# Accelerated design of *Escherichia coli* genomes with reduced size using a whole-cell model and machine learning surrogate

Ioana M. Gherman[1], Joshua Rees-Garbutt[1,3], Wei Pang[2,*], Zahraa S. Abdallah[1,*], Thomas E. Gorochowski[3,4,*], Claire S. Grierson[3,4,*], Lucia Marucci[1,4,*]

[1] Department of Engineering Mathematics, University of Bristol
[2] School of Mathematical and Computer Sciences, Heriot-Watt University, Edinburgh, EH14 4AS, UK
[3] School of Biological Sciences, University of Bristol, 24 Tyndall Avenue, Bristol BS8 1TQ, UK
[4] BrisEngBio, School of Chemistry, University of Bristol, Cantock's Close, Bristol BS8 1TS, UK
[*] These authors should be considered joint senior authors

## Abstract

Whole-cell models (WCMs) are multi-scale computational models that aim to accurately simulate the function of all genes and biological processes within a cell. While WCMs offer deeper insights into why cells behave and respond in specific ways, they also require significant computational resources to run, making their development and use challenging. To address this limitation it is possible to use simpler machine learning (ML) surrogates that can learn to approximate specific behaviours of larger and more complex models, while requiring only a fraction of the computational resources. Here, we show how ML surrogates can be trained on WCM outputs to accurately predict whether cells divide successfully when a subset of genes are removed (knocked out). We used these surrogates and a genome-design algorithm to generate a reduced *E. coli* cell *in silico*, where 39% of the genes included in the WCM were removed, a task made possible by our ML surrogate that performs simulations up to 16 times faster than the original WCM. These results demonstrate the value of adopting WCMs and ML surrogates for enabling genome-wide engineering of living cells, offering promising new routes for biologists to understand better how cellular phenotypes emerge from genotypes.

## Introduction

Whole-cell models (WCMs) are comprehensive computational models that aim to capture the dynamics of all core biomolecular processes and components of a cell, as well as their interactions with the wider environment. The development of whole-cell modelling is considered a grand challenge of the 21st century and is seen as particularly useful for understanding the complex relationship between the genotype and phenotype of individual cells [1]. The construction of WCMs has been aided by the rapid growth in the availability of experimental data and high-throughput assays that allow researchers to gain a more holistic understanding of biological processes. There are currently three published WCMs: one for *Mycoplasma genitalium* [2], one for *Escherichia coli* and one for the minimal cell JCVI-syn3A [3]. The *E. coli* WCM is represented mechanistically both at the single-cell [4, 5] and at the population level [6, 7].

Numerous biological discoveries have arisen from the development and utilization of WCMs. For example, Sanghvi et al. used a WCM in conjunction with experimental data to examine the growth rates of *M. genitalium* and make predictions concerning kinetic parameters of specific enzymes [8]. By cross-validating numerous *E. coli* datasets during the construction of the WCM, Macklin et al. identified several inconsistencies in the literature, which upon resolution, enabled the model to accurately predict *in vitro* protein half-lives [4]. Additionally, the *E. coli* WCM has recently been applied to better understand the differences between *in vivo* and *in vitro* measurements of tRNA aminoacylation [9]. The population-level *E. coli* WCM, which considers many independent cells within a shared environment, has been instrumental in assessing antibiotic responses and identifying sub-generationally expressed genes that significantly impact cell survival [7]. Design-build-test-learn cycles can also be optimized using WCMs, with the *M. genitalium* WCM being used to predict minimal genomes *in silico* [10, 11, 12]. Furthermore, the relationship between gene knock outs, cellular subsystems, and phenotypes has been better understood through the application of WCMs [9, 13, 14].

The comprehensive nature of WCMs enables detailed spatial-temporal tracking of cellular processes and components. However, this detail also imparts a substantial computational burden. Simulating a single generation of the *M. genitalium* WCM can take between 5 and 12 hours using a single CPU and 8 GB of RAM, as reported by [10]. In comparison, simulating the doubling time of *E. coli* using the single-cell model only takes approximately 15 minutes on a similar configuration [4]. Despite the *E. coli* WCM being much faster than the *M. genitalium* model for simulation, the combinatorics when studying biological systems still prohibit simulating numerous different genotypes over many generations. The

reason for such long simulation time is caused primarily by the time-series data generated by the WCM for tens of thousands of variables. Nevertheless, in many applications of WCMs, researchers are only interested in a specific subset of these outputs (e.g., the overall cellular mass). ML surrogates represent a solution to address this computational burden by exclusively generating the outputs of interest after being trained on the data generated by a WCM.

ML surrogates, or emulators as they are sometimes known, are black-box models trained to approximate the behaviour of complex and computationally expensive mathematical models [15]. The major advantage of their use is the reduced computational resources required, thus allowing simulation time to fall from hours or days to seconds. ML surrogates are trained using data simulated by mechanistic models, taking as input their initial conditions and/or parameters and predicting some or all of their outputs. The choice of ML algorithm depends on the nature of the problem, the amount and type of available data, and the desired accuracy. ML surrogates have been used to emulate several biomedical mechanistic models [16, 17, 18] and some molecular biology systems [19, 20], enabling efficient exploration of a mechanistic model's parameter space [21, 22], as well as supporting model optimization [23] and sensitivity analysis [20]. To date, ML surrogates have emulated mechanistic models based on Partial Differential Equations (PDEs) [16, 17, 18], with Ordinary Differential Equations (ODE)-based models only used to show how a surrogate can be trained, without a significant improvement in computational time [19, 20].

In this study, we use a random forest-based ML surrogate to simulate reduced genomes (i.e. genomes containing fewer genes) of *E. coli*. Experimental [24, 25, 26] and *in silico* [10, 3] studies have aimed to design minimal genomes for different organisms, removing as many genes as possible, while ensuring the cell remains viable. WCMs have already been used to design minimal genomes *in silico* for *M. genitalium* [10], which was possible due to the small size of the genome (401 genes modelled) and the fact that the cells were simulated for only one generation (division) [10]. In contrast, the *E. coli* WCM used here, includes the functionality of 1,219 genes and can be run for as many cellular divisions as the user chooses. Using a similar approach to [10] for designing an *in silico* reduced genome, we estimate a simulation time of approximately 14 years using one CPU and 8 GB of RAM. In this study, we show how this number could be reduced to approximately 2.6 months by using an ML surrogate. To the best of our knowledge, the present study is the first to show how an ML surrogate can emulate a large and complex biological model, that integrates different mathematical modelling techniques and contains over 10,000 equations, such as the *E. coli* single-cell WCM.

