



UCL CDT DIS Note

22nd April 2020



UK Atomic
Energy
Authority

Draft version 0

2 Surrogate Modelling of 3 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 mod-
7 ern 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 ac-
16 curacy 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	11
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 1 Introduction

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

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

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

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

76 1.1 Problem Description

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

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

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



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

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

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

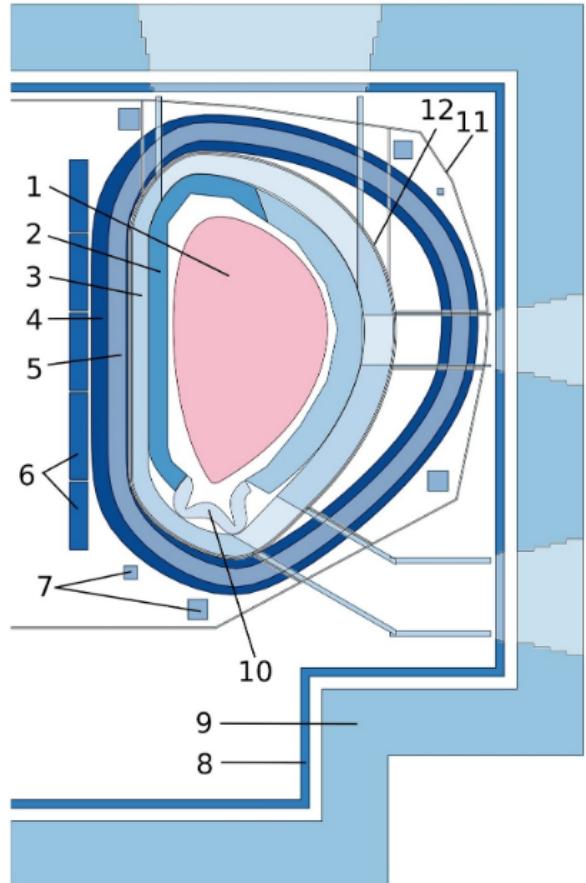


Figure 1: Typical single-null reactor configuration as specified by BLUEPRINT [6]: 1 — plasma, 2 — breeding blankets

122 2 Data Exploration

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

128 2.1 Expensive Model Description

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

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

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

145 2.2 Dataset Generation

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

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

#	Samples	Batch division	Total run time	MET [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. Total run time includes waiting in the processing queue. MET stands for mean evaluation time (per single sampled point).

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

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.

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

173 2.3 Dimensionality Reduction

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

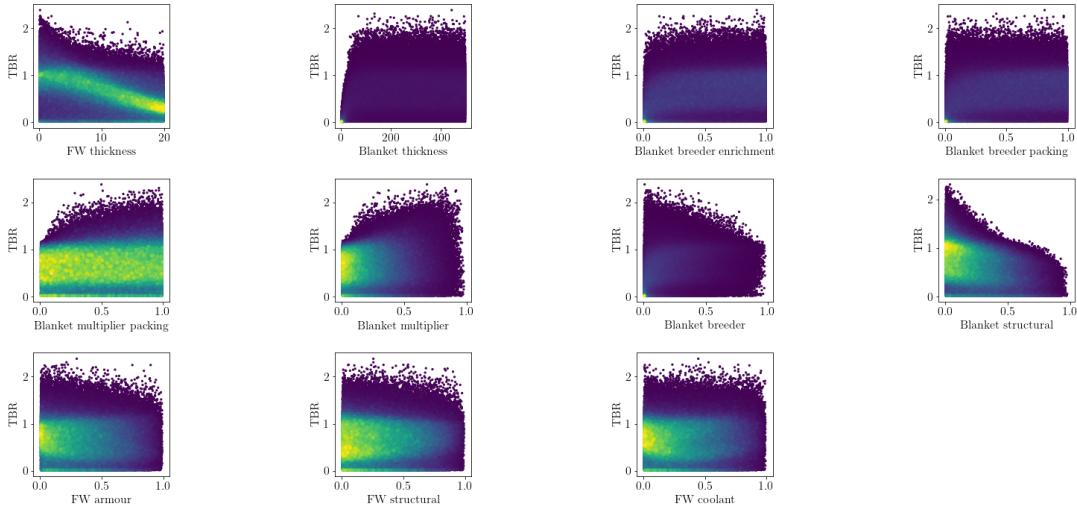


Figure 2: Marginalised dependence of TBR on the choice of continuous features.

182 2.3.1 Principal Component Analysis

183 To pursue linear feature extraction, principal component analysis (PCA) [8] was performed via
 184 SciKit Learn [9] on a set of 300 000 uniform samples of the MC TBR model.

