



UCL CDT DIS Note

27th April 2020



Draft version 0

Surrogate Modelling of the Tritium Breeding Ratio

4 Petr Mánek^a and Graham Van Goffrier^a

5 ^aUniversity College London

6 The tritium breeding ratio (TBR) is an essential quantity for the design of modern
7 and next-generation Tokamak nuclear fusion reactors. Representing the ratio
8 between tritium fuel generated in breeding blankets at the boundary of the reactor,
9 and fuel consumed during reactor runtime, the TBR depends in a complex manner
10 on reactor geometry and material properties. We explored the training of surrogate
11 models to produce a cheap but high-quality approximations for a Monte Carlo TBR
12 model in use at the UK Atomic Energy Authority. We investigated possibilities for
13 dimensional reduction on the parameter space of this model, and reviewed N classes
14 of surrogate models for potential applicability. Here we present the performance and
15 scaling properties of these surrogate models, the best of which demonstrated an accu-
16 racy of X and a mean prediction time of X , representing a relative speedup X with
17 respect to the MC model. We further present a novel adaptive sampling algorithm,
18 QASS, capable of interfacing with any of the individual studied models. Our prelim-
19 inary testing on a toy TBR theory has demonstrated the efficacy of this algorithm
20 for speeding up the surrogate modelling process.

22 **Contents**

23	1 Introduction	3
24	1.1 Problem Description	3
25	2 Data Exploration	5
26	2.1 Expensive Model Description	5
27	2.2 Dataset Generation	6
28	2.3 Dimensionality Reduction	7
29	2.3.1 Principal Component Analysis	7
30	2.3.2 Variogram Computations	8
31	2.3.3 Autoencoders	8
32	3 Methodology	9
33	3.1 Metrics	10
34	3.2 Evaluation of Supervised Learning Surrogates	10
35	3.2.1 Experiments	11
36	3.3 Adaptive Sampling	13
37	4 Results	14
38	4.1 Evaluation of Supervised Learning Surrogates	14
39	4.1.1 Hyperparameter Tuning	14
40	4.1.2 Scaling Benchmark	15
41	4.1.3 Competitive Surrogate Training	15
42	4.2 Results of Adaptive Sampling	16
43	5 Conclusion	18
44	Appendices	20
45	A Detailed Results of Supervised Models	20
46	B Overview of Online Resources	21

47 1 Introduction

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

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

64 In this internship project, in coordination with the UK Atomic Energy Authority (UKAEA)
65 and Culham Centre for Fusion Energy (CCFE), we sought to develop a surrogate model for the
66 tritium breeding ratio (TBR) in a Tokamak nuclear fusion reactor. Our expensive model was a
67 MC-based neutronics simulation [3], itself a spherical approximation of the Joint European Torus
68 (JET) at CCFE, which returns a prediction of the TBR for a given reactor configuration. We
69 took an empirical approach to the construction of this surrogate, and no results described here
70 are explicitly dependent on prior physics knowledge.

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

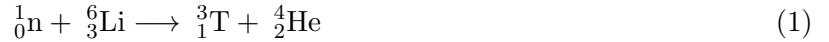
79 1.1 Problem Description

80 Nuclear fusion technology relies on the production and containment of an extremely hot and
81 dense plasma. In this environment, by design similar to that of a star, hydrogen atoms attain
82 energies sufficient to overcome their usual electrostatic repulsion and fuse to form helium [4].
83 Early prototype reactors made use of the deuterium (^2H , or D) isotope of hydrogen in order
84 to achieve fusion under more accessible conditions, but lead to limited success. The current
85 frontier generation of fusion reactors, such as JET and the under-construction International
86 Thermonuclear Experimental Reactor (ITER), make use of tritium (^3H , or T) fuel for further
87 efficiency gain. Experimentation at JET dating back to 1997 [5] has made significant headway

88 in validating deuterium-tritium (D-T) operations and constraining the technology which will be
 89 employed in ITER in a scaled up form.

90 However, tritium is much less readily available as a fuel source than deuterium. While at least one
 91 deuterium atom occurs for every 5000 molecules of naturally-sourced water, and may be easily
 92 distilled, tritium is extremely rare in nature. It may be produced indirectly through irradiation
 93 of heavy water (D_2O) during nuclear fission, but only at very low rates which could never sustain
 94 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



95 where T represents tritium and 7Li , 6Li are the
 96 more and less frequently occurring isotopes of lith-
 97 ium, respectively. 6Li has the greatest tritium
 98 breeding cross-section of all tested isotopes [4], but
 99 due to magnetohydrodynamic instability of liquid
 100 lithium in the reactor environment, a variety of
 101 solid lithium compounds are preferred.

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

112 The input parameters of the computationally-
 113 expensive TBR model therefore fall into two
 114 classes. Continuous parameters, including material
 115 thicknesses and packing ratios, describe the geo-
 116 metry of a given reactor configuration. Discrete
 117 categorical parameters further specify all relevant
 118 material sections, including coolants, armours, and
 119 neutron multipliers. One notable exception is the
 120 enrichment ratio, a continuous parameter denoting
 121 the presence of 6Li . Our challenge, put simply, was
 122 to produce a TBR function which takes these same
 123 input parameters and approximates the MC TBR
 124 model with the greatest achievable accuracy.

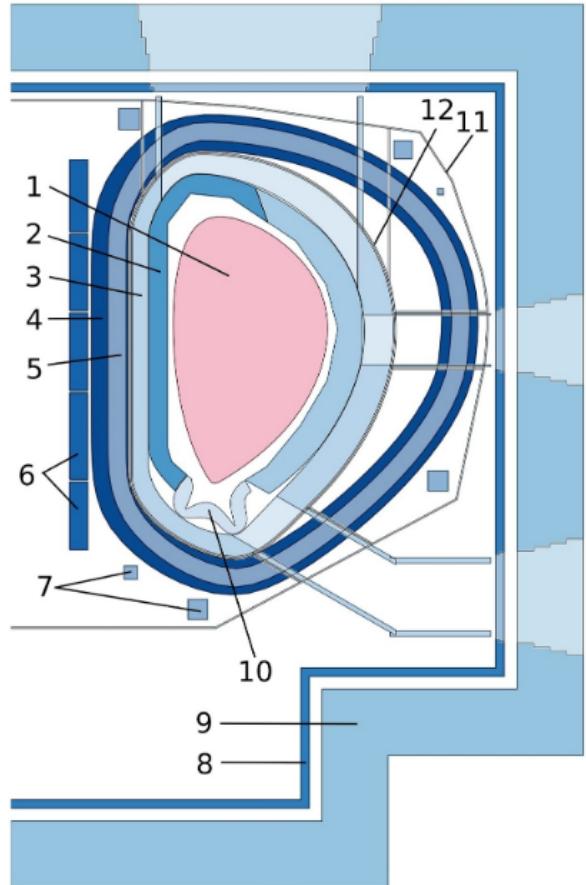


Figure 1: Typical single-null reactor configuration as specified by BLUEPRINT [6]: 1 — plasma, 2 — breeding blankets

125 2 Data Exploration

126 The initial step of our work is the study of the existing MC TBR model and its behaviour.
 127 Following the examination of its features (simulation parameters), we present efficient means of
 128 evaluating this model on large sets of points in high-performance computing (HPC) environment,
 129 preprocessing techniques designed to adapt collected datasets for surrogate modelling, and our
 130 attempts at feature space reduction to achieve the lowest possible number of dimensions.

131 2.1 Expensive Model Description

132 The MC TBR model that we understand to be the expensive function for our surrogate modelling
 133 is fundamentally a Monte Carlo simulation based on the OpenMC framework [7]. At input the
 134 software expects a set of 19 parameters, discrete and continuous, that are fully listed in Table 1.
 135 Following a brief period of time (usually on the order of tens of seconds), during which a fixed
 136 number of events is generated, the simulation outputs the mean and the standard deviation of
 137 the TBR aggregated over the simulated run. The former of these two we accept to be the output
 138 TBR value that is subject to approximation.

	Parameter Name	Acronym	Type	Domain	Description
Blanket	Breeder fraction [†]	BBF	Continuous	[0, 1]	TODO
	Breeder ⁶ Li enrichment fraction	BBLEF	Continuous	[0, 1]	
	Breeder material	BBM	Discrete	{Li ₂ TiO ₃ , Li ₄ SiO ₄ }	
	Breeder packing fraction	BBPF	Continuous	[0, 1]	
	Coolant fraction [†]	BCF	Continuous	[0, 1]	
	Coolant material	BCM	Discrete	{D ₂ O, H ₂ O, He}	
	Multiplier fraction [†]	BMF	Continuous	[0, 1]	
	Multiplier material	BMM	Discrete	{Be, Be ₁₂ Ti}	
	Multiplier packing fraction	BMPF	Continuous	[0, 1]	
	Structural fraction [†]	BSF	Continuous	[0, 1]	
First wall	Structural material	BSM	Discrete	{SiC, eurofer}	
	Thickness	BT	Continuous	[0, 500]	
	Armour fraction [‡]	FAF	Continuous	[0, 1]	
	Armour material	FAM	Discrete	{tungsten}	
	Coolant fraction [‡]	FCF	Continuous	[0, 1]	
	Coolant material	FCM	Discrete	{D ₂ O, H ₂ O, He}	
	Structural fraction [‡]	FSF	Continuous	[0, 1]	
	Structural material	FSM	Discrete	{SiC, eurofer}	
	Thickness	FT	Continuous	[0, 20]	

Table 1: Input parameters supplied to the MC TBR simulation in alphabetical order. Fractions marked with superscripts^{†‡} are independently required to sum to one.

