Nuclear Engineering and Technology 54 (2022) 1902–1908

Contents lists available at ScienceDirect

Nuclear Engineering and Technology

journal homepage: www.elsevier.com/locate/net

Technical Note

Performing linear regression with responses calculated using Monte Carlo transport codes

Dean Price*, Brendan Kochunas

Department of Nuclear Engineering and Radiological Science, University of Michigan, Ann Arbor, 2355 Bonisteel Blvd., Ann Arbor, MI, 48109, USA

ARTICLE INFO

Article history:
Received 18 August 2021
Received in revised form
2 November 2021
Accepted 3 November 2021
Available online 12 November 2021

Keywords:
Linear regression
Sensitivity analysis
Monte Carlo
Uncertainty quantification

ABSTRACT

In many of the complex systems modeled in the field of nuclear engineering, it is often useful to use linear regression-based analyses to analyze relationships between model parameters and responses of interests. In cases where the response of interest is calculated by a simulation which uses Monte Carlo methods, there will be some uncertainty in the responses. Further, the reduction of this uncertainty increases the time necessary to run each calculation. This paper presents some discussion on how the Monte Carlo error in the response of interest influences the error in computed linear regression coefficients. A mathematical justification is given that shows that when performing linear regression in these scenarios, the error in regression coefficients can be largely independent of the Monte Carlo error in each individual calculation. This condition is only true if the total number of calculations are scaled to have a constant total time, or amount of work, for all calculations. An application with a simple pin cell model is used to demonstrate these observations in a practical problem.

1. Introduction

Often in nuclear engineering, it is desired to obtain the sensitivity of some response of interest to changes in model parameters. Linear regression based methods are often used to perform these analyses, and many tools used in the nuclear engineering community for uncertainty quantification and sensitivity analysis contain them [1–3]. The general steps to perform an analysis using a linear regression based method is to first specify a relevant parametric domain. For example, the parametric domain may be selected to reflect the uncertainty ranges of different model parameters. This is often where linear regression-based methods are most useful because in these narrow parametric ranges the linearity assumption is the strongest. Nevertheless, after the parametric domain is selected, it is sampled to generate many relationships between points in the parametric domain and values in the response of interest. Finally, some form of linear regression model is selected and a fit is performed. From different sampling strategies, to linear model forms and fitting methods, a wealth of literature exists [4–7] that details a wide array of methods which generally follow these three simple steps. However, in nuclear engineering where many calculations are carried out using Monte Carlo codes, some uncertainty associated with the calculated model responses arises strictly from the Monte Carlo calculation method [8]. Fortunately, most of these codes report some estimate of this uncertainty along with the approximations of the response of interest.

This paper presents the equations that clearly show the impact of this uncertainty on the linear model parameters. However, when applying these findings to ones own sensitivity study, one question remains: *How should the Monte Carlo sampling error be selected when generating samples for linear regression analysis to minimize the impact of the Monte Carlo uncertainty on calculated regression coefficients?* Although, it may be desirable to pick high sampling parameters (e.g. a large number of particles or batches) and achieve low uncertainties in calculated quantities, these calculations are more expensive and take longer to compute. Alternatively, selecting sampling parameters that are too low may negatively impact the convergence of sensitivity estimates despite each calculation running much quicker. This note presents a clear observation regarding the selection of the Monte Carlo error in calculated responses that are used to construct linear model fits with ordinary least squares (OLS). This observation is built on some statistical approximations allow for no full Monte Carlo model evaluations to be required. Instead, only information on the input sampling distributions and the proposed error in the response of interest are

* Corresponding author.
E-mail address: deanrp@umich.edu (D. Price).

https://doi.org/10.1016/j.net.2021.11.003

needed which allow for easy applications of these equations to any problem. And hence, generalized conclusions from these equations. In short, the Monte Carlo error associated with each sample has a minimal impact on computed sensitivity coefficients when the number of samples is adjusted to have a constant total computation time. In fact, in the limit that the model startup time (time the code takes to process the model before beginning the transport sequence) goes to zero, the error in the sensitivity coefficients from the Monte Carlo error is a strict function of total time dedicated to generating samples. Finally, the predicted observation is confirmed by numerical calculations using a pin cell model.

