Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Flow-Based Surrogates for High-Dimensional Likelihoods in Experimental Neutrino Physics

Mathias El Baz

mathias.elbaz@unige.ch

University of Geneva https://orcid.org/0000-0002-9562-3897

Lorenzo Giannessi

https://orcid.org/0000-0001-6899-7116

Adrien Blanchet

Federico Sánchez

University of Geneva https://orcid.org/0000-0003-0320-3623

Article

Keywords: Neutrino physics, Long-baseline oscillation experiments, Systematic uncertainties, Likelihood-based inference, Normalizing flows

Posted Date: August 19th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10319007/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: There is **NO** Competing Interest.

Flow-Based Surrogates for High-Dimensional Likelihoods in Experimental Neutrino Physics

Mathias El Baz [1*†], Lorenzo Giannessi [1*†], Adrien Blanchet [2] and Federico Sánchez [1]

[1*]Section de Physique, DPNC, University of Geneva, Geneva, Switzerland.
[2]Laboratoire de Physique Nucléaire et de Hautes Énergies, LPNHE, CNRS/IN2P3, Sorbonne Université, Université Paris Cité, Paris, France.

*Corresponding author(s). E-mail(s): mathias.elbaz@unige.ch; lorenzo.giannessi@unige.ch;
Contributing authors: adrien.blanchet@lpnhe.in2p3.fr;
†These authors contributed equally to this work.

Keywords: Neutrino physics, Long-baseline oscillation experiments, Systematic uncertainties, Likelihood-based inference, Normalizing flows

Precision long-baseline neutrino experiments use near-detector data to constrain systematic uncertainties on the unoscillated neutrino flux, a prerequisite for accurate oscillation-parameter measurements at the far detector. When the constrained likelihood is high-dimensional and non-Gaussian, this procedure demands advanced statistical treatment. Here we show that normalizing flows provide faithful and portable likelihood models for this problem. Leveraging an initial Gaussian approximation of the likelihood, we train a hybrid architecture combining coupling transformations and autoregressive spline flows. We demonstrate the method on a representative near-detector likelihood replica with 110 systematic uncertainty parameters, 10 of which explicitly introduce non-Gaussianities in the constrained likelihood. The trained model achieves a relative effective sample size of 98%, compared with about 5% for the Gaussian approximation, and reproduces a Markov Chain Monte Carlo reference while remaining closed-form, samplable, and pointwise evaluable, making it suitable to uncertainty propagation in downstream physics analyses.

1 Introduction

Long-baseline neutrino-oscillation experiments aim to measure the leptonic mixing angles, the neutrino mass ordering, and the charge–parity (CP) violating phase that may help explain the matter–antimatter asymmetry of the Universe [1]. As these measurements sharpen, the experiments have entered a precision regime in which the limiting factor is often no longer statistical power alone, but the faithful modeling and propagation of systematic uncertainties. Flux predictions, neutrino-nucleus interaction models, detector efficiency and reconstruction effects are constrained through likelihood fits, in which nuisance parameters are adjusted so that the predicted event rates best describe control data. These fits can involve large numbers of nuisance parameters, whose impact on predicted spectra is often correlated and non-Gaussian [2–9]. Moreover, these fits depart from the familiar setting in which a few parameters of interest are isolated by marginalizing away the rest. Here both the parameters of interest and the nuisance parameters are numerous, so eliminating the nuisances is a genuinely hard problem [10], and the constrained uncertainties it yields must enable or enhance downstream correlated analyses, as happens in the near-detector–far-detector fit scheme typical of long-baseline experiments. In this context, the quantity that must be carried from one stage of the analysis to the next is the constrained likelihood over the systematic parameters.

Within a single long-baseline experiment, near-detector constraints must be propagated to the far-detector prediction before the oscillation parameters are extracted. More demanding still, recent combinations across experiments, such as the joint T2K-NOvA oscillation analysis and the Super-Kamiokande-T2K atmospheric-accelerator fit, require the constraints obtained in one analysis to enter the likelihood of another [11–14]. In such scenarios, a fit result is only as useful as the fidelity with which its constrained likelihood can be represented and reused. This mirrors a broader effort in particle physics to publish full statistical models rather than summary results [15], so that the information content of a fit is preserved.

The question addressed here is therefore how this constrained likelihood should be exported for downstream tasks. By a portable likelihood representation, we mean a compact object that can be distributed independently of the original analysis software, evaluated at arbitrary points in parameter space, sampled efficiently, and inserted into downstream likelihoods without rerunning the upstream fit or accessing the original data. Such a representation must preserve the relevant non-Gaussian structure of the original likelihood while remaining practical to use in far-detector predictions, joint fits, or global analyses.

In current oscillation analyses, this propagation is often performed by replacing the constrained likelihood with a local Gaussian approximation around its maximum [6, 8, 9]. This so-called post-fit Gaussian is centered on the best-fit nuisance parameters, with a covariance matrix inferred from the curvature of the likelihood at that point. This representation is compact and easy to reuse, offering a continuous closed-form likelihood, but it assumes that the log-likelihood is quadratic and that the relevant systematic variations are adequately described by an ellipsoid in parameter space. Non-linear parameter dependencies, physical boundaries, limited-statistics

effects [16], and degeneracies can instead generate asymmetric marginals, curved correlations, and non-Gaussian tails. Markov Chain Monte Carlo methods [17, 18] provide a more faithful sampling of such likelihoods, but a finite chain is not a closed-form continuous likelihood function, therefore limiting the practicality of its use for constraint propagation. Full experiment-specific likelihood implementations provide the most faithful representation, but require access to dedicated software, detailed analysis input, and experiment-specific infrastructure, making them difficult to distribute or insert into independent downstream analyses [19]. This leaves a methodological gap between Gaussian summaries, which are portable but potentially biased, MCMC samples, which are faithful but not directly evaluable, and full likelihood implementations, which are accurate but heavy.

In this work, we address this gap using normalizing flows [20, 21] (NF) as explicit surrogate models for high-dimensional systematic likelihoods. Normalizing flows transform a simple base distribution through a sequence of invertible learnable maps, producing a density model that is both directly samplable and pointwise evaluable. Once trained, the flow is not only a generator of nuisance-parameter samples, but a continuous surrogate for the constrained likelihood itself: it can be sampled, evaluated, and inserted into downstream likelihoods on which gradient-based optimization can be performed, without rerunning the original fit nor accessing the original data. Flow-based and, more broadly, simulation-based inference methods have become a widely adopted tool across the physical sciences [22–25]. Applications include posterior estimation in gravitational-wave astronomy and cosmology [26–28], fast and high-fidelity detector simulation [29, 30], and asymptotically exact sampling of lattice field theories [31–33]. Within neutrino physics, related ideas have been explored for oscillation inference, event generation and detector-calibration likelihood modeling [34–39].

We demonstrate this approach on a realistic high-dimensional statistical-analysis replica inspired by the near-detector fit of the currently running T2K experiment, where the modeling of systematic uncertainties involves $\mathcal{O}(10^2)$ fit parameters constrained by near-detector data. This benchmark provides a controlled setting in which the target likelihood is known, non-Gaussian features can be isolated, and direct comparisons can be made with both the post-fit Gaussian approximation and MCMC sampling. We study the accuracy of the trained flow as a likelihood surrogate, its sampling efficiency, and its impact on propagated flux predictions. Finally, we discuss its possible use as a portable constrained likelihood model for downstream oscillation analyses.

2 Results

2.1 Normalizing flows as likelihood surrogates

Our aim is to find a continuous density model of the likelihood function that can be sampled, evaluated, and used for downstream tasks. The target density is the normalized systematic likelihood,

$$p(\boldsymbol{\eta}) \propto \mathcal{L}(\boldsymbol{\eta}), \tag{1}$$

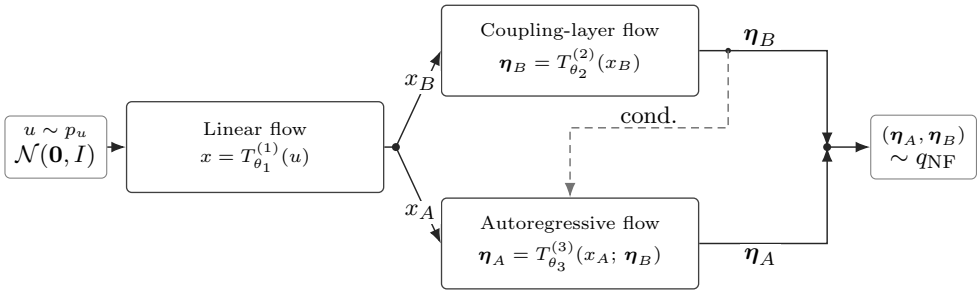


Fig. 1: Hybrid normalizing-flow architecture with level-of-detail approach used to model $q_{\mathrm{NF}}(\boldsymbol{\eta}_A, \boldsymbol{\eta}_B) = q_{\mathrm{AR}}(\boldsymbol{\eta}_A \,|\, \boldsymbol{\eta}_B)\, q_{\mathrm{CL}}(\boldsymbol{\eta}_B)$. Base samples $u \sim \mathcal{N}(\mathbf{0}, I)$ pass through a linear flow $T^{(1)}_{\theta_1}$, optionally initialized from the Gaussian post-fit $(\hat{\boldsymbol{\eta}}, C)$ via $\mu_{\theta_1} = \hat{\boldsymbol{\eta}}$ and $L_{\theta_1} L^{\top}_{\theta_1} = C$. Its output is partitioned as $x = (x_A, x_B)$. The high-dimensional easy block is mapped to $\boldsymbol{\eta}_B$ by a coupling-layer flow $T^{(2)}_{\theta_2}$; the low-dimensional difficult block is mapped to $\boldsymbol{\eta}_A$ by a conditional autoregressive flow $T^{(3)}_{\theta_3}$, with $\boldsymbol{\eta}_B$ feeding the autoregressive conditioner networks (dashed arrow).

where $\boldsymbol{\eta}$ denotes the vector of systematic uncertainty parameters. The normalizing-flow surrogate [20] learns an invertible map $f_\theta : \mathbb{R}^D \to \mathbb{R}^D$ that transports a simple base distribution to this target density. In practice, it is convenient to use the post-fit Gaussian approximation as a starting point. It captures the local structure of the likelihood around the best-fit region and offers an efficient way to generate an initial training dataset. The post-fit Gaussian therefore plays a dual role: it serves both as an initial proposal distribution and as a source of training samples. However, the method does not intrinsically rely on training samples drawn from the Gaussian approximation. The samples used to initialize the training can be chosen according to the expected structure of the constrained likelihood, and may instead be obtained from other sources, such as an existing MCMC chain. In the present application, samples from the post-fit Gaussian provide a convenient starting point.

