Project Report

Alexis Park, Jonathan Chen, Kevin Xie
CSE515T: Bayesian Methods in Machine Learning

December 4, 2019

Data visualization

The Branin function is defined as follows:

$$f(\boldsymbol{x}) = a(x_2 - bx_1^2 + cx_1 - r)^2 + s(1-t)\cos(x_1) + s \tag{0.1}$$

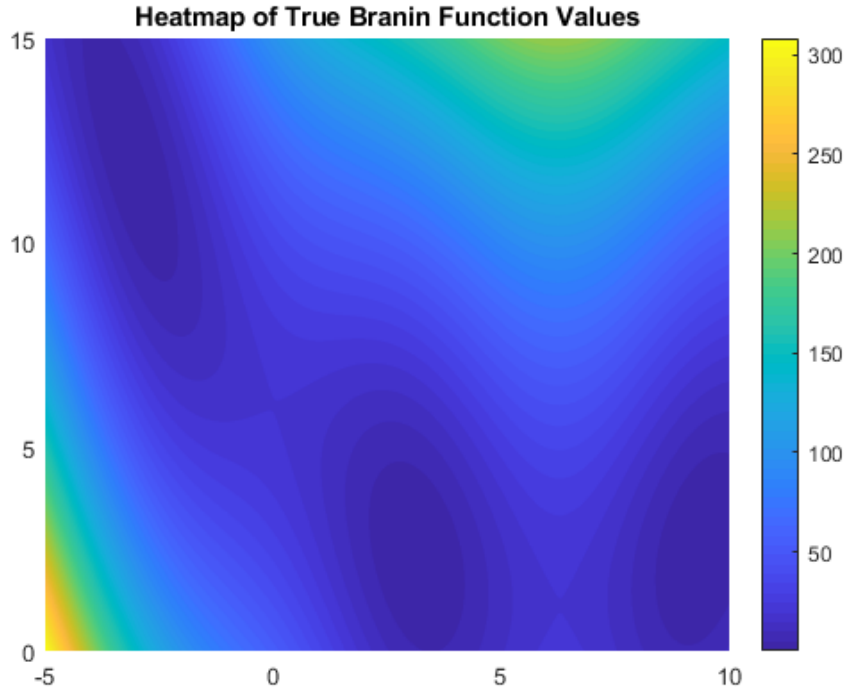

Figure 1: Heatmap of True Branin Function values with domain $\mathcal{X} = [-5, 10] \times [0, 15]$ with 1000 values per dimension.

From 1, one can see that the function's values fluctuate in a somewhat sinusoidal manner, with a large minimal "trench" spanning diagonally across the center of the domain with steadly increasing regions along the trench's sides. Therefore the function does not appear to be stationary.

In order to make the function more stationary, we tried couple of transformations. By analyzing the equation, we note that the sinusoidal fluctuations can be attributed to both the added

cosine term and the 4th order polynomial term. We thus attempted to pass the data through a transformation that combined an inverse cosine function (*arc cos*) with the cumulative density function of the normal probability distribution (*normal cdf*). However, taking the *log* of the data gave us the most stationary result as shown in 2.

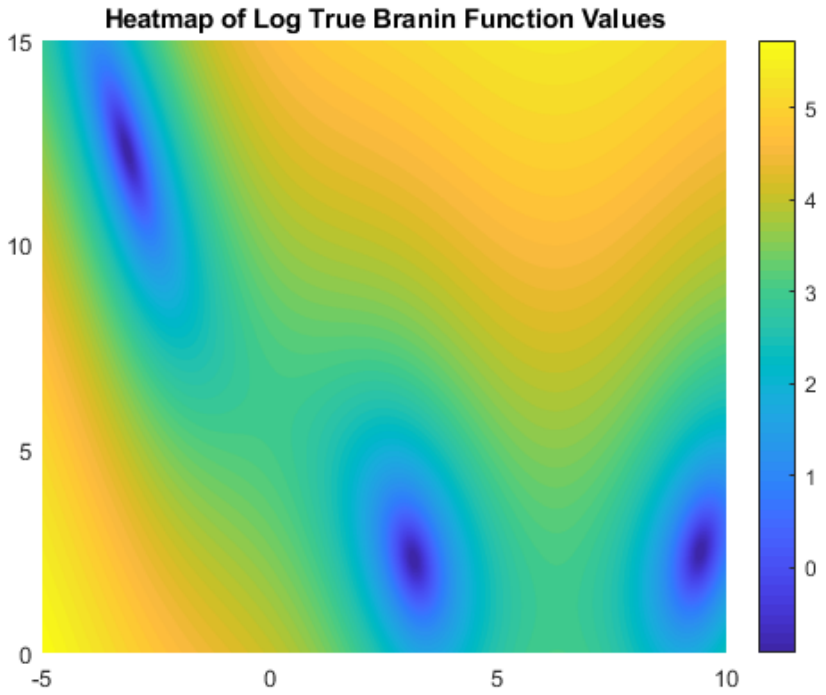

Figure 2: Heatmap of logged Branin Function values with domain $\mathcal{X} = [-5, 10] \times [0, 15]$ with 1000 values per dimension. The plot shows the behavior of the Branin Function relatively more constant throughout the domain compare to the original function behavior.