# Results

## Data preparation for the ML surrogate

Before training and using the ML surrogate, it is necessary to perform some data pre-processing on the outputs of the WCM. We need to decide how long the simulations should be run for (e.g., the number of cell divisions) and what features/states from the WCM's output will be used to train the ML surrogate. The ML surrogate presented in this study is trained on data produced by the *E. coli* single-cell WCM and aims to predict whether a cell will divide when a subset of its genes are removed. Preliminary experiments where we simulated cell growth for four generations showed that in some cases the cells divided but also exhibited significant growth defects, especially in protein mass (Figures S.1 and S.2). This suggests that more than four cell divisions/generations should be simulated. The analysis presented in Supplementary Section S.1 shows that when running eight generations simulations, 96% of the time, cells start failing to divide during the first six generations (Figure S.3) and therefore this number of divisions was used when generating the ML surrogate training data.

Additional decisions had to be made regarding the features to be used. Due to the high dimensionality of the WCM (it contains over 10,000 equations and 19,000 parameters), we began by assessing the values of 22 different macromolecular masses (e.g. of protein, RNA, DNA, membrane), that could capture a summary of the cell's state. We tested computationally the accuracy of ML surrogates using the values of these variables at the beginning of the simulations ($t = 0$, generation 1, Table S.1) and at the end of generation 1 just before the first cell division (Table S.3). The results showed an improvement of approximately 16 percentage points in the F1-score of the training set and 18 percentage points for the test set, when using values at the last time step of generation 1 as features. Therefore, for the rest of this study, we will use as features the state of the cell at the last time step before the first division. A summary of the training procedure that will be used in this study can be seen in Figure 1 and the data preparation techniques are summarised in Supplementary Section S.1. An overview of the simulations run to train the ML surrogates is presented in Table S.2.

Upon further investigation, we found that many of the macromolecular masses used as features to train the ML model were highly correlated, suggesting that not all 22 features are necessary to make predictions. Figure S.4 shows the correlation matrix between all the 22 features. These values were used to select only those features with a Pearson correlation $< 0.95$, allowing us to reduce the number of features used to train the ML model to 12. This reduction did not affect the accuracy of the ML surrogate.

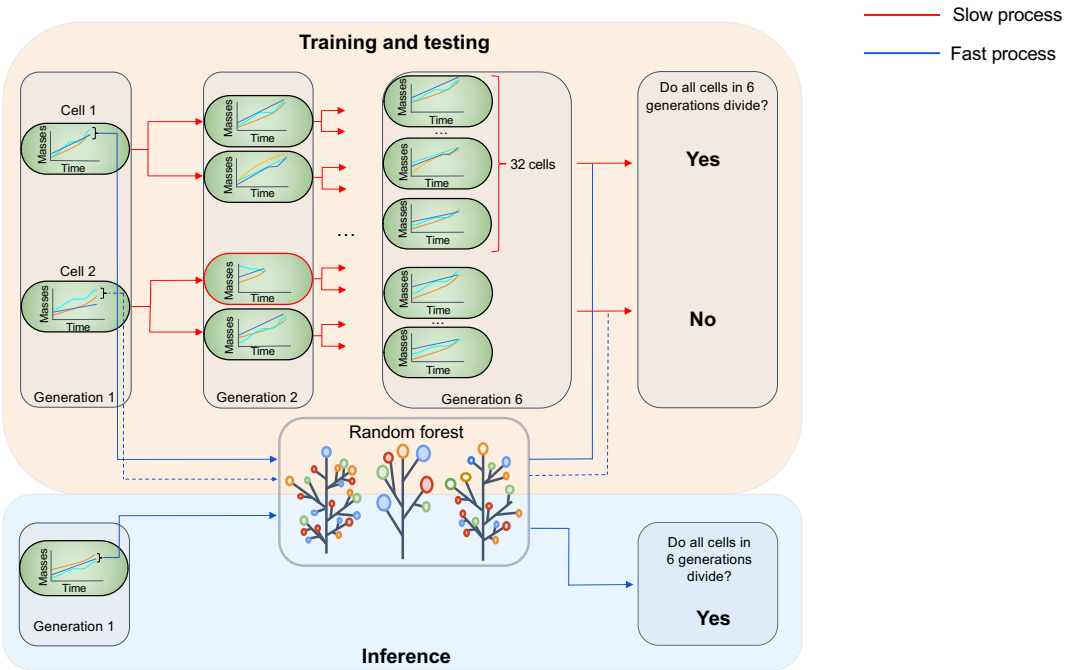


Figure 1: A random forest model is trained to emulate the behaviour of the *E. coli* WCM. To obtain the training data for the ML surrogate, the WCM is run for six generations. Simulating the WCM for six generations is computationally expensive and time consuming (the red arrows in the orange square). Therefore, we use the macromolecular masses at the end of generation one to train a random forest model that predicts whether all the cells in six generations will divide or not. Once the model is trained, for future use, only one generation of the WCM will have to be run (instead of six), and the ML surrogate can be used to predict whether all the cells in six generations divide. This process is significantly faster (the blue arrows in the blue square).

## Training the ML surrogate

We ran 924 single gene knock out simulations (corresponding to the knock out of 462 genes, each run twice), of which 738 were used to train the ML surrogate and 186 to test it. The 12 key macromolecular masses at the point of the first cell division were then used by a random forest classifier to predict whether all cells in 6 generations divide (Figure 1).

A random forest classifier was chosen due to its simplicity, explainability and performance for classification tasks in diverse contexts [27]. To assess the random forest model proposed here we used $k$-fold cross-validation on the training set to tune the model's parameters and avoid overfitting. We also tested the model's performance on a hold-out test set. We obtain an F1-score of 0.92 on the training set, 0.9 when averaging the cross-validation results, and 0.93 on the hold-out test set (Table S.3). The corresponding confusion matrices show that when the ML model predicts that not all cells in 6 generations divide (i.e., the gene knock out impacted viability), that is true for 259 out of 261 (99%) of the data points in the training set, and 74 out of 76 (98%) of the data points in the test set (Figure 2.A). In the

context of minimal genome design, this suggests that it is possible to limit the number of essential genes (i.e. genes that upon removal from the genome cause failure in the division of at least one cell in six generations) that we remove from the genome, guided by the ML surrogate.