139 In our work, we often reference TBR points or samples. These are simply vectors in the feature
 140 space generated by Cartesian product of domains of all features—parameters from Table 1.
 141 Since most surrogate models that we employ assume overall continuous inputs, we take steps to
 142 unify our feature interface in order to attain this property. In particular, we eliminate discrete
 143 features by embedding each such feature into a finite multitude of continuous features using
 144 standard one-hot encoding. This option is available to us since discrete domains that generate

our feature space are finite in cardinality and relatively small in size. And while it renders all features continuous, the modification comes at the expense of increasing the dimensionality of the feature space to 27 features in total. This is further discussed in Section 2.3.

2.2 Dataset Generation

While we deliberately make no assumptions about the internal properties of the MC TBR simulation, effectively treating it as a black box model, our means of studying its behaviour are limited to inspection of its outputs at various points in the feature space, and inference thereof. We thus require sufficiently large and representative quantities of samples to ensure statistical significance of our findings.

With grid search in high-dimensional domain clearly intractable, we selected uniform pseudo-random sampling to generate large amounts of feature configurations that we consider to be independent and unbiased. For evaluation of the expensive MC TBR model, we utilise parallelisation offered by the HPC infrastructure available at UCL computing facilities. To this end, we designed and implemented the Approximate TBR Evaluator—a Python software package capable of sequential evaluation of the multi-threaded OpenMC simulation on batches of previously generated points in the feature space. We deployed ATE at the UCL Hypatia cluster RCIF partition, which consists of 4 homogeneous nodes, each containing 40 cores.¹ We completed three data generation runs, which are summarised in Table 2.

#	Samples	Batch division	t_{run}	$\bar{t}_{\text{eval.}} \text{ [s]}$	Description
0	100 000	100×1000	2 days, 23 h	7.88 ± 2.75	Testing run using older MC TBR version.
1	500 000	500×1000	13 days, 20 h	7.78 ± 2.81	Fully uniform sampling in the entire domain.
2	400 000	400×1000	10 days	7.94 ± 2.60	Mixed sampling, discrete features fixed.

Table 2: Parameters of sampling runs. Here, t_{run} denotes the total run time (including waiting in the processing queue), and $\bar{t}_{\text{eval.}}$ is the mean evaluation time of MC TBR (per single sampled point).

Skipping run zero, which was performed using older, fundamentally different version of the MC TBR software, and was thus treated as a technical proof-of-concept, we generated the total of 900 000 samples in two runs. While the first run featured fully uniform sampling of the unrestricted feature space, the second run used more elaborate strategy. Interested in further study of relationships between discrete and continuous features, we selected four assignments of discrete features (listed in Table 3) and fixed them for all points, effectively slicing the feature space into four corresponding subspaces. In order to achieve comparability between these *slices*, the remaining unassigned features were uniformly sampled only in the first slice, and reused in the other three.

Since some surrogate modelling methods applied in this work are not scale-invariant or perform suboptimally with arbitrarily scaled problems, all run results were standardised prior to further use. In this commonly used statistical procedure, features and regression outputs are independently scaled and offset to attain zero mean and unit variance.

¹ Hypatia RCIF nodes use Intel® Xeon® Gold 6148 CPU with frequency 2.40 GHz and 376 GB RAM.

Batches	Discrete feature assignment						
	BBM	BCM	BMM	BSM	FAM	FCM	FSM
0-99	Li ₄ SiO ₄	H ₂ O	Be ₁₂ Ti	eurofer	tungsten	H ₂ O	eurofer
100-199	Li ₄ SiO ₄	He	Be ₁₂ Ti	eurofer	tungsten	H ₂ O	eurofer
200-299	Li ₄ SiO ₄	H ₂ O	Be ₁₂ Ti	eurofer	tungsten	He	eurofer
300-399	Li ₄ SiO ₄	He	Be ₁₂ Ti	eurofer	tungsten	He	eurofer

Table 3: Selected discrete feature assignments corresponding to slices in run 2.

176 2.3 Dimensionality Reduction

177 Model training over high-dimensional parameter spaces may be improved in many aspects by
 178 carefully reducing the number of variables used to describe the space. For many applications,
 179 feature selection strategies succeed in identifying a sufficiently representative subset of the original
 180 input variables; however, all given variables were assumed to be physically relevant to the MC
 181 TBR model. Feature extraction methods, on the other hand, aim to identify a transformation
 182 from the parameter space which decreases dimensionality; even if no individual parameter is
 183 separable from the space, some linear combinations of parameters or nonlinear functions of
 184 parameters may be.

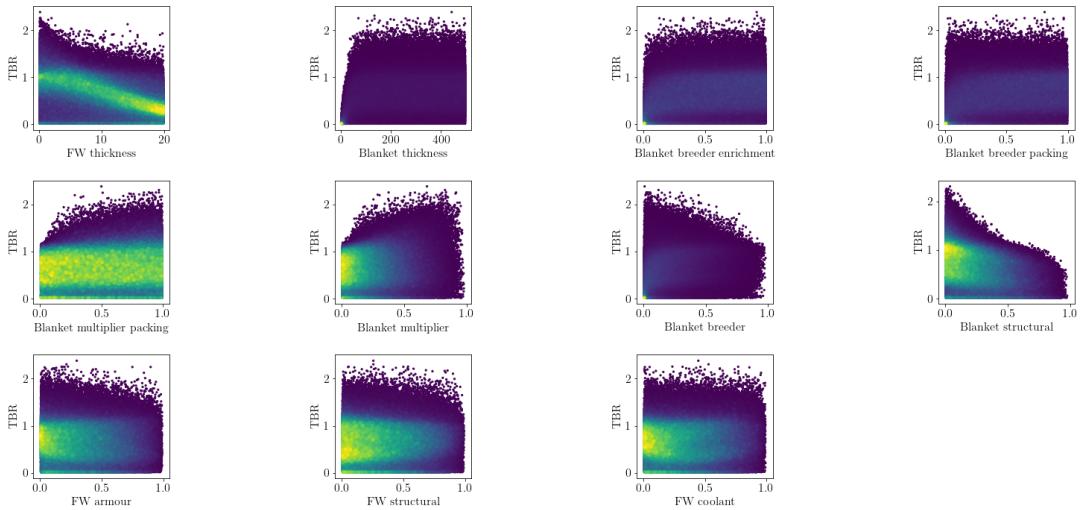


Figure 2: Marginalised dependence of TBR on the choice of continuous features.

185 2.3.1 Principal Component Analysis

186 To pursue linear feature extraction, principal component analysis (PCA) [8] was performed via
 187 SciKit Learn [9] on a set of 300 000 uniform samples of the MC TBR model.
 188 Figure 3 shows the resultant cumulative variance of PCA-identified feature vectors. The similar
 189 share of variance among all features reveals irreducibility of the TBR model by linear methods.

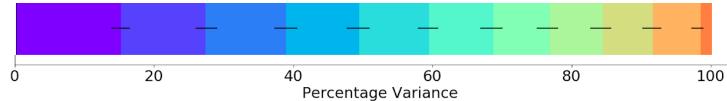


Figure 3: Cumulative variance for optimal features identified by PCA

190 2.3.2 Variogram Computations

191 Kriging is a geostatistical surrogate model-
 192 ling technique which relies on correlation func-
 193 tions over distance (lag) in the feature space
 194 [10]. Although kriging performed poorly for
 195 our use case due to high dimensionality, these
 196 correlation measures gave insight into similar-
 197 ities between discrete-parameter slices of the
 198 data.

199 Figure 4 shows the Matheron semivariance [11]
 200 for three discrete slices with coolant material
 201 varied, but all other discrete parameters fixed.
 202 Fits [12] to the Matérn covariance model con-
 203 firmed numerically that the coolant material is
 204 the discrete parameter with the greatest dis-
 205 tinguishability in the MC TBR model.