185 Figure 3 shows the resultant cumulative variance of PCA-identified feature vectors. The similar
 186 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

187 2.3.2 Variogram Computations

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

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

203 2.3.3 Autoencoders

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

219 In particular, we consider two equally-sized sets¹ of samples: (i) a
 220 subset of data obtained from run 1 and (ii) a subset of a single slice
 221 obtained from run 2. Our expectation was that while the former case
 222 would provide meaningful insights into correlations within the feature
 223 space, the latter would validate our autoencoder implementation by
 224 analysing a set of points that are trivially reducible in dimensionality
 225 due to a number of fixed discrete features.

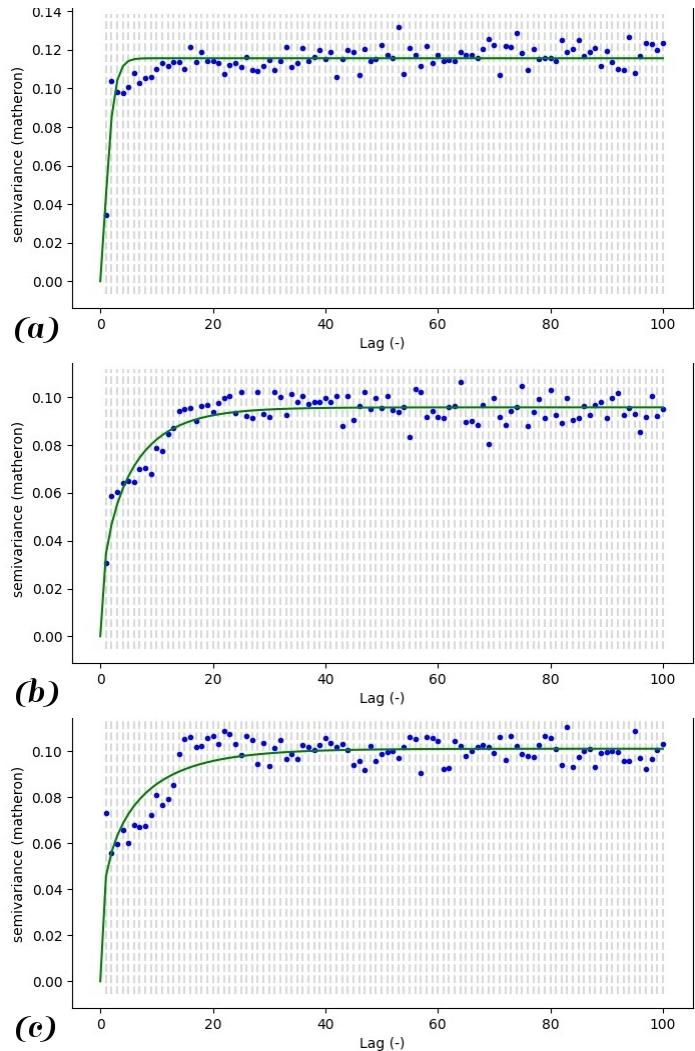


Figure 4: Semivariograms for MC TBR data with coolant materials: (a) He, (b) H₂O, (c) D₂O

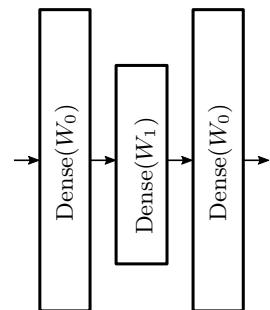


Figure 5: Autoencoder with input width W_0 and bottleneck width W_1 .

¹ Each set contained 100 000 samples from batches 0-99 of the corresponding runs.

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

233 Confirming our expectation that in the latter, trivial
 234 case the autoencoder should achieve greater dimensional
 235 reduction, we are inclined to believe that our implementa-
 236 tion is indeed operating as intended. However, we must
 237 also conclude that in both examined cases this method
 238 failed to produce a reduction that would be superior
 239 to a naïve approach.² This is consistent with previous
 240 results obtained by PCA and kriging.

241 3 Methodology

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

249 Labeling the expensive MC TBR model $f(x)$, a surro-
 250 gate is a mapping $\hat{f}(x)$ that yields similar images as $f(x)$.
 251 In other words, $f(x)$ and $\hat{f}(x)$ minimise a selected sim-
 252 ilarity metric. Furthermore, in order to be considered
 253 viable, surrogates are required to achieve expected eval-
 254 uation time lower than that of $f(x)$.

