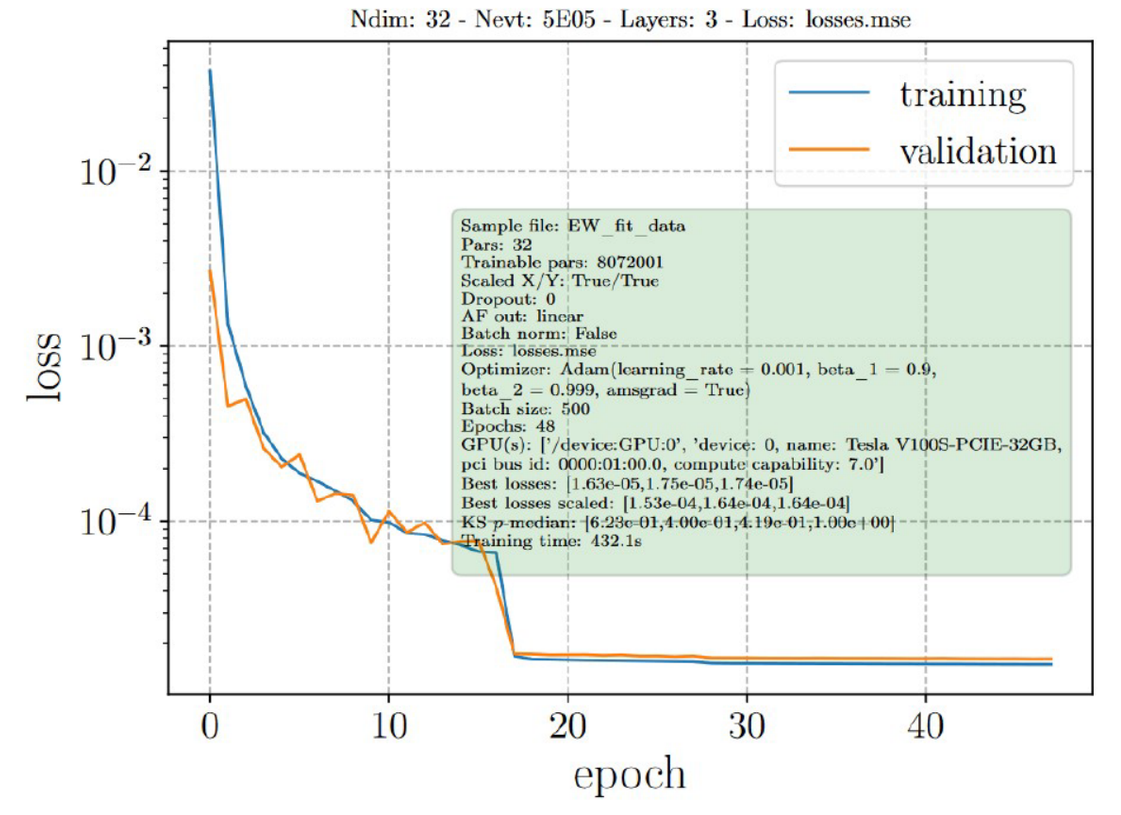
83 parameters (6 poi, 77 nuis), highly non-Gaussian/multimodal