206 2.3.3 Autoencoders

207 Autoencoders [13] are a family of approaches
 208 to dimensionality reduction driven by arti-
 209 ficial neural networks (ANNs). Faced with
 210 many possible alternatives, we utilised a con-
 211 ventional autoencoder, which relies on a three-
 212 layer network that is trained to replicate the
 213 identity mapping. While it follows that the in-
 214 put and output layers of such network are sized
 215 to accommodate the analysed dataset, the hid-
 216 den layer, also called *the bottleneck*, allows for
 217 variable number of neurons that represent a
 218 smaller subspace. By scanning over a range
 219 of bottleneck widths and investigating relative
 220 changes in the validation loss, we assess the
 221 potential for dimensional reduction.

222 In particular, we consider two equally-sized sets² of samples: (i) a
 223 subset of data obtained from run 1 and (ii) a subset of a single slice
 224 obtained from run 2. Our expectation was that while the former case

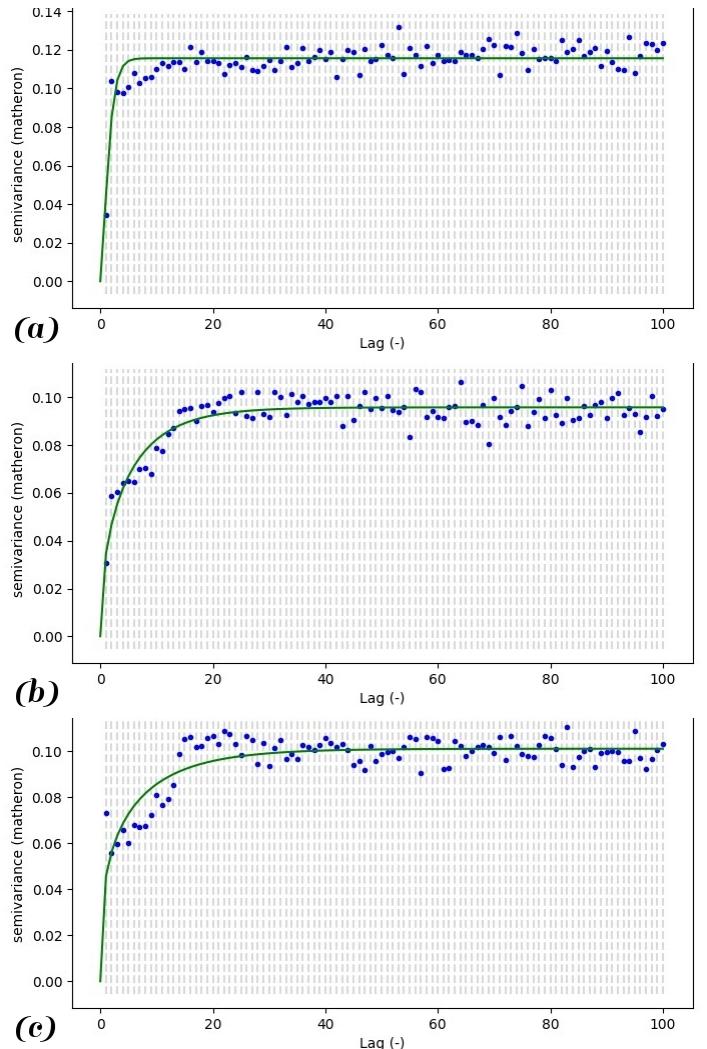
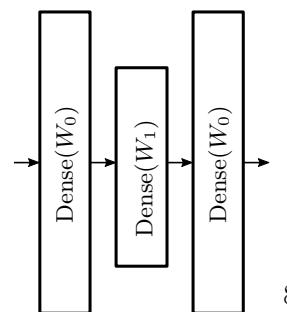


Figure 4: Semivariograms for MC TBR data with coolant materials: (a) He, (b) H₂O, (c) D₂O

² Each set contained 100 000 samples from batches 0-99 of the corresponding runs.



would provide meaningful insights into correlations within the feature space, the latter would validate our autoencoder implementation by analysing a set of points that are trivially reducible in dimensionality due to a number of fixed discrete features.

The results of both experiments are shown in Figure 6. Consistent with our motivation, in each plot we can clearly identify a constant plateau of low error in the region of large dimensionality followed by a point, from which a steep increase is observed. We consider this *critical point* to mark the largest viable dimensional reduction without significant information loss. With this approach we find that the autoencoder was able to reduce the datasets into a subspace of 18 dimensions in the first case and 10 dimensions in the second case.

Confirming our expectation that in the latter, trivial case the autoencoder should achieve greater dimensional reduction, we are inclined to believe that our implementation is indeed operating as intended. However, we must also conclude that in both examined cases this method failed to produce a reduction that would be superior to a naïve approach.³ This is consistent with previous results obtained by PCA and kriging.

3 Methodology

Assuming that data is appropriately treated to eliminate redundant features, we proceed to propose surrogate models and criteria used for their evaluation. The task all presented surrogates strive to solve can be formulated using the language of conventional regression problems. In this section, we focus on interpretation in the scheme of supervised and unsupervised learning.

Labeling the expensive MC TBR model $f(x)$, a surrogate is a mapping $\hat{f}(x)$ that yields similar images as $f(x)$. In other words, $f(x)$ and $\hat{f}(x)$ minimise a selected similarity metric. Furthermore, in order to be considered *viable*, surrogates are required to achieve expected evaluation time lower than that of $f(x)$.

In the supervised learning setting, we first gather a sufficiently large training set of samples $\mathcal{T} = \{(x^{(i)}, f(x^{(i)}))\}_{i=1}^N$ to describe the behaviour of $f(x)$ across its domain. Depending on specific model class and appropriate choice of its hyperparameters, surrogate models $\hat{f}(x)$ are trained to minimise

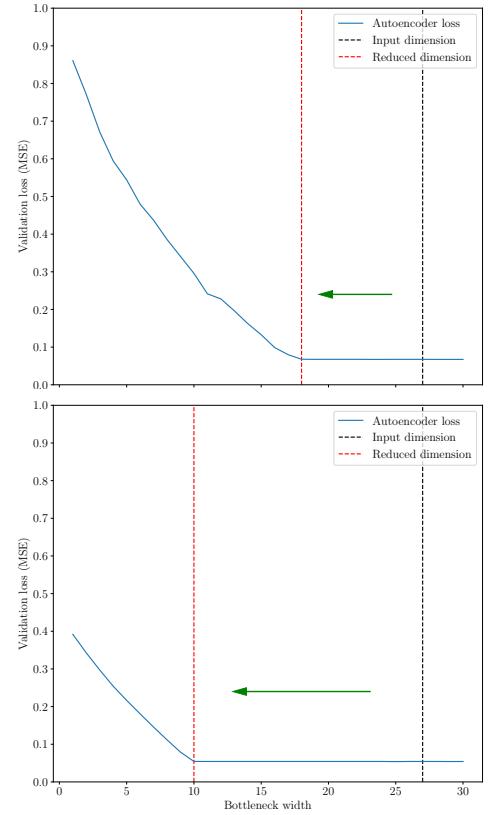


Figure 6: Autoencoder loss scan on the full feature space (top) and a single slice (bottom). Dimensional reduction is indicated by a green arrow.

³ In both tested cases we can trivially eliminate 7 dimensions due to overdetermined one-hot-encoded features and 2 dimensions due to sum-to-one constraints. Furthermore, in the single slice case we may omit 7 additional dimensions due to artificially fixed features.

263 empirical risk with respect to \mathcal{T} and a model-specific loss function \mathcal{L} , where empirical risk is
264 defined as $R_{\text{emp.}}(\hat{f} \mid \mathcal{T}, \mathcal{L}) = \frac{1}{N} \sum_{i=1}^N \mathcal{L}(\hat{f}(x^{(i)}), f(x^{(i)}))$.

265 The unsupervised setting can be viewed as an extension of this method. Rather than fixing
266 the training set \mathcal{T} for the entire duration of training, multiple sets $\{\mathcal{T}_k\}_{k=0}^K$ are used, such that
267 $\mathcal{T}_{k-1} \subset \mathcal{T}_k$ for all $k > 1$. The first set \mathcal{T}_0 is initialised randomly to provide a *burn-in*, and
268 is repeatedly extended in epochs, whereby each epoch trains a new surrogate on \mathcal{T}_k using the
269 supervised learning procedure, evaluates its performance, and forms a new set \mathcal{T}_{k+1} by adding
270 more samples to \mathcal{T}_k . This permits the learning algorithm to condition the selection of new
271 samples on the evaluation results in order to maximise improvement of surrogate performance
272 over complex regions within the domain.

273 3.1 Metrics

274 Aiming to provide objective comparison of a diverse set of surrogate model classes, we define a
275 multitude of metrics to be tracked during experiments. Following the motivation of this work, two
276 desirable properties of surrogates arise: (i) their capability to approximate the TBR MC model
277 well and (ii) their complexity. An ideal surrogate would maximise the former while minimising
278 the latter.

