Fast Regression of the Tritium Breeding Ratio in Tokamak Fusion Reactors

G Van Goffrier[1] and P Mánek[1,2], V Gopakumar[3], N Nikolau[1], J Shimwell[3], I Waldmann[1]

[1] Department of Physics and Astronomy, University College London, Gower Street, London WC1E 6BT, UK

[2] Institute of Experimental and Applied Physics, Czech Technical University, Husova 240/5, Prague 110 00, Czech Republic

[3] UK Atomic Energy Authority, Culham Science Centre, OX14 3DB Abingdon, UK

E-mail: graham.vangoffrier.19@ucl.ac.uk, petr.manek.19@ucl.ac.uk

Abstract. The tritium breeding ratio (TBR) is an essential quantity for the design of modern and next-generation Tokamak nuclear fusion reactors. Representing the ratio between tritium fuel generated in breeding blankets and fuel consumed during reactor runtime, the TBR depends on reactor geometry and material properties in a complex manner. In this work, we explored the training of surrogate models to produce a cheap but high-quality approximation for a Monte Carlo TBR model in use at the UK Atomic Energy Authority. We investigated possibilities for dimensional reduction of its feature space, reviewed 9 families of surrogate models for potential applicability, and performed hyperparameter optimisation. Here we present the performance and scaling properties of these models, the fastest of which, an artificial neural network, demonstrated $R^2 = 0.985$ and a mean prediction time of $0.898\,\mu s$, representing a relative speedup of $8 \cdot 10^6$ with respect to the expensive MC model. We further present a novel adaptive sampling algorithm, Quality-Adaptive Surrogate Sampling, capable of interfacing with any of the individually studied surrogates. Our preliminary testing on a toy TBR theory has demonstrated the efficacy of this algorithm for accelerating the surrogate modelling process.

Index terms— magnetic moment, solar neutrinos, astrophysics Submitted to: *J. Phys. G: Nucl. Part. Phys.*

1. Introduction

This section was copied from the report "as is" and needs to be shortened.

The analysis of massive datasets has become a necessary component of virtually all technical fields, as well as the social and humanistic sciences, in recent years. Given that rapid improvements in sensing and processing hardware have gone hand in hand with the data explosion, it is unsurprising that software for the generation and interpretation of this data has also attained a new frontier in complexity. In particular, simulation procedures such as Monte Carlo (MC) event generation can perform physics predictions even for theoretical regimes which are not analytically tractable. The bottleneck for such procedures, as is often the case, lies in the computational time and power which they necessitate.

Surrogate models, or metamodels, can resolve this limitation by replacing a resource-expensive procedure with a much cheaper approximation [1]. They are especially useful in applications where numerous evaluations of an expensive procedure are required over the same or similar domains, e.g. in the parameter optimisation of a theoretical model. The term "metamodel" proves especially meaningful in this case, when the surrogate model approximates a computational process which is itself a model for a (perhaps unknown) physical process [2]. There exists a spectrum between "physical" surrogates which are constructed with some contextual knowledge in hand, and "empirical" surrogates which are derived purely from samples of the underlying expensive model.

In this project, in coordination with the UK Atomic Energy Authority (UKAEA), we sought to develop a surrogate model for the tritium breeding ratio (TBR) in a Tokamak nuclear fusion reactor. Our expensive model was an MC-based neutronics simulation, *Paramak*‡, which returns a prediction of the TBR for a given configuration of a spherical Tokamak. We took an empirical approach to the construction of this surrogate, and no results described here are explicitly dependent on prior physics knowledge.

For the remainder of Section 1, we will define the TBR and set the context of this work within the goals of the UKAEA. In Section 2 we will describe our datasets generated from the expensive model for training and validation purposes, and the dimensionality reduction methods employed to develop our understanding of the parameter domain. In Section 3 we will present our methodologies for the comparison testing of a wide variety of surrogate modelling techniques, as well as a novel adaptive sampling procedure suited to this application. After delivering the results of these approaches in Section 4, we will give our final conclusions and recommendations for further work.

‡ Provided by collaborator Jonathan Shimwell, at UKAEA.

