Fast Regression of the Tritium Breeding Ratio in Fusion Reactors

P Mánek[1,2], G Van Goffrier[1], V Gopakumar[3], N Nikolaou[1], J Shimwell[3] and 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: petr.manek.19@ucl.ac.uk, graham.vangoffrier.19@ucl.ac.uk

Abstract. The tritium breeding ratio (TBR) is an essential quantity for the design of modern and next-generation D-T fueled 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— machine learning, surrogate model, regression, fast approximation, Monte Carlo simulation, OpenMC, Paramak, tritium breeding, adaptive sampling

Submitted to: *Nucl. Fusion*

1. Introduction

Surrogate models were developed to resolve computational limitations in the analysis of massive datasets 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 work, we develop an empirical surrogate model for the tritium breeding ratio (TBR) in an inertial confinement fusion (ICF) reactor. The expensive model that our surrogate model approximates is a Monte Carlo (MC) neutronics simulation, Paramak [3], which returns a prediction of the TBR for a given configuration of a spherical ICF reactor.

Paramak facitilates simulation via OpenMC neutronics workflow that is enclosed in a portable Docker container, which conveniently exposes an HTTP API using the Python 3 `flask` package. For the purposes of this work, we used this setup to simulate a point source with a Muir energy distribution [4] around 14.06 MeV to approximate a Deuterium-Tritium (D-T) plasma neutron source. Illustrated in Figure 1, the simulated geometry was deliberately left tunable, so that dependency of TBR on various parameters may be investigated.

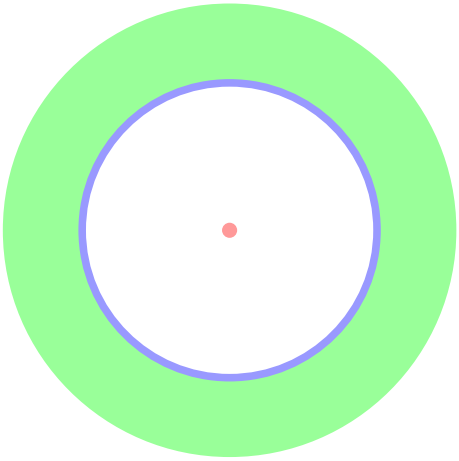

Figure 1. Diagram of the simple sphere geometry (not to scale) where the blanket is , the first wall is and the neutron point source is .

For the remainder of Section 1, we will define the TBR and further motivate our research. In Section 2 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 3, we will give our final conclusions and recommendations in Section 4.

1.1. Problem Description

Nuclear fusion technology relies on the production and containment of an extremely hot and dense plasma containing enriched Hydrogen isotopes. 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 both tritium and deuterium fuel. 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

$$^{1}_{0}\mathrm{n} + {}^{6}_{3}\mathrm{Li} \longrightarrow {}^{3}_{1}\mathrm{T} + {}^{4}_{2}\mathrm{He} \quad (1)$$

$$^{1}_{0}\mathrm{n} + {}^{7}_{3}\mathrm{Li} \longrightarrow {}^{3}_{1}\mathrm{T} + {}^{4}_{2}\mathrm{He} + {}^{1}_{0}\mathrm{n} \quad (2)$$

where T represents tritium and ^{7}Li, ^{6}Li are the more and less common isotopes of lithium, respectively. The TBR is defined as the ratio between tritium generation in the breeding blanket and consumption in the reactor, whose description in Paramak is facilitated by 2 classes of parameters (exhaustively listed in Table 1). While the geometry of a given reactor is described by continuous parameters, material selections are specified by discrete categorical parameters.

In our work, we set out to produce a fast TBR function which takes these same input parameters and approximates the MC model used in Paramak 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 2 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 2, these implementations define 9 surrogate families that are later reviewed in Section 3. We note that each

Table 1. Input parameters supplied to Paramak and surrogates in alphabetical order. Groups of fractions marked†‡ are independently required to sum to 1.

	Parameter name	Domain
Blanket	Breeder fraction†	$[0, 1]$
	Breeder ^{6}Li enrichment fraction	$[0, 1]$
	Breeder material	$\{Li_2TiO_3, Li_4SiO_4\}$
	Breeder packing fraction	$[0, 1]$
	Coolant fraction†	$[0, 1]$
	Coolant material	$\{D_2O, H_2O, He\}$
	Multiplier fraction†	$[0, 1]$
	Multiplier material	$\{Be, Be_{12}Ti\}$
	Multiplier packing fraction	$[0, 1]$
	Structural fraction†	$[0, 1]$
	Structural material	$\{SiC, eurofer\}$
	Thickness	$[0, 500]$
First wall	Armour fraction‡	$[0, 1]$
	Coolant fraction‡	$[0, 1]$
	Coolant material	$\{D_2O, H_2O, He\}$
	Structural fraction‡	$[0, 1]$
	Structural material	$\{SiC, eurofer\}$
	Thickness	$[0, 20]$

Table 2. Considered surrogate model families, their selected abbreviations and implementations. $\mathcal{H}$ denotes the set of hyperparameters.

