Uncertainty Quantification Mini Project

Ben Rickard

1 Introduction to the Model

The model that I will be emulating and practising other UQ techniques on is the Friedman function in the form as shown below:

$$f : \mathbb{R}^5 \to \mathbb{R}$$

$$f(x) = 10\sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5 \qquad [1]$$

This function is regularly used to test statistical modelling techniques, or to generate artificial datasets.[2] It is clearly linear in variables x_4 and x_5, quadratic in x_3, and symmetrically sinusoidal in x_1 and x_2. Moreover, there is a mixture of linear and non-linear relationships between the five input variables and the scalar output. I will assume the input variables are over the hypercube with $x_i \in [0, 1]$ for all $i \in [1, 5]$.

First, I explored how the model depends on the different input variables. I did this by in turn only allowing one of the input variables to vary across $[0, 1]$, whilst fixing the others to 0.5 which is shown in Figure 1 below.

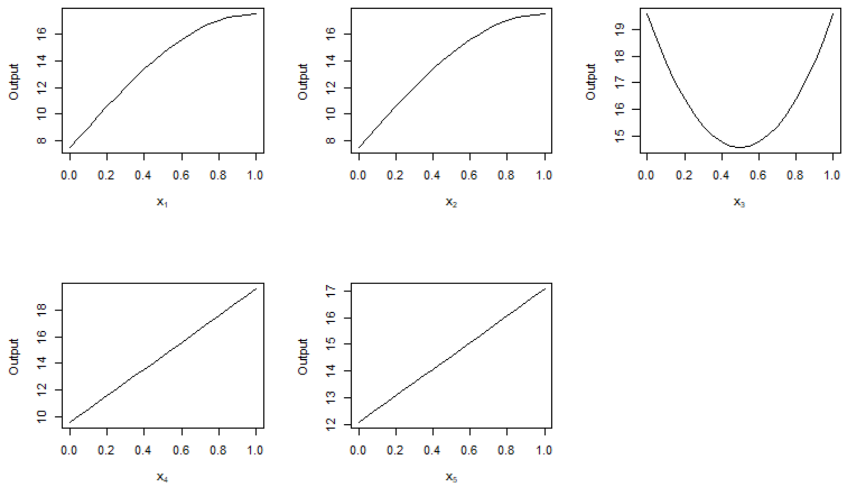

Figure 1: Initial Plots of Friedman function

The five plots agree with our initial statements about the Friedman function and its arguments. It is clear that x_1 and x_2 are identical and are part of a sine wave, x_3 is indeed quadratic, and x_4 and x_5 are linear with x_4 having a gradient twice that of x_5.

2 Emulation in 1 Dimension

The purpose of emulation is to attempt to learn the entire shape of a function, by only evaluating the function at a few points. This is an incredibly useful technique to explore how a complex function behaves without needing to evaluate it over the whole input space. This report will focus on Bayes Linear emulators, which are performed in a Bayesian manner by first specifying prior expectation, variance and covariance. Then, the updated expectation and variance of the function at a new input x can be easily calculated by:

$$\mathbb{E}_D[f(x)] = \mathbb{E}[f(x)] + \mathbb{C}\text{ov}[f(x), D]\mathbb{V}\text{ar}[D]^{-1}(D - \mathbb{E}[D])$$

$$\mathbb{V}\text{ar}_D[f(x)] = \mathbb{V}\text{ar}[f(x)] - \mathbb{C}\text{ov}[f(x), D]\mathbb{V}\text{ar}[D]^{-1}\mathbb{C}\text{ov}[D, f(x)]$$

where D is the vector containing $f(x)$ for the initially performed runs x.[3]

I am going to first emulate the Friedman function in 1 dimension by fixing all variables at 0.5 except from x_3 which I allow to vary in $[0, 1]$. The emulation was done by first evaluating the function at six evenly spaced locations $\{0, 0.2, 0.4, 0.6, 0.8, 1\}$. For this emulator, I set the prior expectation $\mathbb{E}[f(x)] = 15$, the prior variance $\mathbb{V}\text{ar}[f(x)] = \sigma^2 = 2$, and used an exponential covariance function of the form:

$$\mathbb{C}\text{ov}[f(x), f(x')] = \sigma^2 \text{exp}\Big\{-\frac{||x - x'||}{\theta^2}\Big\} citeFunctionWebsite$$

I set the correlation length $\theta = 0.25$, which controls the strength of the dependency of new inputs to the initial runs. I could then evaluate the emulator's updated expectation and variance over 201 evenly spaced points in $[0, 1]$. This is plotted in Figure 2, alongside the true Friedman function for comparison.