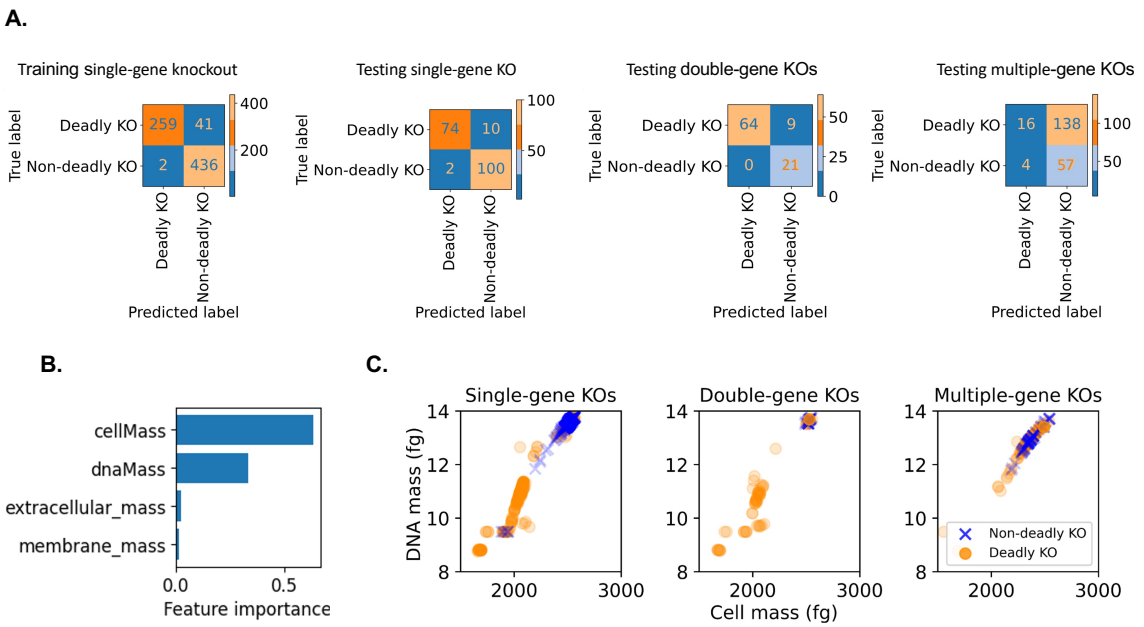


Figure 2: Results of the initial ML surrogate trained on single gene knock out data. (A) The confusion matrices for the training of the ML surrogate on single gene knock outs, and testing on single, double and multiple gene knock outs. The true label is based on the output of the WCM and the predicted label is represented by the output of the ML surrogate. (B) The feature importance as outputted by the random forest algorithm trained on single gene knock outs. (C) Scatter plots of the two main features used for learning, showing different data sets containing single, double and multiple gene knock outs.

Our random forest classifier was trained, validated and tested repeatedly on these selected 12 features and the results showed that only four of them were necessary for making an accurate prediction. One of the advantages of using a random forest classifier is that it also ranks the importance of the features used for a prediction. Figure 2B shows that the most important feature is the cell mass, followed by the DNA mass, the extracellular mass and the membrane mass. More details about the impact of the features on the classifier can be found in Supplementary Section S.2.

This initial random forest surrogate was trained on single gene knock outs and tested on both single and double gene knock outs. The F1-score when testing the surrogate on double gene knock outs was 0.93 and there were no false positives (Figure 2A, Table S.3). However, when searching for a reduced genome, we know that we will have to use the surrogate to predict cell division for multiple gene knock out simulations (i.e., where more than two genes are removed). Therefore, we tested this initial ML surrogate also on multiple gene knock outs and we obtained an F1-score of 0.62 (Figure 2A, Table S.3),

suggesting that the model does not generalise well to this type of simulations. Figure 2C shows a scatter plot of the main two features used by the classifiers (i.e at the end of generation one, just before cell division). We observe that when deleting more genes, the cell and DNA masses become more similar among the deadly and non-deadly knock outs. Specifically, both the DNA and cell mass of the deadly knock outs are higher compared to when we knock out single genes. This suggests that in order to classify multiple gene knock outs, the surrogate has to be retrained using more features and more data representing multiple gene knock outs.

The surrogate was thus retrained by adding to the initial training set of 738 single gene knock outs, 155 additional multiple gene knock outs (each ran once for 6 generations). The F1-score when testing on single and double gene knock outs remained the same, but when testing on the hold-out set of multiple gene knock outs (that were not used for training) the F1-score is 0.89, representing a 0.27 increase compared to the initial surrogate (Table S.3, Figure 3A). This shows that the retrained ML surrogate is able to make accurate predictions for multiple gene knock outs as well, despite dealing with a much more challenging task compared to classifying single gene knock outs. When adding the multiple gene knock outs to the training data, we also reduced the features involved to 11 uncorrelated macromolecular masses; the instantaneous growth rate was removed due to its similarity to the growth output perceived by the random forest. Figure 3B shows that the cell mass, the protein mass, the DNA mass, the tRNA mass and the extracellular mass are important for making accurate predictions.

## Genome reduction

To demonstrate the value of the trained surrogate model, we show that it can be used to design a reduced *E. coli* genome that still functions (i.e. divides) *in silico*. To do this, we used Minesweeper, an algorithm inspired by laboratory fragment-cassette-fraction genome engineering [26] that employs a divide and conquer approach and which we used previously to find minimal genomes using the *M. genitalium* WCM [10]. Minesweeper requires running single gene knock outs and tens of thousands of multiple gene knock out simulations. To do this, we used our trained surrogate models.

Figure 4 shows the framework used to obtain the reduced genome using Minesweeper [10]. This algorithm is trying to identify the genes that can be removed from the genome, without losing the cell's capability to divide successfully in six generations. Stage one of Minesweeper predicts the essentiality of every single gene in the WCM. This was done using the initial random forest surrogate trained only on single gene knock outs for all 1219 genes considered for deletion. This choice was made because at

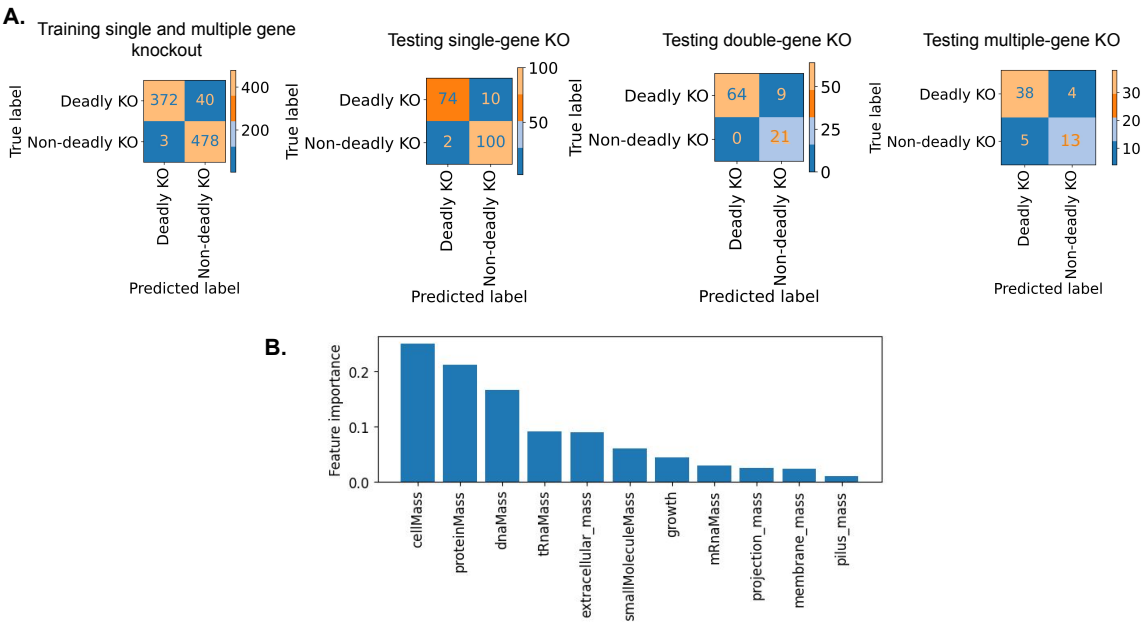


Figure 3: Results of the updated ML surrogate trained on single and multiple gene knock outs data. (A) The confusion matrices for the training and testing of the ML surrogate on single, double and multiple gene knock outs. The true label is based on the output of the WCM and the predicted label is represented by the output of the ML surrogate. (B) The feature importance as outputted by the random forest algorithm trained on single and multiple gene knock outs.