Now, we plotted kernel density estimate (KDE) of LDA and SVM benchmarks (Fig. 3A and 3C). By analyzing probability density graph, we identified tht both KDEs show log-normal distribution. Also, both estimates have relatively similar behavior but on significantly different scales.

Since both KDEs show log-normal distribution, taking log of values gave us the normal distribution, which will give better performance (Fig. 3B and 3D). Both KDEs are more distributed throughout the graph and relatively less concentrated around the peak.

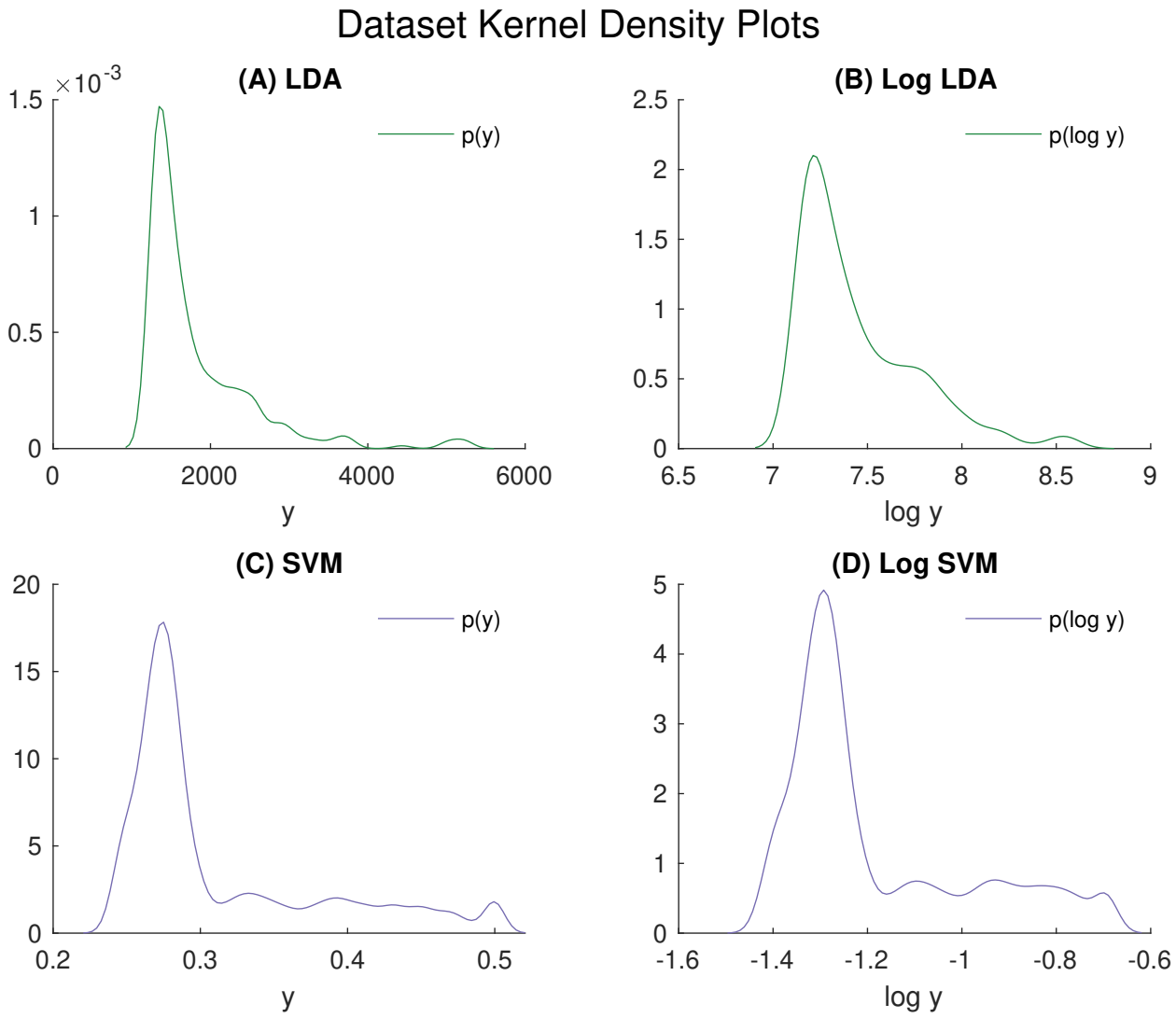

Figure 3: Kernel density estimates for LDA (green, A and B) and SVM (purple, C and D) datasets, along with estimates for their logs.