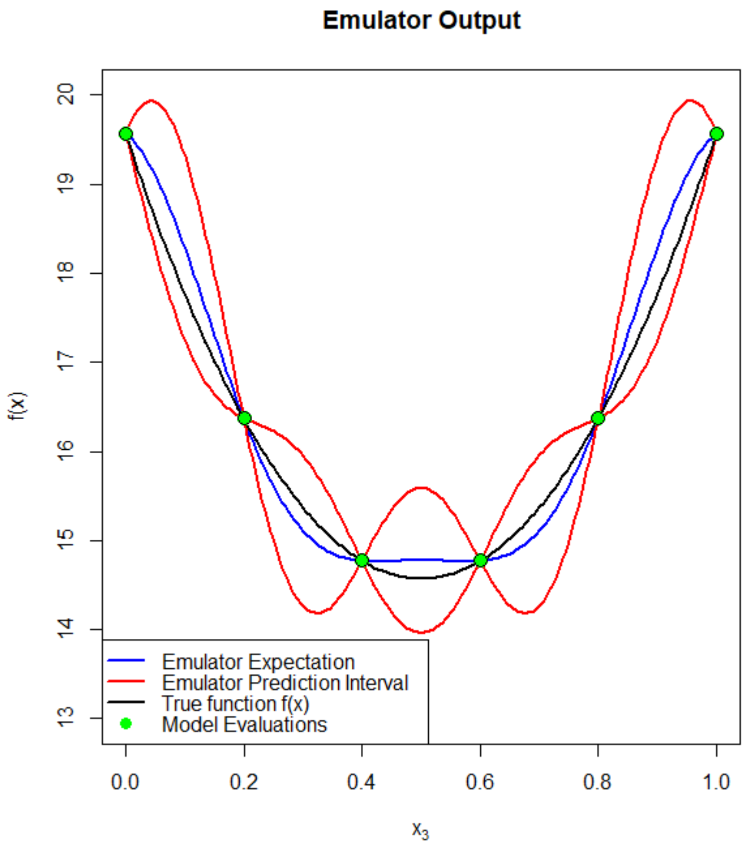

Figure 2: 1D Emulator Ouput with the True Function

One of the key things to note from the emulator in Figure 2, is that there is a very low variance around the points where we evaluated the function, and higher variance in the spaces between. This follows what our intuition tells us to expect. It seems the emulator prediction does a good job at following the true function, since the true function is always within the emulator prediction interval $\mathbb{E}_D[f(x)] \pm 3\mathbb{V}\text{ar}_D[f(x)]$ based on Pukelsheim's 3-sigma rule. Hence, it seems the initial simple emulation in 1 dimension has been relatively successful.

I then performed some further analysis of the 1-dimensional emulator by adjusting some of the prior specifications, to understand further their effect on emulator output and results. Figure 3 depicts the effect of varying the prior variance σ^2. As expected, as σ^2 increases, the prediction interval grows wider since there is more uncertainty in the emulator prediction. It is also clear that if σ^2 is too small, the emulator becomes too confident and the true function leaves the prediction interval. Moreover, the emulator performs poorly if $\sigma^2 = 1$.