In the rest of the paper, Section 2 provides some brief background on the fundamentals of OLS. In Section 3, the relationship between the Monte Carlo uncertainties and covariance matrix of the regression coefficients is derived under certain assumptions. These assumptions are used to simplify the derivation, and do not need to hold in general. Next, in Section 4, we introduce a model to relate the Monte Carlo calculation time to the uncertainty of the calculated result. Section 5 discusses how the selected error effects the error in the linear model. Finally in Section 6, we demonstrate these theoretical propositions using a PWR pin cell model.

2. Linear regression with OLS

This section is fairly brief because linear regression with OLS is well-documented [9]. Let a set of N total model inputs be taken as some random variable in the form of a vector-valued quantity, $\boldsymbol{x}$, and is distributed according to a multivariate normal distribution given as N, with mean $\boldsymbol{\mu_x}$ and covariance matrix Σ_x:

$$x \sim N(\boldsymbol{\mu_x}, \Sigma_x). \tag{1}$$

The distribution should be selected to reflect the input domain being explored in the specific application scenario. However, for the purposes of this note, it will be assumed that each element in $\boldsymbol{x}$ is scaled such that $\boldsymbol{\mu_x} = 0$. Each realization of $\boldsymbol{x}$ from its distribution is denoted with some subscript i as $\boldsymbol{x_i}$. Using a physics simulation denoted as the general function f, the response of interest y_i can be calculated for each sampled $\boldsymbol{x_i}$ as:

$$y_i = f(\boldsymbol{x_i}). \tag{2}$$

The objective of linear regression is to best approximate y_i using a summation of the elements in $\boldsymbol{x_i}$, each weighted with values contained in some vector, $\boldsymbol{\beta}$, of fitted parameters. In the following notation, an intercept term is included $\boldsymbol{\beta}$, so a 1 should be prepended to the vector $\boldsymbol{x_i}$. This intercept term is denoted with β_0. The linear approximation of the function f is denoted as $\widehat{f}$ and is given below:

$$\widehat{f}(\boldsymbol{x}) = \boldsymbol{x^T}\beta. \tag{3}$$

In order to find the $\boldsymbol{\beta}$ which best approximates f, some total number of samples I can be obtained by performing I evaluations of f on realizations of $\boldsymbol{x}$. Then, an overconstrained system of equations can be formed as:

$$X\boldsymbol{\beta} = y \tag{4}$$

$$X = \begin{bmatrix} - & \boldsymbol{x_1^T} & - \\ - & \boldsymbol{x_2^T} & - \\ - & \vdots & - \\ - & \boldsymbol{x_I^T} & - \end{bmatrix}, \tag{5}$$

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_I \end{bmatrix} \tag{6}$$

and

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_N \end{bmatrix}. \tag{7}$$

To obtain a solution for $\boldsymbol{\beta}$ which best maps X to $\boldsymbol{y}$, the normal equations [10] can be used and manipulated to obtain a solution for $\boldsymbol{\beta}$:

$$\boldsymbol{\beta} = \left(X^T X\right)^{-1} X^T y. \tag{8}$$

This solution of $X\boldsymbol{\beta}$ minimizes the sum-of-squared differences between $\boldsymbol{y}$ and $X\boldsymbol{\beta}$.

3. Impact of Monte Carlo uncertainty in calculated quantities

There is some influence that Monte Carlo uncertainty in y_i has on estimates for $\boldsymbol{\beta}$. Although it can certainly be said that there may be correlations between model results associated with different executions of a Monte Carlo code, for this paper it will be assumed that each model evaluation is completely independent of all other model evaluations. Moreover, in this line of analysis, the same variance in the response of interest, σ_y^2, is assumed across a set of samples. This assumption is not required but it does lead to more interpretable expressions later. It is worth noting that the σ_y^2 reported in most Monte Carlo neutron transport codes is actually a biased estimate for the true uncertainty from the Monte Carlo calculation method [11]. Nevertheless, the error in $\boldsymbol{y}$ can be accounted for mathematically by writing it as a random variable with mean (μ_y) and error (σ_y^2) reported by the Monte Carlo code as:

$$y \sim N(\boldsymbol{\mu_y}, \Sigma_y). \tag{9}$$

Here,

$$\Sigma_y = \sigma_y^2 I_N \tag{10}$$

where I_N is the identity matrix of size $N \times N$.