Model fitting

Model fitting with Branin Function

For model fitting, we used 32 Sobol Sequence Branin Function training points and fit the data to the Gaussian Process model with constant mean value of 0, squared exponential covariance with constant value of length scale and output scale of 1. After maximizing the marginal likelihood, we got the following values of hyperparameters:

- mean: 137.1865
- cov: [1.3864 4.7366] length: 4.004 output: 114.046

During the procedure, we also had to fit likelihood, which we retrieve 0.001 noise likelihood. **Q: Do they agree with your expectations given your visualization?** Based on the 4, the mean

should be around 150 which is reflected in our hyperparameter. The covariance value is also around ...INSERT INTERPRETATION

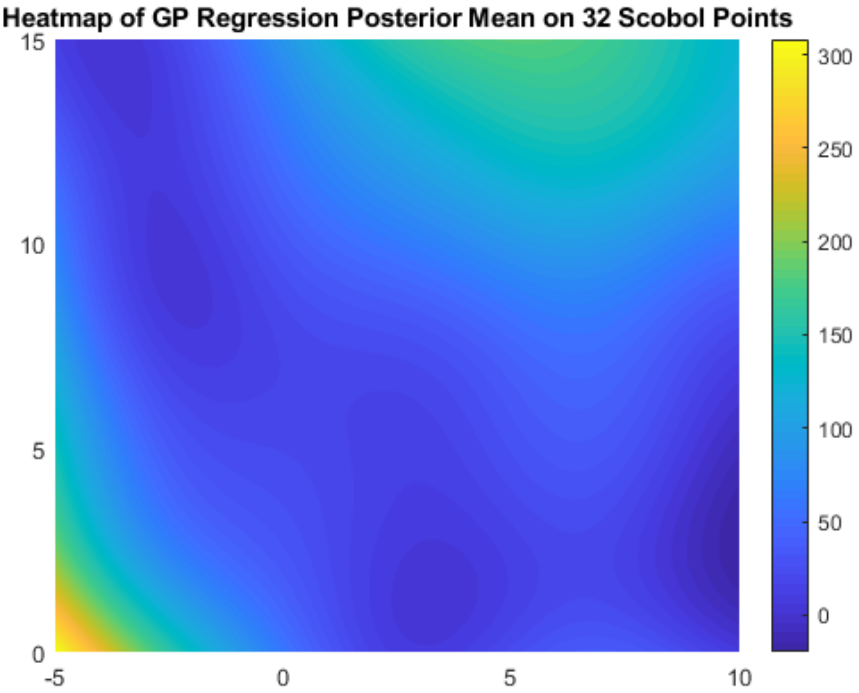

Figure 4: Heatmap of GP Posterior Mean

Comparing the predicted GP Posterior Mean and the True Branin Function values, 5 shows that the true points have the darkest spots on the heatmap, which represents that the predicted and true values are almost alike. **Do you see systematic errors?** On the heatmap, places where we don't have the Branin Function values, as it gets farther away from the points, there are more absolute differences exist between predicted and true values.

