AMSI VACATION RESEARCH SCHOLARSHIPS 2022–23

SET YOUR SIGHTS ON RESEARCH THIS SUMMER

Deep Learning Surrogate for Fast and Accurate Bayesian Inference of Complex Mathematical Models

Dang Khuong Tran

Supervised by Dr Christopher Drovandi & Dr Simon Denman

Queensland University of Technology

AMSI AUSTRALIAN MATHEMATICAL SCIENCES INSTITUTE

1 Abstract

Solving inverse problems is a crucial process in a wide range of applications, from understanding scientific phenomena to building effective models for prediction. This class of problem involves inferring parameters of a deterministic model from observed data. Bayesian inference is a powerful tool for solving inverse problems using stochastic processes where the parameters are random variables from a probabilistic distribution, called the posterior distribution. The Markov chain Monte Carlo (MCMC) method provides a way to find the posterior distribution numerically. However, MCMC algorithms can become impractical if the model is computationally expensive due to its requiring many evaluations of the aforementioned model. To address this challenge, a deep neural network (DNN) is proposed as a surrogate model to help reduce the computational cost of Bayesian inference. In this research, we inferred the parameters of a model of nitrogen mineralisation in soil from the Algricultural Production Systems Simulator (APSIM), using simulated data with artificial errors. The initial samples used for training the DNN were generated from a fast but approximate called component-wise iterative ensemble Kalman inversion method. We find that by using a well-trained surrogate model, the MCMC algorithm was able to achieve similar predictive performance to an exact Bayesian method, namely sequential Monte Carlo (SMC), a popular inference method, at significantly reduced computational cost.

2 Introduction

By definition, "simulation is the use of a model to develop a conclusion that provides insight on the behaviour of any real-world elements" (McHaney, 1991). With the rapid growth of computing power, the world has seen the development and application of increasingly complex simulations in multiple aspects of life, for example:

1. Weather forecasting: Meteorologists use computer simulations to predict the weather by considering factors such as temperature, humidity, wind speed, and pressure (Hollis and Kariko, 2006). Such simulations help provide more accurate weather forecasts, allowing better preparation for severe weather conditions.

2. Traffic management: Transportation engineers use computer simulations to model the flow of traffic (Ramamohanarao et al., 2016). This allows them to optimise traffic lights and road layouts.

3. Medical imaging: Doctors use medical imaging simulations to model the behaviour of certain biomedical products in the body (Viceconti et al., 2016). This helps to better understand the effects of these products, allowing faster development and regulation.

In each case, it is clear that computer simulations facilitate accurate predictions and improve our understanding of complex systems.

To create a simulation that resembles the real world as closely as possible, it is important to find a suitable set of parameters that allows the model to fit experimental data. This class of problem is called an inverse problem. An inverse problem involves finding the underlying parameters of the model, based on the observed data.

A major challenge introduced with inverse problems is the fact that no model can perfectly represent reality as there is always a degree of error and uncertainty associated with them. This error can stem from multiple sources including, but not limited, to measurement errors, limitations in the data used to build the model, and inherent complexities of the target system.

One powerful method for solving inverse problems, taking into account the uncertainty, is Bayesian inference. This is a statistical approach that treats the model parameters and the error as random variables, and it uses prior knowledge about the model along with observed data to make predictions and update the beliefs about the system being modelled. While the analytical form of the posterior is not always known, it is possible to sample from this distribution using a numerical method, such as one of the Markov chain Monte Carlo (MCMC, Robert and Casella, 2004) algorithms. However, these methods generally involve numerous evaluations of the model, hence inferring computationally expensive models becomes impractical.

In this research, we propose the use of a deep neural network (DNN) as a surrogate model for Bayesian inference, as inspired by Angione et al., 2022 and Chen, 2022, allowing significantly reduced computational cost. DNNs have been found to be a universal approximator, meaning they are capable of approximating any function to a certain degree of accuracy (Hornik et al., 1989). By using a DNN as a surrogate, it is possible to solve the inverse problem more efficiently, while still getting accurate results compared to traditional models.