this point we did not have multiple gene knock out simulations and the F1-score of the initial ML surrogate was high for single gene knock outs. The algorithm predicted that 1047 genes are non-essential and 172 genes are essential. Only the 1047 genes predicted as non-essential are considered for future deletions.

In stage two, we create 56 groups containing different percentages of genes to be removed from the genome (this is a modification compared to the previous version of Minesweeper, see Methods for details). The simulations corresponding to the removal of these groups were run using the WCM (simulating six generations). The resulting data was then used to test the initial ML surrogate presented in the previous section and retrain a random forest surrogate using multiple gene knock out data, whose results are summarised in Figure 3. Out of the 56 deletion groups, the removal of 20 of them produced non-deadly cells. These 20 groups correspond to two 12.5% group, four 10% groups and 14 5% groups.

Next, stage three takes the three largest segments that the WCM suggests can be deleted in stage two (two 12.5% groups and one 10% group). It then creates a powerset of combinations obtained from each of these groups and all corresponding remaining non-overlapping segments from stage two, that when deleted, the cell maintained division. We then test whether all cells in six generations divide

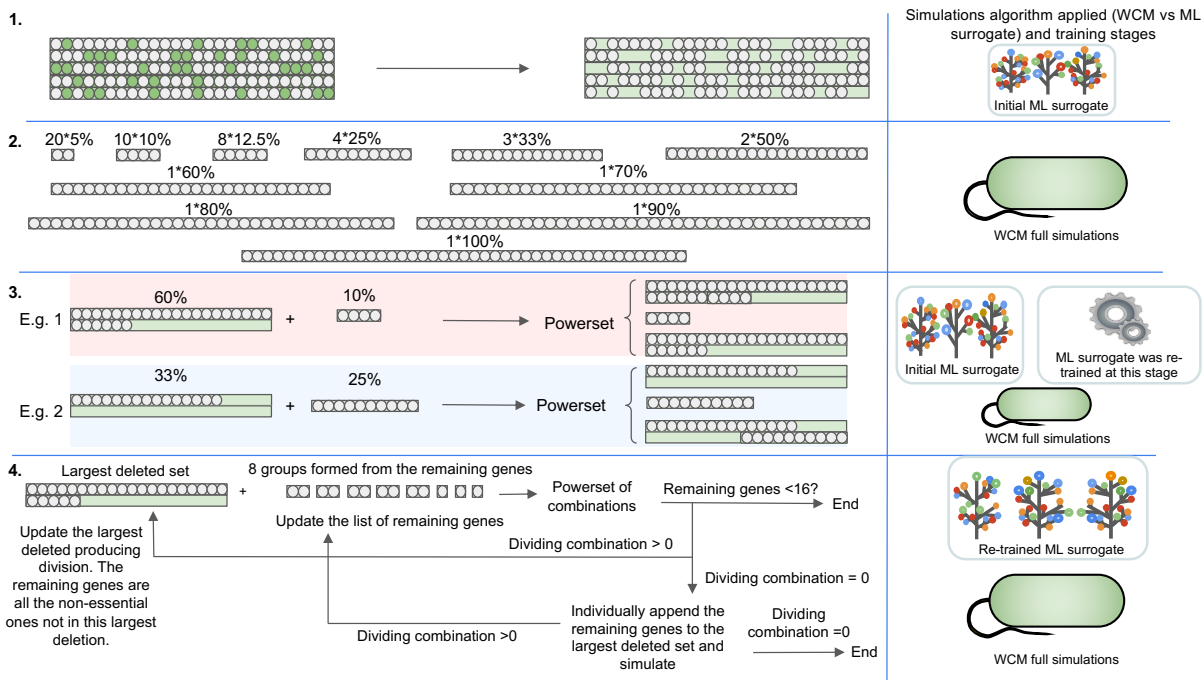


Figure 4: Summary of the framework used to generate the minimal genome. This is an adaptation of the Minesweeper framework [10], using the *E. coli* WCM [4] and the ML surrogates. The process starts by using the ML surrogate trained on single gene knock out data to predict the essentiality of every single gene knock out. Only the genes characterised as non-essential go to stage two (the grey dots). In stage two, 56 groups representing different percentages of the non-essential genes are deleted individually from the genome, and the corresponding simulations are run for 6 generations using the WCM. In stage three, we show examples where we take the two largest deletion segments that when removed produce cells that divide (red and blue) and append all combinations of the non-overlapping segments from stage two. Then we create a powerset of combinations from these segments and run the corresponding simulations where they are removed (not all powerset combinations could be shown here). All these simulations are run using the ML surrogate trained on single gene knock outs and the WCM is used to test the dividing genomes. With the multiple knock out data obtained up to this stage, we re-train the ML surrogate. The fourth stage is cyclical and predictions are made using the retrained ML surrogate, randomly testing the dividing genomes using the six generations WCM simulations.

when these combinations are deleted. These simulations were run using the initial ML surrogate and dividing genomes were tested using the WCM. The resulting data emphasised the need to retrain the ML surrogate using multiple gene knock out simulations. Therefore the data used to retrain the ML surrogate presented in the previous section was generated using simulations coming from stages two and three of Minesweeper. From this stage onwards, only the retrained ML surrogate and the WCM were used to make predictions.

Stage four is cyclical. It starts by taking the largest segment from stage three which when removed, produces dividing cells in six generations, and it forms eight groups from the remaining non-essential genes (not included in the segment). After that, it creates a powerset of combinations between these groups of genes and the current largest deleted segment. If some deletants formed by the removal of

these combinations divide, it updates the largest deletion segment and the remaining genes accordingly. If none of the combinations divide, it appends individual genes to the largest deletion segment that produced dividing cells (rather than combinations of the eight groups). If some of these divide, it updates the list of remaining genes. If none divides, the algorithm stops. We ran this cycle using mainly the ML surrogate, with the WCM used to randomly check if the dividing cells were indeed dividing. These cycles produced several reduced cells with genome sizes varying between 959 genes at the beginning of stage four simulations and 742 genes at the end.