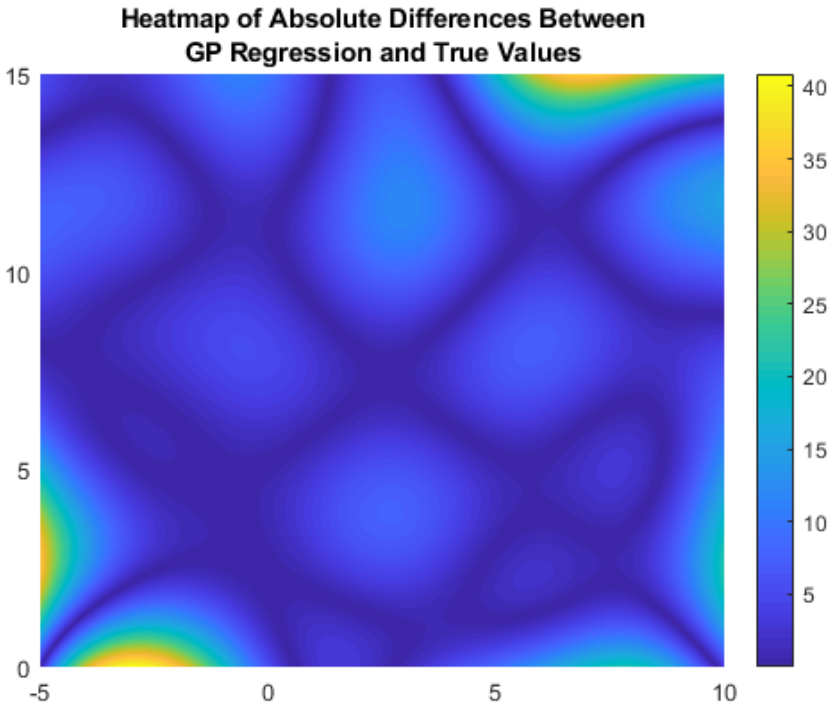

Figure 5: Heatmap of GP Posterior Mean and True Branin values

Figure 6: Heatmap of the GP Posterior standard deviation

Q: Do the values make sense? Does the scale make sense? Does the standard deviation drop to near zero at your data points?

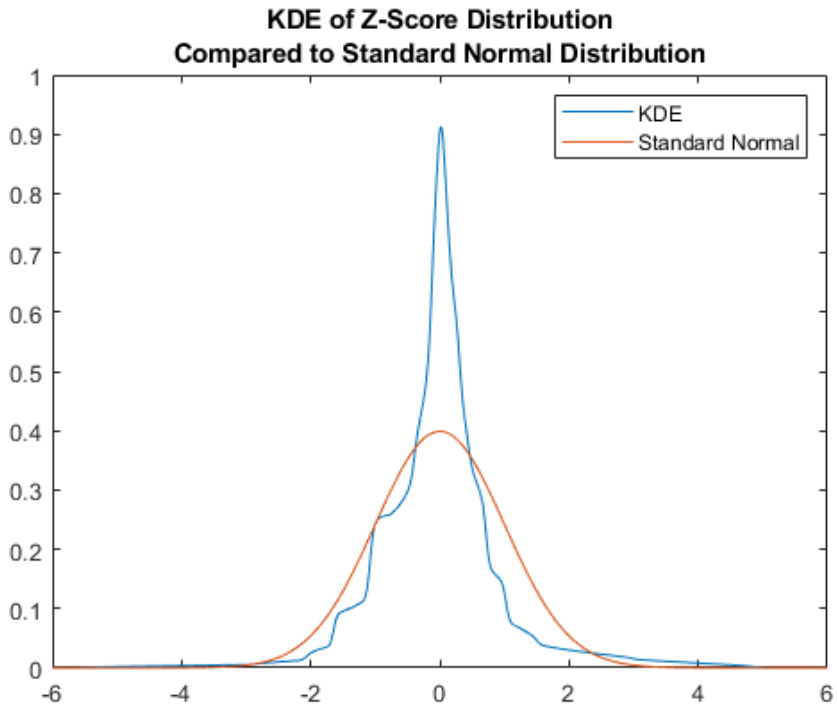

Figure 7: Kernel Density Estimate of the z-scores of the residuals between the GP posterior mean and true values

Based on 7, the KDE of Z-score distribution follows approximately standard normal distrubtion with more concentrated peak in the middle.

Model fitting with log transformation of Branin Function

Now, we repeated model fitting using a log transformation to the output of the Branin function. After maximizing the marginal likelihood with same condition as previous, we got the following values of hyperparameters:

- mean: 3.6227
- cov: [1.2007 0.4896] length: 3.3224 output: 1.6316

During the procedure, we also had to fit likelihood, which we retrieve 0.001 noise likelihood.