In this work, a surrogate was created for the Agricultural Production Systems Simulator (APSIM, Holzworth et al., 2014), a model capable of predicting nitrogen mineralisation in soil. While this model has numerous parameters with most fixed at measured constants (Probert et al., 1998), only five chosen parameters were used as inputs for the surrogates, and later inferred using the Metropolis-Hasting (Hastings, 1970) algorithm, one of the simplest implementations of MCMC. The data used for inference is simulated from APSIM with added artificial error to mimic experimental data.

To evaluate the effectiveness of this method, we visually compare the posterior distributions from the Metropolis-Hasting algorithm on the surrogate against those from the sequential Monte Carlo (SMC, Del et al., 2006) method on the original model. Additionally, the predictive performance of this method is also evaluated against that of SMC. Here, SMC solves the inverse problem exactly, in that it does not rely on the surrogate, and is more computationally expensive.

3 Statement of Ownership

The contributors to this project includes:

- Dan Tran designed the deep neural network architecture and implemented it using the Tensorflow (Python) library; collected training data from CW-EIKI and performed data cleaning; implemented Metropolis-Hasting using the surrogate and produced all outputs using Python; produced all results and the discussion presented in this report.
- Dr Christopher Drovandi supervised the project; suggested the use of a Deep Learning Surrogate for

Bayesian Inference; provided parts of the code to interact with the original APSIM model in MATLAB, as well as to perform SMC, which hosts a Metropolis-Hasting algorithm.

- Dr Simon Denman supervised the project; provided reviews and clarification in building the Deep Learning Surrogate.
- Imke Botha provided the posteriors from the component-wise iterative ensemble Kalman inversion (CW-IEKI) and SMC.

4 Case Study: Predicting Nitrogen Mineralisation

4.1 Problem Definition

The method developed in this work is applied to APSIM's prediction on nitrogen mineralisation in soil, with an error defined by Vilas et al., 2021 as:

$$y^r_{t_j} \sim \mathcal{N}(x_{t_j}, (\zeta^r_{t_j})^2 + \sigma^2), \quad x_{t_j} = G(\theta, t_j), \tag{1}$$

for $j = 1, ..., T$ and $r = 1, ..., R$, where T is the number of time points, and R is the number of replicates per time point. The function $G(\theta, t_j)$ represents the deterministic prediction of APSIM, version 7.10 (Holzworth et al., 2014) at time point t_j for a set of parameters θ. Note that while $G(\cdot)$ has numerous parameters with most fixed at measured values (Probert et al., 1998), θ represents a considered subset of these parameters. We consider five parameters: θ = (fbiom, finert, ef_biom = ef_hum, rd_biom and rd_hum).

For each measurement, the error is assumed to be Gaussian, characterised by the variance $(\zeta^r_{t_j})^2 + \sigma^2$. This error structure is composed of a known term $\zeta^r_{t_j}$ - the standard deviation characterising the measurement error, which is set to 4% of the observation $y^r_{t_j}$; and an unknown term σ - the standard deviation for error of unknown origin. Under this assumption, σ is treated as an additional parameter to be estimated along with θ.

The model is fitted to a synthetic dataset, simulated using θ = (fbiom, finert, ef_biom = ef_hum, rd_biom and rd_hum) = (0.1, 0.6, 0.3, 0.0025, 0.0005) and $\sigma = 8$, with $T = 301$ and $R = 4$. To allow stochastic simulation, the known portion of the error $\zeta^r_{t_j}$ was set to 4% of the deterministic prediction at time point t_j (Botha et al., 2022), resulting in:

$$y^{*,r}_{t_j} \sim \mathcal{N}(x_{t_j}, (0.04 \cdot x_{t_j})^2 + \sigma^2). \tag{2}$$

We denote the truncated univariate normal distribution as $\mathcal{N}(x|\mu, \sigma^2, a, b)$ where (μ, σ) are the means and standard deviation, and (a, b) are the lower and upper bounds. The assumed prior distributions for each of the parameters are $\mathcal{N}(\text{fbiom}|0.093, 0.025^2, 0.05, 0.15)$, $\mathcal{N}(\text{fbiom}|0.58, 0.1^2, 0.4, 0.8)$, Uniform(ef_biom = ef_hum|0, 1), Uniform(rd_biom|0.001, 0.01) and Uniform(rd_hum|0, 0.0001) and Uniform(σ|0, 20) (Botha et al., 2022).