Surrogate family	Abbr.	Impl.	$\lvert\mathcal{H}\rvert$
Support vector machines [5]	SVM	SciKit [6]	3
Gradient boosted trees [7, 8, 9]	GBT	SciKit	11
Extremely randomised trees [10]	ERT	SciKit	7
AdaBoosted decision trees[a] [11]	ABT	SciKit	3
Gaussian process regression [12]	GPR	SciKit	2
k nearest neighbours	KNN	SciKit	3
Artificial neural networks	ANN	Keras [13]	2
Inverse distance weighing [14]	IDW	SMT [15]	1
Radial basis functions	RBF	SMT	3

[a]Note that ABTs can be viewed as a subclass of GBTs.

Table 3. 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. The mean $\overline{y} = N^{-1}\sum_{i=1}^{N} y^{(i)}$ and P is the number of input features.

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

[b]This corresponds to evaluation of Paramak on all points of the test set. In surrogates, the equivalent time period is referred to as "prediction time."

presented algorithm defines hyperparameters that may influence its performance. Their problem-specific optimal values are searched within the scope of this work, in particular in Experiments 1 & 2 that are outlined in Section 2.1.

To compare the quality of the produced surrogates, we define a number of metrics listed in Table 3. For regression performance analysis, we include a selection of absolute metrics to assess the models' 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 are invariant with respect to the selected domain and image space. For analysis of computational complexity, surrogates are assessed in terms of wall time (captured by the Python 3 `time` package). This is motivated by common practical use cases of our work, where models are trained and used as drop-in replacements for Paramak. All times reported (training, test, evaluation) are normalized by the corresponding dataset size, i.e. correspond to "time to process a single datapoint."

Even though some surrogates support acceleration by means of parallelisation, we used non-parallelized implementations. The only exception to this are ANNs, which require a considerable amount of processing power for training on conventional CPU architectures. Lastly, to prevent undesirable bias by training set selection, all reported metrics are obtained via 5-fold cross-validation. In this setting, a sample set is uniformly divided into 5 disjoint folds, each of which is used as a test set for models trained on the remaining 4. Having repeated the same experiment for each division, the overall value of individual metrics is reported in terms of their mean and standard deviation over all folds.

2.1. Decoupled Approach

Experiments related to the decoupled approach are organised in 4 parts, further described in this section. In summary, we aim to optimise the hyperparameters of each surrogate family separately and later use the best found results to compare surrogate families among themselves.

The objective of Experiment 1 is to simplify the regression task for surrogates prone to suboptimal performance in discontinuous spaces. To this end, training points are filtered to a single selected discrete feature assignment, and surrogates are trained only on the remaining continuous features. This is repeated several times to explore variances in behaviour, particularly in 4 distinct assignments that are obtained by setting blanket and first wall coolant materials to one of: $\{H_2O, He\}$. Experiment 2 conventionally measures surrogate performance on the full feature space without any parameter restrictions. In both experiments, hyperparameter tuning is facilitated by Bayesian optimisation [16], where we select the hyperparameter configuration that produces the model that maximises R^2. The process is terminated after 1000 iterations or 2 days, whichever condition is satisfied first. The results of Experiments 1 & 2 are depicted in Figures 3 & 4 respectively, and described in Section 3.1.1.

In Experiment 3, the 20 best-performing hyperparameter configurations per each model family are used to train surrogates on sets of various sizes to investigate their scaling properties. In particular, we consider sets of sizes $\{1, 2, 5, 10, 12, 15, 20\}$ in thousands of samples to better characterize the relationship of each family between sample count and the metrics of interest listed in Table 3. The results of this experiment are shown in Figure 5 and discussed in Section 3.1.2.

Finally, Experiment 4 aims to produce surrogates suitable for practical use by retraining selected well-scaling instances on large training sets in order to satisfy the goals of this work. The results of this process are displayed in Figure 6 and in Table 4, and summarized in Section 3.1.3.