279 Table 4 provides exhaustive listing and description of metrics recorded in the experiments. For
280 regression performance analysis, we include a selection of absolute metrics to assess the ap-
281 proximation capability of surrogates, and set practical bounds on the expected accuracy of their
282 predictions. In addition, we also track relative measures that are better-suited for model compar-
283 ison between works as they maintain invariance with respect to the selected domain and image
284 space. For complexity analysis, surrogates are assessed in terms of wall time elapsed during
285 training and prediction. This is motivated by common practical use cases of our work, where
286 models are trained and used as drop-in replacements for the expensive MC TBR model. Since
287 training set sizes remain to be determined, all times are reported per a single sample. Even
288 though some surrogates support acceleration by means of parallelisation, measures were taken to
289 ensure sequential processing of samples to achieve comparability between considered models.⁴

290 To prevent undesirable bias in results due to training set selection, all metrics are collected
291 in the scheme of k -fold cross-validation with a standard choice of $k = 5$. Herein, a sample
292 set is subdivided into 5 disjoint folds which are repeatedly interpreted as training and testing
293 sets, maintaining a constant ratio of samples between the two.⁵ In each such interpretation
294 experiments are repeated, and the overall value of each metric is reported as the mean across all
295 folds.

296 3.2 Evaluation of Supervised Learning Surrogates

297 In our experiments, we evaluate and compare surrogates in effort to optimise against metrics
298 described in Section 3.1. To attain meaningful and practically usable results, we require a suffi-

⁴ The only exception to this are artificial neural networks, which are notoriously slow to train on conventional CPU architectures in serialised setting.

⁵ Unless explicitly stated otherwise, we use 1 fold for testing and the $k - 1$ remaining folds for training. This gives 80% to 20% train-test ratio.

Regression performance metrics	Mathematical formulation / description	Ideal value [units]
Mean absolute error (MAE)	$\sum_{i=1}^N y^{(i)} - \hat{y}^{(i)} /N$	0 [TBR]
Standard error of regression S	$\text{StdDev}_{i=1}^N \left\{ y^{(i)} - \hat{y}^{(i)} \right\}$	0 [TBR]
Coefficient of determination R^2	$1 - \sum_{i=1}^N (y^{(i)} - \hat{y}^{(i)})^2 / \sum_{i=1}^N (y^{(i)} - \bar{y})^2$	1 [rel.]
Adjusted R^2	$1 - (1 - R^2)(N - 1)/(N - P - 1)$	1 [rel.]
Complexity metrics		
Mean training time $\bar{t}_{\text{trn.}}$	(wall training time of $\hat{f}(x)/N_0$)	0 [ms]
Mean prediction time $\bar{t}_{\text{pred.}}$	(wall prediction time of $\hat{f}(x)/N$)	0 [ms]
Relative speedup σ	(wall evaluation time of $f(x)/(N\bar{t}_{\text{pred.}})$)	$\rightarrow \infty$ [rel.]

Table 4: Metrics recorded in supervised learning experiments. In formulations, we work with training set of size N_0 and testing set of size N , TBR values $y^{(i)} = f(x^{(i)})$ and $\hat{y}^{(i)} = \hat{f}(x^{(i)})$ denote images of the i th testing sample in the expensive model and the surrogate respectively. Furthermore, the mean $\bar{y} = \sum_{i=1}^N y^{(i)}/N$ and P is the number of input features.

ciently large and diverse pool of surrogate classes to draw from. This is described by the listing in Table 5. The presented selection of models includes basic techniques suitable for linear regression enhanced by the kernel trick or dimension lifting, methods driven by decision trees, instance-based learning models, ensemble regressors, randomized algorithms, artificial neural networks and mathematical approaches developed specifically for the purposes of surrogate modelling. For each of these classes, a state-of-the-art implementation was selected and adapted to operate with TBR samples.

Surrogate	Acronym	Implementation	Hyperparameters
Support vector machines [14]	SVM	SciKit Learn [9]	3
Gradient boosted trees [15–17]	GBT	SciKit Learn	11
Extremely randomized trees [18]	ERT	SciKit Learn	7
AdaBoosted decision trees [19]	ABT	SciKit Learn	3
Gaussian process regression [20]	GPR	SciKit Learn	2
k nearest neighbours	KNN	SciKit Learn	3
Artificial neural networks	ANN	Keras (TensorFlow) [21]	2
Inverse distance weighing [22]	IDW	SMT [23]	1
Radial basis functions	RBF	SMT	3
Kriging [24–27]	KRG	SMT	4

Table 5: Considered surrogate model classes.

In some rows of Table 5, a single class in reality represents a family of fundamentally similar approaches. Good examples of this are kriging methods and neural networks. In the case of the former, multiple algorithms such as kriging based on partial least squares and gradient-enhanced kriging are considered. In the latter, a host of parametric graphs are defined to realise simplistic network architecture search (see Figure 7 for details). Discrimination between such options is considered an additional hyperparameter of the corresponding surrogate class.

3.2.1 Experiments

The presented surrogate candidates are evaluated in four experimental cases:

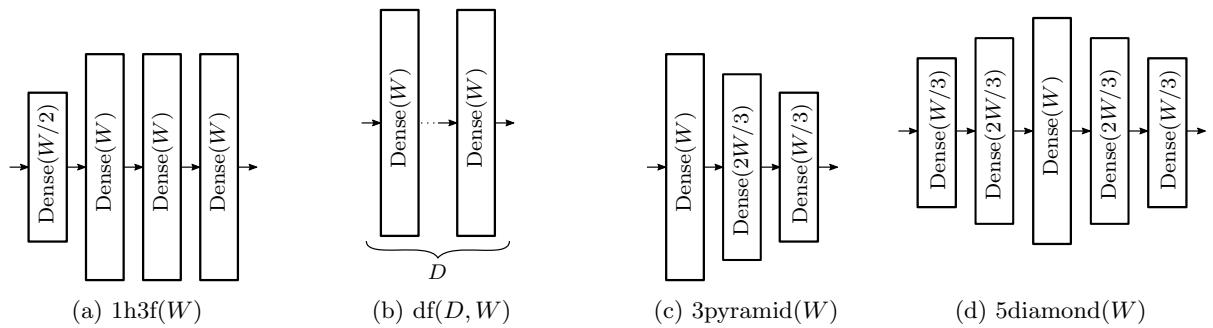


Figure 7: Selected parametric neural network architectures. All layers except the last use ReLU activation. Prediction information flow is indicated by arrows. Connections between neurons are omitted for clarity.

- 314 1. Hyperparameter tuning in simplified domain,
315 2. Hyperparameter tuning in full domain,
316 3. Scaling benchmark,
317 4. Competitive surrogate training.

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 is employed as a standard hyperparameter tuning algorithm. We set its objective to maximise R^2 and perform 1000 iterations.⁶

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 follows the notion that, if desired, accuracy of such models may be replicated at large scale by decomposing the TBR domain into continuous subsets, and learning ensemble surrogate comprised of identical models that are trained on each subset independently, incurring exponential complexity penalty in the process. 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.

334 The second experiment conventionally measures surrogate performance on the full feature space.
335 Here, in extension of the previous case, surrogates work with samples comprised of discrete as
336 well as continuous features.

The goal of the last two experiments is to exploit the information gathered by hyperparameter tuning for study of scaling properties. In the third experiment, 20 best-performing hyperparameter choices of each class (in R^2) are used to learn surrogates using progressively larger training sets. Following that, the fourth experiment attempts to produce competitive surrogates by re-training several selected well-scaling models on the largest available data set.

⁶ Tuning of each surrogate class was artificially terminated after 2 days. Instances which reached this limit are marked in the results.

3.3 Adaptive Sampling

All of the surrogate modelling techniques studied in this project face a shared 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 incrementally so as to best suit the model under consideration.

Adaptive sampling techniques are widespread in the literature and have been specialised for surrogate modelling. Garud's [28] "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 [29] for a review).

Many researchers have embedded surrogate methods into MCMC strategies for parameter optimisation [30, 31], in particular the ASMO-PODE algorithm [32] 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 8) to iteratively increment the training/test set with sample points which maximise surrogate error and minimise a crowding distance metric (CDM) [33] in feature space. On each iteration following an initial training of the surrogate on N uniformly random samples, the surrogate was trained and AE 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.