255 In the supervised learning setting, we first gather
 256 a sufficiently large training set of samples $\mathcal{T} =$
 257 $\{(x^{(i)}, f(x^{(i)}))\}_{i=1}^N$ to describe the behaviour of $f(x)$
 258 across its domain. Depending on specific model class
 259 and appropriate choice of its hyperparameters, surrogate models $\hat{f}(x)$ are trained to minimise
 260 empirical risk with respect to \mathcal{T} and a model-specific loss function \mathcal{L} , where empirical risk is
 261 defined as $R_{\text{emp.}}(\hat{f} | \mathcal{T}, \mathcal{L}) = \frac{1}{N} \sum_{i=1}^N \mathcal{L}(\hat{f}(x^{(i)}), f(x^{(i)}))$.

262 The unsupervised setting can be viewed as an extension of this method. Rather than fixing
 263 the training set \mathcal{T} for the entire duration of training, multiple sets $\{\mathcal{T}_k\}_{k=0}^K$ are used, such that

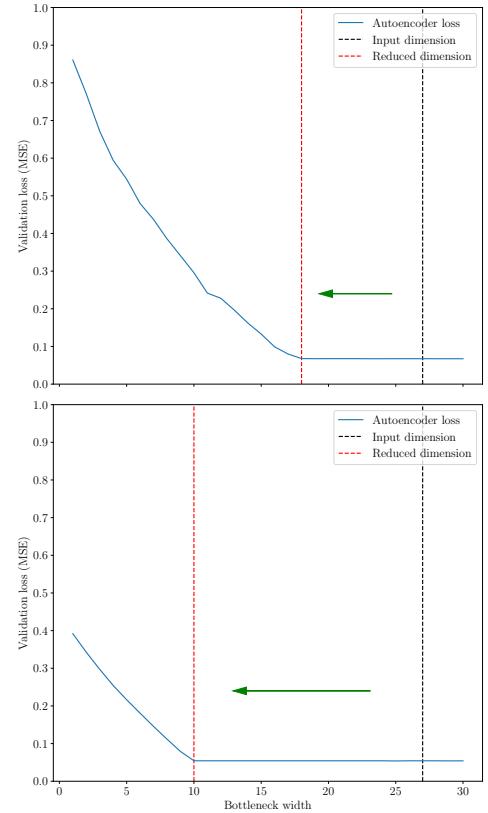


Figure 6: Autoencoder loss scan on the full feature space (top) and a single slice (bottom). Dimensional reduction is indicated by a green arrow.

² 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.

²⁶⁴ $\mathcal{T}_{k-1} \subset \mathcal{T}_k$ for all $k > 1$. The first set \mathcal{T}_0 is initialised randomly to provide a *burn-in*, and
²⁶⁵ is repeatedly extended in epochs, whereby each epoch trains a new surrogate on \mathcal{T}_k using the
²⁶⁶ supervised learning procedure, evaluates its performance, and forms a new set \mathcal{T}_{k+1} by adding
²⁶⁷ more samples to \mathcal{T}_k . This permits the learning algorithm to condition the selection of new
²⁶⁸ samples on the evaluation results in order to maximise improvement of surrogate performance
²⁶⁹ over complex regions within the domain.

²⁷⁰ 3.1 Metrics

²⁷¹ Aiming to provide objective comparison of a diverse set of surrogate model classes, we define a
²⁷² multitude of metrics to be tracked during experiments. Following the motivation of this work,
²⁷³ two desirable properties of surrogates arise: (i) their capability to approximate the TBR MC
²⁷⁴ model well and (ii) their time of evaluation. An ideal surrogate would maximise the former while
²⁷⁵ minimising the latter.

²⁷⁶ Table 4 provides exhaustive listing and description of metrics recorded in the experiments. For
²⁷⁷ regression performance analysis, we include a selection of absolute metrics to assess the ap-
²⁷⁸ proximation capability of surrogates, and set practical bounds on the expected accuracy of their
²⁷⁹ predictions. In addition, we also track relative measures that are better-suited for model compar-
²⁸⁰ ison between works as they maintain invariance with respect to the selected domain and image
²⁸¹ space. For analysis of evaluation time, surrogates are assessed in terms of wall time elapsed
²⁸² during training and prediction. This is motivated by common practical use cases of our work,
²⁸³ where models are trained and used as drop-in replacements for the expensive MC TBR model.
²⁸⁴ Since training set sizes remain to be determined, all times are reported per a single sample. Even
²⁸⁵ though some surrogates support acceleration by means of parallelisation, measures were taken to
²⁸⁶ ensure sequential processing of samples to achieve comparability between considered models.³

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	$1 - \sum_{i=1}^N \left(y^{(i)} - \hat{y}^{(i)} \right)^2 / \sum_{i=1}^N \left(y^{(i)} - \bar{y} \right)^2$	1 [rel.]
Adjusted R^2	$1 - (1 - R^2)(N - 1)/(N - P - 1)$	1 [rel.]
Evaluation time 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	(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.