4.2 Deep Neural Network Surrogate

The major challenge faced by numerical Bayesian inference methods is the requirement for many evaluations of the model $G(\cdot)$, making such methods impractical when $G(\cdot)$ is computationally expensive to evaluate. This is especially problematic when researching and experimenting with different numerical Bayesian inference methods and different versions of the model. We seek to take advantage of the data produced by these methods to train a deep learning approximator, allowing more efficient future optimisation.

4.2.1 Dataset and Transformations

For the DNN to sufficiently approximate the model, it is essential to have an appropriate dataset to train on. We obtain the data from the component-wise iterative ensemble Kalman inversion (CW-IEKI, Botha et al., 2022). An ensemble Kalman inversion algorithm simulates an initial ensemble from the prior, and iteratively updates this to approximate the posterior (Iglesias et al., 2012). By extracting data by evaluating the ensemble for each iteration, we possess sufficient training data. In this case study, the CW-IEKI was evaluated with 1,000 particles per iteration for a total of 11 iterations to produce a total of 11,000 samples of model parameters. Note that since APSIM is a time-series simulation, we collected 7 model predictions at 7 time points for each set of parameters.

To ensure the neural network can generalise well to unseen data, the 11,000 data points were randomly split into a training set (80%), a validation set (10%), and a test set (10%). Before training the neural network, the data was also standardised to ensure that all inputs were on the same scale.

4.2.2 Model Architecture

The neural network is designed to take in 6 input parameters, including 5 model parameters with an added time dimension, to produce a single cumulative nitrogen mineralisation output. The network consist of 3 hidden fully-connected (also called 'dense') layers, each containing 256 neurons and using the hyperbolic tangent (tanh) activation function. While this architecture was not optimised via a neural architecture search or similar, it was found to be sufficient for the task at hand. Figure 1 illustrates the architecture of this deep neural network.

To achieve the regression task, the neural network was trained with a Root Mean Squared Error (RMSE) loss function, defined as the standard deviation of prediction errors:

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{N}(G(\theta, t_j) - \hat{G}(\theta, t_j))^2}{N}}, \tag{3}$$

where $G(\theta, t_j)$ is the original model's prediction for parameters θ and time t_j and $\hat{G}(\theta, t_j)$ is the neural network's prediction for the same parameters and time. For training, the model that best minimised the RMSE of the validation data set was chosen, and the model was set trained until the validation loss did not improve for 25 epochs.