The final and smallest reduced genome, EMine-742, contained 742 genes out of the 1219 modelled by the WCM. This represents a 39% reduction of the WCM's genome in minimal M9 media supplemented with 0.4% glucose. To obtain this result, we simulated 787 genomes running the WCM for 6 generations and over 20,000 genomes running the WCM for one generation and predicting division using the ML surrogates. Assuming that we are running simulations using one CPU on a standard desktop computer and that all cells in six generations would have divided, running all simulations would have taken approximately two years, whilst without the ML surrogate it would have taken approximately 38.4 years. Therefore, the ML surrogate reduces the computational time by 94.8%.

## Reduced *in silico* genome (EMine-742) analysis

### Reduced genome growth analysis

The reduced genome was simulated 50 times for six generations (each dividing cell producing two daughters) using the WCM to assess its stability. The results show that for all repetitions, all cells in six generations divide. We compared the growth and masses of the 50 repetitions of EMine-742 to those of 50 repetitions of wildtype cells (Figure 5). The results show that the average cell mass of the EMine-742 cells is 1385.4 fg and for wildtype cells is 1669.56 fg. Furthermore, the reduced genome cells grow slower and therefore take longer to divide compared to wildtype, which might be caused by the fact that the cells simulated by the WCM need to reach a certain mass before they divide. The RNA and protein mass of the reduced genome are often higher than those of wildtype cells, with the average protein mass over six generations 244.6 fg for EMine-742 and 230 fg for wildtype cells.

To further assess the stability of the EMine-742 genome we also ran 10 generation simulations for 50 repetitions (seeds), where each dividing cell produced only one daughter (to avoid computational burden). The results show that for 96% of these 50 repetitions, all cells in 10 generations divided. In comparison after running 10 generations of wildtype *E. coli* simulations, for 98% of the genomes, all cells in 10

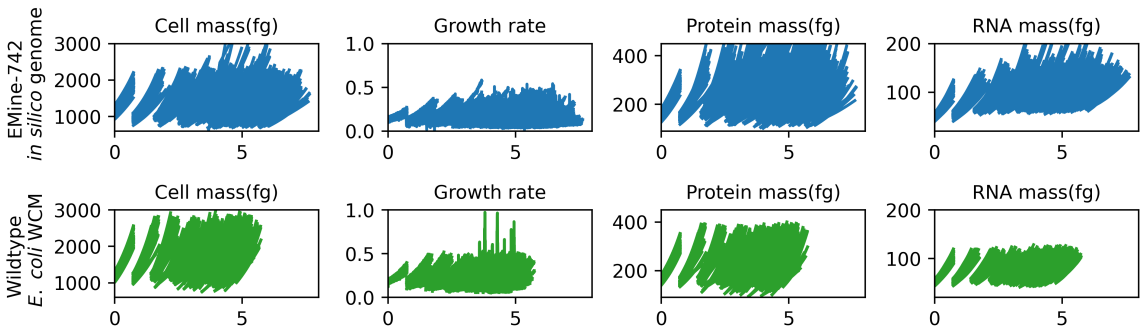


Figure 5: Data from fifty *in silico* repetitions of EMine-742 (first row) and wildtype *E. coli* WCM (second row), run for 6 generations.

generations divided. This suggests that the result is still stable when running 10 generations.

**Reduced genome literature comparison**

The construction of the *E. coli* WCM involved the integration of data from three distinct laboratory strains: K-12 MG1655 (where most of the high-throughput data originated), B/r (used for the growth rate optimisation and detailed composition data) and BW25113 [4]. This approach was necessary due to the insufficient amount of data available from a singular strain for the development of a comprehensive WCM. This multi-strain data integration strategy might potentially constrain the capacity for direct comparison with other reduced *E. coli* genomes that are documented in the scientific literature.

A number of reviews have compared existing laboratory reductions of the *E.coli* genome (primarily in K-12 MG1655 and K-12 W3110 strains) [28, 29, 14], but only three studies tested the reduced MG1655 genomes in minimal media [30, 31, 32]. First, the MDS12 strain built by Kolisnychenko et al. in 2002 deleted 12 genomic segments from MG1655 in minimal media (8.1% deletion) without impacting growth rates [30]. Then Posfai et al. removed additional regions from the genome to obtain the MDS43 strain (15.3% of the genome was removed) [31]. Later, Park et al. removed another 13 genomic regions to obtain the MS56 strain with 23% of the original genome deleted [32]. Since MS56 contained also the gene deletions from MDS12 and MDS43, we will only compare EMine-742 to MS56.

The *E. coli* WCM includes only 15.2% of the genes removed in MS56 (152 genes). From these, 66 were also removed from the *in silico* reduced genome EMine-742. There are 411 genes removed from EMine-742 that are not removed from MS56. Figure 6.A shows that the genes removed *in silico* are more widely spread across the genome, this being one of the advantages of *in silico* approaches that allow a wider search.

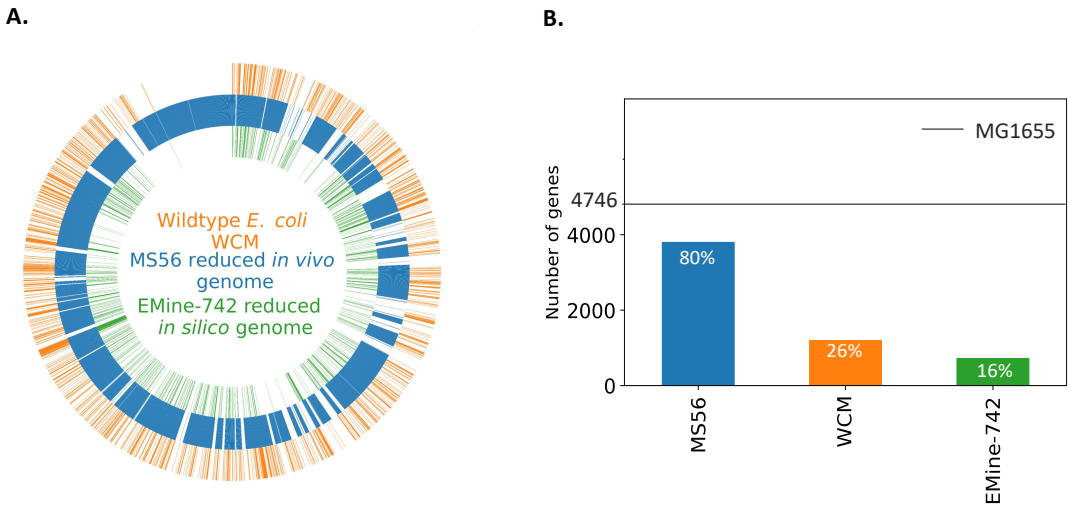


Figure 6: (A) Genes included in the *E. coli* WCM (in orange), the genes included in MS56 [32] (in blue), and the genes included in EMine-742 (in green). (B) Number of genes included in each of the genomes. The percentages are calculated relative to the size of the MG1655 genome.