²⁸⁷ To prevent undesirable bias in results due to training set selection, all metrics are collected
²⁸⁸ in the scheme of k -fold cross-validation with a standard choice of $k = 5$. Herein, a sample

³ The only exception to this are artificial neural networks, which are notoriously slow to train on conventional CPU architectures in serialised setting.

289 set is subdivided into 5 disjoint folds which are repeatedly interpreted as training and testing
 290 sets, maintaining a constant ratio of samples between the two.⁴ In each such interpretation
 291 experiments are repeated, and the overall value of each metric is reported as the mean across all
 292 folds.

293 3.2 Evaluation of Supervised Learning Surrogates

294 In our experiments, we evaluate and compare surrogates in effort to optimise against metrics
 295 described in Section 3.1. To attain meaningful and practically usable results, we require a suffi-
 296 ciently large and diverse pool of surrogate classes to draw from. This is described by the listing
 297 in Table 5. The presented selection of models includes basic techniques suitable for linear re-
 298 gression enhanced by the kernel trick or dimension lifting, methods driven by decision trees,
 299 instance-based learning models, ensemble regressors, randomized algorithms, artificial neural
 300 networks and mathematical approaches developed specifically for the purposes of surrogate mod-
 301 elling. For each of these classes, a state-of-the-art implementation was selected and adapted to
 302 operate with TBR samples.

Surrogate	Acronym	Implementation	Hyperparameters
Support vector machines	SVM	SciKit Learn [9]	3
Gradient boosted trees	GBT	SciKit Learn	11
Extremely randomized trees	ERT	SciKit Learn	7
AdaBoosted decision trees	AB	SciKit Learn	3
Gaussian process regression	GPR	SciKit Learn	2
k nearest neighbours	KNN	SciKit Learn	3
Artificial neural networks	ANN	Keras (TensorFlow) [14]	2
Inverse distance weighting	IDW	SMT [15]	1
Radial basis functions	RBF	SMT	3
Stochastic gradient descent	SGD	SciKit Learn	13
Ridge regression	RR	SciKit Learn	4
Kriging	KRG	SMT	4

Table 5: Considered surrogate model classes.

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

309 3.2.1 Experiments

310 The presented surrogate candidates are evaluated in four experimental cases:

- 311 1. Hyperparameter tuning in simplified domain,

⁴ 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.

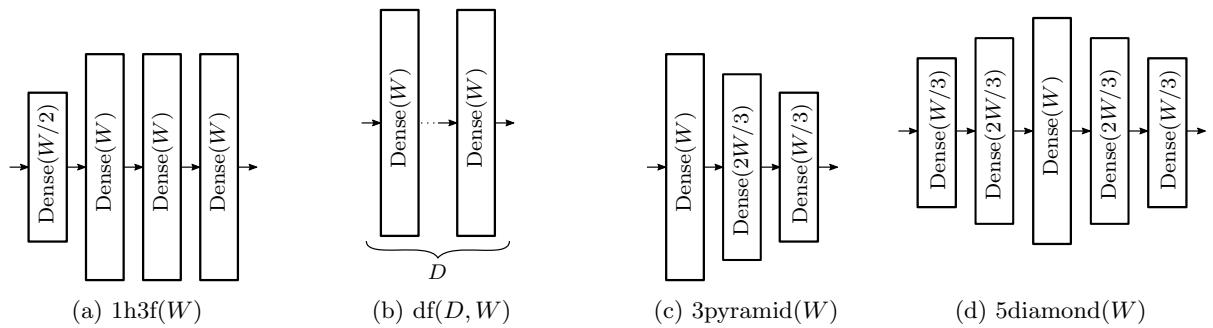


Figure 7: Selected parametric neural network architectures. All layers except the last use ReLU activation. Prediction information flow is indicated by arrows. Connections between neurons are omitted for clarity.

- 312 2. Hyperparameter tuning in full domain,
313 3. Scaling benchmark,
314 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.⁵

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.

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

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

⁵ Tuning of each surrogate class was artificially terminated after 2 days. Instances which reached this limit are marked in the results.