I then also varied the value of θ in the covariance function as seen in Figure 4. As θ grows, the influence of the known runs increases too, leading to the emulator being more sure and the prediction interval shrinks. The emulator expectation function also becomes smoother

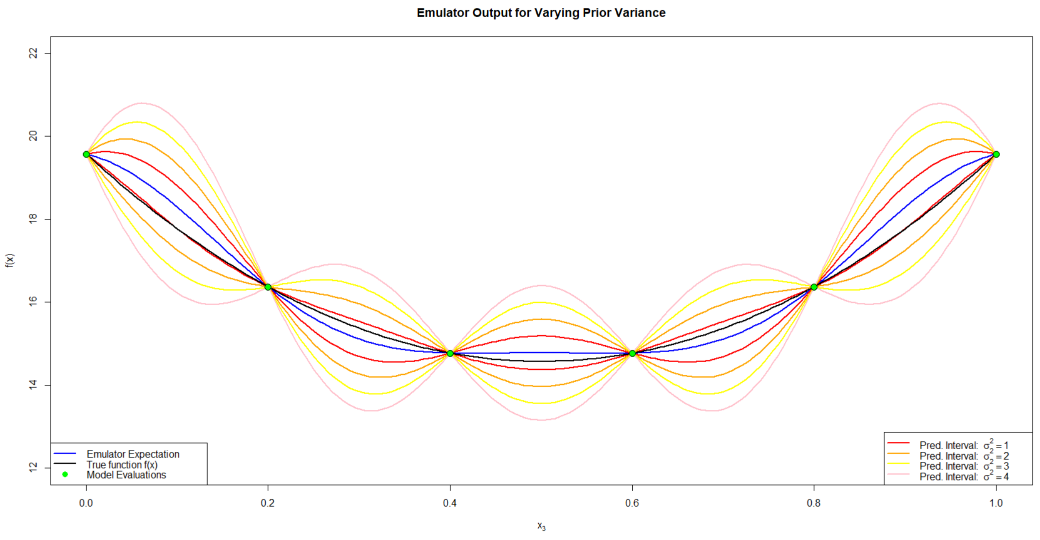

Figure 3: 1D Emulator with Varying Prior Variance

and less wobbly which is a desired feature for this emulator. However, when $\theta = 0.4$, the emulator becomes extremely confident and the emulator prediction interval is more narrow than realistically desired. Hence, it could be thought that the emulator performs poorly here.

3 Emulation in 2 Dimensions

Next, I constructed an emulator for 2-dimensional input by allowing x_2 and x_3 to vary on the unit-square $[0, 1] \times [0, 1]$ and fixing every other variable at 0.5. I first evaluated the Friedman function at 16 points in an evenly-spaced square grid. I again specified the prior expectation $\mathbb{E}[f(x)] = 15$, but I increased the prior variance to $\mathbb{V}\text{ar}[f(x)] = \sigma^2 = 3$ as there is now a more complex function. I still used an exponential covariance function, but changed θ to 0.5. I then evaluated the emulator expectation and variance on a 50×50 grid. Figure 5 depicts the emulator expectation and the true Friedman function.

Figure 5 shows that the emulator expectation matches the shape of the true function well. The minima, maxima and saddle points are all well represented. However, it is easier to see how successful the emulator has been by viewing the emulator variance and the emulator diagnostics, $S_D[f(x)]$, which is calculated by:

$$S_D[f(x)] = \frac{f(x) - \mathbb{E}_D[f(x)]}{\sqrt{\mathbb{V}\text{ar}_D[f(x)]}} \qquad [3]$$

Figure 6 depicts the plots for the emulator variance and diagnostics. It shows the same expected effect with the emulator variance, in that it is lowest near the points we initially ran the function, and it increases in the gaps. It also depicts that there are some areas of failure in the diagnostics as there are some white spaces at the edges where $|S_D[f(x)]| > 3$.