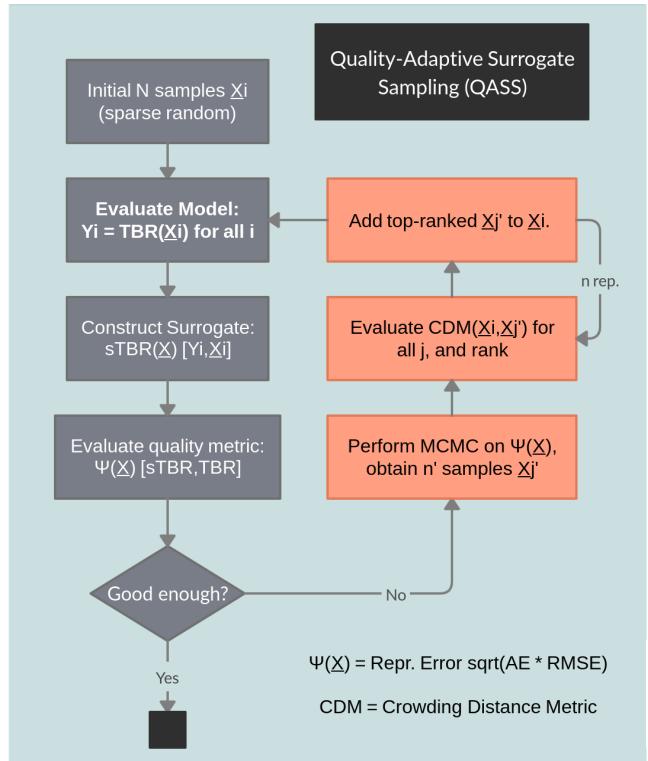


Figure 8: Schematic of QASS algorithm

386 **4 Results**

387 TODO

388 **4.1 Evaluation of Supervised Learning Surrogates**

389 We begin by evaluating a diverse set of surrogate classes that we proposed earlier. In particular,
 390 we aim to study considered models in terms of regression performance and evaluation complexity.
 391 Following Section 3.2.1, where we proposed four experiments that accomplish this task, we present
 392 and discuss our results in the next several sections.

393 **4.1.1 Hyperparameter Tuning**

394 The first two experiments perform Bayesian optimisation to maximise R^2 in cross-validated
 395 setting as a function of model hyperparameters. While in the first experiment we limit training
 396 and testing sets to the scope of four slices of the feature space, in the second experiment we lift
 397 this restriction.

398 The results displayed in Figure 9 indicate that in the first experiment, gradient boosted trees
 399 clearly appear to be the most accurate as well as the fastest surrogate class in terms of mean pre-
 400 diction time. Following that, we note that extremely randomised trees, support vector machines
 401 and artificial neural networks also achieved satisfactory results with respect to both examined
 402 metrics. While the remainder of tested surrogate classes does not exhibit problems in complexity,
 403 its regression performance falls below average.

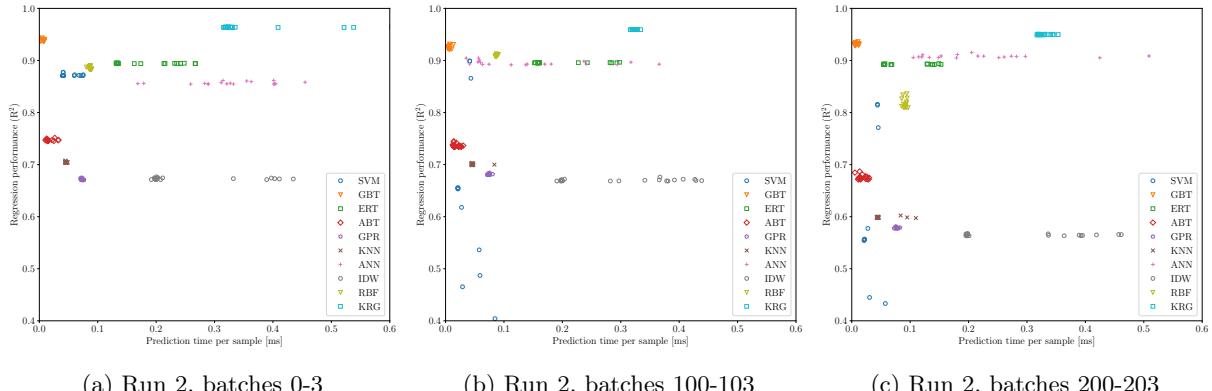


Figure 9: 20 best-performing surrogates per each considered class, plotted in terms of complexity (as $\bar{t}_{\text{pred.}}$) and regression performance (as R^2) on selected slices of run 2, evaluated in experiment 1.

404 TODO: multislice description

405 4.1.2 Scaling Benchmark

406 In the next experiment we examine surrogate scaling properties
 407 correlating metrics of interest with progressively increasing
 408 training set size. The results shown in Figure 11 seem to suggest
 409 that in terms of regression performance, methods based on
 410 decision trees and artificial neural networks offer the best ac-
 411 curacy on larger sets overall. This is a curious extension of the
 412 previous findings, where gradient boosted trees were observed
 413 to significantly dominate over the rest of the examined meth-
 414 ods. With increasing training set size, their relative advantage
 415 is clearly gradually diminished.

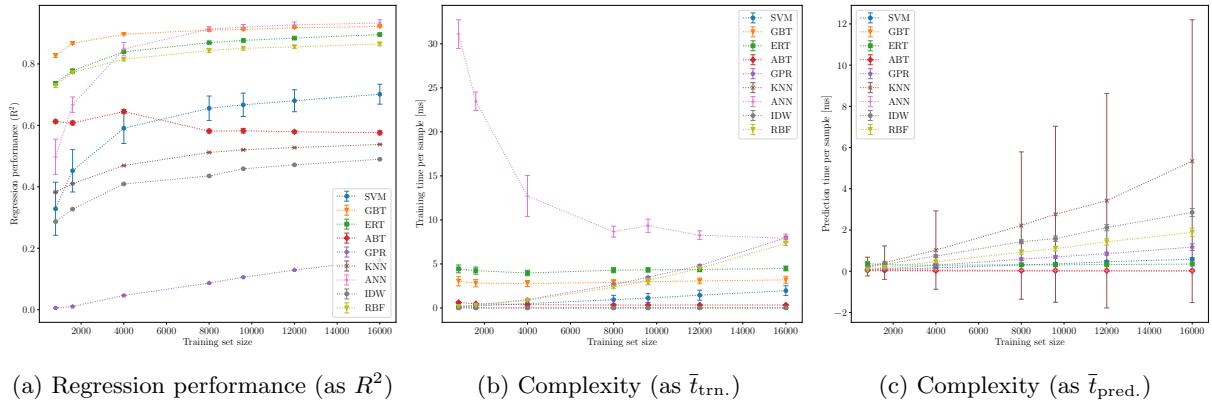


Figure 11: Various metrics collected during experiment 3 (scaling benchmark) displayed as a function of training set size.

416 According to our experiment, the lowest mean training time is generally achieved by instance-
 417 based learning methods, which seem to offer near-constant scaling characteristics at the expense
 418 of significant performance increase later during prediction. Following that, we observe that the
 419 majority of tree-based methods also exhibit desirable properties. The notable exception here
 420 appear to be artificial neural networks, which are the only model to utilise parallelisation. As
 421 such, their constant synchronisation overhead presumably hinders performance on small training
 422 sets, producing misleading results when divided by the number of samples.

423 In terms of mean prediction time, all tested surrogates except previously mentioned instance-
 424 based learning methods scale exceptionally well. Tree-based generally models appear to perform
 425 the fastest.

426 4.1.3 Competitive Surrogate Training

427 TODO

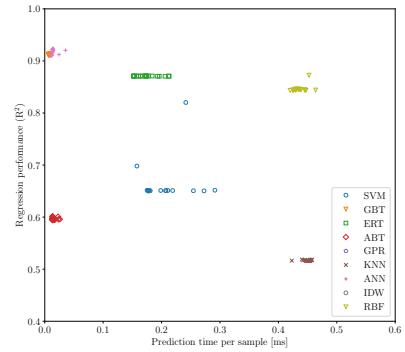


Figure 10: Results of experiment 2, plotted analogously to Figure 9.

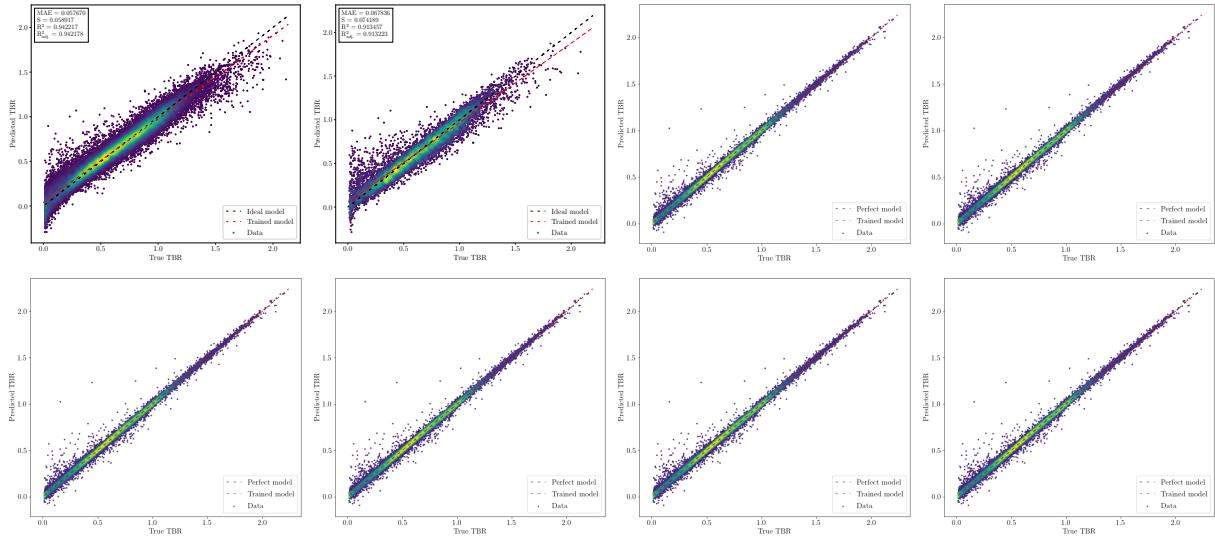


Figure 12: Regression performance of models 1-4 (row 1, from the left) and 5-8 (row 2) trained in experiment 4 (competitive surrogate training), viewed as true vs. predicted TBR on a test set of a selected cross-validation fold. Due to large set sizes, points are randomly subsampled and coloured by density.