339 3.3 Adaptive Sampling

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

347 Adaptive sampling techniques are widespread in the literature and have been specialised for
 348 surrogate modelling. Garud's [16] "Smart Sampling Algorithm" achieved notable success by
 349 incorporating surrogate quality and crowding distance scoring to identify optimal new samples,
 350 but was only tested on a single-parameter domain. We theorised that a nondeterministic sample
 351 generation approach, built around Markov Chain Monte Carlo methods (MCMC), would fare
 352 better for high-dimensional models by more thoroughly exploring all local optima in the feature
 353 space. MCMC produces a progressive chain of sample points, each drawn according to the
 354 same symmetric proposal distribution from the prior point. These sample points will converge
 355 to a desired posterior distribution, so long as the acceptance probability for these draws has a
 356 particular functional dependence on that posterior value (see [17] for a review).

357 Many researchers have embedded surrogate methods into MCMC strategies for parameter optimi-
 358 sation [18, 19], in particular the ASMO-PODE algorithm [20] which makes use of MCMC-based
 359 adaptive sampling to attain greater surrogate precision around prospective optima. Our novel
 360 approach draws inspiration from ASMO-PODE, but instead uses MCMC to generate samples
 361 which increase surrogate precision throughout the entire parameter space.

362 We designed the Quality-Adaptive Sur-
 363 rogative Sampling algorithm (QASS, Fig-
 364 ure 8) to iteratively increment the train-
 365 ing/test set with sample points which
 366 **maximise surrogate error** and minimise a
 367  crowding distance metric (CDM) [21] in
 368 feature space. On each iteration following
 369 an initial training of the surrogate on N
 370 uniformly random samples, the surrogate
 371 was trained and **AE** calculated. MCMC
 372 was then performed on the error func-
 373 tion generated by performing nearest-
 374 neighbor interpolation on these test er-
 375 ror points. The resultant samples were
 376 culled by 50% according to the CDM, and
 377 then the n highest-error candidates were
 378 selected for reintegration with the train-
 379 ing/test set, beginning another training
 380 epoch. Validation was also performed
 381 during each iteration on independent,
 382 uniformly-random sample sets.