Figure 1. Typical single-null reactor configuration as specified by BLUEPRINT [5]: 1 — plasma, 2 — breeding blankets

1.1. Problem Description

Nuclear fusion technology relies on the production and containment of an extremely hot and dense plasma. In this environment, by design similar to that of a star, hydrogen atoms attain energies sufficient to overcome their usual electrostatic repulsion and fuse to form helium [3]. Early prototype reactors made use of the deuterium (^{2}H, or D) isotope of hydrogen in order to achieve fusion under more accessible conditions, but lead to limited success. The current frontier generation of fusion reactors, such as the Joint European Torus (JET) and the under-construction International Thermonuclear Experimental Reactor (ITER), make use of tritium (^{3}H, or T) fuel for further efficiency gain. Experimentation at JET dating back to 1997 [4] has made significant headway in validating deuterium-tritium (D-T) operations and constraining the technology which will be employed in ITER in a scaled-up form.

However, tritium is much less readily available as a fuel source than deuterium. While at least one deuterium atom occurs for every 5000 molecules of naturally-sourced water, and may be easily distilled, tritium is extremely rare in nature. It may be produced indirectly through irradiation of heavy water (D_2O) during nuclear fission, but only at very low rates which could never sustain industrial-scale fusion power.

Instead, modern D-T reactors rely on tritium breeding blankets, specialised layers of material which partially line the reactor and produce tritium upon neutron bombardment, e.g. by

$$^1_0\mathrm{n} + {}^6_3\mathrm{Li} \longrightarrow {}^3_1\mathrm{T} + {}^4_2\mathrm{He} \tag{1}$$

$$^1_0\mathrm{n} + {}^7_3\mathrm{Li} \longrightarrow {}^3_1\mathrm{T} + {}^4_2\mathrm{He} + {}^1_0\mathrm{n} \tag{2}$$

where T represents tritium and ^{7}Li, ^{6}Li are the more and less frequently occurring isotopes of lithium, respectively. ^{6}Li has the greatest tritium breeding cross-section of all tested isotopes [3], but due to magnetohydrodynamic instability of liquid lithium in the reactor environment, a variety of solid lithium compounds are preferred.

The TBR is defined as the ratio between tritium generation in the breeding blanket per unit time and tritium fuel consumption in the reactor. The MC neutronics simulations previously mentioned therefore must account for both the internal plasma dynamics of the fusion reactor and the resultant interactions of neutrons with breeding blanket materials. Neutron paths are traced through a CAD model (e.g. Figure 1) of a reactor with modifiable geometry.

The input parameters of the computationally-expensive TBR model therefore fall into two classes. Continuous parameters, including material thicknesses and packing ratios, describe the geometry of a given reactor configuration. Discrete categorical parameters further specify all relevant material sections, including coolants, armours, and neutron multipliers. One notable exception is the enrichment ratio, a continuous

parameter denoting the presence of ^{6}Li. Our challenge, put simply, was to produce a fast TBR function which takes these same input parameters and approximates the MC TBR model with the greatest achievable regression performance.

2. Methodology

Labeling the expensive MC TBR model $f(x)$, a surrogate is a mapping $\hat{f}(x)$ such that $f(x)$ and $\hat{f}(x)$ minimise a selected dissimilarity metric. In order to be considered *viable*, $\hat{f}(x)$ is required to achieve expected evaluation time lower than that of $f(x)$. In this work, we consider two methods of producing viable surrogates: (1) a conventional decoupled approach, which evaluates $f(x)$ on a set of randomly sampled points and trains surrogates in a supervised scheme, and (2) an adaptive approach, which attempts to compensate for localised regression performance insuficiencies by interleaving multiple epochs of sampling and training.

For both methods, we selected state-of-the-art regression algorithms to perform surrogate training on sampled point sets. Listed in table 1, these implementations define 9 surrogate families that are later reviewed in section 3. We note that each presented algorithm defines hyperparameters that may influence its performance. Since their optimal values for this problem are unknown, we explore their assignments prior to other experiments.

Table 1. Considered surrogate model families. $\mathcal{H}$ denotes hyperparameter set.

Surrogate	Acronym	Implementation	$\lvert\mathcal{H}\rvert$
Support vector machines [6]	SVM	SciKit Learn [7]	3
Gradient boosted trees [8, 9, 10]	GBT	SciKit Learn	11
Extremely randomised trees [11]	ERT	SciKit Learn	7
AdaBoosted decision trees [12]	ABT	SciKit Learn	3
Gaussian process regression [13]	GPR	SciKit Learn	2
k nearest neighbours	KNN	SciKit Learn	3
Artificial neural networks	ANN	Keras (TensorFlow) [14]	2
Inverse distance weighing [15]	IDW	SMT [16]	1
Radial basis functions	RBF	SMT	3