EMine-742 was further reduced by also removing from it the genes deleted in MS56, that were modelled in the *E. coli* WCM. The simulations corresponding to this genome did not produce dividing cells. This result is expected, since the deletion pathways are different and it is likely that the activity of some genes removed to obtain EMine-742 are complementary to some of the genes removed to obtain MS56, creating synthetic lethal gene knock out combinations.

## Discussion

Our results show that by using an ML-based surrogate to emulate the predictions of the *E. coli* WCM, it is possible to greatly accelerate the search for *in silico* genomes. Specifically, we showed that a search that on a typical desktop PC would take 38 years, can be reduced to two years. In this study, the computational time was further accelerated by using a supercomputer and running simulations in parallel. Here, the ML surrogate model predicts a binary variable (cell division), but we are confident that this approach can be expanded to predict continuous variables such as metabolic yield, or to forecast time series corresponding to some variables of interest. To predict continuous variables, it would be necessary to train regression-based surrogate models, while for time series forecasting, deep learning approaches such as Long Short Term Memory (LSTM) networks could be used. These methods could be useful when using the *E. coli* WCM for metabolic engineering, particularly when many simulations need to be run.

Typically, ML surrogates are trained on initial conditions and/or the parameters of mechanistic models.

In this study, this was not possible due to the high-dimensionality and complexity of the WCM. We would have needed more than 10,000 training examples to avoid having more features than training data. Instead, the surrogates presented here were trained using as features variables that summarise the state of the cells (i.e. macromolecular masses). Moreover, rather than using the initial state of these variables, we chose to use the state at the end of the first generation of each genome simulation, which increased the F1-score of the classifier by 16 percentage points, as shown in the data preparation section. This suggests that there is not enough information regarding the cellular dynamics at the beginning of each simulation to make an accurate prediction. Even with this approach, the number of cells that had to be simulated in six generations was reduced from 63 using only the *E. coli* WCM to one using the ML surrogate (assuming all daughter cells in six generations divide).

The good performance of the surrogate (e.g. F1-score of 0.93 on single gene knock outs test set) suggests that the cell mass, DNA mass, extracellular mass and membrane mass at the end of generation one hold enough information to predict the future viability of the daughter cells for other 5 generations. This is indeed surprising and it would be interesting to be assessed experimentally as well. Regarding the false negative predictions made by the ML surrogate, in most cases, it is likely that they are due to the fact that the decrease in cell and DNA mass is not obvious enough at the end of generation one and therefore the ML surrogate classifies the knock out as non-deadly, when in fact it should be deadly. Given the genes corresponding to these mistakes, it is not clear what causes this behaviour, this being something that could be investigated further in the future. However, including these false negatives in the Minesweeper analysis means that these genes will be considered for deletion instead of being ignored after stage one of Minesweeper. This is not a problem in the context of reducing the size of the genome since it is just increasing the pool of possible gene deletions. In contrast, having a high number of false positives would be more problematic because it would mean that we are decreasing the pool of genes to be deleted when searching for a minimal genome. The ML surrogates presented in this study have a small false positive rate of 0.6-1.9% on single gene knock outs.

The genome obtained by using a combination of six generations of WCM simulations and the ML surrogates was reduced by 39%, from 1219 genes to 742. A direct comparison to the literature *in vivo* results is not possible because the WCM only includes the functionality of a fraction of *E. coli's* MG1655 genome. Despite this limitation, the 411 genes deleted only in the EMine-742 genome (i.e. not removed as part of MS56's genome reduction) should be analysed further and tested experimentally. Mathematical models are not perfect and we are aware of the current limitations of the *E. coli* WCM, therefore we

do not expect that when the genome proposed here is tested experimentally, the cell will remain viable. However, we believe that this data could guide future experimental avenues for obtaining smaller *E. coli* genomes.

The increase in protein and RNA mass of the reduced *in silico* genome is another avenue worth investigating in future experimental and computational studies (Figure 5). One of the main reasons for building reduced genomes is to have organisms that are highly efficient at utilizing the resources available [29]. Consequently, the organism might be able to optimize resource allocation more efficiently toward the synthesis of proteins and RNAs, which could potentially yield heightened expression levels of particular genes. This can therefore cause a greater production of proteins and RNAs per given unit of cellular resources. One avenue for future work that could take advantage of the richness of information outputted by WCMs, could involve analysing their massive time series outputs to understand which genes are being overexpressed.

As more genes are added to the WCM, the number of algorithmic iterations needed to search for reduced genomes increases. This suggests that as WCMs become more powerful and better at matching cellular phenotypes, it is essential to address their computational complexity in order to be able to use them for genome design tasks, making ML surrogates timely and relevant.

## Methods

### The *E. coli* whole-cell model simulations

In this study, we are using the single-cell version of the *E. coli* whole-cell model, which can be thought of as a large system of Ordinary Differential Equations (ODEs) where the state variables of the ODEs correspond to different cellular states, and the differential equations correspond to cellular processes [4]. Simulating a cell is similar to numerical integration of the ODE system. The resources stored in cellular states must be shared with all cellular processes to ensure mass conservation [4]. In addition, the model has two main stochastic components: initializing RNA/protein synthesis and degrading RNA/protein. This is implemented by using a multinomial and Poisson distribution, respectively. The model is implemented in Python, using Cython and is available at https://github.com/CovertLab/wcEcoli.

The model simulates the log phase growth of each cell and it is possible to do so for several generations. In this study, we used the default M9 minimal media supplemented with 0.4% glucose. For each cell simulation, the model outputs 219,207 time series after approximately 15 minutes of running time on a

CPU. Given the large number of experiments needed for this study, the simulations were run in parallel on the University of Bristol's supercomputers BlueCrystal phase 4 and BluePebble, and on Oracle Cloud. The *E. coli* WCM also uses Fireworks, a workflow management tool, to automate the execution of different jobs.

For the scope of this project, simulations were run for six generations, unless during this process a cell was not dividing in which case the simulations were sometimes stopped after four generations. Unless otherwise specified in the results section or in the supplementary material, the simulations were run for one repetition.

### ML surrogate

A machine learning surrogate model is generally used to emulate the behaviour of a mechanistic model, with the added advantage that once trained, it can be used to make predictions several orders of magnitude faster than the original mechanistic model. The process for training such a machine learning model involves running the mechanistic model several times in order to generate the training and test sets. After that, the ML model is trained using this data, and once the accuracy is satisfactory, it can be used to replace the original mechanistic model in future simulations. Some common techniques used to create surrogate models include regression models [20, 33], Gaussian processes [22], support vector machines [33], decision tree-based models [17, 18] and neural networks [18, 16].

Typically, a machine learning surrogate model is trained on the parameters and/or initial conditions of the mechanistic model. However, due to the high number of parameters and variables of the WCM, this was not possible in this study due to the curse of dimensionality. Therefore, the ML surrogate is trained on the final state (just before division) of the first cell simulated for each repetition, and it predicts whether all the cells in that and the future five generations will divide. The advantage of this approach is that once it is trained, the mechanistic WCM has to be run for one generation, rather than six, and the data from this first generation is used to predict the fitness of all the cells in the future five generations.