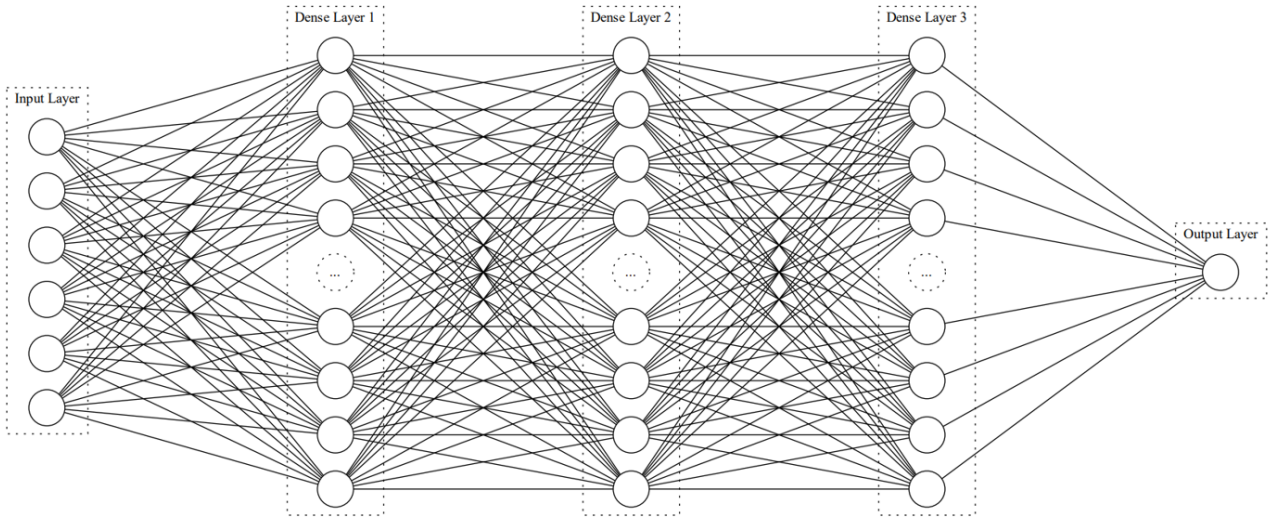


Figure 1: Deep Neural Network Architecture for the surrogate model. Each hidden layer consists of 256 neurons and uses the hyperbolic tangent (tanh) activation function.

4.3 Metropolis-Hasting MCMC

We use the Metropolis-Hasting MCMC algorithm (shown in Algorithm 1, Hastings, 1970) to perform Bayesian inference using the surrogate model. The initial samples used by this algorithm are the 1,000 samples generated from the final iteration of CW-IEKI (Botha et al., 2022) instead of from the prior, as we seek to reduce the number of iterations required until convergence.

4.4 Results

4.4.1 Surrogate Accuracy

Figure 2 shows the surrogate model performance by comparing the surrogate model predictions with the original model predictions. We observe that the surrogate model was able to replicate APSIM's behavour almost perfectly, except for two data points at the higher end, and achieved an overall RMSE of 0.53. These exceptions are considered insignificant due to there being very few parameter samples in that region, meaning these parameter samples were discarded by CW-EIKI in an early iteration.

4.4.2 Inference of Model Parameters

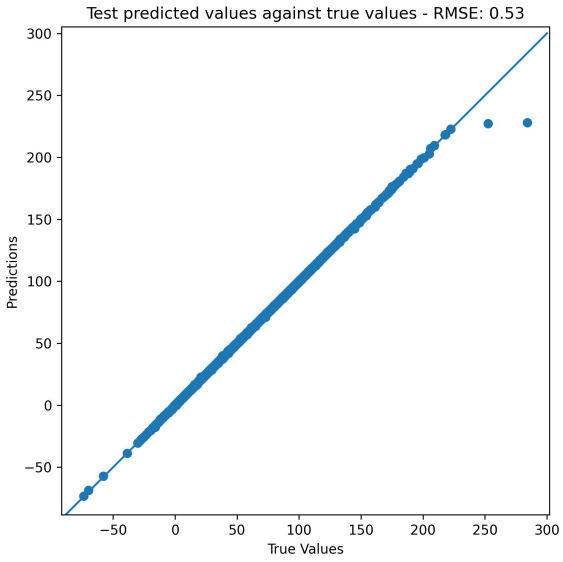
Figure 3 shows the marginal posterior density plots of the six parameters after 5000 iterations of Metropolis-Hasting MCMC replacing true model predictions with the DNN surrogate (simply referred to as MCMC hereafter). The parameters (fbiom, finert and rd_hum) see little to no significant deviation from CW-IEKI since this method already produced similar posteriors to SMC. On the other hand, (ef_biom = ef_hum, rd_biom, σ) show clear signs of convergence to SMC, thus producing a more accurate approximation than CW-IEKI. However, we also observe that MCMC could not replicate the bimodality of ef_biom = ef_hum.