Now that $\boldsymbol{y}$ is defined to behave as a random variable, the definition for $\boldsymbol{\beta}$ must be updated as well to include random behavior. Following the solution $\boldsymbol{\beta}$ given in Equation (8), the expected value for $\boldsymbol{\beta}$, $\boldsymbol{\mu_\beta}$, can be written as:

$$\boldsymbol{\mu_\beta} \equiv E[\boldsymbol{\beta}] = \left(X^T X\right)^{-1} X^T \boldsymbol{\mu_y} \tag{11}$$

The derivation for an approximation for the covariance matrix of

$\boldsymbol{\beta}$, Σ_β, is a bit more complicated but does not require anything beyond algebra of random variables:

$$\Sigma_\beta \equiv Cov[\boldsymbol{\beta}] = \left(X^T X\right)^{-1} X^T \Sigma_y X \left(X^T X\right)^{-1}, \tag{12}$$

$$\Sigma_\beta = \sigma_y^2 \left(X^T X\right)^{-1}. \tag{13}$$

Before moving on, some discussion on this expression is warranted. First, it should be unsurprising that the variance in the estimated $\boldsymbol{\beta}$ scales linearly with the variance in the response of interest introduced from the Monte Carlo calculation method, σ_y^2. This linear relationship between error in $\boldsymbol{\beta}$ and σ_y^2 is the driving force for the content of this paper because when actually performing calculations, σ_y^2 reduces roughly as $\mathrm{O}\left(1/\sqrt{t}\right)$, where t is calculation time. This can lead to large computational costs if σ_y^2 is low. Therefore, it may be more efficient to allow σ_y^2 to remain relatively high and instead attempt to reduce Σ_β through the $(X^T X)^{-1}$ factor using intelligent selections for X.

On that note, because each element in $\boldsymbol{x}$ was taken from a zero-centered distribution ($\boldsymbol{\mu_x} = 0$), the quantity $\frac{1}{I-1} X^T X$ can be used as an approximation of the covariance matrix of $\boldsymbol{x}$, denoted earlier as Σ_x [12, Ch. 14]. Furthermore, $(I-1)(X^T X)^{-1}$ is an estimate for the inverse of the covariance matrix (Σ_x)—which is sometimes called the precision matrix. Making the approximation that $X^T X = I\Sigma_x$, yields the following:

$$\Sigma_\beta \approx \frac{\sigma_y^2}{I} \Sigma_x^{-1}. \tag{14}$$

Although this leads to a weak estimate of Σ_β when I is small, this has a negligible effect on this work and simplifies the final expression. Furthermore, in many multivariate analyses, many samples will need to be run to obtain good linear estimates for the trends within the sampled domain regardless of the status of the error in model parameters due to Monte Carlo error. This work is intended to inform readers of the optimal Monte Carlo simulation parameters to minimize the uncertainties in their results and this particular approximation leads to the development of an expression for the relationship between I, σ_y^2 and Σ_β which does not even require the use a computer to draw samples from $\boldsymbol{x}$. The suggestions given in this note are intended to be non-intrusive and provide insight on how the Monte Carlo error selected in each sample effects the error in the regression coefficients. As such, further analysis on the relationships between Σ_β and Σ_x^{-1} which may require adjustments to sampling methods for error minimization are not discussed. To summarize, the main assumptions made to get Equation (14) are as follows:

- The variance in y is the variance reported by the Monte Carlo method is σ_y^2. In reality, it is likely that the reported variance underestimates the true σ_y^2.
- σ_y^2 is constant across all calculations of the response of interest.
- $\frac{1}{I} X^T X$ can reasonably approximate Σ_x^{-1} due to the large sample sizes typically associated with Monte Carlo uncertainty quantification and sensitivity analyses.

It is useful here to define an error scaling factor $\tilde{e}$ which, in the instance that both Σ_β and Σ_x are known, can be calculated with:

$$\tilde{e} = \frac{||\Sigma_\beta||_2}{\left|\left|\Sigma_x^{-1}\right|\right|_2}. \tag{15}$$