2.2. Adaptive Approach

Adaptive sampling techniques appear frequently in the literature and have been specialised for surrogate modelling, where precision is implicitly limited by the quantity of training samples which are available from the expensive model. Garud's [17] "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 each sample points according to a shared proposal distribution from the prior point. These sample points will converge to a desired posterior distribution, so long as the acceptance probability meets certain statistical criteria (see [18] for a review).

Many researchers have embedded surrogate methods into MCMC strategies for parameter optimisation [19, 20], in particular the ASMO-PODE algorithm [21] which makes use of MCMC-based adaptive sampling. Our 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) [22] 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.

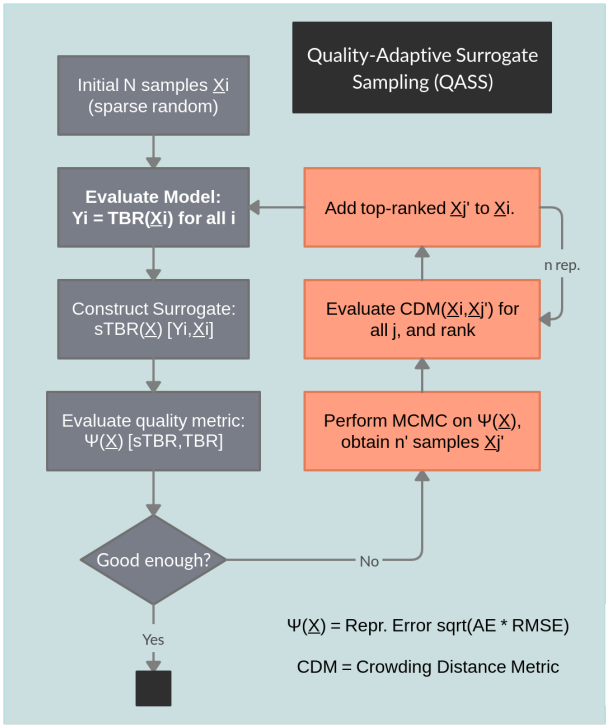

Figure 2. Schematic of QASS algorithm

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

3.1. Decoupled Approach

3.1.1. Hyperparameter Tuning The results displayed in Figure 3 indicate that in the first, simplified case GBTs clearly appear to be the most accurate as well as the fastest surrogate family in terms of mean prediction time. Following that, we note that ERTs, SVMs and ANNs also achieved satisfactory results with respect to both examined metrics. While the remainder of tested surrogate families does not exhibit prohibitive complexity, its regression performance falls below average.

Comparing these results with those of the second, unrestricted experiment (shown in Figure 4), we observe that many surrogate families consistently underperform. The least affected models appear to be GBTs, ANNs and ERTs, which are known to be capable of capturing relationships involving mixed feature types that were deliberately withheld in the first experiment. With only negligible differences, the first two of these families appear to be tied for the best performance as well as the shortest prediction time. We observe that ERTs and RBFs also demonstrated satisfactory results, clearly outperforming the remaining surrogates in terms of regression performance, and in some cases also in prediction time.

Following both hyperparameter tuning experiments, we conclude that while domain restrictions employed in the first case have proven effective in improving the regression performance of some methods, their performance fluctuates considerably depending on the selected slices. Furthermore, in all instances the best results are achieved by families of surrogates that do not benefit from this restriction (GBTs, ANNs, ERTs).

3.1.2. Scaling Benchmark The results shown in Figure 5 suggest that in terms of regression performance the most accurate families from the previous experiments consistently maintain their relative advantage over others, even as more training points are introduced. While such families achieve nearly comparable performance on the largest dataset, in the opposite case tree-based ensemble approaches clearly