To compare quality of produced surrogates, we define a number of metrics listed in table 2. For regression performance analysis, we include a selection of absolute metrics to assess their approximation capability and set practical bounds on the expected uncertainty of their predictions. In addition, we also track relative measures that are better-suited for comparison between this work and others as they maintain invariance with respect to the selected domain and image space. For complexity analysis, surrogates are assessed in terms of wall time (captured by the Python `time` package). This is motivated by common practical use cases of our work, where models are trained and used as drop-in replacements for the expensive MC TBR model. Since training set sizes remain to be determined, all times are reported per a single datapoint. Even though

some surrogates support acceleration by means of parallelisation, sequential processing of samples was ensured to achieve comparability between considered models. The only exception to this are ANNs, which require considerable amount of processing power for training on conventional CPU architectures. Lastly, to prevent undesirable bias by training set selection, all metrics are collected in the scheme of 5-fold cross-validation.

Table 2. Metrics recorded in experiments. In formulations, we work with a training set of size N_0 and a test set of size N, values $y^{(i)} = f(x^{(i)})$ and $\hat{y}^{(i)} = \hat{f}(x^{(i)})$ denote images of the ith testing sample in the MC TBR model and the surrogate respectively. Furthermore, the mean $\overline{y} = \sum_{i=1}^{N} y^{(i)}/N$ and P is the number of input features.

Regression perf. metrics	Notation	Mathematical formulation
Mean absolute error	MAE	$\sum_{i=1}^{N} \lvert y^{(i)} - \hat{y}^{(i)} \rvert / N$
Standard error of regression	S	$\mathrm{StdDev}_{i=1}^{N} \left\{ \lvert y^{(i)} - \hat{y}^{(i)} \rvert \right\}$
Coefficient of determination	R^2	$1 - \sum_{i=1}^{N} \left(y^{(i)} - \hat{y}^{(i)} \right)^2 / \sum_{i=1}^{N} \left(y^{(i)} - \overline{y} \right)^2$
Adjusted R^2	$R^2_{\mathrm{adj.}}$	$1 - (1 - R^2)(N-1)/(N-P-1)$
Complexity metrics		
Mean training time	$\overline{t}_{\mathrm{trn.}}$	(wall training time of $\hat{f}(x)$)$/N_0$
Mean prediction time	$\overline{t}_{\mathrm{pred.}}$	(wall prediction time of $\hat{f}(x)$)$/N$
Relative speedup	ω	(wall evaluation time of $f(x)$)$/(N\overline{t}_{\mathrm{pred.}})$

2.1. Decoupled Sampling

This section needs to be shortened

The presented surrogate candidates are evaluated in four experimental cases:

(i) Hyperparameter tuning in a simplified domain.
(ii) Hyperparameter tuning in full domain.
(iii) Scaling benchmark.
(iv) Model comparison.

The aim of the initial experiments is to use a relatively small subset of collected TBR samples to determine hyperparameters of considered surrogates. Since this process requires learning the behaviour of an unknown, possibly expensive mapping – here a function that assigns cross-validated metrics to a point in the hyperparameter domain – it in many aspects mirrors the primary task of this work with the notable extension of added utility to optimise. In order to avoid undesirable exponential slowdown in exhaustive searches of a possibly high-dimensional parameter space, Bayesian optimisation [17] is employed as a standard hyperparameter tuning algorithm. We set its objective to maximise R^2 and perform 1000 iterations.§

In the first experiment, efforts are made to maximise the possibility of success in surrogates that are prone to suboptimal performance in discontinuous spaces. This

§ Hyperparameter tuning of each surrogate family was terminated after 2 days. Instances that reached this limit may be identified in Table **??** in the Appendix.

follows the notion that, if desired, performance of such models may be replicated by training separate instances to model each continuous subregion of the domain independently. To this end, data are limited to a single slice from run 2, and discrete features are completely withheld from evaluated surrogates. This is repeated for each of the four available slices to investigate variance in behaviour under different discrete feature assignments. The second experiment conventionally measures surrogate performance on the full feature space. Here, in extension of the previous case, surrogates work with samples comprised of discrete as well as continuous features.

The objective of the last two experiments is to exploit the information gathered by hyperparameter tuning. In the third experiment, the 20 best-performing hyperparameter configurations of each family (with respect to R^2) are used to perform training on progressively larger sets to investigate their scaling properties. Following that, the fourth experiment aims to produce surrogates suitable for practical use by retraining selected well-scaling instances on large training sets to satisfy the goals of this work.