Q: Do they agree with your expectations given your visualization

Figure 8: Heatmap of GP Posterior Mean of log transformed Branin Function values

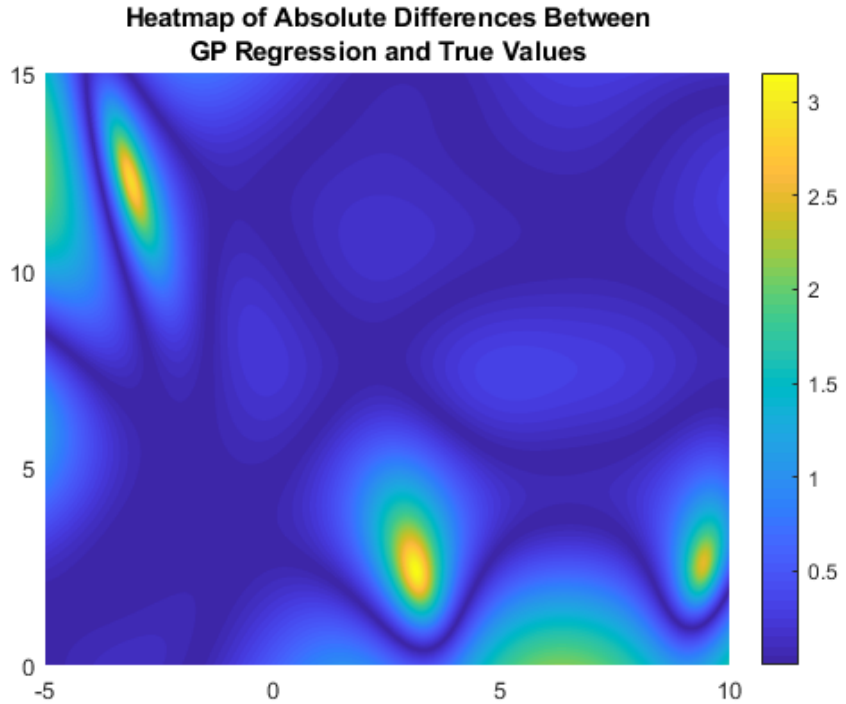

Figure 9: Heatmap of GP Posterior Mean and True log transformed Branin Function values

Q: Compare the predicted values with the true values. Do you see systematic errors?

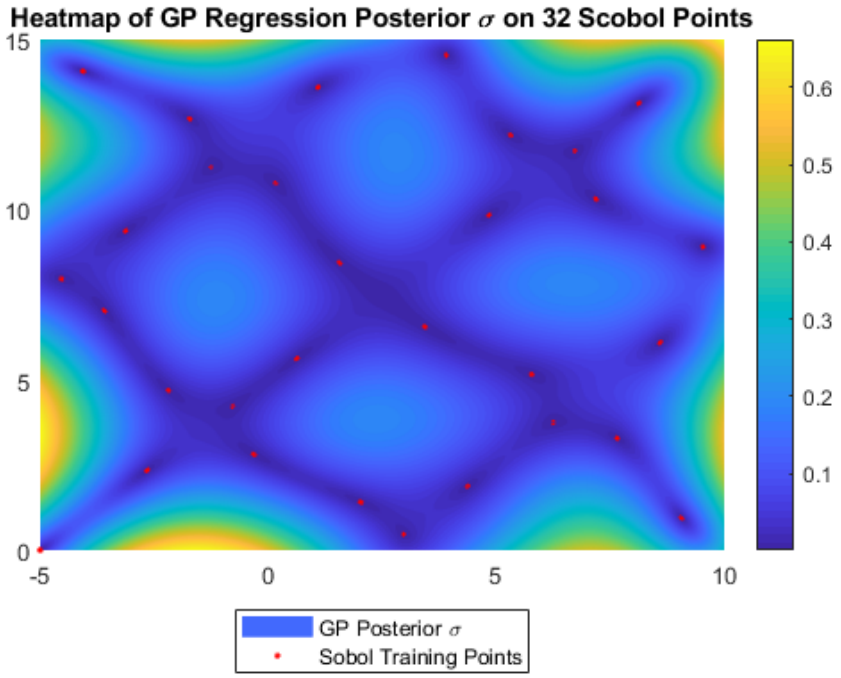

Figure 10: Heatmap of GP Posterior Standard Deviation of log transformed Branin Funciton values

Q: Do the values make sense? Does the scale make sense? Does the standard deviation drop to near zero at your data points?