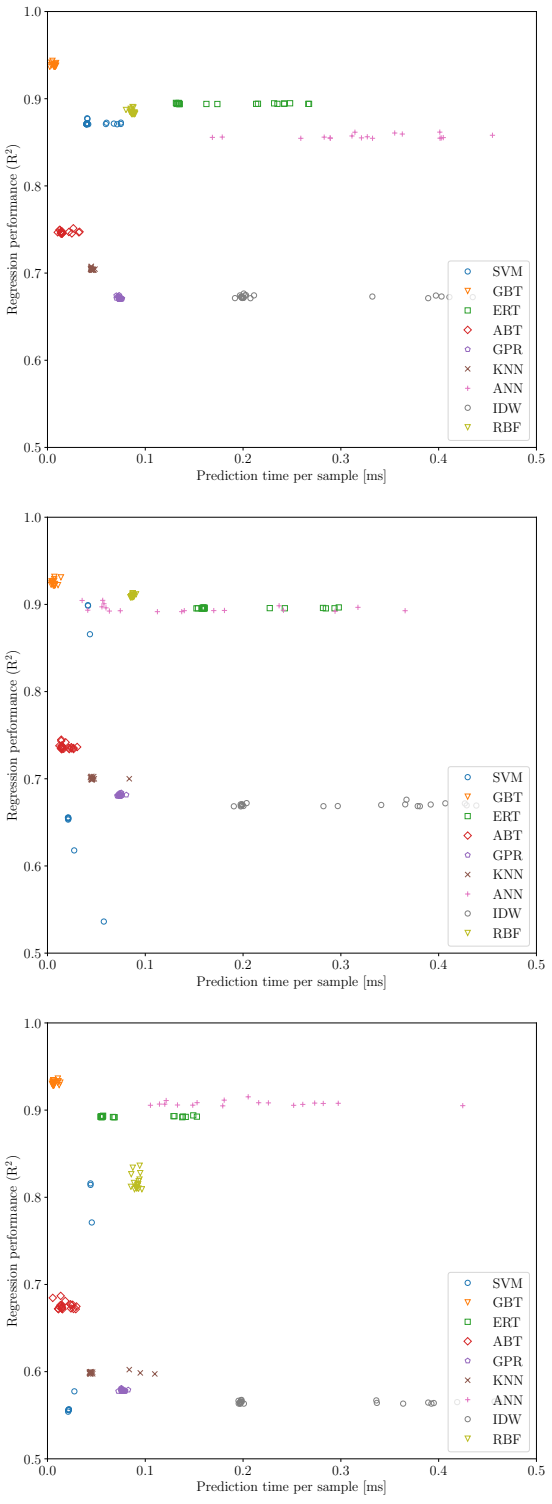

Figure 3. Results of Experiment 1, 20 best-performing surrogates per each considered family, plotted in terms of computational complexity (as $\bar{t}_{\text{pred.}}$) and regression performance (as R^2) with 3 selected discrete feature assignments out of 4.

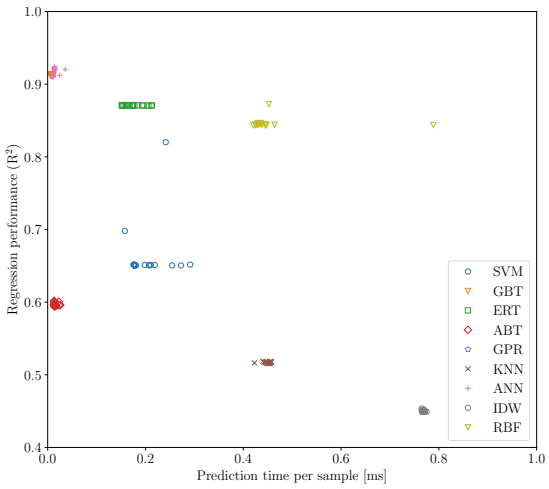

Figure 4. Results of Experiment 2, plotted analogously to Figure 3.

outperform ANNs. This can be observed particularly on sets of sizes up to 6000.

Consistent with our expectation, the shortest training times were achieved by instance-based learning methods (KNN, IDW) that are trained trivially at the expense of increased lookup complexity later during prediction. Furthermore, we observe that the majority of tree-based ensemble algorithms also perform and scale well, unlike RBFs and GPR which appear to behave superlinearly. We note that ANNs, which are the only family to utilise parallelisation during training, show an inverse scaling characteristic. We suspect that this effect may be caused by a constant multi-threading overhead that dominates the training process on relatively small sets.

Finally, all tested families with the exception of previously mentioned instance-based models offer desirable prediction times. Analogous to previous experiments, GBTs, ABTs and ANNs appear to be tied, as they not only exhibit comparable times but also similar scaling slopes. Following those, we note a clear hierarchy of ERTs, SVMs, GPR and RBFs, trailed by IDW and KNNs.

