# Fast Regression of the Tritium Breeding Ratio in Tokamak Fusion Reactors

**P Mánek[1,2], G Van Goffrier[1], V Gopakumar[3], N Nikolau[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 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***— machine learning, surrogate model, nuclear fusion, 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 [**?**]. 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 [**?**]. 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.

We sought to develop an empirical surrogate model for the tritium breeding ratio (TBR) in a Tokamak nuclear fusion reactor. Our expensive model was a Monte Carlo (MC) neutronics simulation, *Paramak*‡, which returns a prediction of the TBR for a given configuration of a spherical Tokamak.

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 for further work.

‡ Provided by collaborator Jonathan Shimwell, at UKAEA.

### *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 input parameters in the-expensive MC model fall into two classes. Continuous parameters, including material thicknesses and packing ratios, describe the geometry of a given reactor configuration. Discrete categorical parameters then specify material selections, including coolants, armours, and neutron multipliers. We set out to produce a fast TBR function which takes these same input parameters and approximates the MC TBR model with the greatest achievable regression performance.

**Table 1.** 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 [**?**] | SVM | SciKit [**?**] | 3 |
| Gradient boosted trees [**?**, **?**, **?**] | GBT | SciKit | 11 |
| Extremely randomised trees [**?**] | ERT | SciKit | 7 |
| AdaBoosted decision trees [**?**] | ABT | SciKit | 3 |
| Gaussian process regression [**?**] | GPR | SciKit | 2 |
| $k$ nearest neighbours | KNN | SciKit | 3 |
| Artificial neural networks | ANN | Keras [**?**] | 2 |
| Inverse distance weighing [**?**] | IDW | SMT [**?**] | 1 |
| Radial basis functions | RBF | SMT | 3 |

## 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 nine 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.

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.

### *2.1. Decoupled Approach*

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

The objective of the first experiment is to simplify the regression task for surrogates prone to suboptimal performance in discon-

**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 $i$th 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 error of regression | $S$ | $\mathrm{StdDev}_{i=1}^{N}\left\{\lvert y^{(i)} - \hat{y}^{(i)}\rvert\right\}$ |
| Coefficient of determination | $R^2$ | $1 - \sum_{i=1}^{N}\left(y^{(i)} - \hat{y}^{(i)}\right)^2\left[\sum_{i=1}^{N}\left(y^{(i)} - \overline{y}\right)^2\right]^{-1}$ |
| Adjusted $R^2$ | $R^2_{\text{adj.}}$ | $1 - (1 - R^2)(N-1)(N-P-1)^{-1}$ |
| **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 | $\omega$ | (wall evaluation time of $f(x)$)$(N\overline{t}_{\text{pred.}})^{-1}$ |

tinuous 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 for 4 distinct assignments to explore variances in behaviour. The second experiment conventionally measures surrogate performance on the full, unrestricted feature space. In both cases, hyperparameter tuning is facilitated by Bayesian optimisation [**?**]. We set its objective to maximise $R^2$ and terminate the process after 1000 iterations or 2 days, whichever condition is satisfied first.

In the third experiment, the 20 best-performing hyperparameter configurations per each family are used to train surrogates on sets of various sizes 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 in order to satisfy the goals of this work.

### *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 [**?**] "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