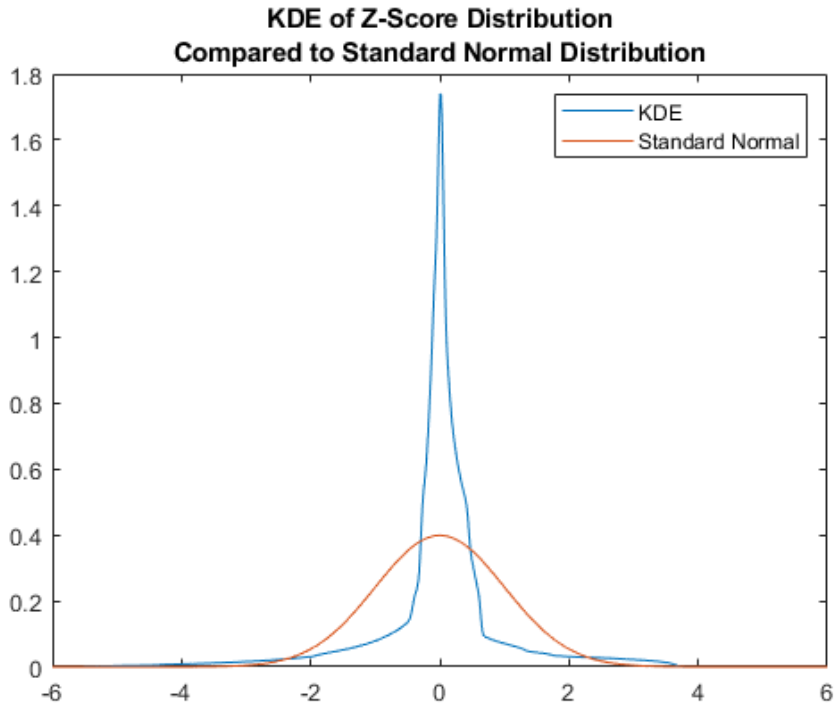

Figure 11: Kernel Density Estimate of z-score of log transformed Branin Function values

Based on 11, the KDE of Z-score distribution follows approximately standard normal distrubtion, but much higher peak.

Q: Does the marginal likelihood improve? Does the model appear better calibrated?

Bayesian Information Criterion (BIC)

In order to compare which model fits better with the dataset, we used BIC to derive a score for how well those choices fit a given dataset. To compute the BIC, we found the values of the hyperparameters maximizing the (log) marginal likelihood using 0.2, and then used 0.3 where $|\theta|$ is the total number of hyperparameters and $|\mathcal{D}|$ is the number of observations. We used 3 for $|\theta|$ and 32 for $|\mathcal{D}|$.

Based on 0.3, we want to minimize the first term as we do not want to overfit the model, but maximize the second term, which is increasing likelihood as higher likelihood is given when the model fits the dataset.

$$\hat{\theta} = \arg\max_{\theta} \log p(y \mid X, \theta) \tag{0.2}$$

$$\text{BIC} = |\theta| \log |\mathcal{D}| - 2 \log p(y \mid X, \hat{\theta}) \tag{0.3}$$

Compute the bic score for the data and model from the last part.

Now considering BIC as a function of the choice of mean and covariance functions (μ, K), we used MeanConst and 4 differenct covariance functions, covSEiso, covRQiso, sum of SE and RQ, and product of SE and RQ. 1 shows the best model and BIC scores we found.

Data	Model	BIC score
Branin	covRQiso	299.367
log Branin	covSEiso	61.471
LDA	covProd	6.777e+06
log LDA	covRQiso	17.298
SVM	covProd	-138.894
log SVM	covRQiso	-77.547

Table 1: Computed BIC scores and best model for different datasets

Bayesian optimization

The best-fitting models were selected from the previous experiments. Specifically, the Branin Model used a log-transformed dataset, a Constant Mean function, and a Squared Exponential covariance function. Similarly, the LDA Model used a log-transformed dataset, a Constant Mean Function, and a Rational Quadratic covariance Function. Finally, the SVM Model used a normal dataset with a Constant Mean Function and a product of the Rational Quadratic and Squared Exponential functions as its covariance function.

We then used the Expected Improvement (EI) Acquisition Function, defined from the course notes as:

$$a_{ei}(\boldsymbol{x}) = (f' - \mu(\boldsymbol{x}))\Phi(f'; \mu(\boldsymbol{x}), K(\boldsymbol{x},\boldsymbol{x})) + K(\boldsymbol{x},\boldsymbol{x})\mathcal{N}(f'; \mu(\boldsymbol{x}), K(\boldsymbol{x},\boldsymbol{x})) \tag{0.4}$$

Where $\Phi(\boldsymbol{x})$ is the Cumulative Probablity Density of the Normal Distribution, and f' is the minimum value of the current observations. Note that the EI acquisition function combines both exploitation and exploration by using both the posterior mean and known minimum value, and the posterior standard deviation, respectively.