The surrogate is designed to exploit the structure of the systematic-parameter likelihood. The large, approximately linear sector of the parameter space is modeled with coupling layers [40], while the smaller sector containing the most non-linear systematic parameters is modeled with a conditional autoregressive spline flow [41, 42]. This keeps the expensive autoregressive structure restricted to the directions where non-Gaussian effects are expected to be strongest, while preserving correlations with the remaining high-dimensional block. The corresponding factorization is

$$q_{\mathrm{NF}}(\boldsymbol{\eta}_A, \boldsymbol{\eta}_B) = q_{\mathrm{AR}}(\boldsymbol{\eta}_A \mid \boldsymbol{\eta}_B) q_{\mathrm{CL}}(\boldsymbol{\eta}_B), \tag{2}$$

where $\boldsymbol{\eta}_A$ denotes the non-linear block and $\boldsymbol{\eta}_B$ the high-dimensional approximately linear block. This architecture is summarized in Figure 1, and the full implementation is given in the Methods section 4.4.

2.2 Replicating a typical near-detector likelihood

To benchmark the method, we construct a realistic near-detector likelihood [9] replica designed to reproduce the statistical structure of a typical long-baseline neutrino near-detector analysis. The purpose of this benchmark is to build a controlled likelihood with similar dimensionality and complexity: a large number of correlated systematic parameters, non-linear event-weight variations, finite-statistics fluctuations, and a post-fit covariance matrix obtained from a maximum-likelihood fit.

The simulated samples consist of neutrino and antineutrino interactions on hydrocarbon, generated with a neutrino-energy spectrum inspired by the T2K off-axis beam. The selected samples mimic the topology-based categories used in near-detector analyses, with three samples in forward horn current (FHC) neutrino mode and three samples in reverse horn current (RHC) antineutrino mode, classified according to the number of detected final-state pions. A statistically independent simulated sample is used as pseudo-data. The pseudo-data and the Monte Carlo model are generated using the same underlying neutrino-nucleus interaction model. Details of the event generation, detector smearing, selection thresholds, sample composition, and likelihood construction are given in the Methods section 4.1.

The likelihood is parametrized with 110 parameters representing a set of hypothetical systematic uncertainties. Of these, 100 act as flux-like systematics: they rescale the predicted event rates through binned (50 bins for each beam mode) weights in true neutrino energy, with the event-rate variation in any given analysis bin being linear in the parameters. The remaining 10 are non-linear parameters that emulate interaction-model uncertainties via event-by-event weights depending on truth-level kinematic variables. Because these weights distort the model histograms in ways that do not need to be linear in $\boldsymbol{\eta}$, they can produce likelihood profiles that depart from Gaussianity. Details on the parametrization of the systematic uncertainties model are provided in the Methods section 4.3.

The scale is large enough to reproduce the non-Gaussian structure of a realistic near-detector likelihood and the separation between a high-dimensional approximately linear bulk and a small non-linear block, while remaining a setup for which a self-contained replica can be built from scratch and a well-converged MCMC reference can be obtained for a controlled comparison. The relevant computational scaling of the NF model is governed primarily by the size of the non-linear block, whose impact is quantified in Section 2.5.

2.3 Fidelity in parameter space

We first verify that the trained flow reproduces the target likelihood in the parameter space. As a faithful reference we use a Markov Chain Monte Carlo (MCMC) sampling of the same likelihood, generated with a Metropolis-Hastings algorithm [17, 18]. Being asymptotically exact, the MCMC provides the reference against which the flow is validated throughout this work. The chain consists of 4.2×10^6 samples, of which the first 2×10^5 are discarded as burn-in. The proposal was tuned during burn-in to target the theoretical optimal acceptance probability of 0.234 for random-walk

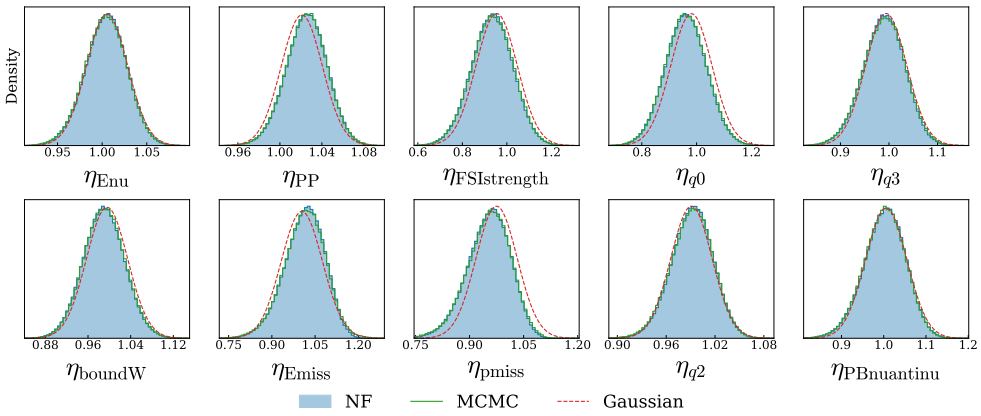


Fig. 2: One-dimensional marginal distributions of the ten non-linear systematic parameters, obtained from 500 000 samples drawn from the trained NF surrogate, 500 000 samples from the MCMC reference, and the post-fit Gaussian approximation. See Methods 4.3 for the parameter definitions.

Metropolis–Hastings [43] (the corresponding optimum is 0.651 for Hamiltonian Monte Carlo [44]).

Figure 2 compares the one-dimensional marginal distributions of the ten non-linear systematic parameters, obtained from 500 000 samples drawn from the trained NF surrogate, 500 000 samples from the MCMC reference, and the analytical post-fit Gaussian approximation. These parameters are the ones expected to develop the strongest departures from Gaussianity. The NF marginals closely follow the MCMC reference across all ten parameters, reproducing their asymmetric tails and modes displaced from the best-fit value, while the Gaussian approximation, symmetric and centered on the best-fit point by construction, fails to capture them.

The non-Gaussian features are most visible in two dimensions: Figure 3, from 500 000 NF samples, shows curved correlations and distorted confidence regions in the off-diagonal panels, for instance between the missing-momentum and missing-energy parameters.

Moreover, the maximum of each marginal distribution of the NF model is shifted with respect to the Gaussian and in agreement with the MCMC. This highlights the potential of the NF model in reducing shortcomings of the Gaussian approximation in terms of the description of the bulk probability mass, reducing inference bias.

2.4 Flux predictions

The trained normalizing flow can be used as a directly samplable surrogate of the full systematic-parameter likelihood, allowing flux predictions to be produced and compared with those obtained from a MCMC sampling of the same likelihood. For each sampled vector of systematic parameters $\boldsymbol{\eta}$, the corresponding systematic response is propagated to the event weights. The reweighted events are then binned in true neutrino energy, separately for samples of neutrino-beam (FHC) and antineutrino-beam

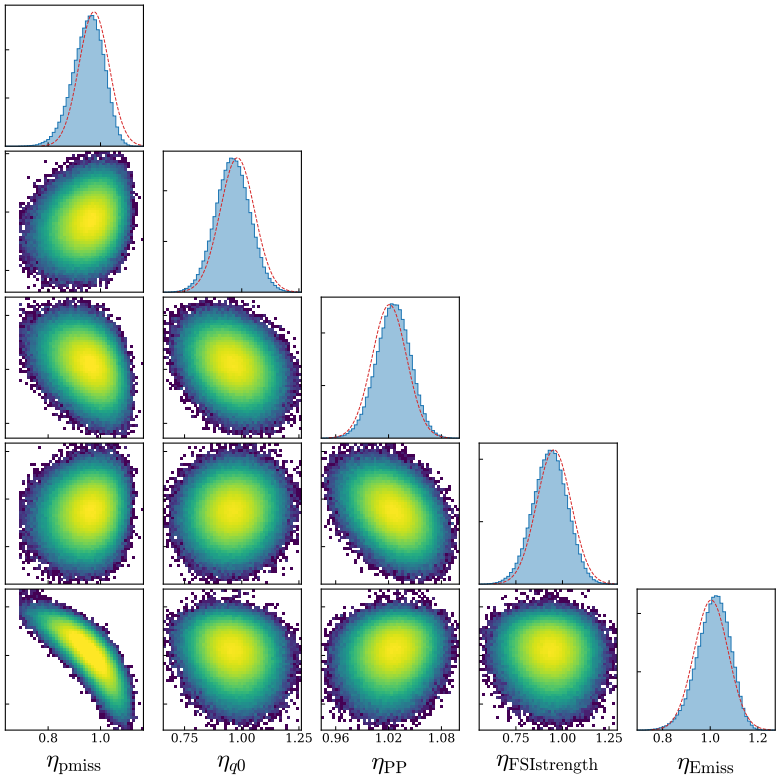


Fig. 3: Corner plot of selected non-linear systematic parameters obtained from 500 000 samples drawn from the NF surrogate. The diagonal panels show the one-dimensional marginal distributions, compared with the post-fit Gaussian approximation shown as a red dashed curve. The off-diagonal panels show the corresponding two-dimensional projections in logarithmic color scale. See Method 4.3 for the parametrization of the nuisance parameters.

(RHC), producing one flux prediction for each systematic throw. Repeating this procedure for samples drawn from the NF surrogate, the post-fit Gaussian approximation, and the MCMC chain gives an ensemble of flux predictions from which the mean, uncertainty bands, and bin-to-bin correlations can be estimated.