428 4.2 Results of Adaptive Sampling

429 In order to test our QASS prototype, several functional
 430 toy theories for TBR were developed as alternatives to
 431 the expensive MC model. By far the most robust of
 432 these was the following sinusoidal theory with adjustable
 433 wavenumber parameter n :

$$\text{TBR} = \text{Mean}_{i \in C} \left[\frac{1 + \sin(2\pi n(x_i - 1/2))}{2} \right] \quad (3)$$

434 plotted in Figure 13 for $n = 1$ and two continuous
 435 parameters C . ANNs trained on this model demon-
 436 strated similar performance to those on the expensive
 437 MC model. QASS performance was verified by train-
 438 ing a 1h3f(256) ANN on the sinusoidal theory for varied
 439 quantities of initial, incremental, and MCMC candidate
 440 samples. Although the scope of this project did not include thorough searches of this hyperpara-
 441 meter domain, sufficient runs were made to identify some likely trends.

442 An increase in MCMC candidate samples was seen to have a positive but very weak effect on
 443 final surrogate precision, suggesting that the runtime of MCMC on each iteration can be limited
 444 for increased efficiency. – Awaiting test results on initial sample quantity –. The most complex
 445 dynamics arose with the adjustment of sample increment, shown in Figure 14. For each tested
 446 initial sample quantity N , the optimal number of step samples was seen to be well-approximated
 447 by \sqrt{N} ; the plotted error trends suggest that incremental samples larger than this optimum give
 448 slower model improvement on both the training and evaluation sets, and a larger minimum error

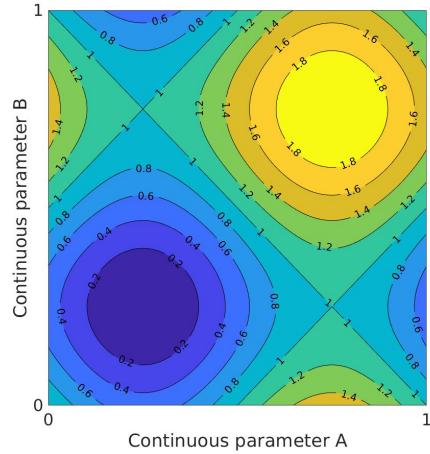


Figure 13: Sinusoidal toy TBR theory over two continuous parameters, wavenumber 1

449 on the evaluation set. This performance distinction is predicted to be even more significant when
 450 trained on the expensive MC model, where the number of sample evaluations will serve as the primary bottleneck for computation time.

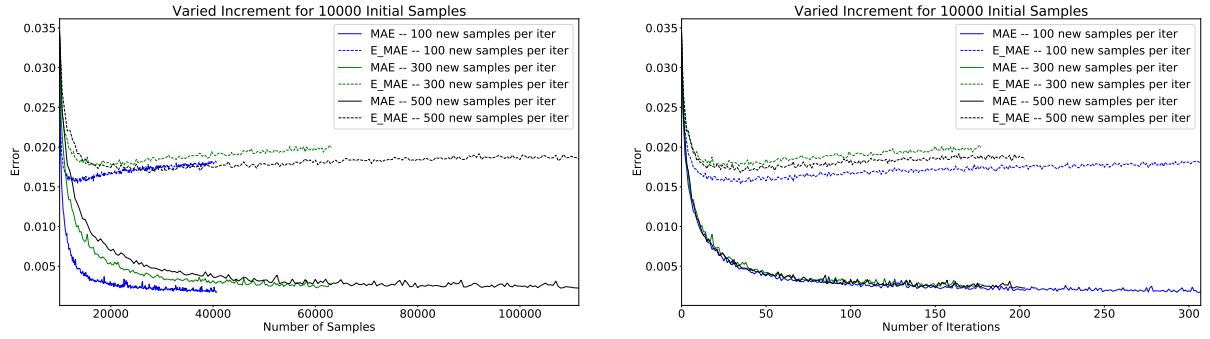


Figure 14: QASS absolute training error over total sample quantity (left) and number of iterations (right). MAE represents surrogate error on the adaptively-sampled training/test set, and E_MAE on the independent evaluation sets.

451
 452 The plateau effect in surrogate error on the evaluation set, seen in Figure 14, was universal to all
 453 configurations and thought to warrant further investigation. At first this was suspected to be a
 454 residual effect of retraining the same ANN instance without adjustment to data normalisation;
 455 a "Goldilocks scheme" for checking normalisation drift was implemented and tested, but did
 456 not affect QASS performance. Schemes in which the ANN is periodically retrained were also
 457 discarded, as the retention of network weights from one iteration to the next was demonstrated
 458 to greatly benefit QASS efficiency. Further insight came from direct comparison between QASS
 459 and a baseline scheme with uniformly random incremental samples, shown in Figure 15.

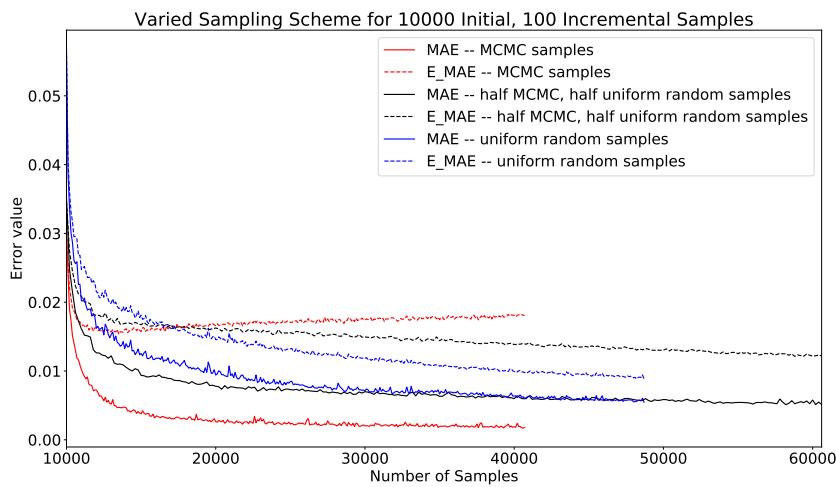


Figure 15: Absolute training error for QASS, baseline scheme, and mixed scheme

460 Such tests revealed that while QASS has unmatched performance on its own adaptively-sampled
 461 training set, it is outperformed by the baseline scheme on uniformly random evaluation sets. We

462 suspected that while QASS excels in learning the most strongly peaked regions of the TBR theory,
463 this comes at the expense of precision in broader, smoother regions where uniformly random
464 sampling suffices. Therefore a mixed scheme was implemented, with half MCMC samples and half
465 uniformly random samples incremented on each iteration, which is also shown in Figure 15.

466 **5 Conclusion**

467 Over the course of this internship project, we employed a broad spectrum of data analysis and
468 machine learning techniques to develop fast and high-quality surrogate models for a MC TBR
469 model in use at UKAEA. We generated 900,000 samples for training and test purposes, evaluated
470 on this expensive MC model. We investigated possibilities for simplification of the parameter
471 space, and concluded that no straightforward reduction was possible. After reviewing N surrogate
472 models (fill in N), and examining their behaviour on (un)constrained feature space and scaling
473 properties, we retrained some of the best-performing surrogates on the full parameter space. The
474 optimum results obtained were an accuracy of X and a mean prediction time of X , representing
475 a relative speedup X with respect to the MC model.

476 After a thorough review of the literature, we also developed a novel adaptive sampling algorithm,
477 QASS, capable of interfacing with any of the individual studied models. Preliminary testing on
478 a toy theory, qualitatively comparable to the MC TBR model, demonstrated the effectiveness
479 of QASS and behavioural trends consistent with the design of the algorithm. *[Insert numerical
480 results for QASS.]* Further optimisation over the hyperparameter space has strong potential to
481 increase this performance, allowing for future deployment of QASS on the MC TBR model in
482 coalition with any of the most effective identified surrogate models.