From Equation (14), it is clear that:

$$\tilde{e} = \frac{\sigma_y^2}{I}. \tag{16}$$

4. Code calculation time model

Next, it is necessary to provide a real estimation for the relationship between calculation time, t, and error due to the Monte Carlo sampling method in the parameter of interest, σ_y. A practical form for an equation to estimate the code wall time, or real time, can be given as:

$$\sigma_y = \frac{k}{\sqrt{t-a}} \tag{17}$$

In this equation, k and a are fitted variables which can be estimated from the runtime of a code for a particular model. The form of this equation comes about by applying a rightward shift of a to the typical $1/\sqrt{t}$ convergence associated with Monte Carlo methods. An optimal convergence rate of $1/\sqrt{t}$ is assumed. a represents the amount of time taken for the code to actually begin spawning particles, it will include the time a code takes to plot the model geometry, load cross-sections and read input files, etc. k is proportional to the time per iteration and can be estimated using a linear regression on the log-error versus log-time relationship. In estimating these parameters from a single code execution, it is recommended that a value for a is directly retrieved from code output and separately k estimated from code-reported error versus time data. The form of this equation is valid in executions where multiprocessing is used. Rearranging for t we get:

$$t = \left(\frac{k}{\sigma_y}\right)^2 + a. \tag{18}$$

From here, the total calculation time (T) required to generate all the contained in $\boldsymbol{y}$ can be obtained by multiplying this expression by I:

$$T = I\left(\frac{k}{\sigma_y}\right)^2 + Ia. \tag{19}$$

5. Total computing time optimization

Now that expressions relating total computing time T with the error in the response of interest, σ_y, exist in Equation (19) and expressions relating σ_y to the model error scaling factor $\tilde{e}$ exist in Equation (16), an expression for T can be obtained by combining these equations in terms of σ_y^2:

$$T = \frac{a}{\tilde{e}} \sigma_y^2 + \frac{k^2}{\tilde{e}}. \tag{20}$$

Two observations can be made based on this equation:

1. If code startup time is negligible ($a = 0$), total computing time to obtain some error in computed regression coefficients is

independent of the Monte Carlo error arising from each neutron transport calculation.
2. As code startup time increases, more efficiency is gained by running calculations to lower Monte Carlo error. However, it is relatively unlikely that a code startup time will be large enough for this efficiency gain to be significant.

To provide some practical understanding of Equation (20) some σ_y vs T curves are provided in Fig. 1 with a value for $k = 0.000577\sqrt{\text{min}}$ obtained from the pin cell model described later in Section 6.2. In this figure, a few different $\tilde{e}$ and a are selected with values based around the pin cell model used to obtain k. In the left subfigure, three different lines with three different values of $\tilde{e}$ are shown with the same $a = 3$s. From this figure, it is clear that $\tilde{e}$ has little effect on the curvature of the trend. For the practical values of a and k, changing $\tilde{e}$ results in an almost vertical translation of the trend because the second term dominates Equation (20) when a realistic a is selected. In the right subfigure, three different lines with three different values of a are shown with the same $\tilde{e} = 10^{-10}$. Here, it is clear that a influences the curvature of the trend. However, to demonstrate this, unrealistic values of a were selected considering that k was fitted using the pin cell model. None of the 5400 pin cell models generated in this study yielded a value of a greater than 5 s. Even selecting a suboptimal σ_y for the samples would only yield a 10% increase in computational time if the pin cell took three times longer to begin the transport sequence. This increase in computational time goes down to about 4% when using the actual code startup time observed. Furthermore, in more complicated models where a would be larger, it is likely that the added complexity would also decrease the efficiency in the convergence of the code yielding a larger k—reinforcing the overall dominance of the second term. Overall, this is to show that in any realistic scenario, T will be dominated by the eventual desired error in the regression coefficients $\tilde{e}$. It is very unlikely for the analysts' choice of σ_y to have a large effect on the total computational time.

In terms of using these observations to guide analyses carried out using linear regression, it is important to assert that some number of samples is actually required to generate strong estimates for linear model parameters even if there is no Monte Carlo error associated with the results. The number of samples required depends on the number of model inputs, the linearity of the response of interest with respect to the model inputs and the domain which is sampled. Therefore, the key takeaway here is not necessarily that each sample should be run to low error. Instead, the key takeaway is that when conducting linear regression analyses, there is little advantage to be gained by carefully selecting a specific Monte Carlo error associated with each sample. The error in linear model coefficients will be dominated by total time dedicated to the calculation. This means that the analyst has more freedom when selecting the Monte Carlo error associated with each sample and can more appropriately cover the sample space with many samples with higher error with only a minor penalty in total runtime to obtain some $\tilde{e}$.