In this paper we assess the accuracy of the ML surrogate using the F1-score metric, which combines the precision and recall scores of the models. We chose this metric due to its benefits when dealing with unbalanced data. A higher F1-score represents a better model.

### Random forest

Random Forest is a popular supervised machine learning algorithm used for classification and regression problems [34]. It trains a large number of individual decision trees that operate as an ensemble. Each of these trees is trained on a random subset of the training data and features. The final predicted class is the one that was output by the majority of the decision trees. This ensemble-based approach reduces overfitting and aids interpretability since it becomes possible to understand which features contributed to the prediction the most [35], by averaging the gini impurity of the trees. Furthermore, compared to more complex deep learning models, random forest can be trained very quickly and it often has comparable accuracy. The ML surrogate model based on Random Forest was built in Python using the scikit learn package.

### Minesweeper

Minesweeper is a four-stage algorithm inspired by laboratory fragment-cassette-fraction genome engineering [26], which is similar to the divide-and-conquer paradigm. It was originally published in [10] to predict minimal genomes using the *M. genitalium* WCM, and it is freely available on https://github.com/GriersonMarucciLab/Minesweeper. Minesweeper starts by running all the single gene knock outs. In the original model [10], during the second stage, all the predicted non-essential genes are split into 26 segments of sizes 100%, 90%, 80%, 70%, 60%, 50%, 33%, 25% and 12.5%. Since the *E. coli* WCM includes three times more genes, the 12.5% deletion segments were too big. To address this, we added 30 additional segments of 10% and 5%, respectively. During the third stage of the algorithm, the largest three deletions that produced dividing cells in six generations during stage two, are combined with smaller, dividing, non-overlapping segments. Powersets of unique combinations of these segments are generated and simulated. The largest segment that when deleted produced dividing cells from each of the three groups moves forward to stage four, where it is combined with the remaining non-essential genes that are split into eight groups. The powersets of these segments are simulated and if none of the reduced cells divide, the remaining genes are appended individually to the largest deletion segment. The genes that are individually added and do not produce a dividing cell are not considered for further combinations. This stage is cyclical and it stops when there are fewer than 16 genes (in [10] this was eight) to be added or if the deletion of none of the combinations obtained after adding individual genes produced dividing cells. The threshold for stopping stage four was changed from eight to 16 genes to reduce the number of repetitions of this cycle, again due to the larger *E. coli* genome compared to *M.*

*genitalium*.

## Acknowledgements

We would like to thank the Advanced Computing Research Centre (ACRC) at the University of Bristol for access to the BlueCrystal and BluePeble supercomputers. Special thanks to the HPC and RDSF teams of the ACRC, particularly Dianaimh Greene and Callum Wright for their help with BlueCrystal and BluePebble. Furthermore, we would like to thank Oracle Cloud for allowing us to run some of the simulations used in this study on their clusters. Special thanks to Dr. Christopher Woods for helping us to set everything up and to Richard Pitts from Oracle. We would also like to thank the members of the Covert lab at Stanford University that have answered all our questions regarding the installation and use of the *E. coli* WCM, especially Dr. Travis Ahn-Horst and Gwanggyu Sun. This work was funded through the UK's Biotechnology and Biological Sciences Research Council (BB/W012235/1 to L.M. and J.R-G, BB/W013770/1 to I.M.G.) and the UK's Engineering and Physical Sciences Research Council (EP/S01876X/1 to L.M.). I.M.G. received an EPSRC DTP scholarship. T.E.G. was supported by a Turing Fellowship from The Alan Turing Institute under EPSRC grant EP/N510129/1, and a Royal Society University Research Fellowship grant URF\R\221008. L.M., C.S.G. and T.E.G. were supported by BrisEngBio, a UKRI-funded Engineering Biology Research Centre grant BB/W013959/1. The funders had no role in study design, data collection and analysis, and decision to publish or preparation of the manuscript.

## References


[1] Masaru Tomita. Whole-cell simulation: a grand challenge of the 21st century. *TRENDS in Biotechnology*, 19(6):205–210, 2001.

[2] Jonathan R Karr, Jayodita C Sanghvi, Derek N Macklin, Miriam V Gutschow, Jared M Jacobs, Benjamin Bolival Jr, Nacyra Assad-Garcia, John I Glass, and Markus W Covert. A whole-cell computational model predicts phenotype from genotype. *Cell*, 150(2):389–401, 2012.

[3] Zane R Thornburg, David M Bianchi, Troy A Brier, Benjamin R Gilbert, Tyler M Earnest, Marcelo CR Melo, Nataliya Safronova, James P Sáenz, András T Cook, Kim S Wise, et al. Fundamental behaviors emerge from simulations of a living minimal cell. *Cell*, 185(2):345–360, 2022.

[4] Derek N Macklin, Travis A Ahn-Horst, Heejo Choi, Nicholas A Ruggero, Javier Carrera, John C Mason, Gwanggyu Sun, Eran Agmon, Mialy M DeFelice, Inbal Maayan, et al. Simultaneous cross-evaluation of heterogeneous e. coli datasets via mechanistic simulation. *Science*, 369(6502):eaav3751, 2020.

[5] Travis A Ahn-Horst, Luis Santiago Mille, Gwanggyu Sun, Jerry H Morrison, and Markus W Covert. An expanded whole-cell model of e. coli links cellular physiology with mechanisms of growth rate control. *NPJ Systems Biology and Applications*, 8(1):30, 2022.

[6] Eran Agmon, Ryan K Spangler, Christopher J Skalnik, William Poole, Shayn M Peirce, Jerry H Morrison, and Markus W Covert. Vivarium: an interface and engine for integrative multiscale modeling in computational biology. *Bioinformatics*, 38(7):1972–1979, 2022.

[7] Christopher J Skalnik, Sean Y Cheah, Mica Y Yang, Mattheus B Wolff, Ryan K Spangler, Lee Talman, Jerry H Morrison, Shayn M Peirce, Eran Agmon, and Markus W Covert. Whole-cell modeling of e. coli colonies enables quantification of single-cell heterogeneity in antibiotic responses. *PLOS Computational Biology*, 19(6):e1011232, 2023.

[8] Jayodita C Sanghvi, Sergi Regot, Silvia Carrasco, Jonathan R Karr, Miriam V Gutschow, Benjamin Bolival Jr, and Markus W Covert. Accelerated discovery via a whole-cell model. *Nature methods*, 10(12):1192–1195, 2013.

[9] Heejo Choi and Markus W Covert. Whole-cell modeling of E. coli confirms that in vitro tRNA aminoacylation measurements are insufficient to support cell growth and predicts a positive feedback mechanism regulating arginine biosynthesis. *Nucleic Acids Research*, 05 2023. gkad435.

