Event Generator Tuning Incorporating Systematic Uncertainty

Jaffae Schroff[1],[*] and *Xiangyang* Ju[2],[**]

[1]Physics Division, University of California, Berkeley, CA 94720
[2]Scientific Data Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720

Abstract. Event generators play an important role in all physics programs at the Large Hadron Collider and beyond. Dedicated efforts are required to tune the parameters of event generators to accurately describe data. There are many tuning methods ranging from expert-based manual tuning to surrogate function-based semi-automatic tuning, to machine learning-based re-weighting. Although they scale differently with the number of generator parameters and the number of experimental observables, these methods are effective in finding optimal generator parameters. However, none of these tuning methods includes the Monte Carlo (MC) systematic uncertainties. That makes the tuning results sensitive to systematic variations. In this work, we introduce a novel method to incorporate the MC systematic uncertainties into the tuning procedure and to quantitatively evaluate uncertainties associated with the tuned parameters. Tested with a dummy example, the method results in a χ^2 distribution that is centered around one, the optimal generator parameters are closer to the true parameters, and the estimated uncertainties are more accurate.

1 Introduction

General-purpose event generators, like Pythia 8 [1], are widely used in High Energy Physics for event generation and physics simulations. They often contain many parameters that must be tuned so that the generated distributions match the data. Dedicated tuning campaigns were launched by the ATLAS and CMS experiments to tune these event generators for the Large Hadron Collider (LHC).

The tuning method evolved from manual tuning to automated tuning. In the beginning, the tuning was performed by domain experts based on their sense of physics and goodness of fit [2]. Later on, the software, Professor [3], made the tuning automated and more objective. It first optimizes a surrogate function that models the relationship between generator parameters and experimental variables (inner-loop optimization), and then optimizes a χ^2 function that measures the differences between simulated data and experimental data (outer-loop optimization). Recently, Apprentice [4], a purely Python-based tool, was developed to leverage High-Performance

---

[*]e-mail: jeffae@berkeley.edu
[**]e-mail: xju@lbl.gov

Computing and introduced rational approximation as an alternative surrogate function.

However, the Monte Carlo (MC) systematic uncertainties are either ignored or artificially compensated. Ref. [2] artificially introduced a 5% uncertainty when calculating the χ^2 function for experimental histograms so that the χ^2 is not too large, while Professor and Apprentices ignored MC uncertainties. Because of the absence of MC uncertainties, these tunings are often sensitive to systematic variations. For example, the latest ATLAS tuning [5] finds that tuning with different parton distribution functions (PDFs) results in different tuned parameters.

Two major sources of MC systematic uncertainties exist: Quantum chromodynamics (QCD) scale and parton distribution functions (PDF). The QCD scale uncertainties stem from the choice of factorize and renormalization QCD scales, while the PDF uncertainties are from either the PDF sets themselves or the differences among PDF sets.

The developments of the Les Houches (LHE) 3 data format [6, 7] automate the estimation of MC systematic uncertainties, thanks to the multiple event weights stored in LHE 3 files. We propose to improve the current MC tuning procedure by taking into account these theoretical uncertainties and estimating the parameter uncertainties based on the χ^2 distribution.

2 Current MC tuning procedure

The current MC tuning procedure is a two-step optimization process, detailed in Refs [3, 4]. In the inner loop, a surrogate function is optimized to model the relationship between the generator parameters and the experimental observables. In the outer loop, the generator parameters are optimized to minimize a χ^2 function. We will describe the two steps and refer to it as `MC-Tune-NoError` in the following sections.

2.1 Inner loop optimization

To illustrate the method, we assume there are n generator parameters and ℓ generator parameters are sampled for simulation. The inner loop optimization is performed for each bin of each experimental observable. Without a loss of generality, we focus on one bin and use the quadratic approximation:

$$f(\boldsymbol{p}) \approx a_0^{(0)} + \sum_{i=1}^{n} a_i^{(1)} p_i + \sum_{i=1}^{n} \sum_{j=i}^{n} a_{ij}^{(2)} p_i p_j \tag{1}$$

$$= M(\boldsymbol{p}) \tag{2}$$

where $M(\boldsymbol{p})$ is the vector of model predictions corresponding to the parameters $\boldsymbol{p}$ and $a_{i(j)}^{(k)}$ are the coefficients of the surrogate function. Equation 1 can be written in a matrix form:

$$f(\boldsymbol{p}) = \boldsymbol{P} \cdot \boldsymbol{A} = \boldsymbol{M} \tag{3}$$

where $\boldsymbol{P}$ is a matrix in which column k contains the parameter variations of model set k:

$$P_{k,1...m} = (1, p_1(k), \ldots, p_n^{(k)}, \ldots, p_1^{(k)2}, \ldots, p_n^{(k)2}, p_1^{(k)} p_2^{(k)}, \ldots, p_{n-1}^{(k)} \cdot p_n^{(k)}).$$

The $m = 1 + n + n(n+1)/2$ coefficients $a^{(0,1,2)}$ of the surrogate function are unknown and determined by fitting Eq. (3) to ℓ simulation distributions ($\ell \geq m$), generated with different parameter settings.