6. Example application

To illustrate a use-case for the information presented above, a sensitivity analysis is performed using a simple 2D pin cell model. Many applications exist for linear regression, it is thought that a dimensionalized sensitivity analysis may be the most straightforward. Often, sensitivity analysis is used in the context of uncertainty quantification to find the contribution of each uncertain input to overall uncertainty associated with a quantity. Here, sensitivity analysis will simply refer to obtaining a linear estimate for the response of interest to perturbations in some input parameters.

6.1. Pin cell model and input parameters

Fig. 2 shows the pin cell model with some nominal dimensions consisting of UO_2 fuel and zirconium cladding surrounded by water. No fuel-clad gap is considered. The inputs whose sensitivity is explored are the U-235 enrichment, density of the coolant surrounding the pin, pin pitch, cladding density and radius of the fuel pellet. The response of interest is the k_∞ of the pin cell configuration and will be calculated using Serpent 2 [13]. The pin cell model

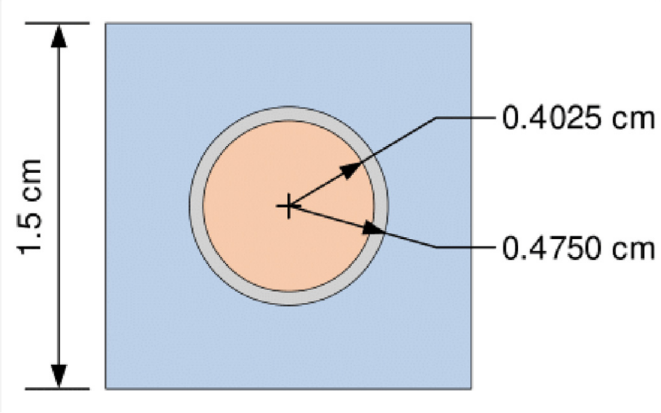

Fig. 2. Nominal dimensions of pin cell model used for example application.

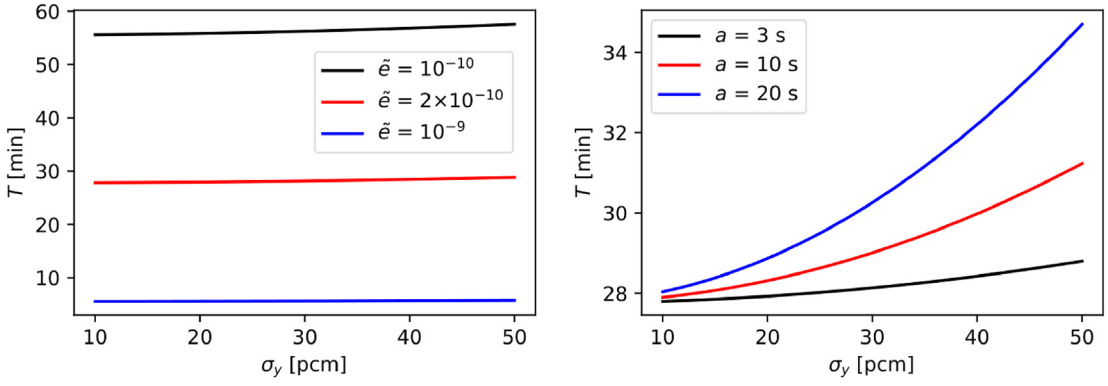

Fig. 1. Visualizations of Equation (20) with different $\tilde{e}$ and a.

is based on one of the Serpent example inputs provided in the code manual.

Table 1 shows the distributions used to generate the samples. In a real application, the domain represented by these sampling distributions may represent the uncertainty ranges in these parameters. Here, obtaining a sensitivity estimate for k_∞ to the input parameters gives a best-estimate linear response of k_∞ to perturbations to these parameters within their uncertainty ranges. In this table, the "2 × CoV" columns refer to twice the coefficient of variation, which can be obtained by dividing twice the standard deviation by the mean. It may be used represent the arbitrarily selected relative uncertainty for these quantities.

The required calculated quantities for the theoretical basis presented in Section 4 are the code runtime parameters shown in Equation (17). To estimate these parameters, a single instance of the model was run with all inputs set to their mean values and the time to start the transport calculations, represented by a, was directly reported by Serpent to be 3.17 s. The calculation was run to a final σ_y of 8 pcm, but the Monte Carlo error was reported periodically. As mentioned earlier, by doing a linear regression between $\log(t)$ and $\log(\sigma_{y(t)})$, k can be estimated as well as the validity of the $1/\sqrt{t}$ convergence can be checked. For these fits, the first 10% of error reports are not used to allow time for the code to perform reasonable estimates for σ_y. Fig. 3 shows the code-reported error as a function of time along with the linear regression fit. This fit resulted in a k value of 0.000577 $\sqrt{\text{min}}$ and a slope of −0.4996, very close to the expected −0.5 from Monte Carlo methods.