Figure 4 shows the unoscillated flux predictions obtained with each likelihood model (NF, MCMC and Gaussian) for the FHC and RHC samples. The upper panels show the mean event yield divided by the bin width, together with bin-by-bin uncertainty bands estimated as the sample standard deviation of the bin event yield over a repetition of toy experiments. The lower panels compare the relative uncertainty widths, defined as

$$\frac{\Delta\sigma}{\bar{\sigma}} = \frac{\sigma_A - \sigma_B}{(\sigma_A + \sigma_B)/2},$$

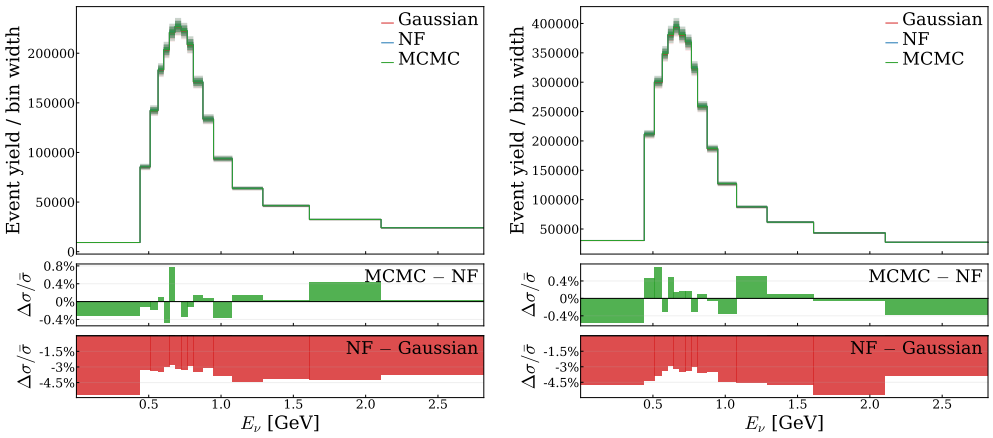


Fig. 4: FHC (left) and RHC (right) flux predictions obtained by propagating systematic-parameter throws to the event weights and filling the resulting E_ν spectra. The predictions are shown as the event yield divided by the bin width, using 500 000 samples drawn from the NF surrogate, 500 000 MCMC steps, and the corresponding post-fit Gaussian mean and standard deviation. The spectra are truncated at $E_\nu = 2.82$ GeV, and the binning is chosen such that the expected nominal population is approximately equal in each bin. The lower panels compare the relative uncertainty widths, $\Delta\sigma/\bar{\sigma}$, between MCMC and NF and between NF and the Gaussian approximation, where $\bar{\sigma}$ is the average of the two uncertainties being compared.

where σ_A and σ_B are the bin-by-bin standard deviations obtained from the flux predictions of two different methods. The NF and MCMC uncertainty estimates agree within approximately $\pm 1\%$, while the post-fit Gaussian approximation overestimates the uncertainty width by about 3.5% compared to both NF and MCMC. This indicates that the NF reproduces the uncertainty propagation obtained from the reference MCMC sample. This has direct consequences for the inference of the parameters of interest: the NF-based propagation provides a more accurate estimate of the systematic uncertainty, and in this particular case leads to a smaller propagated uncertainty than the post-fit Gaussian approximation.

Figure 5 provides a complementary bin-by-bin comparison using split violin plots. Each violin shows the relative deviation of the bin event yields with respect to the Gaussian mean prediction. The two distributions are in very good agreement in each bin: they exhibit the same shift, a similar spread, and a similar distribution shape. In particular, both the MCMC and NF distributions exhibit a median shifted by approximately $+0.2\%$ relative to the Gaussian reference, indicating an underestimation of the median event rate by the Gaussian approximation. This shows that using the post-fit Gaussian as a surrogate for the systematic likelihood could introduce a bias in the predictive estimate of the flux after systematic uncertainty propagation, while the NF reproduces the MCMC results at the bin-by-bin distribution level.

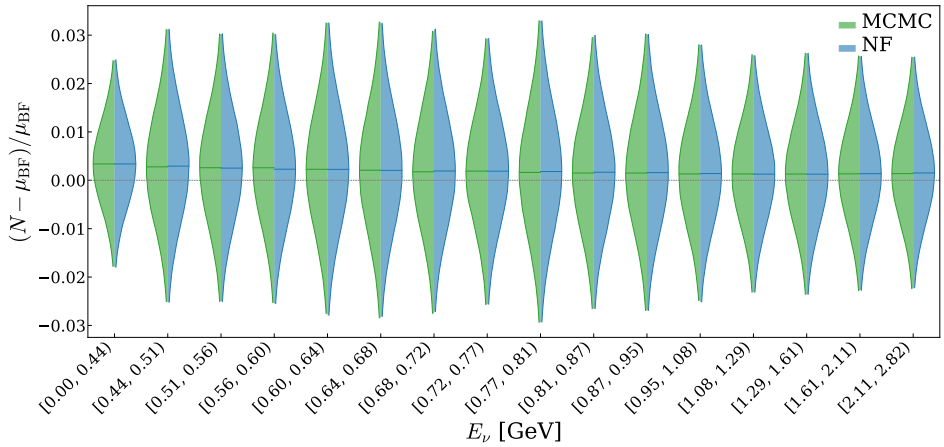


Fig. 5: Split violin representation of the FHC flux predictions in the same E_ν bins, using 500 000 samples drawn from the NF surrogate and 500 000 samples from the MCMC chain. For each systematic-parameter throw, the event yield N in a given bin is compared with the fixed post-fit Gaussian prediction μ_{BF} in that bin. Each violin shows the resulting distribution of the relative deviation $(N-\mu_{\mathrm{BF}})/\mu_{\mathrm{BF}}$ over throws, with the MCMC distribution shown on the left side and the NF distribution on the right side of the violin. The horizontal segment drawn inside each half-violin indicates the median of the corresponding distribution, while the dashed horizontal line at zero marks the post-fit Gaussian reference.

2.5 Sampling efficiency and computational cost

Figure 6 shows the evolution of the sampling quality during training, quantified by the relative effective sample size,

$$\mathrm{rESS} = \frac{1}{N}\frac{\left(\sum_{i=1}^{N} w_i\right)^2}{\sum_{i=1}^{N} w_i^2},$$

where N is the number of samples and w_i is the importance weight associated with the i-th sample. For samples drawn from a proposal distribution q, the weights are defined as

$$w_i = \frac{p(\boldsymbol{\eta}_i)}{q(\boldsymbol{\eta}_i)},$$

where p denotes the target likelihood distribution. During training, q corresponds to the distribution represented by the current normalizing flow, while the Gaussian baseline is obtained by taking $q = g(\boldsymbol{\eta}; \boldsymbol{\eta}_{\text{best-fit}}, C)$ where $\boldsymbol{\eta}_{\text{best-fit}}$ and C are the post-fit maximum likelihood estimator and covariance matrix.

The rESS obtained with the Gaussian approximation is shown as a dashed line for comparison. When the training is stopped after 64 000 epochs, the NF reaches a rESS of 98%, compared to only 5% for the Gaussian proposal. This corresponds to an overall improvement of approximately a factor 20 in sampling efficiency when

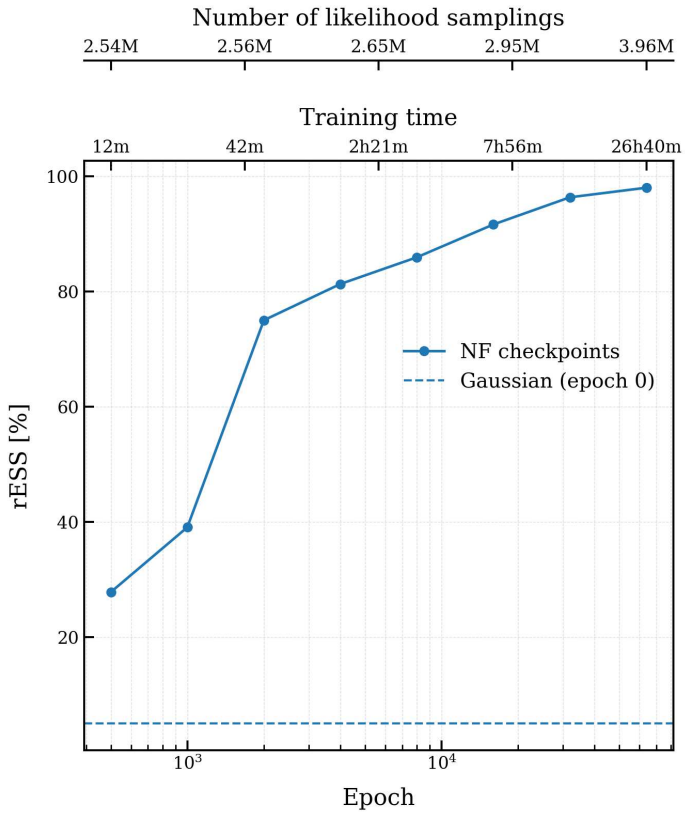


Fig. 6: Relative effective sample size of a set of 500 000 samples from the current NF model at different times during training.

replacing the initial post-fit Gaussian proposal with the trained normalizing flow. The corresponding distribution of importance weights is shown in Figure 7, obtained using 500 000 samples drawn from the trained NF and 500 000 samples drawn from the post-fit Gaussian approximation. While the Gaussian proposal produces weights spanning more than 10 orders of magnitude, the NF weights are much more concentrated. This shows that the trained NF provides a much closer approximation to the target likelihood distribution, leading to substantially more efficient importance sampling.

Moreover, once trained, the surrogate becomes essentially independent of the expensive likelihood evaluation. For the benchmark configuration with $D_A = 10$, both sampling and density evaluation are analytical and fast, reaching approximately 38 million samples and density evaluations per GPU-hour, or 960 000 per CPU-hour. This should be compared with the original likelihood, for which only about 10 000 evaluations per CPU-hour are possible on the same hardware.

As anticipated in Section 2.1, the sampling cost of the trained flow is mainly controlled by the autoregressive block. It therefore scales with D_A, the number of non-linear parameters modeled autoregressively, rather than with the total dimension. By

D_A	CPU [samples/hr]	GPU [samples/hr]
10	9.6×10^5	3.8×10^7
20	2.3×10^5	1.4×10^7
30	1.2×10^5	7.3×10^6
60	5.6×10^4	2.0×10^6
100	2.2×10^4	8.4×10^5

Table 1: Sampling speed of the trained flow as a function of the dimension D_A of the autoregressive block, at fixed total dimension.

contrast, the coupling block is evaluated in a single pass. The relevant quantity for a given analysis is therefore the number of genuinely non-linear systematics, which is typically a small fraction of the total: the T2K 2023 oscillation analysis, for instance, used 59 non-linear parameters out of 711 [6]. Table 1 reports the sampling throughput as D_A is increased up to this scale. In the worst case scenario $D_A = 100$, the trained surrogate generates of order 10^4 samples per CPU-hour and 10^6 per GPU-hour, still exceeding the $\sim 10^4$ evaluations per CPU-hour of the original likelihood.

The computational cost of training the NF model is dominated by the evaluation of the likelihood, which also drives the computational cost of likelihood sampling methods like MCMC. Figure 6 highlights that the number of likelihood evaluations required to obtain an efficient NF model is comparable to the number of points sampled by the benchmark MCMC. Unlike MCMC, the NF samples are not autocorrelated, and the training cost is paid only once: the trained flow is a closed-form density that can be sampled and evaluated thereafter at marginal cost.