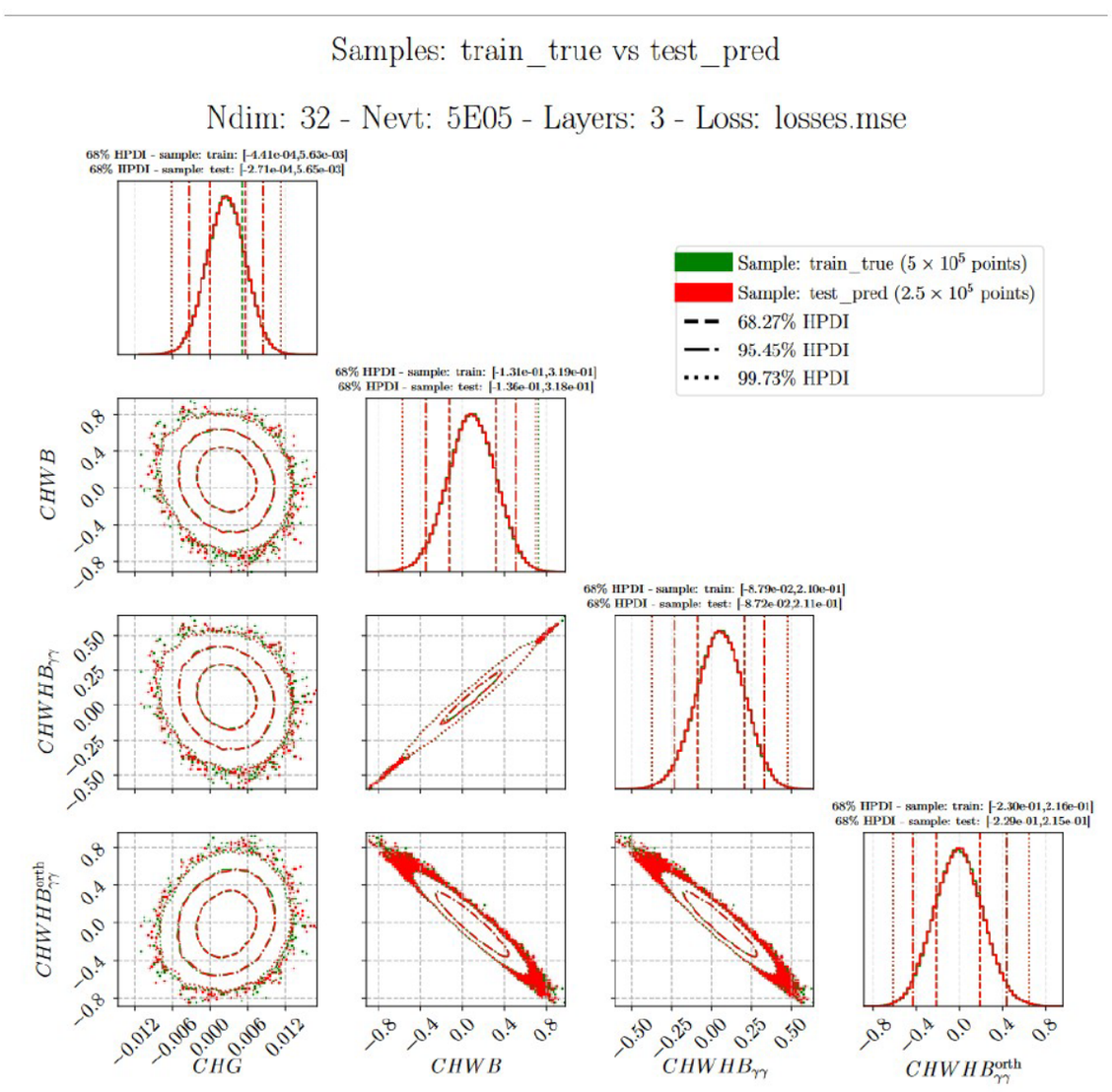
Real-life example 2

Master Thesis, Alberto Scibilia, 2022

Likelihood for EFT interpretation of EW observables (available and sampled through HEPFit)

32 parameters (17 poi, 15 nuis, highly correlated observables)





Conclusions

- Great effort in the HEP community to bring ML tools into our workflow
- The DNNLikelihood framework provides a solution to encode, distribute, combine, analyze, and preserve high dimensional and complicated Likelihood functions
- It works extremely well without the need of too much hyperparameters tuning or advanced techniques (even in the highly correlated, highly non-Gaussian, and multimodal cases we studied)
- Together with the DNN models our code always produces several auxiliary files keeping track of all parameters, metrics, results, etc. so that each model is carefully self-documented
- A comprehensive Python package that allows to sample likelihoods, build DNN models, optimize them, and analyze the results within different statistical tools is in the last stage of development. It can be found on GitHub and is documented here.
- Real world examples confirm the simplicity and efficiency of the approach
- Extension to unsupervised DNNLikelihood (based on Normalizing Flows for density estimation) is under study

THANK YOU!