Algorithm 1 Metropolis-Hastings Algorithm

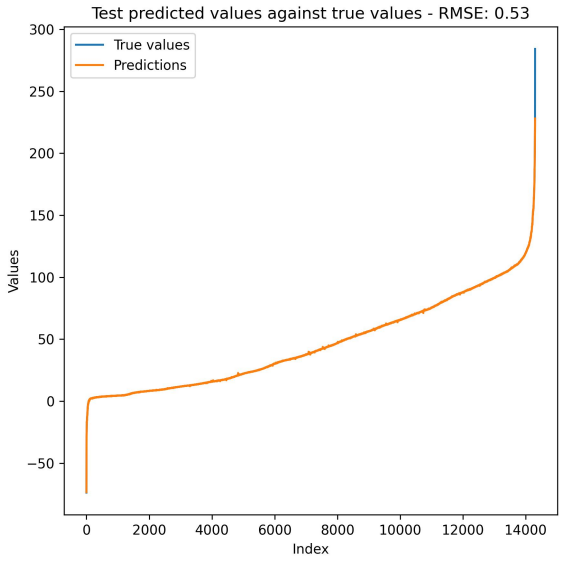
Require: data y, initial ensembles $\theta_0^{1:N}$, number of iterations M

Ensure: N samples from approximate posterior distribution $\hat{p}(\theta|y)$

```
Calculate the covariance matrix Σ from θ_0^{1:N}
for m ← 1 to M do
    for n ← 1 to N do
        Sample θ_m^{n,*} from q(·|θ_m^n) = N(θ_m^n, Σ)
        Calculate the acceptance probability α = min(1, [p(y|θ_m^{n,*}) / p(y|θ_m^n)] [p(θ^{n,*}) / p(θ_m^n)])
        Draw a random number u ~ Uniform(0, 1)
        if α < u then
            Set θ_m^n = θ_m^{n,*}
        end if
    end for
end for
return θ_M^{1:N}
```



(a) Comparison of surrogate predictions against original model predictions plot. The more the data points line along the diagonal line, the better the surrogate model performs.



(b) Surrogate predictions and original predictions plot for data points by index. The better the two predictions lines line up, the better the surrogate model is performing.

Figure 2: Surrogate model performance in approximating the original APSIM model.