3 Discussion

We have shown that a normalizing flow can accurately reproduce the high-dimensional constrained systematic likelihood of a realistic T2K-inspired near-detector benchmark with 110 parameters, including its asymmetric and non-quadratic features. Once trained, the normalizing flow turns the fitted systematic likelihood into a portable object: it can be sampled, evaluated, and propagated without rerunning the original fit. As in the case of a post-fit Gaussian approximation, the original data and full fitting machinery are no longer needed for downstream propagation, but the NF preserves the non-Gaussian structure of the likelihood with much higher accuracy. The important step is a change of representation: the fit output is promoted to an explicit density model that can be efficiently evaluated and sampled wherever it is needed.

This places the NF in a useful middle ground between MCMC samples and Gaussian likelihood surrogates. Compared with MCMC, it is lightweight, explicit, and directly samplable: new systematic throws can be generated on demand, without storing large chains or recomputing the original likelihood. Compared with a Gaussian approximation, it retains skewness, curved correlations, asymmetric marginals, and non-Gaussian tails, and therefore avoids the bias that can appear when a likelihood is propagated through an overly restrictive quadratic model.

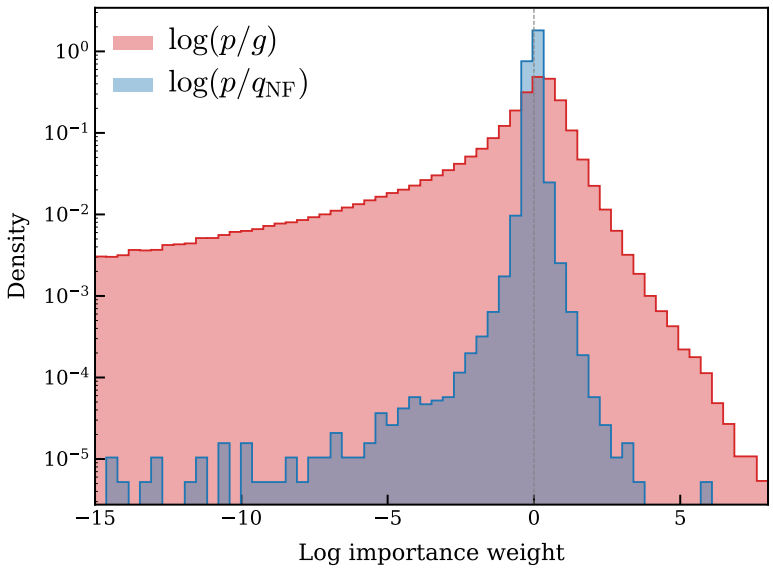


Fig. 7: Distribution of the log-importance weights for 500 000 samples drawn from the post-fit Gaussian proposal g (red) and from the trained normalizing flow q_{NF} (blue).

Such a representation is particularly useful when the constrained systematic likelihood from one analysis must be propagated into another, as is the case for oscillation-parameter measurements using a two-step near-detector/far-detector approach, as well as joint experiment fits, or global combinations involving several detectors, experiments, or data samples, fundamental for precision oscillation measurements in the near future [6, 12, 13]. In such scenarios, an NF surrogate can be inserted as an explicit, systematics density model: it can be sampled for uncertainty propagation, used for nuisance marginalization, or evaluated as a prior or constraint term in a downstream likelihood on which gradient descent can be performed, all without relying on a Gaussian approximation of the original fit. As these analyses grow in scale and ambition, the ability to carry a likelihood in its full complexity will be no less essential than the precision of the fit that produced it.

4 Methods

4.1 Benchmark near-detector likelihood

We benchmark the method by simulating an end-to-end near-detector-like analysis. We generate 500 000 neutrino and 500 000 antineutrino interactions on hydrocarbon using the NEUT [45] framework and formatted as NUISANCE [46] flat trees [47]. Since the benchmark is inspired by T2K near-detector analyses, the simulation follows the most recent publicly available T2K off-axis neutrino-energy spectrum [48], peaking at approximately 0.6 GeV. Detector response is simulated heuristically by applying Gaussian smearing to the final-state muon kinematics of each simulated event. We use

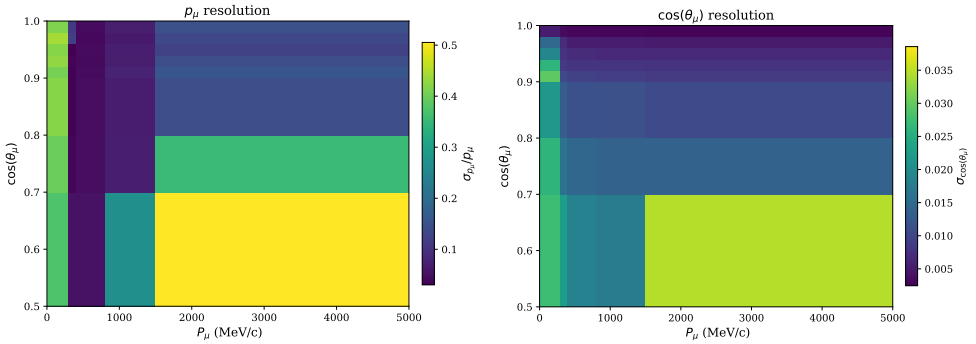


Fig. 8: Simulated muon-kinematics detector resolution as a function of p_μ and $\cos\theta_\mu$. Left: muon momentum resolution. Right: muon angular resolution.

Particle	Momentum	$\cos(\theta)$
Muon	10 MeV	0.5
Proton	450 MeV	0.5
Pion	200 MeV	0.5

Table 2: Particle detection thresholds.

resolutions on the muon momentum magnitude $|\vec{p}_\mu|$ and on the cosine of the muon scattering angle $\cos\theta_\mu$ similar to those of the T2K Near Detector (ND280) [2, 49, 50], as shown in Figure 8.

Typical near-detector data samples are produced using topological event selections, where each event is assigned to a sample based on the number of detected particles in the final state. In order to simulate detector-efficiency effects, we introduce fixed thresholds for the detection of final-state charged particles, summarized in Table 2.

Six samples are defined: three samples in neutrino mode, FHC for Forward Horn Current, and three equivalent samples in antineutrino mode, RHC for Reverse Horn Current. The samples are defined according to the number of detected charged pions in the final state: no pion, one pion, or more than one pion. Exactly one detected muon (antimuon) is required in each neutrino (antineutrino) sample to pass the selection. The total efficiency of the selection is around 55%. Table 3 summarizes the number of selected events in each sample. The binning is optimized for each sample based on expected event rates, with higher granularity in high-statistics regions.

To emulate the procedure of fitting a Monte Carlo model to data, we repeat the same generation and selection procedure on an independent sample containing the same number of neutrino and antineutrino interactions, which we label as pseudo-data.

Sample	# of events
FHC ν_μ CC 0π	188 305
FHC ν_μ CC 1π	43 154
FHC ν_μ CC Other	4 272
RHC $\bar{\nu}_\mu$ CC 0π	280 545
RHC $\bar{\nu}_\mu$ CC 1π	27 666
RHC $\bar{\nu}_\mu$ CC Other	1 100
Total	545 042

Table 3: Summary of selected events per Monte Carlo sample.

4.2 Likelihood construction

The likelihood is written as a function of the systematic-parameter vector $\boldsymbol{\eta}$,

$$-2\ln\mathcal{L}(\boldsymbol{\eta}) = -2\ln\mathcal{L}_{\text{stat}}(\boldsymbol{\eta}) + \chi^2_{\text{syst}}(\boldsymbol{\eta}), \tag{3}$$

where $\mathcal{L}_{\text{stat}}$ compares the observed bin counts to the predicted event rates, and χ^2_{syst} encodes the prior constraints on the systematic parameters. With prior mean $\boldsymbol{\eta}_0$ and prior covariance matrix V, the systematic penalty is

$$\chi^2_{\text{syst}}(\boldsymbol{\eta}) = (\boldsymbol{\eta} - \boldsymbol{\eta}_0)^\top V^{-1}(\boldsymbol{\eta} - \boldsymbol{\eta}_0). \tag{4}$$

The statistical term is built from a binned likelihood including Monte Carlo statistical uncertainties through the "lite" Barlow–Beeston prescription [16, 51]. The likelihood is evaluated with the GUNDAM analysis framework [52], a tool used for oscillation and cross-section analyses in the T2K, ICARUS and DUNE experiments, which computes it for a given systematic-parameter vector $\boldsymbol{\eta}$.

The statistical term compares observed counts n_i to the predicted mean $\mu_i(\boldsymbol{\eta})$ in each sample bin. The total statistical contribution is the sum over all bins,

$$-2\ln\mathcal{L}_{\text{stat}}(\boldsymbol{\eta}) = -2\sum_i \ln\mathcal{L}^i_{\text{BB}}(\boldsymbol{\eta}), \tag{5}$$

where $\mathcal{L}^i_{\text{BB}}$ denotes the "lite" Barlow–Beeston likelihood in bin i, as described in Ref. [51].

The likelihood's shape depends both on the statistical term in Eq. (3) and on the way in which the parameters $\boldsymbol{\eta}$ modify the event rates.

The 110 parameters are associated with an equal number of random variables, whose prior distribution is described by a 110×110 covariance matrix V, which enters the systematic term of the likelihood (Eq. (4)).

4.3 Systematic-uncertainty parametrization

The systematic uncertainties are implemented as a set of parameters $\boldsymbol{\eta}$ that modify the underlying model histograms as a function of physically relevant quantities. The model

is modified through event-by-event weights that depend on the value of the systematic parameters and on the true-level kinematics of the event, denoted generically as

$$w(\boldsymbol{\eta}, \text{kinematics}).$$

This provides model control through physics-motivated systematic parameters.

In a realistic analysis, such weights can be computed from variations of the cross-section model with respect to the baseline prediction through cross-section ratios,

$$w(\eta_i) = \frac{\sigma_{\text{var}}(\eta_i, \text{kinematics})}{\sigma_{\text{baseline}}(\text{kinematics})}.$$

The function $w(\eta_i)$ is obtained by evaluating the cross-section ratio at different values of η_i and interpolating the result, for instance with polynomial splines. In the present benchmark, this behavior is mimicked with custom continuous functions of the systematic parameter and of the event kinematics.

The shape of the likelihood function $\mathcal{L}(\boldsymbol{\eta})$ is affected by the shape of the total event weight,

$$w(\boldsymbol{\eta}) = \prod_i w(\eta_i).$$

Even in the limit of large statistics, the likelihood function is not necessarily Gaussian in $\boldsymbol{\eta}$ when some parameters enter the prediction through non-linear weight functions.