Solving equation (3) is to minimize the loss function:

$$\mathcal{L} = ||\boldsymbol{M} - \boldsymbol{P} \cdot \boldsymbol{A}||^2.$$

It can be solved by inverting the matrix $\vec{P}$. Often is the case that there are more experimental runs than the number of coefficients, making the equation over-determined. Therefore, a simple matrix inversion based on singular value decomposition may not be robust. We find adding a penalty term such as lasso or ridge helps to stabilize the optimization process.

2.2 Outer loop optimization

After the surrogate function is optimized, the next step is to optimize the generator parameters by minimizing the χ^2 function:

$$\chi^2 = \sum_i^B \frac{[d_i - f(\boldsymbol{p}, x_i)]^2}{\sigma_{d_i}^2}$$

where i loops over all B bins, $f_i(\boldsymbol{p})$ is the surrogate function for the i-th bin, d_i is the experimental measurement, and σ_{d_i} is the uncertainty of the measurement. Throughout the procedure, no MC uncertainties are taken into account.

3 Tuning with MC uncertainties

A straightforward method to incorporate the MC uncertainties is to use another surrogate function $g(\boldsymbol{p})$ to model the relationship between the generator parameters and the MC uncertainties e_{MC} for each bin. This surrogate function can be obtained similarly to the one for the nominal values. Then, the χ^2 function can be modified to include the MC uncertainties:

$$\chi^2 = \sum_i^B \frac{[d_i - f(\boldsymbol{p}, x_i)]^2}{\sigma_{d_i}^2 + g(\boldsymbol{p}, x_i)^2}$$

This method is referred to as `MC-Tune-Error` in the following sections.

We propose propagating the MC uncertainties to the surrogate function and incorporating that in the χ^2 function. We minimize the following loss function to obtain the best estimates of the surrogate function coefficients $\boldsymbol{A}$ and the covariance matrix Σ of the coefficients:

$$\mathcal{L} = ||\boldsymbol{M} - \boldsymbol{P} \cdot \boldsymbol{A}||^2 / \boldsymbol{M}_{\mathrm{error}}^2$$

where $\boldsymbol{M}_{\mathrm{error}}$ is the MC uncertainties associated with the event generators. The covariance matrix Σ of the coefficents of the surrogate function is then used to estimate the surrogate function uncertainties σ_{f_i} in the χ^2 function:

$$\sigma_f^2(\boldsymbol{p}) = \boldsymbol{J} \Sigma \boldsymbol{J}^T$$

where $\boldsymbol{J}$ is the Jacobian matrix of the surrogate function. Now, we can modify the χ^2 function to take into account the MC uncertainties:

$$\chi^2 = \sum_i^n \frac{[d_i - f(\boldsymbol{p}, x_i)]^2}{\sigma_{d_i}^2 + \sigma_{f_i}^2(\boldsymbol{p})}$$

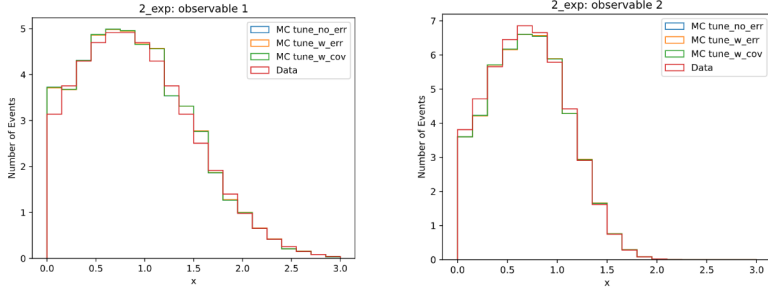This method is referred to as `MC-Conv-Tune` in the following sections.

Figure 1. Toy observables. The red curve labeled as "Data" is the target distribution. Other curves, labeled as "MC" are the distributions generated by generator parameters tuned with different methods as detailed in the text.

4 Toy data setup

We create toy data to evaluate the effectiveness of our method. We define two observables following exponential functions:

$$y_0 = e^{ax_0 + bx_0^2}, \quad y_1 = e^{ax_1 + bx_1^3}$$

a and b are two generator parameters that control the observable distributions. Because generator parameters are often bounded by physical constraints, we also set the boundaries on these tuning parameters in our toy data. Specifically, a is bounded to be $[1, 2]$ and b to be $[-1.2, 0.8]$. The two toy observables are binned into histograms with 20 bins, shown as 'observable 1' and 'observable 2' in Fig. 1.

Following the tuning procedure outlined in Section 2, we randomly sample 30 independent pairs of (a, b) with 100,000 events for each pair. We then use the 3rd-order polynomial function as the surrogate function for all three tuning methods.