[10] Joshua Rees-Garbutt, Oliver Chalkley, Sophie Landon, Oliver Purcell, Lucia Marucci, and Claire Grierson. Designing minimal genomes using whole-cell models. *Nature communications*, 11(1):836, 2020.

[11] Joshua Rees-Garbutt, Oliver Chalkley, Claire Grierson, and Lucia Marucci. Minimal genome design algorithms using whole-cell models. *Computational Methods in Synthetic Biology*, pages 183–198, 2021.

[12] Lucia Marucci, Matteo Barberis, Jonathan Karr, Oliver Ray, Paul R Race, Miguel de Souza Andrade, Claire Grierson, Stefan Andreas Hoffmann, Sophie Landon, Elibio Rech, et al. Computer-aided whole-cell design: taking a holistic approach by integrating synthetic with systems biology. *Frontiers in Bioengineering and Biotechnology*, 8:942, 2020.

[13] Sophie Landon, Oliver Chalkley, Gus Breese, Claire Grierson, and Lucia Marucci. Understanding metabolic flux behaviour in whole-cell model output. *Frontiers in molecular biosciences*, page 1114, 2021.

[14] Sophie Landon, Joshua Rees-Garbutt, Lucia Marucci, and Claire Grierson. Genome-driven cell engineering review: in vivo and in silico metabolic and genome engineering. *Essays in biochemistry*, 63(2):267–284, 2019.

[15] Ioana M Gherman, Zahraa S Abdallah, Wei Pang, Thomas E Gorochowski, Claire S Grierson, and Lucia Marucci. Bridging the gap between mechanistic biological models and machine learning surrogates. *PLoS Computational Biology*, 19(4):e1010988, 2023.

[16] Liang Liang, Minliang Liu, Caitlin Martin, and Wei Sun. A machine learning approach as a surrogate of finite element analysis–based inverse method to estimate the zero-pressure geometry of human thoracic aorta. *International journal for numerical methods in biomedical engineering*, 34(8):e3103, 2018.

[17] Yaghoub Dabiri, Alex Van der Velden, Kevin L Sack, Jenny S Choy, Ghassan S Kassab, and Julius M Guccione. Prediction of left ventricular mechanics using machine learning. *Frontiers in physics*, 7:117, 2019.

[18] Li Cai, Lei Ren, Yongheng Wang, Wenxian Xie, Guangyu Zhu, and Hao Gao. Surrogate models based on machine learning methods for parameter estimation of left ventricular myocardium. *Royal Society open science*, 8(1):201121, 2021.

[19] Shangying Wang, Kai Fan, Nan Luo, Yangxiaolu Cao, Feilun Wu, Carolyn Zhang, Katherine A Heller, and Lingchong You. Massive computational acceleration by using neural networks to emulate mechanism-based biological models. *Nature communications*, 10(1):1–9, 2019.

[20] Marissa Renardy, Tau-Mu Yi, Dongbin Xiu, and Ching-Shan Chou. Parameter uncertainty quantification using surrogate models applied to a spatial model of yeast mating polarization. *PLoS computational biology*, 14(5):e1006181, 2018.

[21] Paolo Di Achille, Ahmed Harouni, Svyatoslav Khamzin, Olga Solovyova, John J Rice, and Viatcheslav Gurev. Gaussian process regressions for inverse problems and parameter searches in models of ventricular mechanics. *Frontiers in physiology*, 9:1002, 2018.

[22] Vinny Davies, Umberto Noè, Alan Lazarus, Hao Gao, Benn Macdonald, Colin Berry, Xiaoyu Luo, and Dirk Husmeier. Fast parameter inference in a biomechanical model of the left ventricle by using statistical emulation. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 68(5):1555–1576, 2019.

[23] Nestor V Queipo, Raphael T Haftka, Wei Shyy, Tushar Goel, Rajkumar Vaidyanathan, and P Kevin Tucker. Surrogate-based analysis and optimization. *Progress in aerospace sciences*, 41(1):1–28, 2005.

[24] Yumi Iwadate, Hirofumi Honda, Haruhiko Sato, Masayuki Hashimoto, and Jun-ichi Kato. Oxidative stress sensitivity of engineered escherichia coli cells with a reduced genome. *FEMS microbiology letters*, 322(1):25–33, 2011.

[25] Yasutaka Hirokawa, Hiroaki Kawano, Kimie Tanaka-Masuda, Noriko Nakamura, Akira Nakagawa, Mikito Ito, Hideo Mori, Taku Oshima, and Naotake Ogasawara. Genetic manipulations restored the growth fitness of reduced-genome escherichia coli. *Journal of bioscience and bioengineering*, 116(1):52–58, 2013.

[26] Clyde A Hutchison III, Ray-Yuan Chuang, Vladimir N Noskov, Nacyra Assad-Garcia, Thomas J Deerinck, Mark H Ellisman, John Gill, Krishna Kannan, Bogumil J Karas, Li Ma, et al. Design and synthesis of a minimal bacterial genome. *Science*, 351(6280):aad6253, 2016.

[27] Jehad Ali, Rehanullah Khan, Nasir Ahmad, and Imran Maqsood. Random forests and decision trees. *International Journal of Computer Science Issues (IJCSI)*, 9(5):272, 2012.

[28] Masaomi Kurokawa and Bei-Wen Ying. Experimental challenges for reduced genomes: The cell model escherichia coli. *Microorganisms*, 8(1):3, 2019.

[29] Donghui Choe, Suhyung Cho, Sun Chang Kim, and Byung-Kwan Cho. Minimal genome: Worthwhile or worthless efforts toward being smaller? *Biotechnology journal*, 11(2):199–211, 2016.

[30] Vitaliy Kolisnychenko, Guy Plunkett, Christopher D Herring, Tamás Fehér, János Pósfai, Frederick R Blattner, and György Pósfai. Engineering a reduced escherichia coli genome. *Genome research*, 12(4):640–647, 2002.

[31] Gyorgy Posfai, Guy Plunkett III, Tamás Fehér, David Frisch, Gunther M Keil, Kinga Umenhoffer, Vitaliy Kolisnychenko, Buffy Stahl, Shamik S Sharma, Monika De Arruda, et al. Emergent properties of reduced-genome escherichia coli. *science*, 312(5776):1044–1046, 2006.

[32] Myung Keun Park, Sang Hee Lee, Kyung Seok Yang, Suk-Chae Jung, Jun Hyoung Lee, and Sun Chang Kim. Enhancing recombinant protein production with an escherichia coli host strain lacking insertion sequences. *Applied microbiology and biotechnology*, 98:6701–6713, 2014.

[33] W Andrew Pruett and Robert L Hester. The creation of surrogate models for fast estimation of complex model outcomes. *PloS one*, 11(6):e0156574, 2016.

[34] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995.

[35] Gérard Biau and Erwan Scornet. A random forest guided tour. *Test*, 25:197–227, 2016.