We distinguish between linear and non-linear systematic parameters. Linear parameters apply bin-by-bin multiplicative corrections to the predicted event distributions. For a parameter value η_i, the event yield in the corresponding bin is scaled by $w_i = \eta_i$. Non-linear weight functions mimic interaction-model uncertainties and are implemented as

$$w = 1 + \alpha\, t(k) \left(\eta^p - \eta^p_{\text{prior}}\right), \tag{6}$$

where η is the value of the systematic parameter. For simplicity, the dependence on event kinematics is taken to be a linear function of one kinematic variable k,

$$t(k) = 2\left(\frac{k - k_{\text{min}}}{k_{\text{max}} - k_{\text{min}}} - 0.5\right). \tag{7}$$

The parameters α, p, and the kinematic range $[k_{\text{min}}, k_{\text{max}}]$ determine how the predicted spectra, and consequently the likelihood, vary as a function of the systematic parameter. They are defined separately for each non-linear parameter. The physically motivated parameters have imposed boundaries, which determine the domain of the likelihood function.

The framework is completed by a positive definite covariance matrix that encodes prior parameter uncertainties and correlations, together with a prior parameter vector $\boldsymbol{\eta}_{\text{prior}}$. This is typical of near-detector analyses, where linear parameters, such as detector and flux systematics, induce fluctuations of the spectra as a function of binned observables and can display correlations between adjacent bins, while correlations between cross-section uncertainties are motivated by the interaction model. No

Parameter name	Kinematic dependence	α
`Enu`	Neutrino energy E_ν	0.3
`PP`	Total energy of final-state pions	0.3
`q0`	Energy transfer q^0	0.1
`q3`	Magnitude of momentum transfer $\lvert\vec{q}\rvert$	0.3
`FSIstrength`	Strength of final-state interactions, measured as the amount of energy lost by the proton before escaping the nucleus	0.3
`W`	Hadronic invariant mass W	0.3
`Emiss`	Nuclear missing energy E_{miss}	0.1
`pmiss`	Nuclear missing momentum p_{miss}	0.1
`Q2`	Four-momentum transfer squared Q^2	0.1
`PBnuantinu`	Event rate modulation at low outgoing proton momentum, with opposite effects on ν and $\overline{\nu}$.	0.1

Table 4: Characteristics of the non-linear systematic-parameter implementation.

prior correlation is assumed between the two sets, although correlations may arise in the post-fit covariance matrix due to the way in which both sets affect the sample spectra [53].

Each non-linear systematic parameter assigns a weight w to each event based on the event kinematics and on the value of the systematic parameter η as described in equations (6) and (7). In a realistic case, the event weight may depend on several kinematic variables, while in this benchmark each parameter depends on a single kinematic variable. Table 4 summarizes the parametrization of the non-linear systematic uncertainties. All prior values are set to 1.0 for simplicity, without loss of generality.

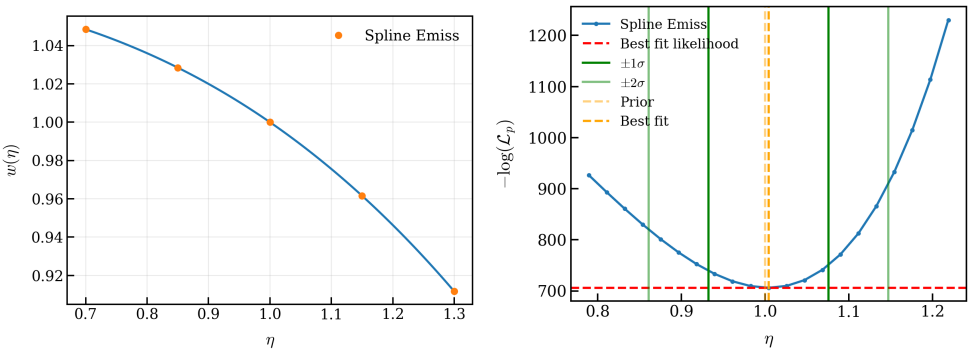
Figure 9 shows how a representative non-linear weight function affects the likelihood through a negative-log-likelihood scan. For non-linear parameters, the NLL profiles can deviate from the quadratic shape expected for a Gaussian likelihood.

4.4 Normalizing-flow model

4.4.1 Autoregressive flows and coupling layers

Normalizing flows rely on invertible transformations that allow for efficient application of the change-of-variable formula. Two types of architectures show the best performance: autoregressive flows and coupling layers. Both approaches exploit triangular (or block-triangular) Jacobians to ensure computational efficiency, with complexity scaling linearly with the dimensionality of the probability space. However, the choice between autoregressive flows and coupling layer-based flows ultimately depends on the application.

In practice, autoregressive flows such as masked autoregressive flow (MAF) [41] are D times slower to invert than to evaluate (or vice-versa), where D is the dimension of the probability space. On the other hand, flows based on coupling layers, such as NICE [54] or RealNVP [40], have an analytic one-pass inverse. Sampling and evaluating the density at the same time requires evaluating both the forward and inverse



(a) (b)

Fig. 9: (a): Event weight $w(\eta)$ as a function of the parameter value η. The points mark the values at which the weight is evaluated, interpolated by the continuous curve. (b): Negative-log-likelihood profile obtained by scanning the E_{miss} parameter around its prior value, with all other parameters fixed. The dashed orange lines mark the prior and best-fit values, the green lines the $\pm1\sigma$ and $\pm2\sigma$ intervals, and the dashed red line the best-fit likelihood.

transformations. Therefore, coupling layers allow both fast sampling and fast density evaluation. However, coupling layer-based flows are generally less expressive than autoregressive flows.

We also exclude continuous normalizing flows [55, 56], including flow-matching variants: although they scale well and can be easier to train, evaluating their log density requires integrating an augmented ODE for the divergence of the learned vector field, which is prohibitively expensive given the many repeated density evaluations our application requires.

4.4.2 Parametrization of the flow transformation

In addition to choosing an architecture (autoregressive or coupling-layer based), a normalizing flow must specify the family of invertible elementary transformations used to map between the base variables and the target parameter space. In this work we employ two complementary parametrizations: (i) Rational Quadratic Neural Spline Flows (RQ-NSF) [57, 58] for the autoregressive (conditional) block, and (ii) affine coupling blocks for the coupling-layer block. This choice reflects a trade-off between expressiveness and computational cost: RQ-NSF provide highly flexible, non-linear monotone transforms well suited to capture non-Gaussian structure, while affine coupling layers provide very fast sampling and density evaluation in high dimensions [59].

Rational Quadratic Neural Spline Flows

RQ-NSF [57] model each transformed coordinate with a strictly monotone, piecewise rational–quadratic spline on a bounded interval $[-B, B]$, acting as the identity outside

it. A neural network outputs the bin widths, heights, and internal-knot derivatives; within each bin $\log|T'(x)|$ is available in closed form and the inverse T^{-1} follows from solving a quadratic, so both density evaluation and inversion are analytic [58].

The D_A-dimensional "non-linear" block $\boldsymbol{\eta}_A$ is transformed one dimension at a time using such 1D spline transformers. Writing u_A for the base variables and η_A for the transformed variables, a conditional autoregressive RQ-NSF layer takes the form

$$\eta_{A,i} \;=\; T_i\Big(u_{A,i};\, \psi_i(\, u_{A,<i},\, \boldsymbol{\eta}_B\,)\Big), \qquad i = 1, \ldots, D_A, \tag{8}$$

where the parameters ψ_i are produced by a conditioner network that depends on previous components (autoregressive masking) and, in our hybrid setting, also on the conditioning variables $\boldsymbol{\eta}_B$. The Jacobian of (8) is triangular, hence the log-determinant reduces to a sum of 1D terms,

$$\log\left|\det\frac{\partial\boldsymbol{\eta}_A}{\partial\mathbf{u}_A}\right| \;=\; \sum_{i=1}^{D_A}\log\left|\frac{\partial T_i(u_{A,i};\psi_i)}{\partial u_{A,i}}\right|. \tag{9}$$

RQ-NSF are particularly attractive because they combine (i) strong expressiveness for non-Gaussian targets with (ii) stable monotonicity constraints and (iii) tractable density evaluation and inversion [57]. They have been successfully applied within T2K for related inference tasks, including cross-section modeling and low-dimensional likelihood estimation [35, 36].

Affine coupling blocks

For the easy, high-dimensional block $\boldsymbol{\eta}_B$, we use affine coupling layers [40], which provide an analytic one-pass inverse and a cheap log-determinant even in very large dimensions. Splitting the input $z \in \mathbb{R}^{D_B}$ into two disjoint subsets B_1 and B_2, an affine coupling block leaves z_{B_1} unchanged and transforms

$$z'_{B_2} = z_{B_2} \odot \exp\big(s_\theta(z_{B_1})\big) + t_\theta(z_{B_1}), \tag{10}$$

where s_θ and t_θ are neural networks and $\odot$ is elementwise multiplication. The block is invertible in a single pass, and its Jacobian is triangular with log-determinant $\sum_{j\in B_2} s_{\theta,j}(z_{B_1})$, linear in the dimension. Coupling blocks are stacked with the split (B_1, B_2) alternated between layers so that all coordinates are eventually transformed [40, 54].

4.4.3 Overall architecture

We target probability spaces with $\mathcal{O}(10^2)$ uncertainty parameters. In this regime, a fully autoregressive architecture over all dimensions can become prohibitively expensive. At the same time, we can exploit strong a priori knowledge about the structure of a typical near-detector likelihood. Only a subset of systematics induces highly non-linear, shape-changing responses through spline-based or otherwise non-trivial

reweighting, while many others act closer to linear. We therefore partition the nuisance parameters into two groups,

$$\boldsymbol{\eta} = (\boldsymbol{\eta}_A, \boldsymbol{\eta}_B), \tag{11}$$

where $\boldsymbol{\eta}_A$ collects the non-linear parameters expected to generate complex, non-Gaussian likelihood features, and $\boldsymbol{\eta}_B$ contains the larger set of linear parameters whose effect on the likelihood is closer to linear.

We model the joint density by combining two flow models that allocate expressivity and computational cost where it matters most, using the conditional probability identity as in Eq. (2).

In practice, the conditioning is implemented by feeding $\boldsymbol{\eta}_B$ to the autoregressive conditioner networks, so that the parameters of the one-dimensional transformers used for $\boldsymbol{\eta}_A$ depend on $\boldsymbol{\eta}_B$. This preserves non-trivial cross-correlations while keeping the computationally expensive autoregressive structure confined to the smaller block $\boldsymbol{\eta}_A$. The implementation is based on the `normflows` package [60], with a custom hybrid structure adapted to the likelihood model.