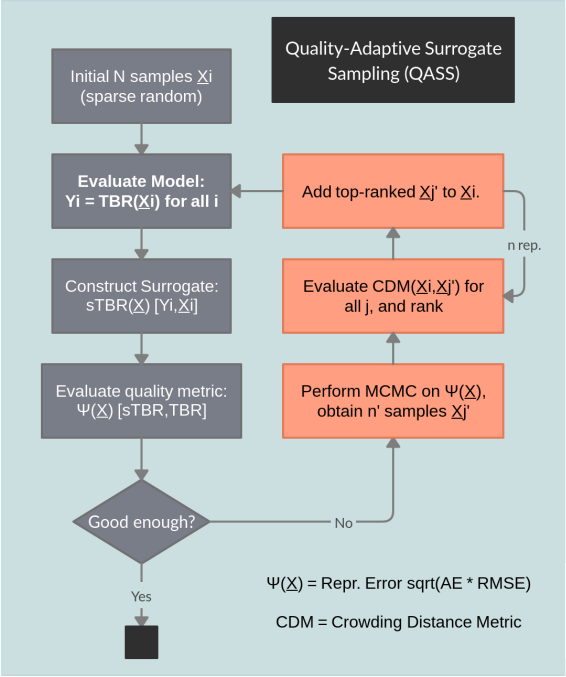


**Figure 1.** Schematic of QASS algorithm

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 [**?**] for a review).

Many researchers have embedded surrogate methods into MCMC strategies for parameter optimisation [**?**, **?**], in particular the ASMO-PODE algorithm [**?**] 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 1) to iteratively increment the training/test set with sample points which maximise surrogate error and minimise a crowding distance metric (CDM) [**?**] 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

### *3.1. Decoupled Approach*

*3.1.1. Hyperparameter Tuning* The results displayed in figure 2 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 problems in complexity, its regression performance falls below average.

Comparing these results with those of the second, unrestricted experiment (shown in figure 3), we observe that many surrogate families consistently underperformed. 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, relatively 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, this result has fluctuated considerably depending on the selected slices. Furthermore, in all instances the best results were achieved by families of surrogates that were nearly unaffected by this modification.

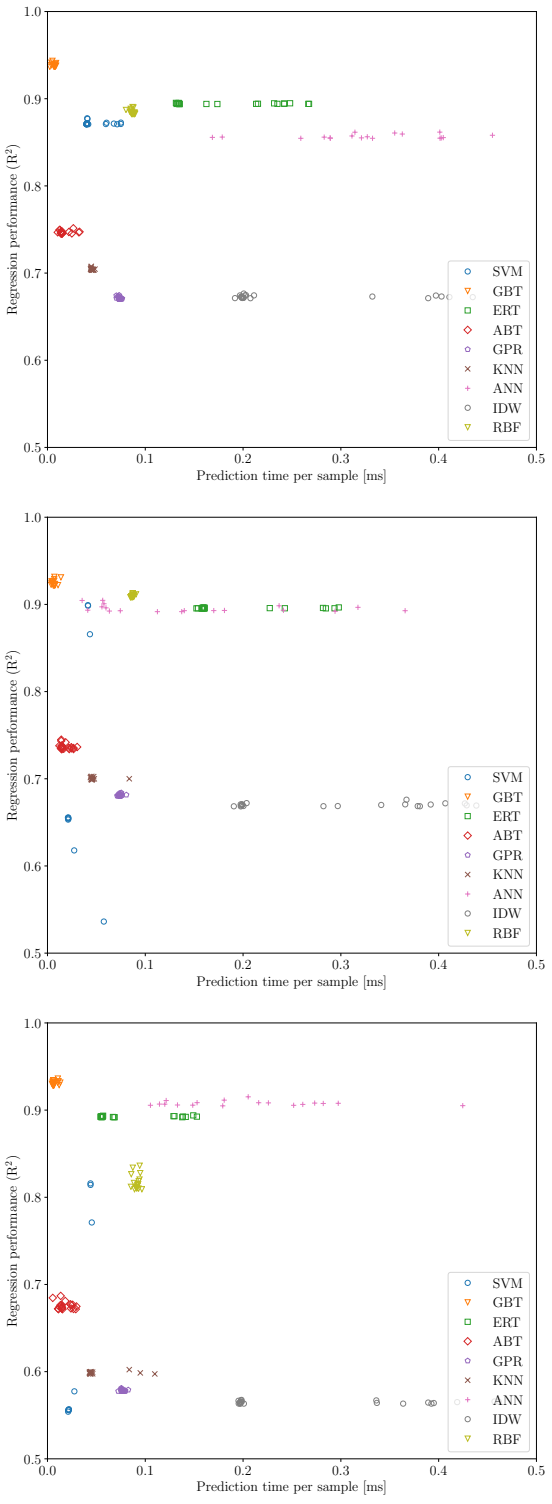


**Figure 2.** 20 best-performing surrogates per each considered family, plotted in terms of complexity (as $\bar{t}_{\text{pred.}}$) and regression performance (as $R^2$) with 3 selected discrete feature assignments.