6.2. Pin cell calculation results

In this section, the results do not rely on algebraic approximations to simplify the calculation of Σ_β or theoretical code-timing models. For these results, 7 sets of 600 samples were generated, each with different Monte Carlo sampling parameters. Furthermore, Σ_β is calculated using Equation (12) which does not rely on the assumption that $X^TX = I\Sigma_x$ or the assumption of constant σ_y. In practice, small variations in σ_y are still observed across samples calculated with identical Monte Carlo sampling parameters. Also, given that in this instance $\Sigma_x$and Σ_β are known, $\tilde{e}$ can be calculated according to Equation (15). Finally, all input parameters are scaled to have means of 0 in the analysis to ensure consistency when making comparisons between this application and the theoretical discussion.

Fig. 4 shows the evolution of $\tilde{e}$ as β is calculated using an I-sized subset of the 600 total samples from each of the 7 sets. As shown in the legend, the mean σ_y is each of the sets is used to uniquely identify the data points associated with each of the sets. In both subfigures, the same data is displayed. However, in the right subfigure, instead of using I on the x-axis, the total computer time to generate the I samples is used. No model is used to approximate computation time, the Serpent reported calculation time for each sample was used. In these plots, there is some noise in the lower sample ranges because it takes a few samples to generate reasonable estimates for Σ_β. It is clear that changing the axis from I to calculation time T removes most of the differences in the trends between the 7 sample sets. The results shown here qualitatively support the observations made in Section 5 that the error in the regression coefficients is dominated almost entirely by calculation time. Hence the lack of difference in trends created using sets of samples with significantly different Monte Carlo error.

Table 1
Sampling distribution used to generate perturbed input models for linear regression.

Parameter	Mean	Standard Deviation	2 × CoV
U-235 Enrichment	2.0%	0.05%	5%
Coolant Density	1.0 g/cm^3	0.025 g/cm^3	5%
Fuel Radius	0.4025 cm	0.004025 cm	2%
Pin Pitch	1.5 cm	0.0225 cm	3%
Cladding Density	6.55 g/cm^3	0.16375 g/cm^3	5%

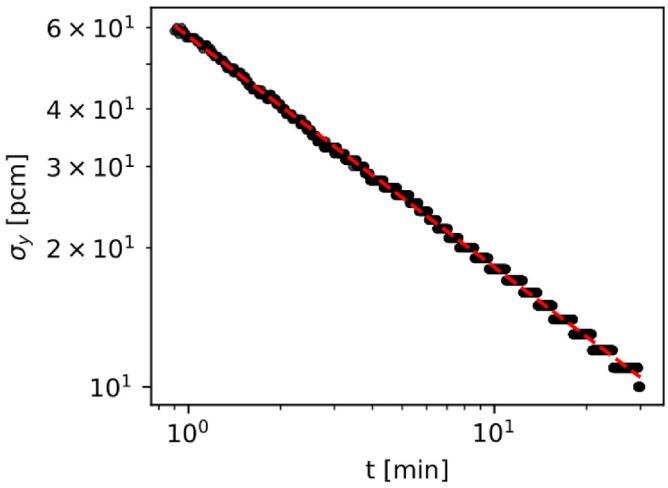

Fig. 3. Code reported Monte Carlo error over time. Regression line used to estimate $k$shown in Equation (17).

6.3. Quantitative comparison with prediction

The methods shown in Section 5 require no calculation beyond a code-timing fit to predict the impact of the Monte Carlo error on calculated regression coefficients as a function of computing time. The use of a pin cell model and many transport code calculations demonstrated that the same conclusions could be reached by actually carrying out a regression analysis. This section will quantitatively compare the relationship in $\tilde{e}$ and T described by Equation (20) to the relationship observed in the pin cell application.