Figure 1 summarizes the corresponding implementation as a composition of three invertible stages applied to base variables u drawn from a tractable distribution. We take

$$u \sim p_u, \qquad p_u(u) = \mathcal{N}(0, I), \tag{12}$$

i.e. a product of independent standard normal Gaussians. We first apply a multivariate linear transformation

$$x \;=\; T^{(1)}_{\theta_1}(u) \;=\; L_{\theta_1}\, u + \mu_{\theta_1}, \tag{13}$$

with $\mu_{\theta_1} \in \mathbb{R}^{D_A+D_B}$ and L_{θ_1} lower-triangular. This layer can be treated as a fixed reparametrization or trained jointly with the remaining flow parameters. We initialize it to match the post-fit Gaussian approximation: MINUIT [61] provides the best-fit point $\hat{\boldsymbol{\eta}}$ and HESSE provides the Hessian H of $-\ln\mathcal{L}$ at $\hat{\boldsymbol{\eta}}$, from which $C = H^{-1}$. We then set

$$\mu_{\theta_1} = \hat{\boldsymbol{\eta}}, \qquad C = L_{\theta_1} L_{\theta_1}^{\top}, \tag{14}$$

so that the first stage reproduces the Gaussian post-fit mean and covariance. Alternatively, μ_{θ_1} and L_{θ_1} can be initialized based on any convenient Gaussian model (e.g. the prior model), and optionally kept fixed.

The subsequent non-linear stages implement the hybrid factorization of Eq. (2) by splitting $x = (x_A, x_B)$ according to the same partition as $\boldsymbol{\eta}$. The easy, high-dimensional block is modeled with a coupling-layer flow,

$$\boldsymbol{\eta}_B \;=\; T^{(2)}_{\theta_2}(x_B), \tag{15}$$

while the difficult block is modeled with a conditional autoregressive flow,

$$\boldsymbol{\eta}_A \;=\; T^{(3)}_{\theta_3}(x_A; \boldsymbol{\eta}_B), \tag{16}$$

where the dependence on $\boldsymbol{\eta}_B$ encodes the conditioning.

The model can be used in both directions required for our application. For sampling, we draw $u \sim p_u$, compute $x = T^{(1)}_{\theta_1}(u)$, then obtain $\boldsymbol{\eta}_B$ and finally $\boldsymbol{\eta}_A$

conditionally, returning $\boldsymbol{\eta} = (\boldsymbol{\eta}_A, \boldsymbol{\eta}_B)$. For density evaluation, Eq. (2) yields the decomposition

$$\log q_{\mathrm{NF}}(\boldsymbol{\eta}_A, \boldsymbol{\eta}_B) = \log q_{\mathrm{AR}}(\boldsymbol{\eta}_A \mid \boldsymbol{\eta}_B) + \log q_{\mathrm{CL}}(\boldsymbol{\eta}_B), \tag{17}$$

so the two contributions are computed with the architectures best suited to their respective blocks. Since in realistic target applications we expect $\dim(\boldsymbol{\eta}_B) \gg \dim(\boldsymbol{\eta}_A)$, this hybrid design retains the expressivity needed to capture strongly non-Gaussian directions, while keeping both sampling and likelihood evaluation tractable in high dimensions.

4.5 Training

4.5.1 Loss function

Normalizing flows are trained by adjusting the parameters θ of a tractable surrogate density q_θ so that it matches a target distribution p. A standard objective is based on the Kullback–Leibler divergence (KL), which quantifies the discrepancy between two probability densities. Since the KL divergence is asymmetric, minimizing it can lead to qualitatively different behaviors depending on the direction.

Minimizing the *forward* KL divergence,

$$D_{\mathrm{KL}}(p\|q_\theta) = \mathbb{E}_{x\sim p}\left[\log \frac{p(x)}{q_\theta(x)}\right], \tag{18}$$

typically yields *mass-covering* (or mean-seeking) solutions: the learned density q_θ is encouraged to assign probability wherever p has support, which can result in overly diffuse tails. By contrast, minimizing the *reverse* KL divergence,

$$D_{\mathrm{KL}}(q_\theta\|p) = \mathbb{E}_{x\sim q_\theta}\left[\log \frac{q_\theta(x)}{p(x)}\right], \tag{19}$$

often produces *mode-seeking* behavior: q_θ may concentrate on a subset of dominant modes while underestimating (or ignoring) other relevant regions of the target.

To mitigate these complementary pathologies, we consider a symmetric discrepancy defined as the average of the two KL divergences,

$$D_{\mathrm{s}}(p, q_\theta) \;=\; \frac{1}{2}\Big(D_{\mathrm{KL}}(p\|q_\theta) + D_{\mathrm{KL}}(q_\theta\|p)\Big). \tag{20}$$

A direct evaluation of Eq. (20) is impractical in our setting. The forward term requires samples from the target p, which are expensive to obtain, while the reverse term requires sampling from q_θ, which can be highly unstable early in training. To obtain a stable estimator, we rely on importance sampling with a proposal density q that is easy to sample from and sufficiently covers the relevant support of the target [62]. The specific choice and adaptation of q are discussed in Section 4.5.2.

Let $x_i \sim q$ for $i = 1, \ldots, N$. By importance sampling, the symmetric KL in Eq. (20) can be approximated using the Monte Carlo estimator:

$$D_{\mathrm{s}}(p, q_\theta) \approx \frac{1}{2N} \sum_{i=1}^{N} \left[\frac{p(x_i)}{q(x_i)} \log \frac{p(x_i)}{q_\theta(x_i)} + \frac{q_\theta(x_i)}{q(x_i)} \log \frac{q_\theta(x_i)}{p(x_i)} \right], \tag{21}$$

where the sampling is performed from the proposal distribution $q(x_i)$, and $p(x_i)/q(x_i)$ and $q_\theta(x_i)/q(x_i)$ are the importance weights.

This estimator is robust provided that q adequately covers the regions where both p and q_θ have significant mass. In practice, such coverage prevents the flow from collapsing onto narrow subsets of the manifold and reduces the risk of missing important modes. Nevertheless, the variance of Eq. (21) can still be large if q is poorly matched to the target, which motivates the iterative construction of increasingly accurate proposals described in Section 4.5.2.

A final subtlety concerns the fact that the target likelihood introduced in Section 4.2 is available only up to a multiplicative constant. Denoting by $\tilde{p}(x)$ the unnormalized target, we have

$$p(x) = \frac{\tilde{p}(x)}{Z}, \qquad Z = \int \tilde{p}(x)\, dx. \tag{22}$$

We estimate the normalizing constant Z via importance sampling under the same proposal q,

$$\hat{Z} = \frac{1}{N} \sum_{i=1}^{N} \frac{\tilde{p}(x_i)}{q(x_i)}. \tag{23}$$

$\hat{Z}$ enters the loss function as a normalizing factor for the target distribution $p(x_i)$.

4.5.2 Iterative training

Considering the importance-sampling-based definition of the loss function (Eq. (21)), and the large dimensionality of the space, careful choice of the proposal distribution is required for efficient training. The target density can be far from analytically simple, and using a fixed proposal (e.g. the prior or posterior Gaussian) can lead to very large-variance importance weights [63]. Sampling directly from the current flow q_θ at every gradient step is also undesirable: it would require evaluating the target likelihood online for each newly generated sample, effectively turning training into an MCMC-like procedure with a prohibitive wall time as dimensionality increases.

We therefore adopt an iterative, asynchronous scheme in which proposal samples and target evaluations are generated on CPU workers from a lagged flow checkpoint, while GPU training proceeds continuously on the most recent aggregated dataset. Concretely, the procedure is as follows.

We maintain a dataset $\mathcal{D}$ consisting of a queue of the last K batches,

$$\mathcal{D} = \bigcup_{j=1}^{K} \mathcal{B}_j, \qquad \mathcal{B}_j = \{x_i^{(j)},\ \tilde{p}(x_i^{(j)}),\ q(x_i^{(j)})\}_{i=1}^{N_j}, \tag{24}$$

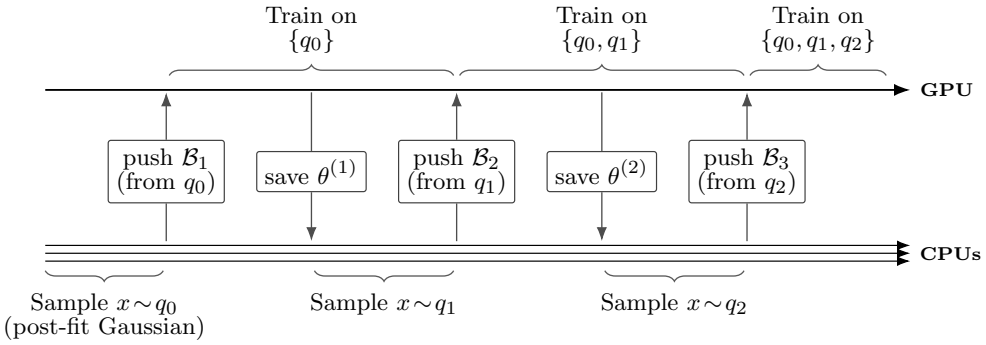


Fig. 10: Timeline of the asynchronous training loop. The GPU (top) trains continuously on the FIFO buffer $\mathcal{D}$ and periodically writes a flow checkpoint $\theta^{(k)}$. CPU workers (bottom) run in parallel: each samples $x \sim q_k$ from the most recent checkpoint, evaluates the unnormalized target $\tilde{p}$, and pushes a batch $\mathcal{B}_j = \{x_i, \tilde{p}(x_i), q_j(x_i)\}_{i=1}^{N_j}$ into $\mathcal{D}$, where q_j is the proposal that produced batch j. We use the shorthand $q_k \equiv q_{\theta^{(k)}}$ for $k \geq 1$; q_0 is the initial post-fit Gaussian. Once $\mathcal{D}$ contains K batches, each new push evicts the oldest.

where each batch $\mathcal{B}_j$ contains proposal samples $x_i^{(j)}$ together with their unnormalized target evaluations $\tilde{p}(x_i^{(j)})$ and proposal densities $q(x_i^{(j)})$ (needed for importance weighting). The queue is updated in FIFO fashion: when a new batch $\mathcal{B}_{K+1}$ is produced, the oldest batch is discarded so that only the last K batches are kept. This controls memory and prevents the training distribution from being dominated by outdated proposals.

Batches are generated asynchronously. At regular intervals, we save a checkpoint of the current flow parameters, denoted θ^{ckpt}. CPU workers load this checkpoint, sample from the corresponding flow, and use that distribution as the proposal,

$$q(x) \equiv q_{\theta^{\text{ckpt}}}(x). \tag{25}$$

For each proposed point $x_i \sim q_{\theta^{\text{ckpt}}}$, the workers evaluate the expensive likelihood to obtain $\tilde{p}(x_i)$. The resulting batch is then pushed into the FIFO queue and becomes available to the trainer.