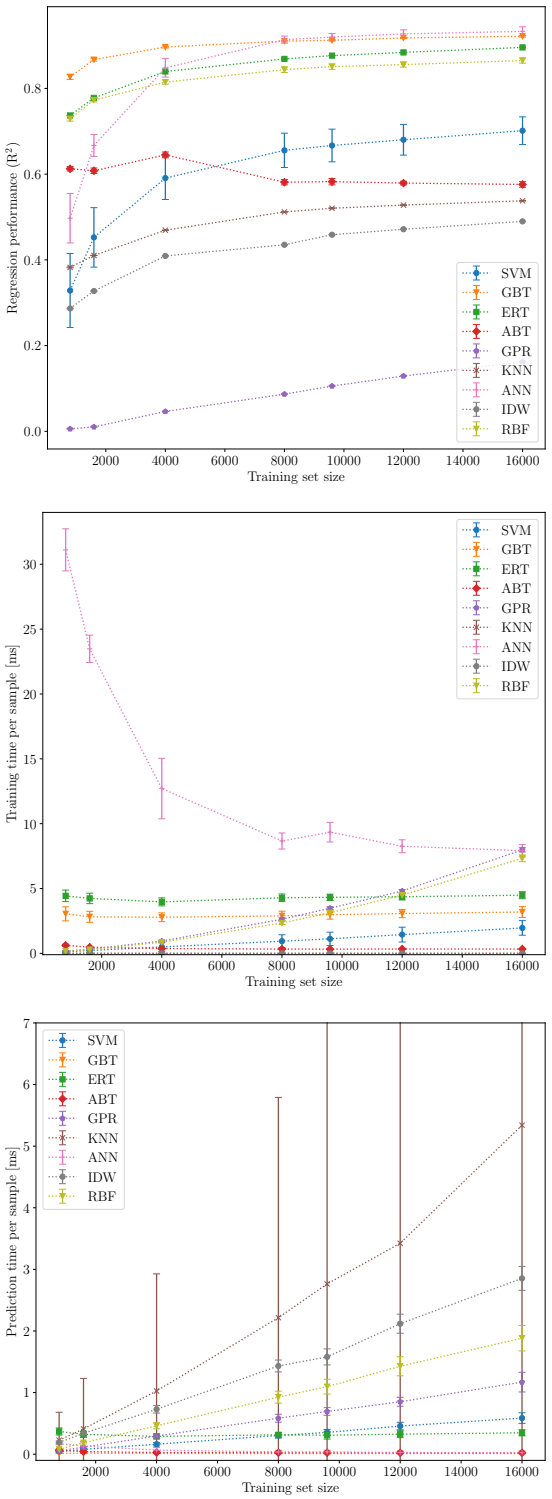

Figure 5. Results of the scaling benchmark displayed as a function of training set size. From top to bottom, regression performance (as R^2), mean training time and mean prediction time.

3.1.3. Model Comparison In Experiment 4, we aim to create models that yield: (a) the best regression performance regardless of other features, (b) acceptable performance with the shortest mean prediction time, or (c) acceptable performance with the smallest training set. To this end, we trained 8 surrogates that are presented in Figure 6 and Table 4. Having selected ANNs, GBTs, ERTs, RBFs and SVMs based on the results of Experiments 2 & 3, we utilised the best-performing hyperparameters. In pursuit of goal (a), the best approximator (Model 1, ANN) achieved $R^2 = 0.998$ and mean prediction time $\bar{t}_{\text{pred.}} = 1.124\,\mu\text{s}$. These correspond to a standard error $S = 0.013$ and a relative speedup $\omega = 6\,916\,416\times$ with respect to the MC TBR evaluation baseline. Satisfying goal (b), the fastest model (Model 2, ANN) achieved $R^2 = 0.985$, $\bar{t}_{\text{pred.}} = 0.898\,\mu\text{s}$, $S = 0.033$ and $\omega = 8\,659\,251\times$. While these surrogates were trained on the entire available set of 500 000 datapoints, to satisfy goal (c) we also trained a more simplified model (Model 4, GBT) that achieved $R^2 = 0.913$, $\bar{t}_{\text{pred.}} = 6.125\,\mu\text{s}$, $S = 0.072$ and $\omega = 1\,269\,777\times$ with a set of size only 10 000.

Overall we found that due to their superior performance, boosted tree-based approaches seem to be advantageous for fast surrogate modelling on relatively small training sets (up to the order of 10^4). Conversely, while neural networks perform poorly in such a setting, they dominate on larger training sets (at least of the order of 10^5) both in terms of regression performance and mean prediction time.

3.2. Adaptive Approach

In order to test our QASS prototype, several functional toy theories for TBR were developed

Table 4. Results of Experiment 4. Here, average values and standard deviations are reported over 5 cross-validation folds, $|\mathcal{T}|$ denotes cross-validation set size ($\times 10^3$) and ω is a relative speedup with respect to $\bar{t}_{\text{eval.}} = 7.777 \pm 2.810$ s measured in the MC TBR model over 500 000 samples. The best-performing method(s) under each are highlighted in bold.