Fig. 5 shows the relationship between T and $\tilde{e}$ predicted by Equation (20) and the same relationship observed in the pin cell case study. In this figure, the red dashed line represents predicted trends, the points represent observations from the pin cell case study. Unlike Fig. 4, total computing time is the x-axis and $\tilde{e}$ is the y-axis. Overall, there is enough agreement between the predicted trend and the observed trend to support the conclusions formed in Section 5. It seems that the code runtime for a particular error is consistently overpredicted by Equation (20). This is likely caused by inaccuracies in the estimation of the timing parameter k given that it is obtained using a single code execution. However, these inaccuracies have a minimal effect on the conclusions drawn from the form of Equation (20).

7. Conclusions

The content of this manuscript can be used to guide the selection of the Monte Carlo sampling parameters used in models that are undergoing a sensitivity analysis using linear regression techniques. The selection of Monte Carlo sampling parameters ultimately determines the uncertainty in code-calculated quantities due to the stochastic nature of the Monte Carlo solution method. A model of this error as a function of code runtime is combined with a statistically-derived expression which relates the error in the linear regression coefficients to the error in code-calculated quantities. The result being an expression which estimates the total computer

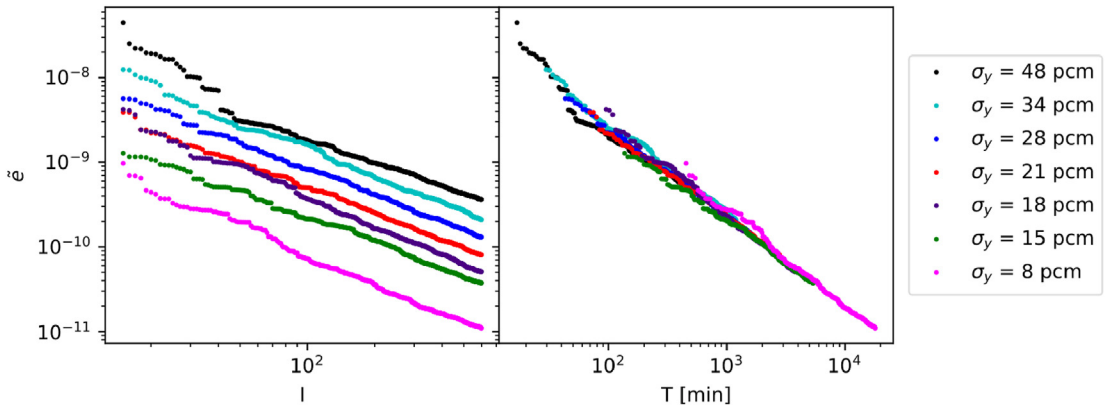

Fig. 4. Σ_β calculated using an I-sized subset of the 600 total samples from each of the 7 sets.

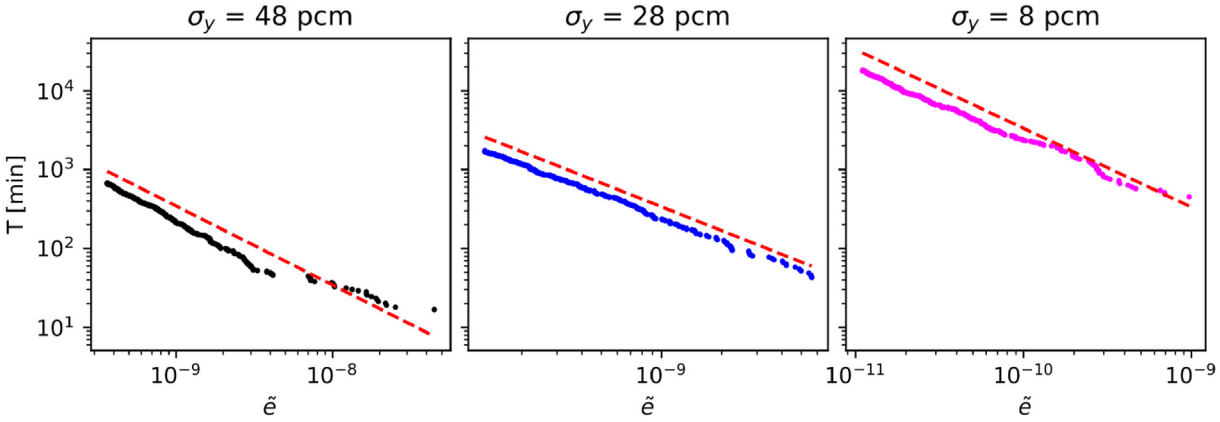

Fig. 5. Comparison of relationship between T and $\tilde{e}$ predicted by Equation (20) and observed in the pin cell case study. The red dashed line represents predicted trends, the points represent observations from the pin cell case study. (For interpretation of the references to colour in this figure legend, the reader is referred to the Web version of this article.)