The GPU trainer continuously optimizes θ using the current dataset $\mathcal{D}$. Since $\mathcal{D}$ is a union of batches that may have been generated by different checkpoints (hence different proposals), the loss is computed as a weighted average over batches:

$$\widehat{D}_{\text{s}}(p, q_\theta; \mathcal{D}) = \sum_{j=1}^{K} \frac{N_j}{\sum_{\ell=1}^{K} N_\ell} \, \widehat{D}_{\text{s}}^{(j)}(p, q_\theta), \tag{26}$$

	Parameter	Value
Architecture	Coupling blocks (q_{CL})	12
	Autoregressive layers (q_{AR})	12
	Spline bins	9
	Spline tail bound B	6
	Coupling conditioner network	1 layer $\times$ 256 units
	Autoregressive conditioner network	2 layers $\times$ 512 units
Training	Optimizer	Adam (AMSGrad)
	Learning rate (constant)	1×10^{-5}
	Sampling batch size N_j	10 000
	FIFO buffer length	50 batches
	Parallel CPU samplers	10

Table 5: Hyperparameters of the hybrid normalizing-flow architecture and of the training procedure. The coupling and autoregressive blocks refer to the two factors of Eq. (2); the spline bins and tail bound B specify the rational-quadratic transform of Eq. (8), which acts as the identity outside the interval $[-B, B]$. Each conditioner network is given as (hidden layers $\times$ units per layer). The FIFO buffer length is the maximum number K of batches of size N_j retained in the training dataset $\mathcal{D}$, and the parallel CPU samplers are the workers that generate proposal points from the asynchronous checkpoint (Section 4.5.2).

with per-batch estimators

$$\widehat{D}_{\mathrm{s}}^{(j)}(p, q_\theta) = \frac{1}{2N_j} \sum_{i=1}^{N_j} \left[\frac{p(x_i^{(j)})}{q_j(x_i^{(j)})} \log \frac{p(x_i^{(j)})}{q_\theta(x_i^{(j)})} + \frac{q_\theta(x_i^{(j)})}{q_j(x_i^{(j)})} \log \frac{q_\theta(x_i^{(j)})}{p(x_i^{(j)})} \right], \tag{27}$$

where q_j denotes the proposal that generated batch $\mathcal{B}_j$ (i.e. the checkpoint flow used by the workers when producing that batch). In particular, when the proposal is taken to be a checkpointed flow, we store $q_j(x_i^{(j)}) = q_{\theta^{(j)}}(x_i^{(j)})$ alongside each sample to avoid re-evaluating old proposals.

This iterative procedure makes the proposal progressively closer to the target as training improves, which reduces the variance of the importance weights and stabilizes optimization. The overall workflow, including the checkpointing, parallel sampling, FIFO dataset update, and continuous optimization, is summarized in Figure 10. Table 5 summarizes the tunable parameters of the iterative training, alongside the other normalizing flows model hyperparameters.

In the present analysis, the initial training dataset is generated by sampling the systematic parameters $\boldsymbol{\eta}$ from $g(\boldsymbol{\eta}; \boldsymbol{\eta}_{\text{best-fit}}, C)$. Since these initial samples are drawn independently from the post-fit Gaussian distribution, their generation and likelihood evaluation are embarrassingly parallelizable and can therefore be performed efficiently. Training starts from an initial dataset of 2.5 million vectors of systematic parameters, together with their corresponding likelihood evaluations. In this procedure, 10 CPU workers continuously generate new points from the NF proposal and compute their

corresponding likelihood values. After 64 000 epochs, corresponding to approximately 26 hours of training and an additional 1.5 million training points, the procedure is stopped and the final NF approximation of the likelihood is obtained.

Data availability. Data supporting the findings of this study are openly available in Zenodo at https://doi.org/10.5281/zenodo.21104524.

Code availability. The code used in this study is openly available on GitHub. The normalizing-flow surrogate, including the hybrid architecture and training loop, is at https://github.com/GunFlows/gunflows, and the benchmark near-detector likelihood and fit are at https://github.com/GunFlows/ToyNDFit. The flow implementation builds on the open-source `normflows` package [60]. Likelihood evaluations are performed with the GUNDAM framework [52].

Acknowledgements. We thank S. Samani, K. Vedantha, S. Bordoni, and L. Amziane for their valuable contributions and support. We thank S. Dolan and C. Wilkinson for providing datasets of simulated neutrino-nucleus interactions used for the demonstrator near-detector fit.

Funding. This work was partially supported by the Swiss National Science Foundation (SNSF) under grants No. 200021E_213196 and No. 200020_204609.

Author contributions. M.E.B. and L.G. conceived the use of normalizing flows as high-dimensional likelihood surrogates, developed the normalizing-flow surrogate model and its training, as well as the benchmark near-detector fit replica, and carried out the analysis. A.B. contributed to the statistical methodology and software implementation, including the interface with the GUNDAM software. F.S. supervised every aspect of the project. All authors discussed the results and wrote, reviewed and approved the manuscript.

Competing interests. The authors declare no competing interests.

References

[1] T2K Collaboration: Constraint on the matter-antimatter symmetry-violating phase in neutrino oscillations. Nature **580**, 339–344 (2020) https://doi.org/10.1038/s41586-020-2177-0 arXiv:1910.03887 [hep-ex]

[2] Abe, K., *et al.*: The T2K Experiment. Nucl. Instrum. Meth. A **659**, 106–135 (2011) https://doi.org/10.1016/j.nima.2011.06.067 arXiv:1106.1238 [physics.ins-det]

[3] Abe, K., *et al.*: Evidence of Electron Neutrino Appearance in a Muon Neutrino Beam. Phys. Rev. D **88**(3), 032002 (2013) https://doi.org/10.1103/PhysRevD.88.032002 arXiv:1304.0841 [hep-ex]

[4] Ankowski, A.M., Mariani, C.: Systematic uncertainties in long-baseline neutrino-oscillation experiments. Journal of Physics G: Nuclear and Particle Physics **44**(5),

054001 (2017) https://doi.org/10.1088/1361-6471/aa61cc arXiv:1609.00258 [hep-ph]

[5] DUNE Collaboration: Deep Underground Neutrino Experiment (DUNE), Far Detector Technical Design Report, Volume II: DUNE Physics (2020)

[6] T2K Collaboration: Measurements of neutrino oscillation parameters from the T2K experiment using 3.6×10^{21} protons on target (2023)

[7] Di Lodovico, F., Patterson, R.B., Shiozawa, M., Worcester, E.: Experimental considerations in long-baseline neutrino oscillation measurements. Annual Review of Nuclear and Particle Science **73**, 69–93 (2023) https://doi.org/10.1146/annurev-nucl-102020-101615

[8] T2K Collaboration: Results from the T2K experiment on neutrino mixing including a new far detector μ-like sample (2025)

[9] T2K Collaboration: Constraining Neutrino Interaction Uncertainties for Neutrino Oscillation Measurements at the T2K Experiment (2026)

[10] Berger, J.O., Liseo, B., Wolpert, R.L.: Integrated likelihood methods for eliminating nuisance parameters. Statistical Science **14**(1), 1–28 (1999) https://doi.org/10.1214/ss/1009211804

[11] Super-Kamiokande Collaboration, T2K Collaboration: First joint oscillation analysis of Super-Kamiokande atmospheric and T2K accelerator neutrino data. Physical Review Letters **134**, 011801 (2025) https://doi.org/10.1103/PhysRevLett.134.011801 arXiv:2405.12488 [hep-ex]

[12] NOvA Collaboration, T2K Collaboration: Joint neutrino oscillation analysis from the T2K and NOvA experiments (2025)

[13] Esteban, I., Gonzalez-Garcia, M.C., Maltoni, M., Martinez-Soler, I., Pinheiro, J.P., Schwetz, T.: NuFit-6.0: Updated global analysis of three-flavor neutrino oscillations (2024)

[14] DUNE Collaboration: Experiment Simulation Configurations Approximating DUNE TDR (2021)

[15] Cranmer, K., Kraml, S., Prosper, H., Bechtle, P., Bernlochner, F., Bloch, I.M., Canonero, E., Chrzaszcz, M., Coccaro, A., Conrad, J., Cowan, G., Feickert, M., Ferreiro, N., Fowlie, A., Heinrich, L.A., Held, A., Kuhr, T., Kvellestad, A., Madigan, M., Mahmoudi, F.N., Morå, K.D., Neubauer, M.S., Pierini, M., Rojo, J., Sekmen, S., Silvestrini, L., Sanz, V., Stark, G.H., Torre, R., Thorne, R., Waltenberger, W., Wardle, N., Wittbrodt, J.: Publishing statistical models: Getting the most out of particle physics experiments. SciPost Physics **12**(1) (2022) https://doi.org/10.21468/scipostphys.12.1.037

[16] Barlow, R., Beeston, C.: Fitting using finite monte carlo samples. Computer Physics Communications **77**(2), 219–228 (1993) https://doi.org/10.1016/0010-4655(93)90005-W

[17] Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E.: Equation of state calculations by fast computing machines. The Journal of Chemical Physics **21**(6), 1087–1092 (1953) https://doi.org/10.1063/1.1699114

[18] Hastings, W.K.: Monte carlo sampling methods using markov chains and their applications. Biometrika **57**(1), 97–109 (1970) https://doi.org/10.1093/biomet/57.1.97

[19] Cranmer, K., Lewis, G., Moneta, L., Shibata, A., Verkerke, W.: HistFactory: A tool for creating statistical models for use with RooFit and RooStats. Technical report, New York U., New York (2012). https://doi.org/10.17181/CERN-OPEN-2012-016 . https://cds.cern.ch/record/1456844

[20] Kobyzev, I., Prince, S.J.D., Brubaker, M.A.: Normalizing flows: An introduction and review of current methods. IEEE Transactions on Pattern Analysis and Machine Intelligence **43**(11), 3964–3979 (2021) https://doi.org/10.1109/tpami.2020.2992934

[21] Papamakarios, G., Nalisnick, E., Rezende, D.J., Mohamed, S., Lakshminarayanan, B.: Normalizing flows for probabilistic modeling and inference. Journal of Machine Learning Research **22**(57), 1–64 (2021)

[22] Cranmer, K., Brehmer, J., Louppe, G.: The frontier of simulation-based inference. Proceedings of the National Academy of Sciences **117**(48), 30055–30062 (2020) https://doi.org/10.1073/pnas.1912789117

[23] Brehmer, J., Cranmer, K.: Simulation-based inference methods for particle physics (2020) arXiv:2010.06439 [hep-ph]. Chapter in Artificial Intelligence for Particle Physics (World Scientific)

[24] Brehmer, J.: Simulation-based inference in particle physics. Nature Reviews Physics **3**, 305 (2021) https://doi.org/10.1038/s42254-021-00305-6