		Regression performance				Computational complexity		
Model	$\lvert\mathcal{T}\rvert$	MAE [TBR]	S [TBR]	R^2 [rel.]	$R^2_{\text{adj.}}$ [rel.]	$\bar{t}_{\text{trn.}}$ [ms]	$\bar{t}_{\text{pred.}}$ [ms]	ω [rel.]
1 (ANN)	500	$\mathbf{0.009 \pm 0.000}$	$\mathbf{0.013 \pm 0.001}$	$\mathbf{0.998 \pm 0.000}$	$\mathbf{0.998 \pm 0.000}$	3.659 ± 0.035	$\mathbf{0.001 \pm 0.000}$	6 916 416×
2 (ANN)	500	0.025 ± 0.001	0.033 ± 0.001	0.985 ± 0.001	0.985 ± 0.001	2.989 ± 0.026	$\mathbf{0.001 \pm 0.000}$	**8 659 251**×
3 (GBT)	200	0.058 ± 0.001	0.059 ± 0.000	0.941 ± 0.001	0.941 ± 0.001	2.221 ± 0.010	0.007 ± 0.000	1 169 933×
4 (GBT)	10	0.071 ± 0.002	0.072 ± 0.003	0.913 ± 0.006	0.912 ± 0.006	$\mathbf{1.621 \pm 0.008}$	0.006 ± 0.000	1 269 777×
5 (ERT)	200	0.051 ± 0.000	0.056 ± 0.000	0.950 ± 0.001	0.950 ± 0.001	2.634 ± 0.010	0.214 ± 0.004	36 308×
6 (ERT)	40	0.068 ± 0.000	0.072 ± 0.000	0.917 ± 0.001	0.917 ± 0.001	2.368 ± 0.005	0.188 ± 0.008	41 370×
7 (RBF)	50	0.068 ± 0.001	0.077 ± 0.002	0.910 ± 0.003	0.910 ± 0.003	3.453 ± 0.019	1.512 ± 0.016	5143×
8 (SVM)	200	0.062 ± 0.000	0.094 ± 0.002	0.891 ± 0.003	0.891 ± 0.003	33.347 ± 0.382	2.415 ± 0.011	3220×

as alternatives to the expensive MC model. By far the most robust of these was the following sinusoidal theory over continuous parameters C, with adjustable wavenumber parameter n:

$$\text{TBR} = |C|^{-1} \sum_{i \in C} [1 + \sin(2\pi n(x_i - 1/2))] \quad (3)$$

ANNs trained on this model demonstrated similar performance to those on the expensive MC model. QASS performance was verified by training an ANN on the sinusoidal theory for varied quantities of initial, incremental, and MCMC candidate samples.

An increase in initial samples with increment held constant had a strong impact on final surrogate precision, an early confirmation of basic functionality. An increase in MCMC candidate samples was seen to have a positive but very weak effect on final surrogate precision, suggesting that the runtime of MCMC on each iteration could be limited for increased efficiency. We also found that an optimum increment exists and is monotonic with initial sample quantity, above or below which models showed slower improvement on both the training and evaluation sets, and a larger minimum error on the evaluation set. This performance distinction will be far more significant for an expensive model such as Paramak, where the number of sample evaluations is the primary computational bottleneck.

A plateau effect in surrogate error on the evaluation set was universal to all configurations, and initially suspected to be a residual effect of retraining the same ANN instance without adjustment to data normalisation. A "Goldilocks scheme" for checking normalisation drift was implemented and tested, but did not affect QASS performance. Schemes in which the ANN is periodically retrained were also discarded, as the retention of network weights from one iteration to the next was demonstrated to greatly benefit QASS efficiency. Further insight came from direct comparison between QASS and a baseline scheme with uniformly random incremental samples, shown in Figure 7.

Such tests revealed that while QASS has unmatched performance on its own adaptively-sampled training set, it is outperformed by the baseline scheme on uniformly-random evaluation sets. We suspected that while QASS excels in learning the most strongly peaked regions of the TBR theory, this comes at the expense of precision in broader, smoother regions where uniformly random sampling suffices. Therefore a mixed scheme was implemented, with half MCMC samples and half uniformly random samples incremented on each iteration,