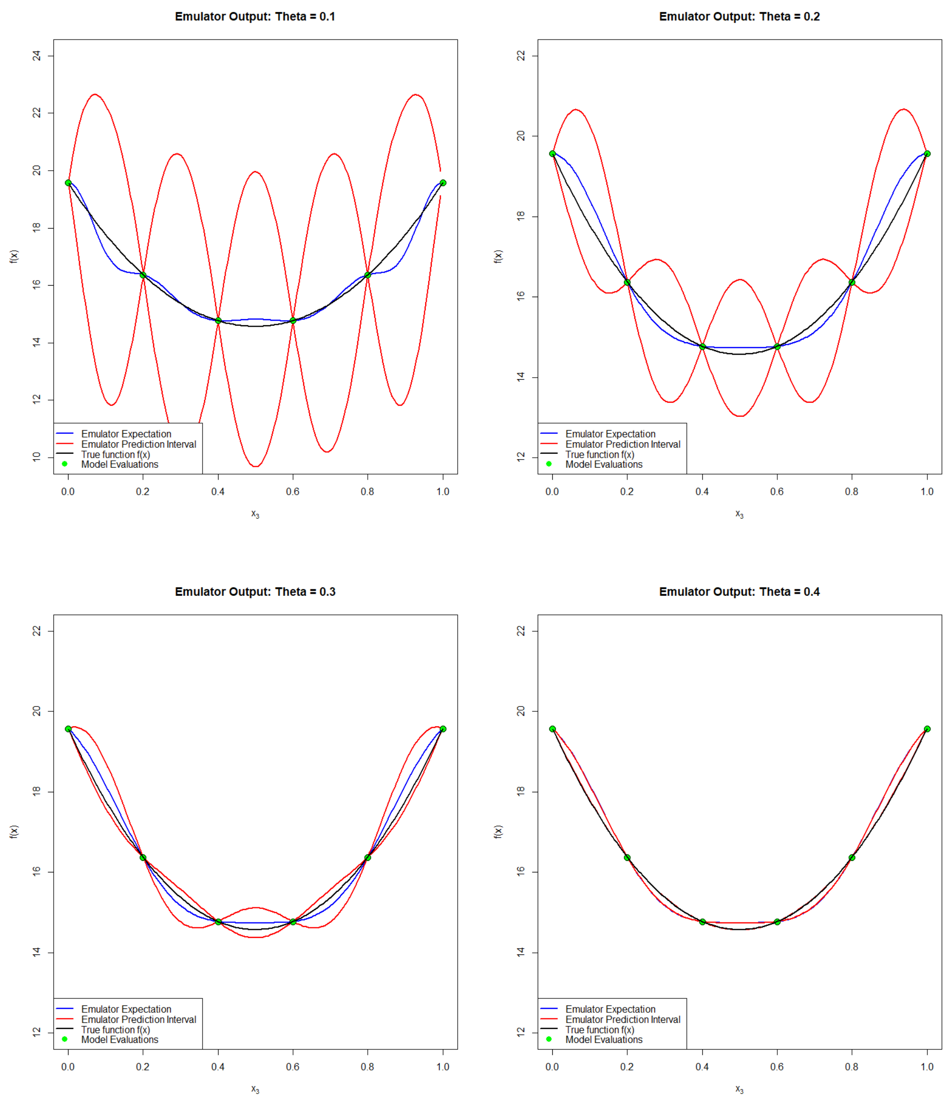

Figure 4: Comparison of 1D Emulator Output for Different Values of θ.

Note that the threshold is chosen by Pukelsheim's 3-sigma rule. This means the emulator has performed relatively poorly in these areas.

The performance could hopefully be improved by a more effective design in the placement of the initial evaluations of the function. Hence, I performed the emulation process again, but with a Maximum Latin Hypercube Design, which is the LHD with the maximum minimum distance between run points.[3] Hence, this should more effectively cover the input space. The Maximum LHD with 16 points, which was randomly generated in R, is shown in Figure