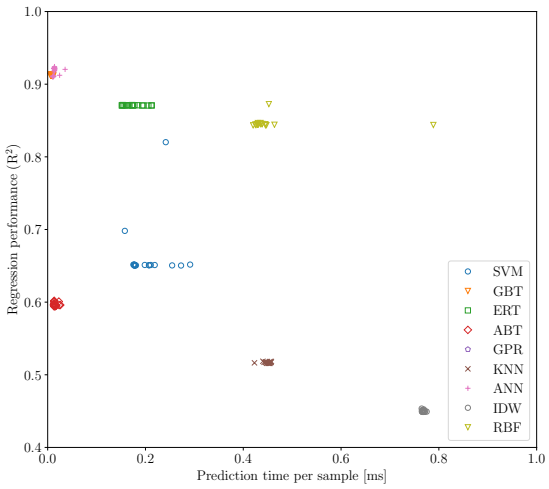


**Figure 3.** Results of experiment 2, plotted analogously to figure 2.

*3.1.2. Scaling Benchmark* The results shown in figure 4 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 approaches clearly 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 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.

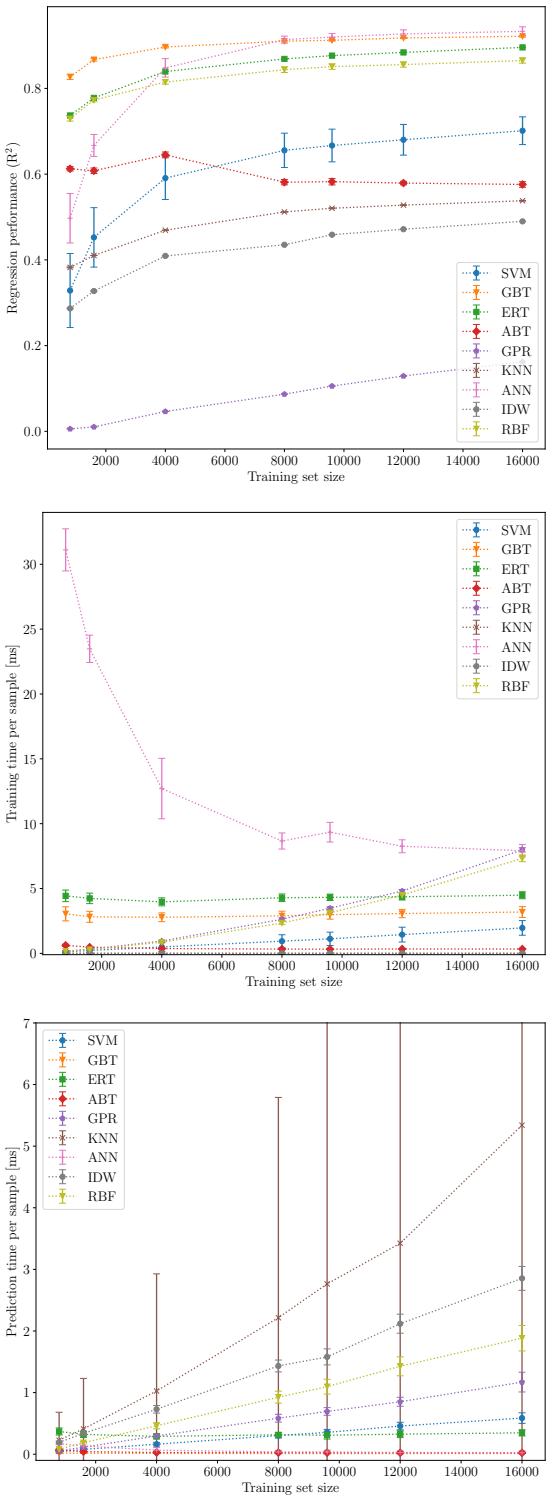


**Figure 4.** Results of the scaling benchmark displayed as a function of training set size. From top to bottom, regression performance (as $R^2$), mean training time and mean prediction time.