runtime required as a function of the desired final error in the regression coefficients, calculation timing parameters associated with the model and the error associated with each individual run of the Monte Carlo code. There were two observations made based on this line of work:

1. If code startup time is negligible total computing time to obtain some error in computed regression coefficients is *independent* of the Monte Carlo error arising from each neutron transport calculation.
2. As code startup time increases, more efficiency is gained by running calculations to lower Monte Carlo error. However, it is relatively unlikely that a code startup time will be large enough for this efficiency gain to be significant.

These observations were supported with an application to a simple light water reactor pin cell. Overall, the exploration contained in this study show that the selection of the error in each Monte Carlo model evaluation has little effect on the uncertainty in the linear regression coefficients given a constant total model runtime.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

We would like to thank Professor Brian Kiedrowski and David Griesheimer for their helpful thoughts during our discussions concerning the subject of this work. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE 1256260.

References

[1] K. Dalbey, M.S. Eldred, G. Geraci, J.D. Jakeman, K.A. Maupin, J.A. Monschke, D.T. Seidl, L.P. Swiler, A. Tran, F. Menhorn, et al., Dakota, a Multilevel Parallel Object-Oriented Framework for Design Optimization, Parameter Estimation, Uncertainty Quantification, and Sensitivity Analysis: Version 6.12theory Manual, SAND2020-4987, Sandia National Laboratory, 2020.
[2] A. Alfonsi, C. Wang, P. Talbot, M.M. Abdo, M. Gamal, D. Mandelli, C. Rabiti, J.J. Cogliati, R.A. Kinoshita, Raven User Manual, Tech. rep., Idaho National Lab.(INL), 2021. Idaho Falls, ID (United States).
[3] M. Reyes-Fuentes, E. del Valle-Gallegos, J. Duran-Gonzalez, J. Or t ız-Villafuerte, R. Castillo-Dur an, A.G. omez-Torres, C. Queral, Aztusia: a new application software for un certainty and sensitivity analysis for nuclear reactors, Reliab. Eng. Syst. Saf. 209 (2021) 107441.
[4] C. Yu, W. Yao, Robust linear regression: a review and comparison, Commun. Stat. Simulat. Comput. 46 (8) (2017) 6261–6282.
[5] W. Zwermann, N. Berner, A. Aures, K. Velkov, Sensitivity and uncertainty analysis for the uam-sfrsub-exercises with linear regression from random sampling, Ann. Nucl. Energy 149 (2020) 107832.
[6] N.R. Draper, R.C. Van Nostrand, Ridge regression and james-stein estimation: review and comments, Technometrics 21 (4) (1979) 451–466.
[7] R. Tibshirani, Regression shrinkage and selection via the lasso: a retrospective, J. Roy. Stat. Soc. B 73 (3) (2011) 273–282.
[8] D. Price, A. Maile, J. Peterson-Droogh, D. Blight, A Methodology for Uncertainty Quantification and Sensitivity Analysis for Responses Subject to Monte Carlo

Uncertainty with Application to Fuel Plate Char-Acteristics in the Atrc, Nuclear Engineering and Technology.
[9] G.A. Seber, A.J. Lee, Linear Regression Analysis, vol. 329, John Wiley & Sons, 2012.
[10] Y. Dodge, The Concise Encyclopedia of Statistics, Springer New York, New York, NY, 2008.
[11] F.B. Brown, A Review of Best Practices for Monte Carlo Criticality Calculations, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2009.
[12] L. Wasserman, All of Statistics: a Concise Course in Statistical Inference, Springer Science & Business Media, 2013.
[13] J. Leppänen, M. Pusa, T. Viitanen, V. Valtavirta, T. Kaltiaisenaho, The serpent Monte Carlo code: status, development and applications in 2013, Ann. Nucl. Energy 82 (2015) 142–150, joint International Conference on Supercomputing in Nuclear Applications and Monte Carlo 2013, SNA + MC 2013. Pluri-and Trans-disciplinarity, Towards New Modeling and Numerical Simulation Paradigms.