[25] Tejero-Cantero, A., Boelts, J., Deistler, M., Lueckmann, J.-M., Durkan, C., Gonçalves, P.J., Greenberg, D.S., Macke, J.H.: sbi: A toolkit for simulation-based inference. Journal of Open Source Software **5**(52), 2505 (2020) https://doi.org/10.21105/joss.02505

[26] Green, S.R., Simpson, C., Gair, J.: Gravitational-wave parameter estimation with autoregressive neural network flows. Phys. Rev. D **102**(10), 104057 (2020) https://doi.org/10.1103/PhysRevD.102.104057 arXiv:2002.07656 [astro-ph.IM]

[27] Dax, M., Green, S.R., Gair, J., Macke, J.H., Buonanno, A., Schölkopf, B.: Real-time gravitational wave science with neural posterior estimation. Phys. Rev. Lett. **127**(24), 241103 (2021) https://doi.org/10.1103/PhysRevLett.127.241103 arXiv:2106.12594 [gr-qc]

[28] Dax, M., Green, S.R., Gair, J., Pürrer, M., Wildberger, J., Macke, J.H., Buonanno, A., Schölkopf, B.: Neural importance sampling for rapid and reliable gravitational-wave inference. Phys. Rev. Lett. **130**(17), 171403 (2023) https://doi.org/10.1103/PhysRevLett.130.171403 arXiv:2210.05686 [gr-qc]

[29] Krause, C., Shih, D.: Fast and accurate simulations of calorimeter showers with normalizing flows. Phys. Rev. D **107**(11), 113003 (2023) https://doi.org/10.1103/PhysRevD.107.113003 arXiv:2106.05285 [physics.ins-det]

[30] Krause, C., Shih, D.: Accelerating accurate simulations of calorimeter showers with normalizing flows and probability density distillation. Phys. Rev. D **107**(11), 113004 (2023) https://doi.org/10.1103/PhysRevD.107.113004 arXiv:2110.11377 [physics.ins-det]

[31] Albergo, M.S., Kanwar, G., Shanahan, P.E.: Flow-based generative models for markov chain monte carlo in lattice field theory. Phys. Rev. D **100**(3), 034515 (2019) https://doi.org/10.1103/PhysRevD.100.034515 arXiv:1904.12072 [hep-lat]

[32] Kanwar, G., Albergo, M.S., Boyda, D., Cranmer, K., Hackett, D.C., Racanière, S., Rezende, D.J., Shanahan, P.E.: Equivariant flow-based sampling for lattice gauge theory. Phys. Rev. Lett. **125**(12), 121601 (2020) https://doi.org/10.1103/PhysRevLett.125.121601 arXiv:2003.06413 [hep-lat]

[33] Albergo, M.S., Kanwar, G., Racanière, S., Rezende, D.J., Urban, J.M., Boyda, D., Cranmer, K., Hackett, D.C., Shanahan, P.E.: Flow-based sampling for fermionic lattice field theories. Phys. Rev. D **104**(11), 114507 (2021) https://doi.org/10.1103/PhysRevD.104.114507 arXiv:2106.05934 [hep-lat]

[34] Pina-Otey, S., Sánchez, F., Gaitán, V., Lux, T.: Likelihood-free inference of experimental neutrino oscillations using neural spline flows. Physical Review D **101**, 113001 (2020) https://doi.org/10.1103/PhysRevD.101.113001 arXiv:2002.09436 [hep-ph]

[35] Pina-Otey, S., Sánchez, F., Lux, T., Gaitan, V.: Exhaustive neural importance sampling applied to monte carlo event generation. Phys. Rev. D **102**, 013003 (2020) https://doi.org/10.1103/PhysRevD.102.013003

[36] El Baz, M., Sánchez, F.: Fast posterior probability sampling with normalizing flows and its applicability in bayesian analysis in particle physics. Phys. Rev. D **109**, 032008 (2024) https://doi.org/10.1103/PhysRevD.109.032008

[37] Gavrikov, A., Serafini, A., Dolzhikov, D., Garfagnini, A., Gonchar, M., Grassi, M., *et al.*: Simulation-based inference for precision neutrino physics through neural monte carlo tuning. Communications Physics (2026) https://doi.org/10.1038/s42005-026-02499-6 arXiv:2507.23297 [hep-ex]

[38] IceCube Collaboration: Neural posterior estimation of the neutrino direction in IceCube using transformer-encoded normalizing flows on the sphere (2026)

[39] El Baz, M., Sánchez, F., Jachowicz, N., Niewczas, K., Jha, A.K., Nikolakopoulos, A.: Efficient monte carlo event generation for neutrino-nucleus exclusive cross sections. Phys. Rev. D **111**, 113001 (2025) https://doi.org/10.1103/v7sz-vn3b

[40] Dinh, L., Sohl-Dickstein, J., Bengio, S.: Density estimation using Real NVP (2017). https://arxiv.org/abs/1605.08803

[41] Papamakarios, G., Pavlakou, T., Murray, I.: Masked autoregressive flow for density estimation. In: Advances in Neural Information Processing Systems, vol. 30, pp. 2338–2347 (2017). https://proceedings.neurips.cc/paper/2017/file/6c1da886822c67822bcf3679d04369fa-Paper.pdf

[42] Winkler, C., Worrall, D., Hoogeboom, E., Welling, M.: Learning Likelihoods with Conditional Normalizing Flows (2019). https://arxiv.org/abs/1912.00042

[43] Roberts, G.O., Gelman, A., Gilks, W.R.: Weak convergence and optimal scaling of random walk Metropolis algorithms. The Annals of Applied Probability **7**(1), 110–120 (1997) https://doi.org/10.1214/aoap/1034625254

[44] Beskos, A., Pillai, N., Roberts, G., Sanz-Serna, J.M., Stuart, A.: Optimal tuning of the hybrid Monte Carlo algorithm. Bernoulli **19**(5A), 1501–1534 (2013) https://doi.org/10.3150/12-BEJ414

[45] Hayato, Y., Pickering, L.: The neut neutrino interaction simulation. The European Physical Journal Special Topics **230**(24), 4469–4481 (2021) https://doi.org/10.1140/epjs/s11734-021-00287-7 arXiv:2106.15809 [hep-ph]

[46] Stowell, P., *et al.*: NUISANCE: a neutrino cross-section generator tuning and comparison framework. JINST **12**(01), 01016 (2017) https://doi.org/10.1088/1748-0221/12/01/P01016 arXiv:1612.07393 [hep-ex]

[47] Dolan, S., Pickering, L., Stowell, P., Wilkinson, C., Wret, C.: CP-violation or Nuclear Excitation: Reviewing the Role of Neutrino Interaction Model Uncertainties on Accelerator-Based Neutrino Oscillation Measurements (2026) arXiv:2605.28671 [hep-ex]

[48] Abe, K., *et al.*: T2K neutrino flux prediction. Phys. Rev. D **87**, 012001 (2013) https://doi.org/10.1103/PhysRevD.87.012001 arXiv:1211.0469 [hep-ex]

[49] Abgrall, N., *et al.*: Time Projection Chambers for the T2K Near Detectors. Nucl. Instrum. Meth. A **637**, 25–46 (2011) https://doi.org/10.1016/j.nima.2011.02.036 arXiv:1012.0865 [physics.ins-det]

[50] Amaudruz, P.A., *et al.*: The T2K Fine-Grained Detectors. Nucl. Instrum. Meth. A **696**, 1–31 (2012) https://doi.org/10.1016/j.nima.2012.08.020 arXiv:1204.3666 [physics.ins-det]

[51] Conway, J.S.: Incorporating Nuisance Parameters in Likelihoods for Multisource Spectra. In: PHYSTAT 2011, pp. 115–120 (2011). https://doi.org/10.5170/CERN-2011-006.115

[52] GUNDAM Collaboration: GUNDAM: Generalized and Unified Neutrino Data Analysis Methods. https://doi.org/10.5281/zenodo.13223262 . https://github.com/gundam-organization/gundam

[53] Chakrani, J., Dolan, S., Avanzini, M.B., Ershova, A., Koch, L., McFarland, K., Megias, G.D., Munteanu, L., Pickering, L., Skwarczynski, K., Nguyen, V.Q., Wret, C.: Parametrized uncertainties in the spectral function model of neutrino charged-current quasielastic interactions for oscillation analyses. Phys. Rev. D **109**, 072006 (2024) https://doi.org/10.1103/PhysRevD.109.072006

[54] Dinh, L., Krueger, D., Bengio, Y.: NICE: Non-linear Independent Components Estimation (2015). https://arxiv.org/abs/1410.8516

[55] Grathwohl, W., Chen, R.T.Q., Bettencourt, J., Sutskever, I., Duvenaud, D.: FFJORD: Free-form Continuous Dynamics for Scalable Reversible Generative Models (2018). https://arxiv.org/abs/1810.01367

[56] Lipman, Y., Chen, R.T.Q., Ben-Hamu, H., Nickel, M., Le, M.: Flow Matching for Generative Modeling (2023). https://arxiv.org/abs/2210.02747

[57] Durkan, C., Bekasov, A., Murray, I., Papamakarios, G.: Neural spline flows. In: Advances in Neural Information Processing Systems, vol. 32, pp. 7511–7522 (2019). https://proceedings.neurips.cc/paper/2019/file/7ac71d433f282034e088473244df8c02-Paper.pdf

[58] Delbourgo, R., Gregory, J.A.: Piecewise rational quadratic interpolation to monotonic data. IMA Journal of Numerical Analysis **3**(3), 141–152 (1983) https://doi.org/10.1093/imanum/3.3.141

[59] Coccaro, A., Letizia, M., Reyes-González, H., Torre, R.: Comparison of affine and rational quadratic spline coupling and autoregressive flows through robust statistical tests. Symmetry **16**(8), 942 (2024) https://doi.org/10.3390/sym16080942

[60] Stimper, V., Liu, D., Campbell, A., Berenz, V., Ryll, L., Schölkopf, B., Hernández-Lobato, J.M.: normflows: A pytorch package for normalizing flows. Journal of Open Source Software **8**(86), 5361 (2023) https://doi.org/10.21105/joss.05361

[61] James, F., Roos, M.: Minuit: A System for Function Minimization and Analysis of the Parameter Errors and Correlations. Comput. Phys. Commun. **10**, 343–367 (1975) https://doi.org/10.1016/0010-4655(75)90039-9

[62] Müller, T., McWilliams, B., Rousselle, F., Gross, M., Novák, J.: Neural importance sampling. CoRR **abs/1808.03856** (2018) 1808.03856

[63] Elvira, V., Martino, L.: Advances in Importance Sampling, pp. 1–14 (2021)