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 that, we note a clear hierarchy of ERTs, SVMs, GPR and RBFs, trailed by IDW and KNNs.

*3.1.3. Model Comparison* In the fourth experiment, 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 5 and table 3. 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 (no. 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 (no. 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 (no. 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

**Table 3.** Results of experiment 4. Here, figures are reported over 5 cross-validation folds, $|\mathcal{T}|$ denotes cross-validation set size ($\times 10^3$) and $\omega$ is a relative speedup with respect to $\bar{t}_{\text{eval.}} = 7.777\,\text{s}$ measured in the MC TBR model during run 1 (see table **??**). The best-performing metrics are highlighted in bold.

| | | Regression performance | | | | 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] | $\omega$ [rel.] |
| 1 (ANN) | 500 | $\mathbf{0.009 \pm 0.000}$ | $\mathbf{0.013 \pm 0.001}$ | $\mathbf{0.998 \pm 0.000}$ | $\mathbf{0.998 \pm 0.000}$ | $3.659 \pm 0.035$ | $0.001 \pm 0.000$ | $6\,916\,416\times$ |
| 2 (ANN) | 500 | $0.025 \pm 0.001$ | $0.033 \pm 0.001$ | $0.985 \pm 0.001$ | $0.985 \pm 0.001$ | $2.989 \pm 0.026$ | $\mathbf{0.001 \pm 0.000}$ | $8\,659\,251\times$ |
| 3 (GBT) | 200 | $0.058 \pm 0.001$ | $0.059 \pm 0.000$ | $0.941 \pm 0.001$ | $0.941 \pm 0.001$ | $2.221 \pm 0.010$ | $0.007 \pm 0.000$ | $1\,169\,933\times$ |
| 4 (GBT) | 10 | $0.071 \pm 0.002$ | $0.072 \pm 0.003$ | $0.913 \pm 0.006$ | $0.912 \pm 0.006$ | $\mathbf{1.621 \pm 0.008}$ | $0.006 \pm 0.000$ | $1\,269\,777\times$ |
| 5 (ERT) | 200 | $0.051 \pm 0.000$ | $0.056 \pm 0.000$ | $0.950 \pm 0.001$ | $0.950 \pm 0.001$ | $2.634 \pm 0.010$ | $0.214 \pm 0.004$ | $36\,308\times$ |
| 6 (ERT) | 40 | $0.068 \pm 0.000$ | $0.072 \pm 0.000$ | $0.917 \pm 0.001$ | $0.917 \pm 0.001$ | $2.368 \pm 0.005$ | $0.188 \pm 0.008$ | $41\,370\times$ |
| 7 (RBF) | 50 | $0.068 \pm 0.001$ | $0.077 \pm 0.002$ | $0.910 \pm 0.003$ | $0.910 \pm 0.003$ | $3.453 \pm 0.019$ | $1.512 \pm 0.016$ | $5143\times$ |
| 8 (SVM) | 200 | $0.062 \pm 0.000$ | $0.094 \pm 0.002$ | $0.891 \pm 0.003$ | $0.891 \pm 0.003$ | $33.347 \pm 0.382$ | $2.415 \pm 0.011$ | $3220\times$ |

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 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 a 1h3f(256) 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 6.

Such tests revealed that while QASS has unmatched performance on its own adaptively-sampled training set, it is outperformed by the