Using the previously selected 32 points, a GP model was fit using the aforementioned optmized settings and the following heatmaps of the posterior mean and standard deviation of the Branin function were created:

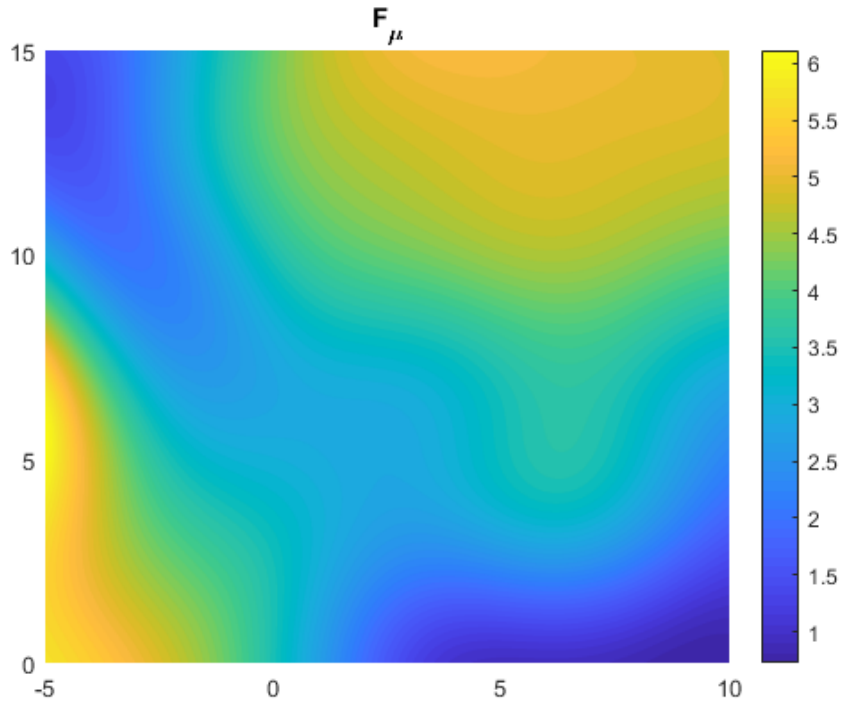

Figure 12: Predictive Posterior Mean of the log Branin Function, calculated using a previously optimized GP model trained on 32 Sobol Sequence points. Warmer colors indicate higher values. Note the predicted minimum areas in dark blue at the top left and bottom right corners of the plot.

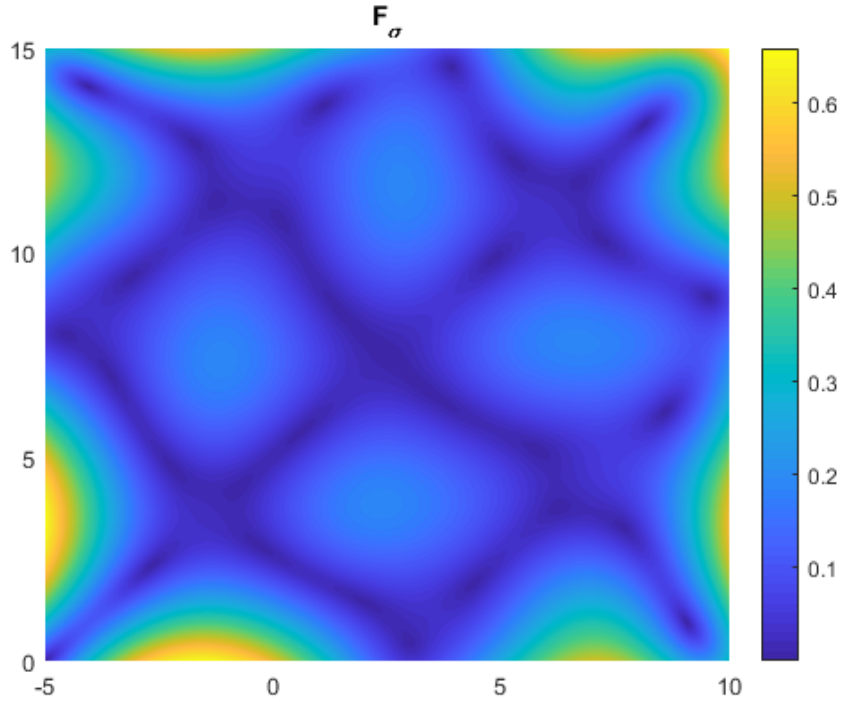

Figure 13: Predictive Posterior Standard Deviation of the log Branin Function, calculated usinga previously optimized GP model trained on 32 Sobol Sequence points. Warmer colors indicate higher deviations and imply greater uncertainty in the prediction.