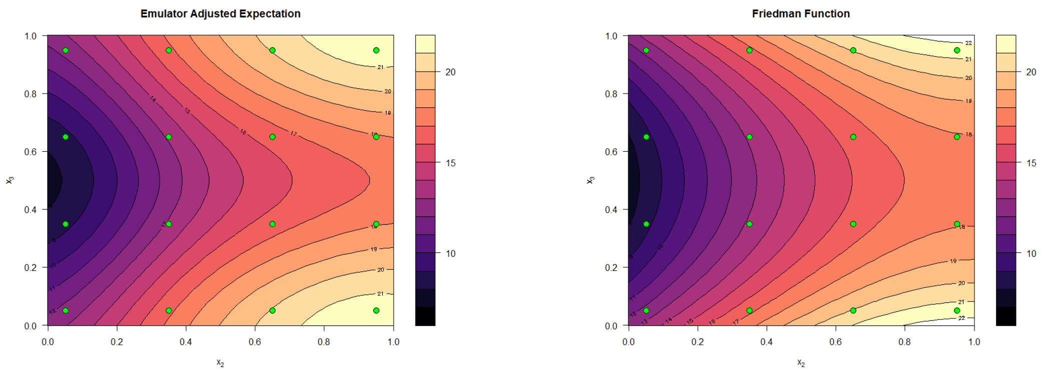

Figure 5: Comparison of Emulator Expectation and the Real Function

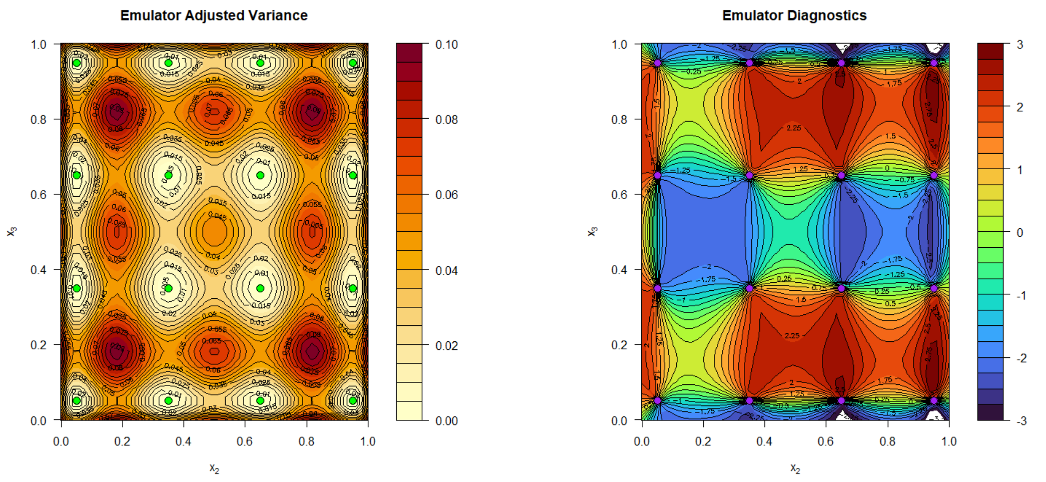

Figure 6: Emulator Variance and Diagnostics

7 by the green dots. The emulation can now be performed with new initial runs, but the same prior hyper-parameters. The new emulator expectation, variance, and diagnostics are also shown in Figure 7 for comparison with the naive grid design.

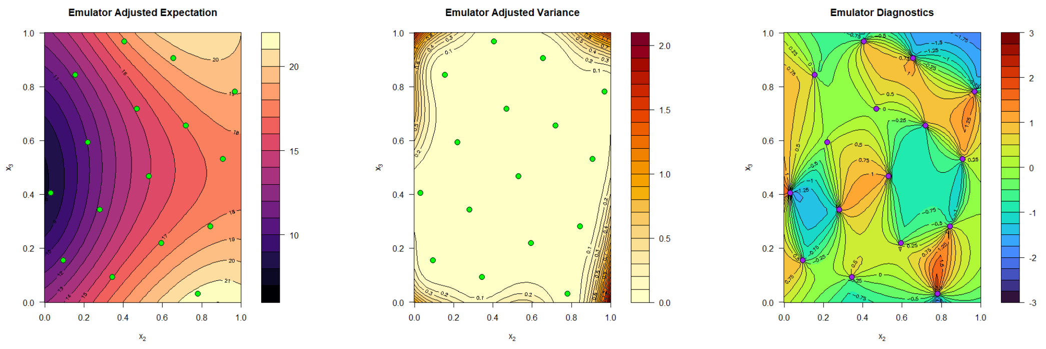

Figure 7: Emulator Expectation, Variance, and Diagnostics from Maximum LHD

It is evident in Figure 7 that the emulator expectation again matches the overall shape of the true function. However, with a more sophisticated emulator design, the variance is more consistent as there are smaller gaps between the points, except in the corners where larger areas are missed by the initial runs. Yet the emulator diagnostics have performed much better with no white areas of failure, and overall there is a lower $|S[f(x)]|$ than with a naive evenly spaced grid design. Hence it can now be said that the 2-dimensional emulator has been successful.