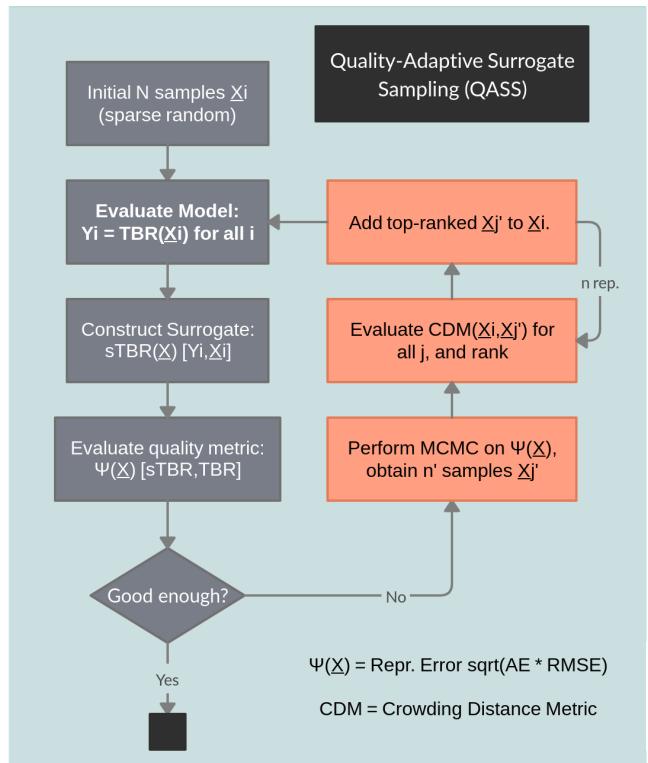


Figure 8: Schematic of QASS algorithm

383 **4 Results**

384 **TODO**

385 **4.1 Evaluation of Supervised Learning Surrogates**

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

390 **4.1.1 Hyperparameter Tuning**

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

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

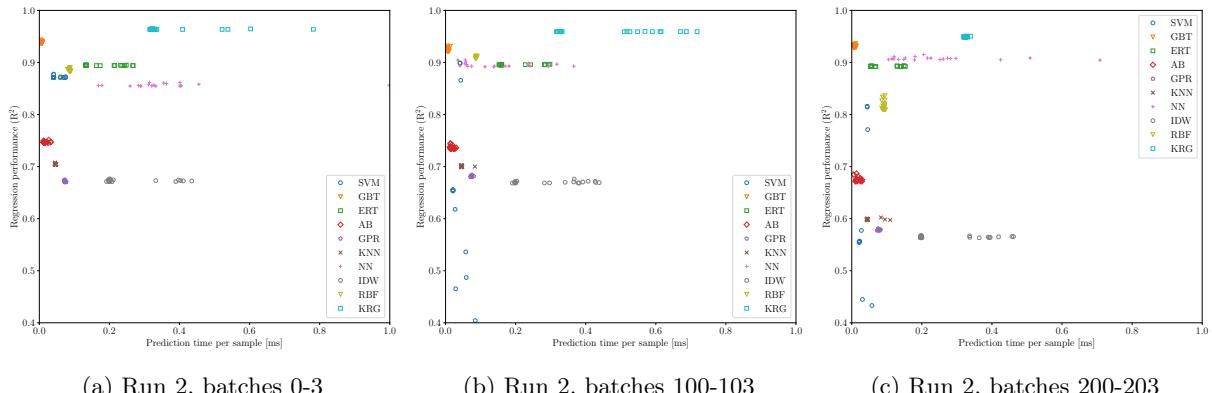


Figure 9: 20 best-performing surrogates per each considered class, plotted in terms of mean prediction time and regression performance (as R^2) on selected slices of run 2, evaluated in experiment 1.

401 TODO: multislice description

402 4.1.2 Scaling Benchmark

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

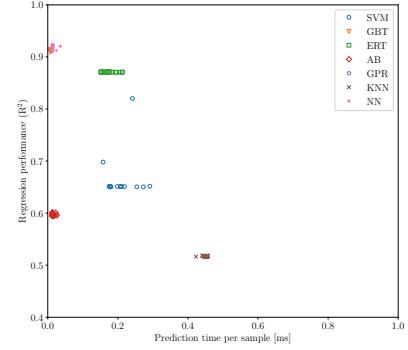


Figure 10: Results of experiment 2, plotted analogously to Figure 9.

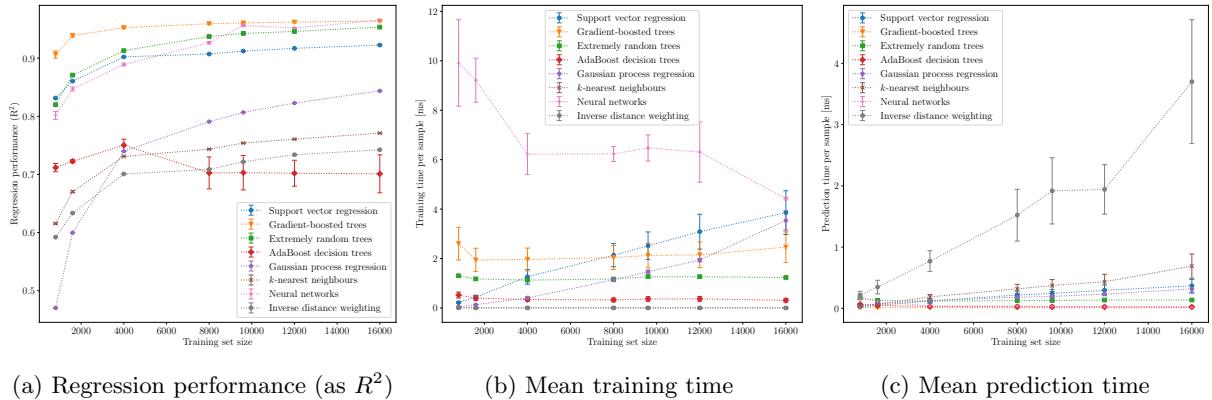


Figure 11: Various metrics collected during experiment 3 (scaling benchmark) displayed as a function of training set size.