2.2. Adaptive Sampling

This section needs to be shortened

Placeholder

Figure 2. Schematic of QASS algorithm

All of the surrogate modelling techniques studied in this project face a common challenge: their accuracy is limited by the quantity of training samples which are available from the expensive MC TBR model. Adaptive sampling procedures can improve upon this limitation by taking advantage of statistical information which is accumulated during the training of any surrogate model. Rather than training the surrogate on a single sample set generated according to a fixed strategy, sample locations are chosen periodically during training so as to best suit the model under consideration.

Adaptive sampling techniques appear frequently in the literature and have been specialised for surrogate modelling. Garud's [18] "Smart Sampling Algorithm" achieved notable success by incorporating surrogate quality and crowding distance scoring to identify optimal new samples, but was only tested on a single-parameter domain. We theorised that a nondeterministic sample generation approach, built around Markov Chain Monte Carlo methods (MCMC), would fare better for high-dimensional models by more thoroughly exploring all local optima in the feature space. MCMC produces a progressive chain of sample points, each drawn according to the same symmetric proposal distribution‖ from the prior point. These sample points will converge to a desired posterior distribution, so long as the acceptance probability for these draws has a particular functional dependence on that posterior value (see [20] for a review).

‖ An adaptive MCMC procedure [19], which adjusts an ellipsoidal proposal distribution to fit the posterior, was also implemented but not fully tested.

Many researchers have embedded surrogate methods into MCMC strategies for parameter optimisation [21, 22], in particular the ASMO-PODE algorithm [23] which makes use of MCMC-based adaptive sampling to attain greater surrogate precision around prospective optima. Our novel approach draws inspiration from ASMO-PODE, but instead uses MCMC to generate samples which increase surrogate precision throughout the entire parameter space.

We designed the Quality-Adaptive Surrogate Sampling algorithm (QASS, Figure 2) to iteratively increment the training/test set with sample points which maximise surrogate error and minimise a crowding distance metric (CDM) [24] in feature space. On each iteration following an initial training of the surrogate on N uniformly random samples, the surrogate was trained and absolute error calculated. MCMC was then performed on the error function generated by performing nearest-neighbor interpolation on these test error points. The resultant samples were culled by 50% according to the CDM, and then the n highest-error candidates were selected for reintegration with the training/test set, beginning another training epoch. Validation was also performed during each iteration on independent, uniformly-random sample sets.

3. Results

4. Conclusion

5. Acknowledgements

6. References

[1] Søndergaard J 2003 *Optimization Using Surrogate Models* Ph.D. thesis Technical University of Denmark
[2] Myers R and Montgomery D 2002 *Response Surface Methodology: Product and Process Optimization Using Designed Experiments* 2nd ed (New York: John Wiley & Sons)
[3] Hernández F and Pereslavtsev P
[4] Keilhacker M
[5] Coleman M and McIntosh S
[6] Fan R E, Chang K W, Hsieh C J, Wang X R and Lin C J 2008 *Journal of machine learning research* **9** 1871–1874
[7] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M and Duchesnay E 2011 *Journal of Machine Learning Research* **12** 2825–2830
[8] Friedman J H 2001 *Annals of statistics* 1189–1232
[9] Friedman J 1999 Stochastic gradient boosting technical report
[10] Hastie T, Tibshirani R and Friedman J 2009 *The elements of statistical learning: data mining, inference, and prediction* (Springer Science & Business Media)
[11] Geurts P, Ernst D and Wehenkel L 2006 *Machine learning* **63** 3–42
[12] Drucker H 1997 Improving regressors using boosting techniques *ICML* vol 97 pp 107–115
[13] Williams C K and Rasmussen C E 2006 *Gaussian processes for machine learning* vol 2 (MIT press Cambridge, MA)
[14] Chollet F *et al.* 2015 Keras `https://keras.io`

[15] Shepard D 1968 A two-dimensional interpolation function for irregularly-spaced data *Proceedings of the 1968 23rd ACM national conference* pp 517–524
[16] Bouhlel M A, Hwang J T, Bartoli N, Lafage R, Morlier J and Martins J R R A 2019 *Advances in Engineering Software* 102662 ISSN 0965-9978
[17] Močkus J 1975 On bayesian methods for seeking the extremum *Optimization techniques IFIP technical conference* (Springer) pp 400–404
[18] Garud S, Karimi I and Kraft M
[19] Zhang J, Chowdhury S and Messac A
[20] Zhou J, Su X and Cui G
[21] Zhang J, Zheng Q, Chen D, Wu L and Zeng L
[22] Gong W and Duan Q
[23] Ginting V, Pereira F, Presho M and Wo S
[24] Solonen A, Ollinaho P, Laine M, Haario H, Tamminen J and Jarvinen H