Further analysis for the 2D emulator can be done by considering different correlation lengths θ in the 2 different dimensions. By considering the emulator expectation function in Figure 7, and also the true Friedman function in Figure 6 (in reality this wouldn't be known), it seems that there is more variation in the x_2 direction than x_3. Hence we may wish to try another emulator, but with a relatively larger θ in the x_3 direction, since it appears that known runs have a larger influence in this direction than in x_2 direction. Figure 8 depicts the emulator expectation, variance, and diagnostics for $\theta = (0.5, 1.4)$. The emulator expectation again seems to match the shape of the true function well, attaining better the maximum in the bottom right. The variance is again consistent across the centre of the grid, but now much lower compared the previously across the whole input space. This has meant that the diagnostics are generally higher with the new θ, although there are no areas of failure. More exploration could be done with adjusting the θ lengths in both directions, but for now it seems the simpler θ generated a better performing emulator overall, due to better performing diagnostics.

4 History Matching in 2 Dimensions

I will now demonstrate history matching in 2-dimensions by assuming that the Friedman function $f(x)$ is a model for a real system y. I will continue to keep x_2 and x_3 free, and

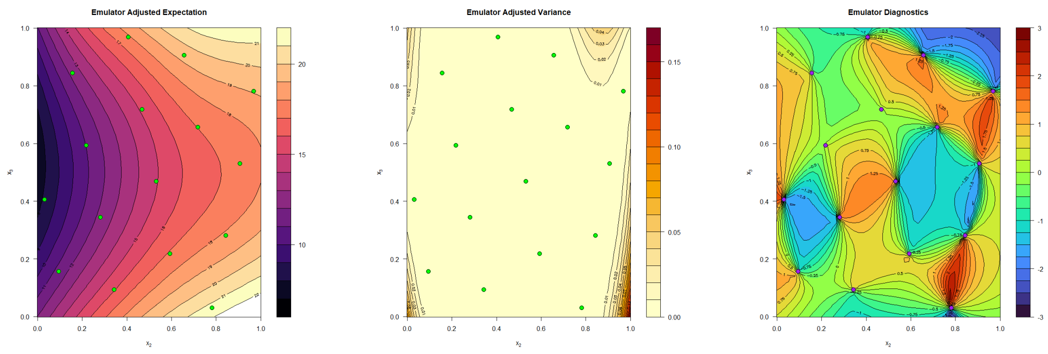

Figure 8: Emulator Expectation, Variance, and Diagnostics with Different Correlation Lengths in Each Dimension

the other variables fixed for this. Since the function is now thought of as a model, there is new uncertainty involved called the model discrepancy, ϵ. This is the difference between the real system, and the model $f(x^*)$ which has been evaluated at the best possible input x^* to match the system. This is denoted as $y = f(x^*) + \epsilon$, where ϵ is a random variable that needs to be specified by the subject experts, and it is normally assumed to be independent of $f(x^*)$.[3] In this History Matching, I will take $\mathbb{E}[\epsilon] = 0$ and $\mathbb{V}\text{ar}[\epsilon] = 0.75^2$.

There is also now observational error in any taken observations of the system y, for example due to uncertainty in any equipment used in obtaining such observations. This is denoted as e and represents the difference between the real system and the imperfect observation z by $z = y + e$.[3] In this History Matching, I will take $\mathbb{E}[e] = 0.1$ and $\mathbb{V}\text{ar}[e] = 0.1^2$.