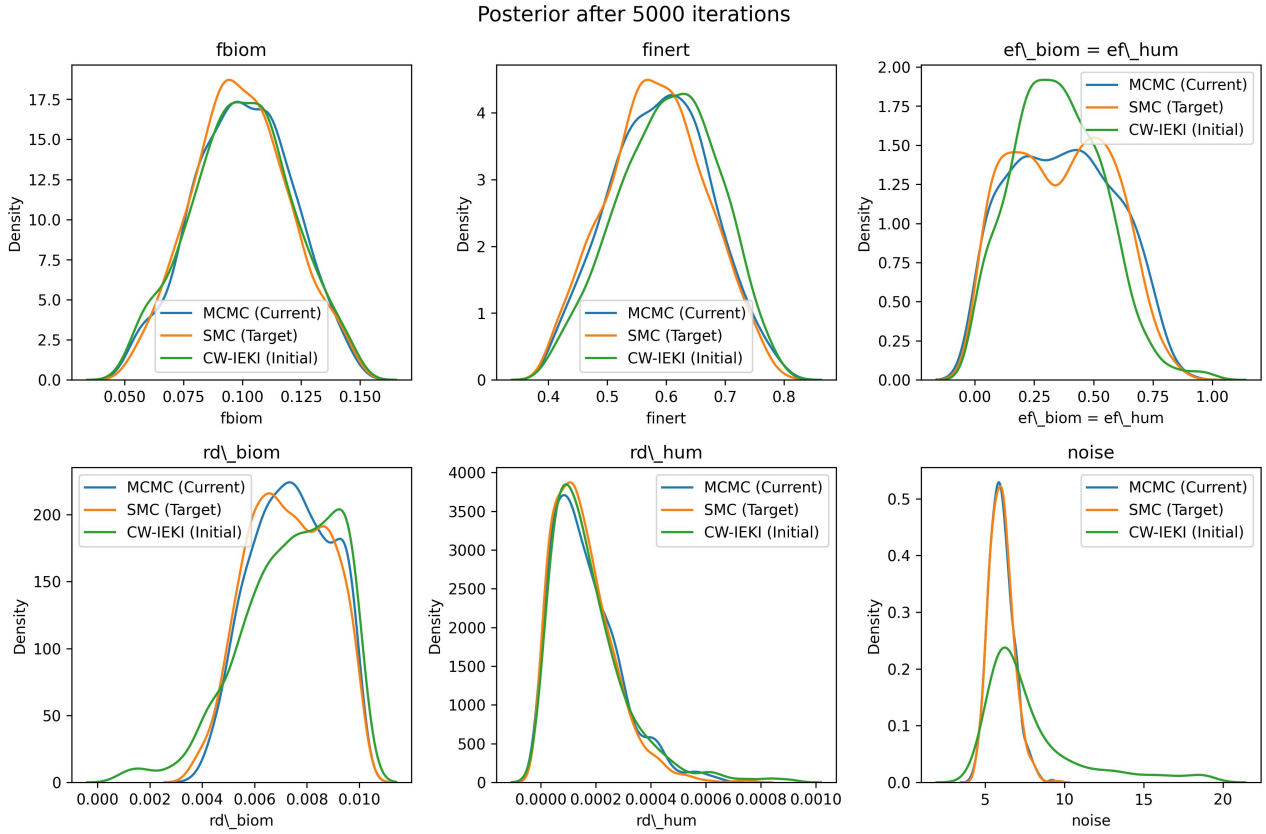


Figure 3: Marginal posterior density plots of the six parameters.

The posterior predictive distribution shown in Figure 4 also illustrates similar predictive power between MCMC and SMC, except that the MCMC posterior results in negative nitrogen mineralisation early in the simulation, which is invalid. The cause of this error is considered out of scope for this research, but is speculated to originate from the parameters not being strictly bounded in the regions defined by the priors.

The computing time for the MCMC method involves the time required to make 11,000 evaluations of the original, expensive model in CW-EIKI; the time to train the deep neural network; and the time to make 5,000,000 predictions using the surrogate in MCMC. We find that on average, the surrogate model is around 1077 times faster than the original model, taking 373 milliseconds to evaluate 1,000 samples, compared to 400 seconds.

5 Discussion and Conclusion

In this paper, we have explored the use of a DNN surrogate model to accelerate the Bayesian inference process on a model to predict time-series nitrogen mineralisation in soil, a module of APSIM. The goal of Bayesian inference was to fit the model to a set of simulated data by optimising five parameters from the model, and a noise parameter. The 11,000 data points used for training were obtained from 11 iterations of the CW-IEKI