5 Results

Figure 1 compares the toy observables between the target distributions and the tuned ones. We see that all three methods can obtain optimal generator parameters that produce distributions that agree with the target distribution. The optimal and true generate parameters are displayed in Fig 2. The `MC-Conv-Tune` finds the optimal parameters closest to the true parameters because it takes into account the MC uncertainties properly. Additionally, we've included a contour plot that shows where the value of the objective function exceeds its minimum value by the number of degrees of freedom. The `MC-Tune-NoError` yields a very narrow contour, indicating the estimated errors are underestimated. That confirms the findings from Ref [3], where the authors did not use the number of degrees of freedom to estimate the errors but instead used an educated guess of the threshold, see the Eigentune method in Ref [3]. On the other hand, the `MC-Tune-Error` yields a very wide contour, indicating the estimated errors are overestimated. This is because the observable values and their errors should not be modeled with independent surrogate functions. The `MC-Conv-Tune` yields a contour that is in between the other two methods and encompasses the true parameters.
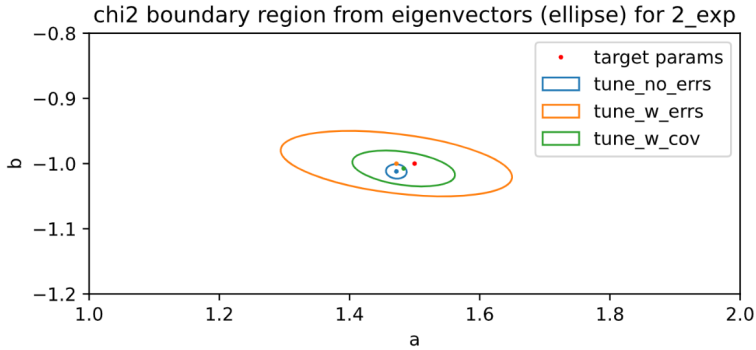
Figure 2. 68% confidence level contour in the a and b plane. The solid dots are the optimal generator parameters obtained with different methods.
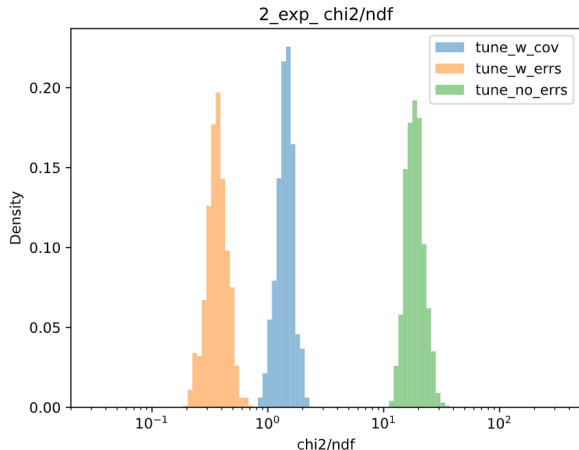

Figure 3. χ^2 over the number of degrees of freedom for different tuning methods obtained with 100 trials.

To check the stability of the tuning methods, we repeat the tuning procedure 100 times. Figure 3 shows the χ^2 distribution over the number of degrees of freedom. All algorithms yield a relatively narrow width. However, the `MC-Tune-Error` and `MC-Tune-NoError` have a slightly larger tail fraction. As inferred from Fig. 2, the `MC-Conv-Tune` peaks around one, while the other two methods yield either much larger or smaller values.

6 Conclusion

We propose a new method to incorporate the MC systematic uncertainties into the MC tuning procedure. We evaluate the method with a toy example and find that the method yields better optimal generator parameters and uncertainty estimations. The method can be easily extended to include different sources of uncertainties.

MC uncertainties are often independent of the experimental uncertainties. Thanks to recent developments of the HepData repo and the support of LHC experiments, the

LHC experiments started to report the breakdown of their measurement uncertainties into theoretical and experimental uncertainties. Within our method, we can properly correlate the MC uncertainties with the reported theoretical uncertainties, and uncorrelate them with the experimental uncertainties. Doing so will further improve the error estimations.

However, this method is computationally expensive. It is much slower than the current MC tuning procedure. We are working on optimizing the processes with GPUs or multithreading in CPUs.

The ATLAS A14 Tune [5] made three different tunings for different PDFs due to the intrinsic differences among these PDFs, which, however, are often treated as systematic uncertainties associated with the PDF. It will be interesting to redo the ATLAS tuning with our method and to check if our method helps to reduce the tuning-related systematic uncertainties.

References

[1] T. Sjöstrand, S. Mrenna, P. Skands, Comput. Phys. Commun. **178**, 852 (2008), `0710.3820`

[2] P. Skands, S. Carrazza, J. Rojo, Eur. Phys. J. C **74**, 3024 (2014), `1404.5630`

[3] A. Buckley, H. Hoeth, H. Lacker, H. Schulz, J.E. von Seggern, Eur. Phys. J. **C65**, 331 (2010), `0907.2973`

[4] M. Krishnamoorthy, H. Schulz, X. Ju, W. Wang, S. Leyffer, Z. Marshall, S. Mrenna, J. Müller, J.B. Kowalkowski, EPJ Web Conf. **251**, 03060 (2021), `2103.05748`

[5] ATLAS Collaboration, *ATLAS Pythia 8 tunes to 7 TeV data*, ATL-PHYS-PUB-2014-021 (2014)

[6] J. Alwall et al., Comput. Phys. Commun. **176**, 300 (2007), `hep-ph/0609017`

[7] J.R. Andersen et al. (2014), `1405.1067`