Now for this History Matching, I assume to have an observation of $z = 11.2$ and I wish to find the acceptable inputs which would give this output. I did this by first running the model at 10 points using a Maximum LHD. Then using the emulator expectation and variance across a 50×50 grid, I calculated the implausibility $I(x)$ across the grid by the equation:

$$I^2(x) = \frac{(\mathbb{E}_D[f(x)] - z)^2}{\mathbb{V}\text{ar}[\mathbb{E}_D[f(x)] - z]} = \frac{(\mathbb{E}_D[f(x)] - z)^2}{\mathbb{V}\text{ar}_D[f(x)] + \mathbb{V}\text{ar}[\epsilon] + \mathbb{V}\text{ar}[e]} \qquad [3]$$

A large value of $I(x)$ indicates that the emulator expectation is relatively far from the observation, so it is unlikely to get a good match between f(x) and z if the model was ran there. Hence, inputs are discarded from the parameter search if $I(x) > c$, where c is chosen by Pukelsheim's 3-sigma rule. Figure 9 shows the emulator expectation, variance, and implausibility for the first wave of History Match with Maximum LHD grid design.

For the second wave of the History Matching, I chose to run the model at points in the blue river-like area at low values of x_2, and in the bottom right corner. This is because this is where points are in the non-implausible region and hence should be investigated further. I ran the model at the points (0.98,0.02), (0.03,0.95), (0.26,0.4), (0.1,0.8), (0.13,0.65), (0.07,0.1), and (0.2,0.24) one at a time, plotting the changed implausibility across the grid based on newly updated expectation and variance. These new points are seen as pink in

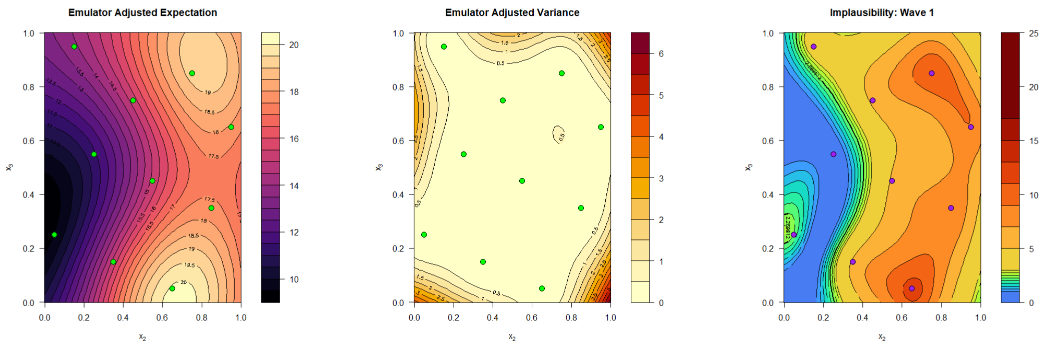

Figure 9: Emulator Expectation, Variance, and Implausibility after Wave 1

the first plot of Figure 10. After finishing the second wave, the implausibility of the true function is plotted to compare how similar the emulator's non-implausible region is to the true function's. This can be seen in Figure 10, where the second wave points become purple once the function has been evaluated there.

Figure 10 depicts the implausibility updating based on every new point the model is evaluated at. This changes the emulator expectation and variance, hence updating the implausibility. The implausibility plot transforms through adding the 7 new points and becomes very similar to the implausibility of the real function. Furthermore, it is clear that there is a good idea of which inputs are likely to lead to a system that gives the observation, and so the History Matching appears to have been successful.

5 Conclusion

To conclude, I have been able to effectively construct emulators for the Friedman function with 1 and 2 dimensional inputs. Further analysis could be done on the effects of adjusting the prior hyper-parameters such as θ or σ^2 in the 2D emulator. I was also able to produce an effective History Matching process with an assumed observation and uncertainties. Again, further analysis could be done here, this time on the effect of changing ϵ and e. Another alternative is I could have explored optimisation techniques in UQ. For example, I could have attempted to find the argument which maximises the Friedman function. This is done by performing n initial runs of the true function as usual, but then deciding the location of the $(n+1)^{\text{st}}$ run by calculating the expectation of improvement across the grid. This is done in a sequential fashion until we are satisfied with the maximum attained.[3]