483 **Acknowledgements**

484 TODO: thank supervisory team

485 References

- 486 [1] Jacob Søndergaard. “Optimization Using Surrogate Models”. PhD thesis. Technical University of Denmark, 2003.
- 487 [2] R.H. Myers and D.C. Montgomery. *Response Surface Methodology: Product and Process Optimization Using Designed Experiments*. 2nd. New York: John Wiley & Sons, 2002.
- 488 [3] Jonathan Shimwell. *collaboration*.
- 489 [4] F.A. Hernández and P. Pereslavtsev. “First principles review of options for tritium breeder and neutron multiplier materials for breeding blankets in fusion reactors”. In: *Fusion Engineering and Design* 137 (Dec. 2018), pp. 243–256. ISSN: 0920-3796.
- 490 [5] M Keilhacker. “JET deuterium: tritium results and their implications”. In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 357 (Mar. 1999), pp. 415–442.
- 491 [6] M. Coleman and S. McIntosh. “BLUEPRINT: A novel approach to fusion reactor design”. In: *Fusion Engineering and Design* 139 (Feb. 2019), pp. 26–38. ISSN: 0920-3796.
- 492 [7] Paul K. Romano et al. “OpenMC: A state-of-the-art Monte Carlo code for research and development”. In: *Annals of Nuclear Energy* 82 (2015). 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, pp. 90–97. ISSN: 0306-4549.
- 493 [8] Ian T Jolliffe and Jorge Cadima. “Principal component analysis: a review and recent developments”. In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 374.2065 (Apr. 2016), p. 20150202.
- 494 [9] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.
- 495 [10] Mohamed Amine Bouhlel and Joaquim Martins. “Gradient-enhanced kriging for high-dimensional problems”. In: *Engineering with Computers* (Feb. 2018).
- 496 [11] Georges Matheron. “Principles of geostatistics”. In: *Economic Geology* 58.8 (Dec. 1963), pp. 1246–1266. ISSN: 0361-0128.
- 497 [12] *Kriging Variogram Model*. URL: https://vsp.pnn.gov/help/Vsample/Kriging%7B%5C_%7DVariogram%7B%5C_%7DModel.htm (visited on 20/04/2020).
- 498 [13] Jürgen Schmidhuber. “Deep learning in neural networks: An overview”. In: *Neural Networks* 61 (2015), pp. 85–117. ISSN: 0893-6080.
- 499 [14] Rong-En Fan et al. “LIBLINEAR: A library for large linear classification”. In: *Journal of machine learning research* 9.Aug (2008), pp. 1871–1874.
- 500 [15] Jerome H Friedman. “Greedy function approximation: a gradient boosting machine”. In: *Annals of statistics* (2001), pp. 1189–1232.
- 501 [16] JH Friedman. *Stochastic Gradient Boosting Technical Report*. 1999.
- 502 [17] Trevor Hastie, Robert Tibshirani and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.
- 503 [18] Pierre Geurts, Damien Ernst and Louis Wehenkel. “Extremely randomized trees”. In: *Machine learning* 63.1 (2006), pp. 3–42.
- 504 [19] Harris Drucker. “Improving regressors using boosting techniques”. In: *ICML*. Vol. 97. 1997, pp. 107–115.
- 505 [20] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*. Vol. 2. 3. MIT press Cambridge, MA, 2006.
- 506 [21] François Chollet et al. *Keras*. <https://keras.io>. 2015.
- 507 [22] Donald Shepard. “A two-dimensional interpolation function for irregularly-spaced data”. In: *Proceedings of the 1968 23rd ACM national conference*. 1968, pp. 517–524.
- 508 [23] Mohamed Amine Bouhlel et al. “A Python surrogate modeling framework with derivatives”. In: *Advances in Engineering Software* (2019), p. 102662. ISSN: 0965-9978.
- 509 [24] Jerome Sacks, Susannah B Schiller and William J Welch. “Designs for computer experiments”. In: *Technometrics* 31.1 (1989), pp. 41–47.
- 510 [25] Herman Wold. “Soft modelling by latent variables: the non-linear iterative partial least squares (NIPALS) approach”. In: *Journal of Applied Probability* 12.S1 (1975), pp. 117–142.
- 511 [26] Mohamed Amine Bouhlel et al. “Improving kriging surrogates of high-dimensional design models by Partial Least Squares dimension reduction”. In: *Structural and Multidisciplinary Optimization* 53.5 (2016), pp. 935–952.
- 512 [27] Mohamed Amine Bouhlel et al. “An improved approach for estimating the hyperparameters of the kriging model for high-dimensional problems through the partial least squares method”. In: *Mathematical Problems in Engineering* 2016 (2016).
- 513 [28] Sushant Garud, Iftekhar Karimi and Markus Kraft. “Smart Sampling Algorithm for Surrogate Model Development”. In: *Computers & Chemical Engineering* 96 (Oct. 2016).
- 514 [29] Jun Zhou, Xiaosi Su and Geng Cui. “An adaptive Kriging surrogate method for efficient joint estimation of hydraulic and biochemical parameters in reactive transport modeling”. In: *Journal of Contaminant Hydrology* 216 (Sept. 2018), pp. 50–57. ISSN: 0169-7722.
- 515 [30] Wei Gong and Qingyun Duan. “An adaptive surrogate modeling-based sampling strategy for parameter optimization and distribution estimation (ASMO-PODE)”. In: *Environmental Modelling & Software* 95 (Sept. 2017), pp. 61–75. ISSN: 1364-8152.

- 596 [31] Jiangjiang Zhang et al. “Surrogate-Based Bayesian Inverse Modeling of the Hydrological System: An Adaptive Approach Considering Surrogate Approximation Error”. In: *Water Resources Research* 56.1 (Jan 2020), e2019WR025721. ISSN: 0043-1397.
- 597 [32] Victor Ginting et al. “Application of the two-stage Markov chain Monte Carlo method for characterization of fractured reservoirs using a surrogate flow model”. In: *Computational Geosciences* 15.4 (2011), p. 691. ISSN: 1573-1499.
- 598 [33] Antti Solonen et al. “Efficient MCMC for Climate Model Parameter Estimation: Parallel Adaptive Chains and Early Rejection”. en. In: *Bayesian Analysis* 7.3 (2012), pp. 715–736. ISSN: 1936-0975.
- 599 [34] Jie Zhang, Souma Chowdhury and Achille Messac. “An adaptive hybrid surrogate model”. In: *Structural and Multidisciplinary Optimization* 46.2 (2012), pp. 223–238. ISSN: 1615-1488.
- 600
- 601
- 602
- 603
- 604
- 605

614 Appendices

615 A Detailed Results of Supervised Models

Surrogate class	#	Regression performance				Complexity	
		MAE [TBR]	S [TBR]	R ² [rel.]	R ² _{adj.} [rel.]	̄t _{trn.} [ms]	̄t _{pred.} [ms]
SVM	1040	0.08 ± 0.01	0.10 ± 0.01	0.86 ± 0.03	0.86 ± 0.03	0.25 ± 0.01	0.04 ± 0.01
GBT	1000	0.06 ± 0.01	0.06 ± 0.01	0.93 ± 0.01	0.93 ± 0.01	3.25 ± 0.62	0.00 ± 0.00
ERT	1000	0.08 ± 0.01	0.08 ± 0.00	0.90 ± 0.00	0.90 ± 0.00	3.26 ± 1.70	0.12 ± 0.04
ABT	1000	0.15 ± 0.02	0.10 ± 0.01	0.73 ± 0.03	0.73 ± 0.03	0.18 ± 0.01	0.01 ± 0.00
GPR	1000	0.15 ± 0.02	0.14 ± 0.02	0.64 ± 0.04	0.63 ± 0.04	0.24 ± 0.00	0.08 ± 0.00
KNN	1000	0.14 ± 0.02	0.14 ± 0.02	0.66 ± 0.05	0.65 ± 0.05	0.00 ± 0.00	0.74 ± 0.03
ANN	461	0.07 ± 0.01	0.08 ± 0.01	0.90 ± 0.02	0.89 ± 0.02	26.21 ± 8.41	0.29 ± 0.03
IDW	1000	0.15 ± 0.02	0.14 ± 0.02	0.63 ± 0.05	0.62 ± 0.05	0.00 ± 0.00	0.29 ± 0.03
RBF	1000	0.08 ± 0.01	0.09 ± 0.01	0.88 ± 0.03	0.88 ± 0.03	0.19 ± 0.00	0.09 ± 0.00
KRG	1000	0.05 ± 0.01	0.05 ± 0.01	0.96 ± 0.01	0.96 ± 0.01	15.90 ± 0.58	0.32 ± 0.02

Table 6: Results of experiment 1 (single slice hyperparameter tuning) as means and standard deviations over 4 tested slices. Column # gives the number of Bayesian optimisation iterations. While regression performance is reported for the best instance (in R²) per surrogate class, complexity is measured over all tested instances.