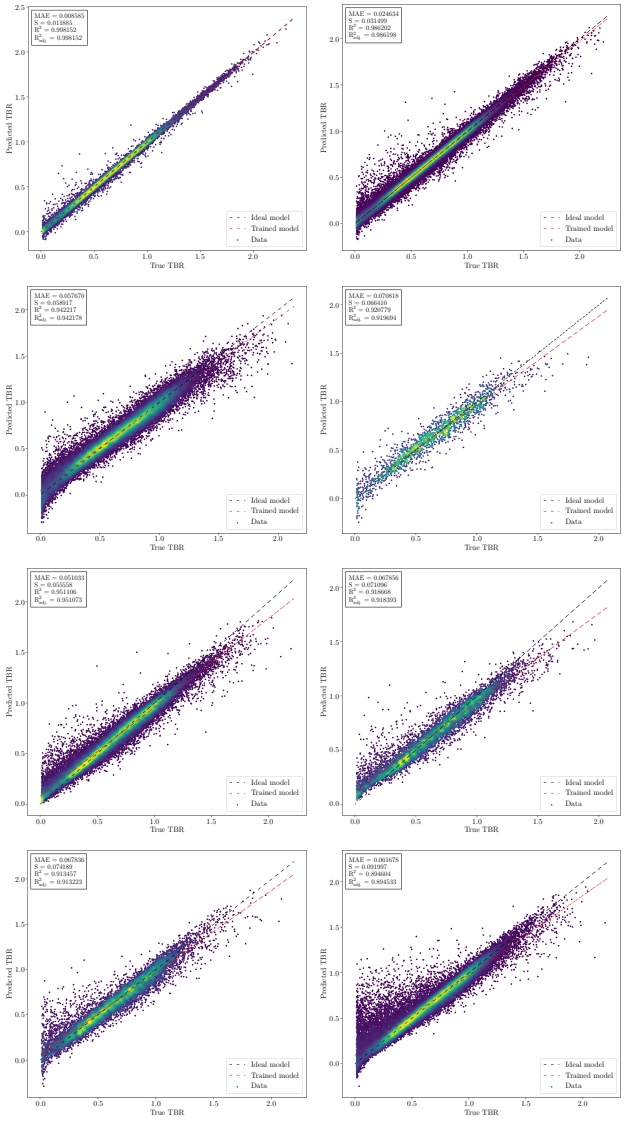

Figure 6. Regression performance of Models 1-8 (from left to right, top to bottom) trained in Experiment 4, viewed as true vs. predicted TBR on a test set of a selected cross-validation fold. Points are coloured by density.

which is also shown in Figure 7. An increase in initial sample size was observed to also resolve precision in these smooth regions of the toy theory, as the initial samples were obtained from a uniform random distribution. As shown in Figure 8, with 100 000 initial samples it was possible to obtain a ∼40% decrease in error as compared to the baseline scheme, from 0.0025 to 0.0015 mean averaged error. Comparing at

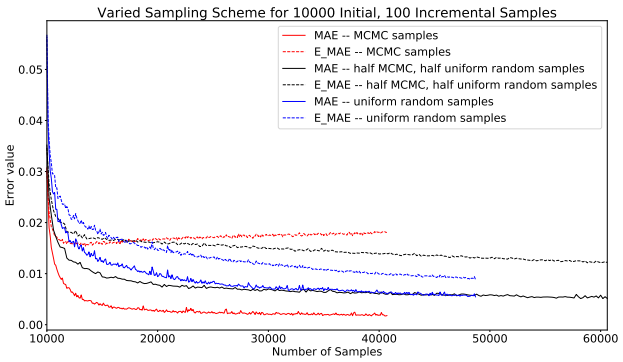

Figure 7. Absolute training error for QASS, baseline scheme, and mixed scheme.

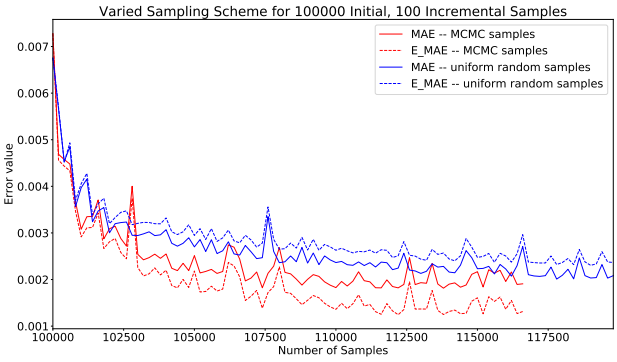

Figure 8. Absolute training error for QASS and baseline scheme, with 100 000 initial samples.

the point of termination for QASS, this corresponds to a ∼6% decrease in the number of total samples needed to train a surrogate with the same error.