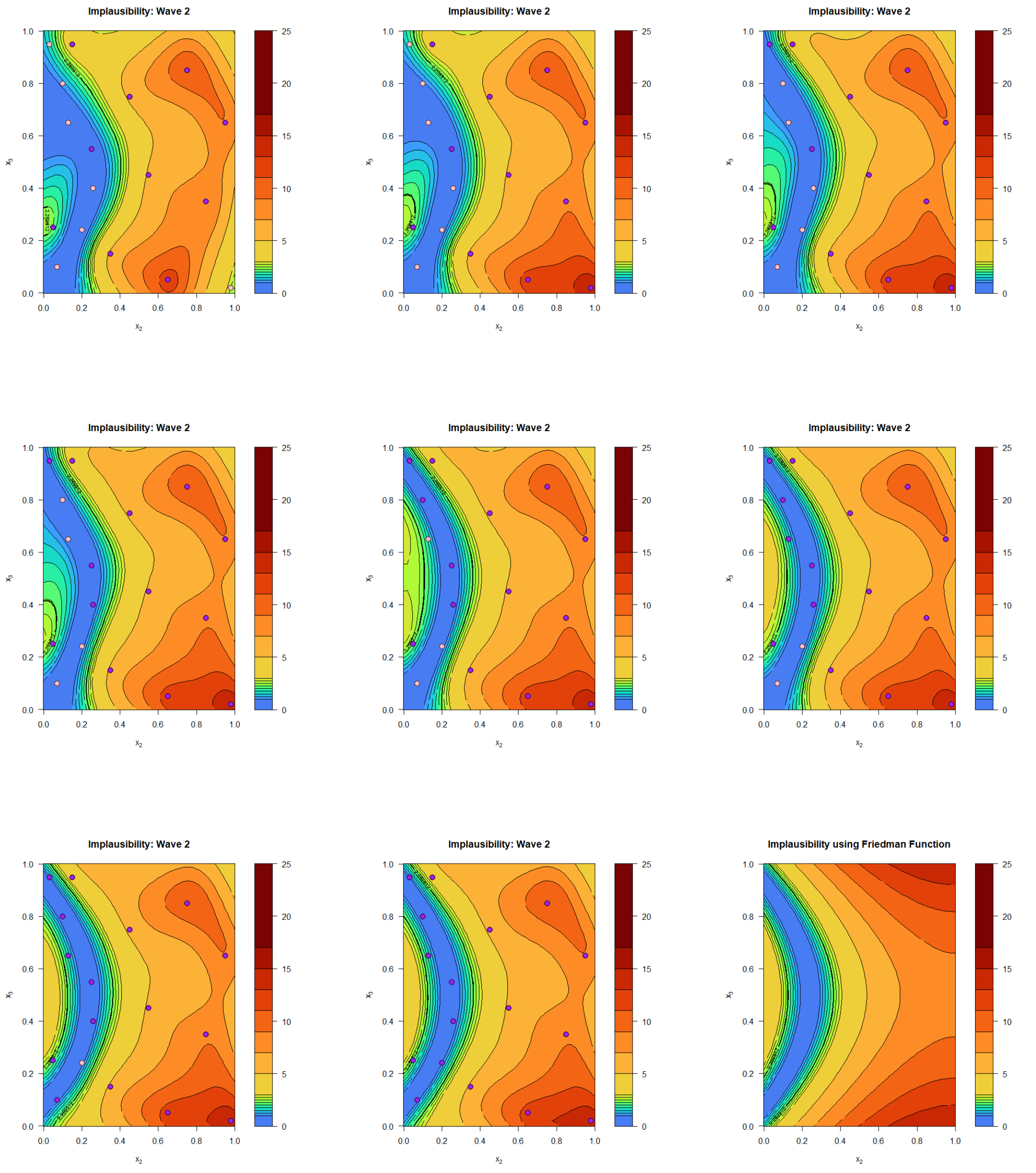

Figure 10: Updated Implausibility for Each New Point, and Implausibility for Real Function

References

[1] S. Surjanovic and D. Bingham (2013), *Friedman Function*, Virtual Library of Simulation Experiments, `https://www.sfu.ca/~ssurjano/fried.html`, Accessed: January 20, 2025.

[2] J. H. Friedman (1991), *Multivariate Adaptive Regression Splines*, *The Annals of Statistics*, **19**, pp.1-67.

[3] I. Vernon, *Uncertainty Quantification 4*, Lecture Notes, Uncertainty Quantification IV, Durham University, Michaelmas 2024.