Surrogate class	#	Regression performance				Complexity	
		MAE [TBR]	S [TBR]	R ² [rel.]	R ² _{adj.} [rel.]	̄t _{trn.} [ms]	̄t _{pred.} [ms]
SVM	1000	0.09	0.11	0.82	0.82	0.52 ± 0.23	0.20 ± 0.04
GBT	581	0.07	0.07	0.91	0.91	14.43 ± 42.08	0.01 ± 0.00
ERT	901	0.09	0.09	0.87	0.87	3.73 ± 12.78	0.17 ± 0.05
ABT	1000	0.18	0.12	0.60	0.60	0.21 ± 0.07	0.01 ± 0.01
GPR	1000	0.27	0.19	0.09	0.08	0.98 ± 0.01	0.28 ± 0.01
KNN	1000	0.18	0.15	0.52	0.51	0.00 ± 0.00	4.53 ± 4.18
ANN	760	0.06	0.07	0.92	0.92	4.45 ± 3.51	0.01 ± 0.00
IDW	1000	0.19	0.16	0.45	0.45	0.00 ± 0.00	0.77 ± 0.01
RBF	1000	0.08	0.09	0.87	0.87	1.01 ± 0.27	0.50 ± 0.14

Table 7: Results of experiment 2 (full feature space hyperparameter tuning), shown analogously to Table 6.

Surrogate class	Regression performance as R^2 [rel.] by cross-validation set size						
	1000	2000	5000	10 000	12 000	15 000	20 000
SVM	0.33 ± 0.09	0.45 ± 0.07	0.59 ± 0.05	0.66 ± 0.04	0.67 ± 0.04	0.68 ± 0.04	0.70 ± 0.03
GBT	0.83 ± 0.01	0.87 ± 0.00	0.90 ± 0.00	0.91 ± 0.00	0.91 ± 0.00	0.92 ± 0.00	0.92 ± 0.00
ERT	0.74 ± 0.00	0.78 ± 0.00	0.84 ± 0.00	0.87 ± 0.00	0.88 ± 0.00	0.88 ± 0.00	0.90 ± 0.00
ABT	0.61 ± 0.01	0.61 ± 0.01	0.64 ± 0.01	0.58 ± 0.01	0.58 ± 0.01	0.58 ± 0.00	0.58 ± 0.01
GPR	0.01 ± 0.00	0.01 ± 0.00	0.05 ± 0.00	0.09 ± 0.00	0.11 ± 0.00	0.13 ± 0.00	0.16 ± 0.00
KNN	0.38 ± 0.00	0.41 ± 0.00	0.47 ± 0.00	0.51 ± 0.00	0.52 ± 0.00	0.53 ± 0.00	0.54 ± 0.00
ANN	0.50 ± 0.06	0.67 ± 0.03	0.85 ± 0.02	0.91 ± 0.01	0.92 ± 0.01	0.93 ± 0.01	0.93 ± 0.01
IDW	0.29 ± 0.00	0.33 ± 0.00	0.41 ± 0.00	0.44 ± 0.00	0.46 ± 0.00	0.47 ± 0.00	0.49 ± 0.00
RBF	0.73 ± 0.01	0.77 ± 0.00	0.81 ± 0.00	0.84 ± 0.01	0.85 ± 0.01	0.86 ± 0.01	0.86 ± 0.01

 Table 8: Results of experiment 3 (scaling benchmark) in terms of R^2 .

Surrogate class	Complexity as $\bar{t}_{\text{trn.}}$ [ms] by cross-validation set size						
	1000	2000	5000	10 000	12 000	15 000	20 000
SVM	0.12 ± 0.06	0.22 ± 0.15	0.50 ± 0.36	0.94 ± 0.49	1.12 ± 0.51	1.45 ± 0.57	1.96 ± 0.56
GBT	3.05 ± 0.54	2.82 ± 0.42	2.78 ± 0.34	2.89 ± 0.36	2.98 ± 0.35	3.07 ± 0.32	3.19 ± 0.42
ERT	4.44 ± 0.45	4.24 ± 0.40	3.97 ± 0.31	4.29 ± 0.29	4.32 ± 0.25	4.37 ± 0.25	4.49 ± 0.27
ABT	0.62 ± 0.08	0.48 ± 0.04	0.37 ± 0.05	0.33 ± 0.03	0.32 ± 0.04	0.34 ± 0.03	0.34 ± 0.04
GPR	0.18 ± 0.01	0.32 ± 0.03	0.95 ± 0.05	2.63 ± 0.11	3.48 ± 0.11	4.81 ± 0.12	7.99 ± 0.16
KNN	0.01 ± 0.00	0.00 ± 0.00	0.01 ± 0.00	0.01 ± 0.00	0.01 ± 0.00	0.01 ± 0.00	0.01 ± 0.00
ANN	31.11 ± 1.62	23.48 ± 1.05	12.71 ± 2.33	8.66 ± 0.62	9.34 ± 0.76	8.26 ± 0.50	7.92 ± 0.47
IDW	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
RBF	0.16 ± 0.01	0.29 ± 0.02	0.86 ± 0.07	2.35 ± 0.12	3.13 ± 0.14	4.49 ± 0.16	7.33 ± 0.26

 Table 9: Results of experiment 3 (scaling benchmark) in terms of $\bar{t}_{\text{trn.}}$, displayed analogously to Table 8.

Surrogate class	Complexity as $\bar{t}_{\text{pred.}}$ [ms] by cross-validation set size						
	1000	2000	5000	10 000	12 000	15 000	20 000
SVM	0.04 ± 0.01	0.08 ± 0.01	0.16 ± 0.03	0.30 ± 0.04	0.35 ± 0.05	0.46 ± 0.06	0.59 ± 0.09
GBT	0.02 ± 0.00	0.02 ± 0.00	0.01 ± 0.00	0.01 ± 0.00	0.01 ± 0.00	0.01 ± 0.00	0.01 ± 0.00
ERT	0.37 ± 0.06	0.32 ± 0.05	0.29 ± 0.04	0.31 ± 0.04	0.31 ± 0.06	0.33 ± 0.06	0.35 ± 0.05
ABT	0.07 ± 0.01	0.04 ± 0.01	0.03 ± 0.00	0.03 ± 0.00	0.02 ± 0.00	0.02 ± 0.00	0.02 ± 0.00
GPR	0.05 ± 0.01	0.11 ± 0.01	0.30 ± 0.02	0.58 ± 0.06	0.69 ± 0.06	0.85 ± 0.07	1.17 ± 0.16
KNN	0.23 ± 0.45	0.41 ± 0.82	1.03 ± 1.90	2.22 ± 3.58	2.76 ± 4.27	3.42 ± 5.20	5.34 ± 6.86
ANN	0.17 ± 0.02	0.13 ± 0.07	0.07 ± 0.04	0.04 ± 0.03	0.04 ± 0.03	0.03 ± 0.02	0.03 ± 0.02
IDW	0.19 ± 0.02	0.34 ± 0.04	0.73 ± 0.06	1.43 ± 0.10	1.58 ± 0.13	2.12 ± 0.15	2.85 ± 0.19
RBF	0.08 ± 0.01	0.19 ± 0.02	0.46 ± 0.05	0.93 ± 0.10	1.10 ± 0.12	1.43 ± 0.15	1.88 ± 0.21

 Table 10: Results of experiment 3 (scaling benchmark) in terms of $\bar{t}_{\text{pred.}}$, displayed analogously to Table 8.

B Overview of Online Resources

- 616 TODO: point reader to various online repositories, briefly describe them
- 617 1. data repository
- 618 2. sampling repository
- 619 3. regression repository

Model	Set size	Regression performance				Complexity		
		MAE [TBR]	S [TBR]	R^2 [rel.]	$R_{\text{adj.}}^2$ [rel.]	$\bar{t}_{\text{trn.}}$ [ms]	$\bar{t}_{\text{pred.}}$ [ms]	σ [rel.]
1 (GBT)	200 000	0.06 ± 0.00	0.06 ± 0.00	0.94 ± 0.00	0.94 ± 0.00	2.22 ± 0.01	0.01 ± 0.00	$1169.933 \times$
2 (RBF)	50 000	0.07 ± 0.00	0.08 ± 0.00	0.91 ± 0.00	0.91 ± 0.00	3.45 ± 0.02	1.51 ± 0.02	$5143 \times$

Table 11: Results of experiment 4 (competitive training) displayed analogously to Table 7. Here, means and standard deviation are reported across 5 cross-validation folds. In addition, we also give used cross-validation set size and relative speedup with respect to the MC TBR model mean evaluation time 7.78 s (per sample) measured during run 1.

- 621 4. hyperopt repository
 622 5. docs repository