4. Conclusion

We employed a broad spectrum of data analysis and machine learning techniques to develop fast and high-quality surrogates for the Paramak TBR model. After reviewing 9 surrogate model families, examining their behaviour on a constrained and unrestricted feature space, and studying their scaling properties, we retrained the best-performing instances to produce properties desirable for

practical use. The fastest surrogate, an artificial neural network trained on 500 000 datapoints, attained an $R^2 = 0.985$ with mean prediction time of 0.898 µs, representing a relative speedup of $8 \cdot 10^6$ with respect to Paramak. Furthermore, we demonstrated the possibility of achieving comparable results using only a training set of size 10 000.

We further developed a novel adaptive sampling algorithm, QASS, capable of interfacing with any of the presented surrogate models. Preliminary testing on a toy theory, qualitatively comparable to Paramak, demonstrated the effectiveness of QASS and behavioural trends consistent with the design of the algorithm. With 100 000 initial samples and 100 incremental samples per iteration, QASS achieved a ∼40% decrease in surrogate error compared to a baseline random sampling scheme. Further optimisation over the hyperparameter space has strong potential to increase this performance, in particular by decreasing the required quantity of initial samples. This will allow for future deployment of QASS on the Paramak TBR model in coalition with any of the most effective identified surrogates.

Relevant source code, model instances and datasets are freely available online as well as a more detailed technical report [23, 24].

5. Acknowledgements

PM & GVG were supported the STFC UCL Centre for Doctoral Training in Data Intensive Science (grant no. ST/P006736/1). This work has been carried out within the framework of the EUROfusion consortium and has received funding from the Euratom research and training programme 2014-2018 and 2019-2020 under grant agreement No 633053. The views and opinions expressed herein do not necessarily reflect those of the European Commission.

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] UKAEA 2020 Paramak URL `https://github.com/ukaea/paramak`

[4] OpenMC 2019 Muir energy spectrum URL `https://docs.openmc.org/en/stable/pythonapi/generated/openmc.stats.Muir.html`

[5] 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

[6] 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

[7] Friedman J H 2001 *Annals of statistics* 1189–1232

[8] Friedman J 1999 Stochastic gradient boosting technical report

[9] Hastie T, Tibshirani R and Friedman J 2009 *The elements of statistical learning: data mining, inference, and prediction* (Springer Science & Business Media)

[10] Geurts P, Ernst D and Wehenkel L 2006 *Machine learning* **63** 3–42

[11] Drucker H 1997 Improving regressors using boosting techniques *ICML* vol 97 pp 107–115

[12] Williams C K and Rasmussen C E 2006 *Gaussian processes for machine learning* vol 2 (MIT press Cambridge, MA)

[13] Chollet F *et al.* 2015 Keras `https://keras.io`

[14] Shepard D 1968 A two-dimensional interpolation function for irregularly-spaced data *Proceedings of the 1968 23rd ACM national conference* pp 517–524

[15] 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

[16] Močkus J 1975 On bayesian methods for seeking the extremum *Optimization techniques IFIP technical conference* (Springer) pp 400–404

[17] Garud S, Karimi I and Kraft M 2016 *Computers & Chemical Engineering* **96**

[18] Zhou J, Su X and Cui G 2018 *Journal of Contaminant Hydrology* **216** 50–57 ISSN 0169-7722 URL https://www.sciencedirect.com/science/article/pii/S0169772218300317?via%3Dihub

[19] Zhang J, Zheng Q, Chen D, Wu L and Zeng L 2020 *Water Resources Research* **56** e2019WR025721 ISSN 0043-1397 URL https://doi.org/10.1029/2019WR025721

[20] Gong W and Duan Q 2017 *Environmental Modelling & Software* **95** 61–75 ISSN 1364-8152 URL https://www.sciencedirect.com/science/article/pii/S1364815216310830

[21] Ginting V, Pereira F, Presho M and Wo S 2011 *Computational Geosciences* **15** 691 ISSN 1573-1499 URL https://doi.org/10.1007/s10596-011-9236-4

[22] Solonen A, Ollinaho P, Laine M, Haario H, Tamminen J and Jarvinen H 2012 *Bayesian Anal.* **7** 715–736 ISSN 1936-0975 URL https://projecteuclid.org:443/euclid.ba/1346158781

[23] Van Goffrier G and Mánek P 2020 UCL TBR Group Project URL https://github.com/ucl-tbr-group-project

[24] Van Goffrier G and Mánek P 2020 Surrogate Modelling of the Tritium Breeding Ratio CDT DIS Note Department of Physics and Astronomy, University College London