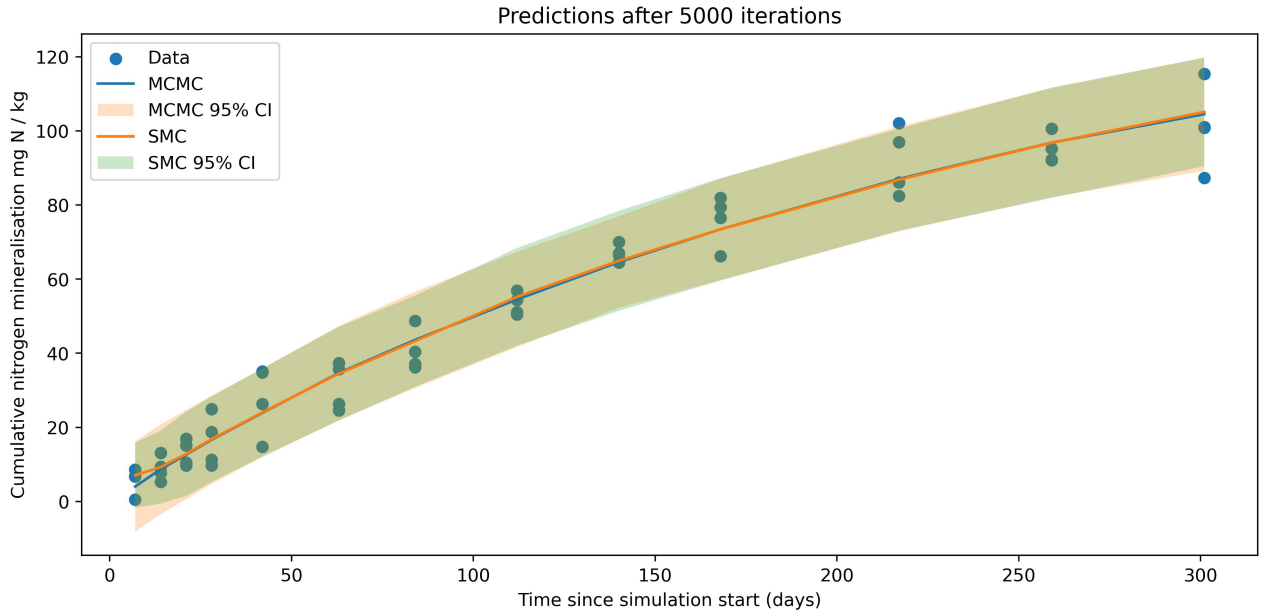


Figure 4: Comparison of the simulated data to the mean and 95% credible intervals for the posterior predictive distribution of cumulative nitrogen mineralisation obtained from the surrogate model fitted to this data. Models were fitted using MCMC (blue - yellow) and SMC (orange - green).

with an ensemble size of 1,000.

We used the dataset to train a fully connected deep neural network with six inputs, including five parameters from the model which we sought to infer, and a time parameter, to output a single cumulative nitrogen mineralisation prediction. The network consisted of three hidden, fully connected layers of 256 neurons each, all using the hyperbolic tangent activation function. Before training, the data set was standardised and split into training (80%), validation (10%) and testing sets (10%). Considering this as a regression model, we used the Root Mean Squared error loss function for training.

We found that the surrogate was able to mimic the original model almost perfectly, except in some rare cases where there was limited training data. The Metropolis-Hasting algorithm (Hastings, 1970) of the Markov chain Monte Carlo (MCMC) class was then applied to perform Bayesian inference on the surrogate model to infer the original model's parameters. We compared the performance of this method against that of the likelihood tempering sequential Monte Carlo (SMC), an exact inference algorithm (Del et al., 2006), on the original model. Most importantly, we found that by using the surrogate, the computational expense was reduced by a factor of over 1000 with a similar final estimation of model parameters achieved.

Since this method involves two parts, being the surrogate and the Bayesian inference algorithm, future work could target either area. Since the neural network used in this paper was not optimised, future work could explore different hyperparameters for this network, i.e. the number of hidden layers, number of neuron per layer; or adapt a different network architecture such as the recurrent neural networks that are capable of producing sequential outputs, exploiting the time-series nature of the original model.

Another area of improvement involves the use of more efficient inference methods instead of a fixed-number of Metropolis-Hasting Monte Carlo iterations. Such methods include the Hamiltonian Monte Carlo algorithm (Duane et al., 1987) and its extension, the No-U-Turn Sampler (Hoffman and Gelman, 2011), assuming it is possible to extract gradient information from the surrogate, and that the surrogate's gradient accurately reflects that of the original model.

Finally, it is of interest to explore the universality and any limitations of this method, by applying it to other, potentially more complex models. As observed in Figure 4, the predictions by APSIM follow a considerably simple curve, making it easy for the surrogate to emulate its behaviours and produce effective parameter estimations. With a more complex model, it is expected that more training data, or a more clever surrogate architecture will be needed to yield similar performance.

6 Acknowledgements

I would like to express my gratitude to Professor Christopher Drovandi and Dr Simon Denman for their valuable guidance and feedback throughout this project. I would also like to thank the Australian Mathematical Sciences Institute (AMSI) for the Vacation Research Scholarship.