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

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

423 4.1.3 Competitive Surrogate Training

424 TODO

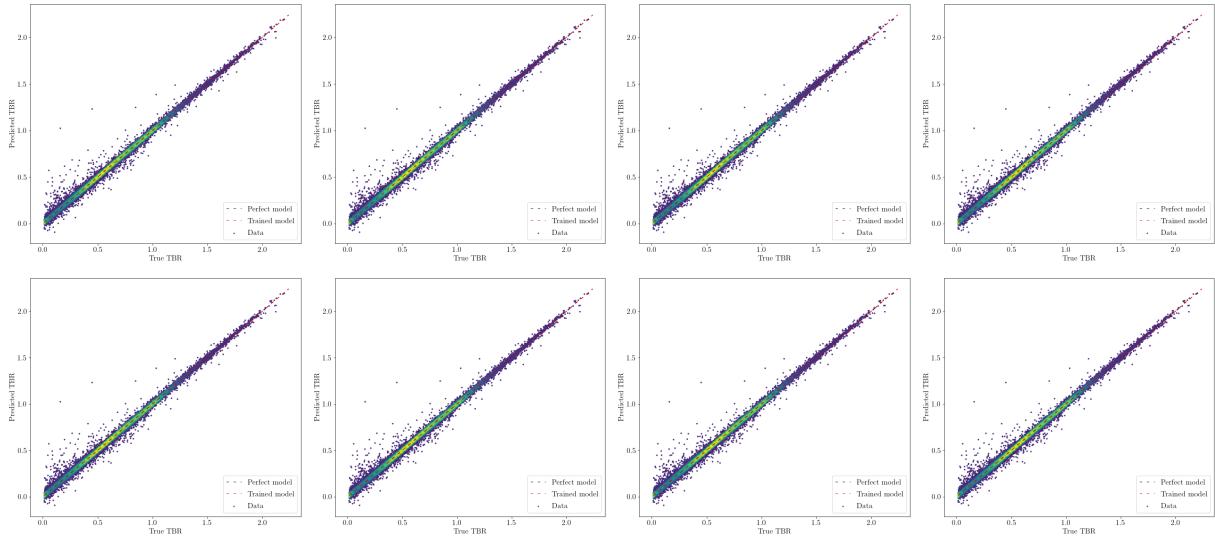


Figure 12: Regression performance of selected best-performing models trained in experiment 4.

4.2 Results of Adaptive Sampling

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 with adjustable wavenumber parameter n :

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

plotted in Figure 13 for $n = 1$ and two continuous parameters C . 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. Although the scope of this project did not include thorough searches of this hyperparameter domain, sufficient runs were made to identify some likely trends.

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 can be limited for increased efficiency. – Awaiting test results on initial sample quantity –. The most complex dynamics arose with the adjustment of sample increment, shown in Figure 14. For each tested initial sample quantity N , the optimal number of step samples was seen to be well-approximated by \sqrt{N} ; the plotted error trends suggest that incremental samples larger than this optimum give slower model improvement on both the training and evaluation sets, and a larger minimum error on the evaluation set. This performance distinction is predicted to be even more significant when

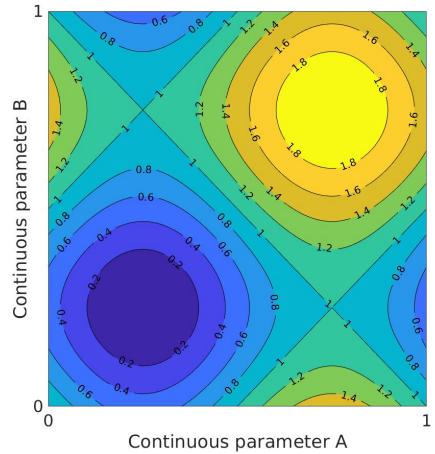


Figure 13: Sinusoidal toy TBR theory over two continuous parameters, wavenumber 1

447 trained on the expensive MC model, where the number of sample evaluations will serve as the primary bottleneck for computation time.

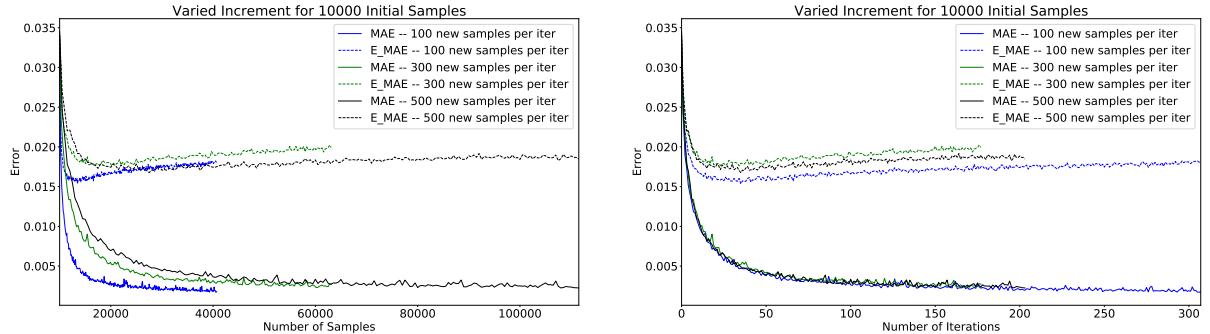


Figure 14: QASS absolute training error over total sample quantity (left) and number of iterations (right). MAE represents surrogate error on the adaptively-sampled training/test set, and E_MAE on the independent evaluation sets.

448
449 The plateau effect in surrogate error on the evaluation set, seen in Figure 14, was universal to all
450 configurations and thought to warrant further investigation. At first this was suspected to be a residual effect of retraining the same ANN instance without adjustment to data normalisation;
451 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
452 to 456 and a baseline scheme with uniformly random incremental samples, shown in Figure 15.