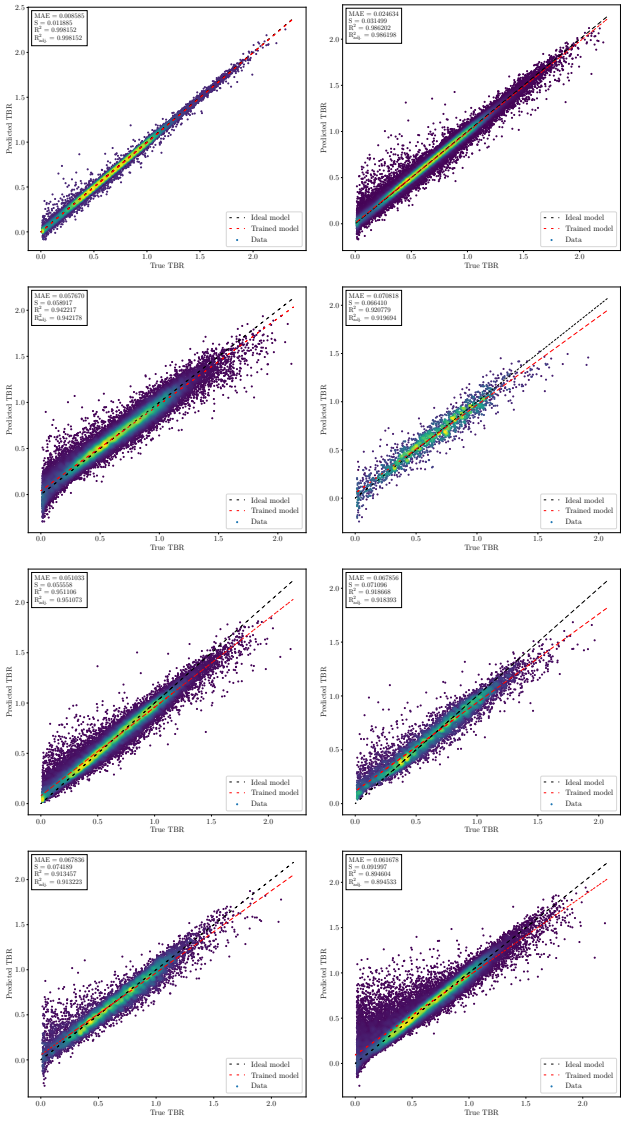


**Figure 5.** Regression performance of models 1-8 (from left to right, top to bottom) trained in the model comparison experiment, viewed as true vs. predicted TBR on a test set of a selected cross-validation fold. Points are coloured by density.

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,

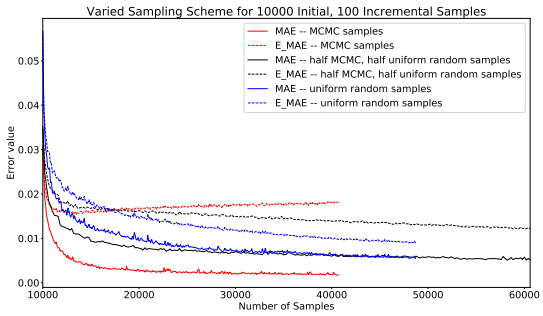


**Figure 6.** Absolute training error for QASS, baseline scheme, and mixed scheme.

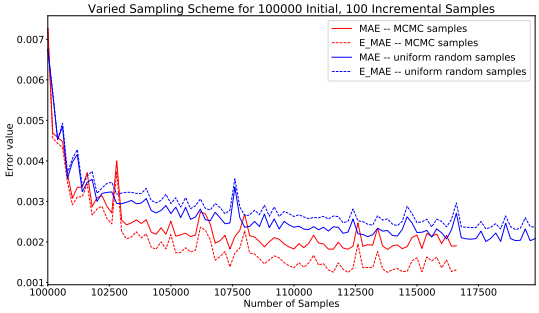


**Figure 7.** Absolute training error for QASS and baseline scheme, with 100k initial samples.

which is also shown in figure 6. 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 7, 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 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 nine surrogate model families, examining their behaviour on 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, featured $R^2 = 0.985$ with mean prediction time of 0.898 µs, representing a relative speedup of $8 \cdot 10^6$ with respect to the MC model. 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 studied surrogates. Preliminary testing on a toy theory, qualitatively comparable to the MC TBR model, 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 as compared with 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.

## 5. Acknowledgements

PM was supported the STFC UCL Centre for Doctoral Training in Data Intensive Science (grant no. ST/P006736/1).

## 6. References