References

Angione, C., Silverman, E., & Yaneske, E. (2022). Using machine learning as a surrogate model for agent-based simulations. *PLOS ONE*, *17*(2), e0263150. https://doi.org/10.1371/JOURNAL.PONE.0263150

Botha, I., Adams, M. P., Tran, D. K., Bennett, F. R., & Drovandi, C. (2022). Component-wise iterative ensemble Kalman inversion for static Bayesian models with unknown measurement error covariance. https://doi.org/10.48550/arxiv.2206.02451

Chen, E. (2022). Deep-learning-based surrogate model for fast and accurate simulation in pipeline transport. *Frontiers in Energy Research*, *10*, 1399. https://doi.org/10.3389/FENRG.2022.979168/BIBTEX

Del, P., Doucet, A., & Jasra, A. (2006). Sequential Monte Carlo samplers. *J. R. Statist. Soc. B*, *68*, 411–436.

Duane, S., Kennedy, A. D., Pendleton, B. J., & Roweth, D. (1987). Hybrid Monte Carlo. *Physics Letters B*, *195*(2), 216–222. https://doi.org/10.1016/0370-2693(87)91197-X

Hastings, W. K. (1970). Monte Carlo Sampling Methods Using Markov Chains and Their Applications. *Biometrika*, *57*(1), 97–109.

Hoffman, M. D., & Gelman, A. (2011). The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, *15*, 1593–1623. https://doi.org/10.48550/arxiv.1111.4246

Hollis, A., & Kariko, A. (2006). BMRC RESEARCH REPORT NO. 123. www.bom.gov.au

Holzworth, D. P., Huth, N. I., deVoil, P. G., Zurcher, E. J., Herrmann, N. I., McLean, G., Chenu, K., van Oosterom, E. J., Snow, V., Murphy, C., Moore, A. D., Brown, H., Whish, J. P., Verrall, S., Fainges, J., Bell, L. W., Peake, A. S., Poulton, P. L., Hochman, Z., ... Keating, B. A. (2014). APSIM – Evolution towards a new generation of agricultural systems simulation. *Environmental Modelling & Software*, *62*, 327–350. https://doi.org/10.1016/J.ENVSOFT.2014.07.009

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, *2*(5), 359–366. https://doi.org/10.1016/0893-6080(89)90020-8

Iglesias, M. A., Law, K. J. H., & Stuart, A. M. (2012). The Ensemble Kalman Filter for Inverse Problems. *Inverse Problems*, *29*(4). https://doi.org/10.1088/0266-5611/29/4/045001

McHaney, R. (1991). Computer Simulation: A Practical Perspective. https://books.google.com.my/books?id=tZl9CeMJXAkC&printsec=frontcover#v=onepage&q&f=false

Probert, M. E., Dimes, J. P., Keating, B. A., Dalal, R. C., & Strong, W. M. (1998). APSIM's water and nitrogen modules and simulation of the dynamics of water and nitrogen in fallow systems. *Agricultural Systems*, *56*(1), 1–28. https://doi.org/10.1016/S0308-521X(97)00028-0

Ramamohanarao, K., Xie, H., Kulik, L., Karunasekera, S., Tanin, E., Zhang, R., Bin Khunayn, E., & Khunayn, E. B. (2016). SMARTS: Scalable microscopic adaptive road traffic simulator. *ACM Trans. Intell. Syst. Technol*, *8*. https://doi.org/10.1145/2898363

Robert, C. P., & Casella, G. (2004). Monte Carlo Statistical Methods. https://doi.org/10.1007/978-1-4757-4145-2

Viceconti, M., Henney, A., & Morley-Fletcher, E. (2016). In silico clinical trials: how computer simulation will transform the biomedical industry. *International Journal of Clinical Trials*, *3*(2), 37–46. https://doi.org/10.18203/2349-3259.IJCT20161408

Vilas, M., Bennett, F., Verburg, K., & Adams, M. (2021). Considering unknown uncertainty in imperfect models: nitrogen mineralization as a case study. *Proceedings of the 24th International Congress on Modelling and Simulation (MODSIM2021)*. https://www.mssanz.org.au/modsim2021/papers/B3/vilas.pdf