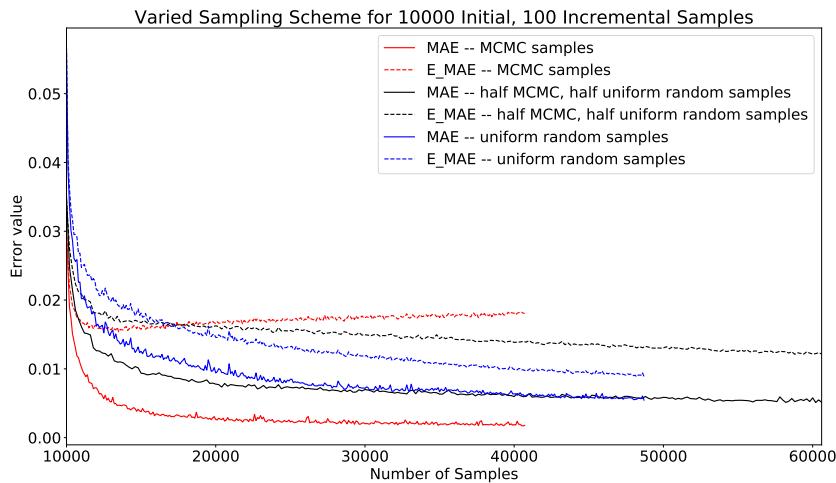


Figure 15: Absolute training error for QASS, baseline scheme, and mixed scheme

457 Such tests revealed that while QASS has unmatched performance on its own adaptively-sampled
458 training set, it is outperformed by the baseline scheme on uniformly random evaluation sets. We
459 suspected that while QASS excels in learning the most strongly peaked regions of the TBR theory,

460 this comes at the expense of precision in broader, smoother regions where uniformly random
461 sampling suffices. Therefore a mixed scheme was implemented, with half MCMC samples and half
462 uniformly random samples incremented on each iteration, which is also shown in Figure 15.

463 **5 Conclusion**

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

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

480 References

- 481 [1] Jacob Søndergaard. “Optimization Using Surrogate Models”. PhD thesis. Technical University of Denmark, 2003. 531
- 482 [2] R.H. Myers and D.C. Montgomery. *Response Surface Methodology: Product and Process Optimization Using Designed Experiments.* 2nd edn New York: John Wiley & Sons, 2002. 535
- 483 [3] Jonathan Shimwell. collaboration. 536
- 484 [4] F.A. Hernández and P. Pereslavtsev. “First principles review of options for tritium breeding and neutron multiplier materials for breeding blankets in fusion reactors”. In: *Fusion Engineering and Design* 137 (Dec. 2018), pp. 243–2561 537
- 485 ISSN: 0920-3796. 542
- 486 [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. 546
- 487 [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. 551
- 488 [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 Transdisciplinarity, Towards New Modeling and Numerical Simulation Paradigms, pp. 90–97. ISSN: 0306-4549. 558
- 489 [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. 564
- 490 [9] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830. 569
- 491 [10] Mohamed Amine Bouhlel and Joaquim Martínez. “Gradient-enhanced kriging for high-dimensional problems”. In: *Engineering with Computers* (Feb. 2018). 573
- 492 [11] Georges Matheron. “Principles of geostatistics”. In: *Economic Geology* 58.8 (Dec. 1963), pp. 1246–1266. ISSN: 0361-0128. 575
- 493 [12] *Kriging Variogram Model*. URL: https://vsp.pnnl.gov/help/Vsample/Kriging%7B%5C_7DVariogram%7B%5C_7DModel.htm (visited on 20/04/2020).
- 494 [13] Jürgen Schmidhuber. “Deep learning in neural networks: An overview”. In: *Neural Networks* 61 (2015), pp. 85–117. ISSN: 0893-6080.
- 495 [14] François Chollet et al. *Keras*. <https://keras.io>. 2015.
- 496 [15] Mohamed Amine Bouhlel et al. “A Python surrogate modeling framework with derivatives”. In: *Advances in Engineering Software* (2019), p. 102662. ISSN: 0965-9978.
- 497 [16] Sushant Garud, Iftekhar Karimi and Markus Kraft. “Smart Sampling Algorithm for Surrogate Model Development”. In: *Computers & Chemical Engineering* 96 (Oct. 2016).
- 498 [17] 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.
- 499 [18] 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.
- 500 [19] 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.
- 501 [20] 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.
- 502 [21] Antti Solonen et al. “Efficient MCMC for Climate Model Parameter Estimation: Parallel Adaptive Chains and Early Rejection”. In: *Bayesian Anal.* 7.3 (2012), pp. 715–736. ISSN: 1936-0975.
- 503 [22] 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.