The EI value was then calculated using the posteriors and identified the point [7.658, 0] as the optimal point to test next. A heatmap of the EI distribution is shown below:

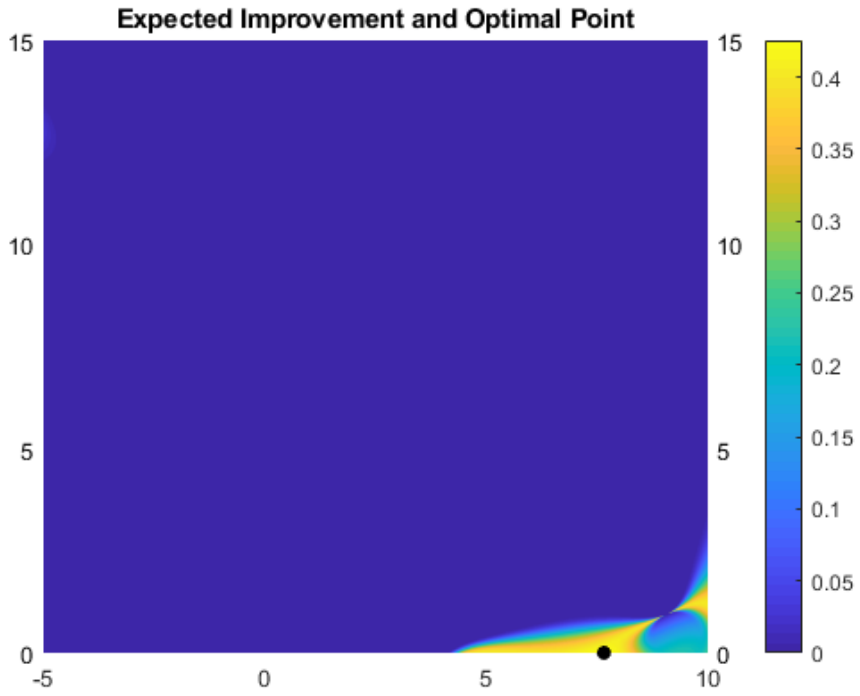

Figure 14: Expected Improvement acquisition values of the optimized GP model trained on 32 Sobol Sequence points. Warmer colors indicate higher expected improvement. The maximum expected improvement is marked as a black point and denotes the optimal location for the next observation.

Analyzing 12, 13, and 14 above, we reason that the proposed optimal testing point is ideal. From the posterior mean, it can be seen that the point [7.658, 0] is within a region of predicted minimal values. Futhermore, from the posterior standard deviation, it can be seen that the point is also within a region of higher uncertainty and therefore reduced predictive confidence. Thus, it is plausible that the EI acquisition function would seek to test this area and point in order to identify a possible global minimum and improve confidence in an area of uncertainty.

The following bayesian active learning experiment was then applied independently to the Branin, LDA and SVM functions:

1. 5 initial observations were randomly selected, constituting the initial dataset $\mathcal{D}$
2. For the Branin Function only, a dense grid of 250,000 points was generated within the domain of the function
3. A GP model using the respective aforementioned optimized settings was fit to $\mathcal{D}$
4. A new point x was found using the EI acquisition function and the GP predictive posterior
5. The function value $f(x)$ was calculated and the point $(x, f(x))$ was added to $\mathcal{D}$
6. Steps 3-5 were repeated 30 times, resulting in a final dataset $\mathcal{D}$ of 35 points

The performance of each of the above experiments was evaluated using the "gap" measure, defined for minimization as:

$$\text{gap} = \frac{f(\text{best found}) - f(\text{best initial})}{f(\text{maximum}) - f(\text{best initial})} \tag{0.5}$$

The gaps for the Branin, LDA, and SVM models were calculated to be 1.0000, 0.7932, and 0.9604, respectively. This implies that EI successfully found a global minimum of the Branin function, but missed the global minimum of the LDA and SVM functions.

The above bayesian active learning experiment was then modified as such:

1. A seed for the random number generator (RNG) was chosen
2. 5 initial observations were randomly selected, constituting the initial dataset $\mathcal{D}$
3. For the Branin Function only, a dense grid of 250,000 points was generated within the domain of the function for use in calculating the GP predictive posterior
4. A GP model using the respective aforementioned optimized settings was fit to $\mathcal{D}$
5. A new point x was found using the EI acquisition function and the GP predictive posterior
6. The function value *f(x)* was calculated and the point $(x, f(x))$ was added to $\mathcal{D}$
7. Steps 4-6 were repeated 150 times, resulting in a final dataset $\mathcal{D}$ of 155 points
8. A new GP model using the respective aforementioned optimized setting was fit to the original initial dataset $\mathcal{D}$, now called $\mathcal{D}$'
9. A new point x was found using the Random Search (RS) acquisition function, which randomly selects a new point
10. The function value *f(x)* was calculated and the point $(x, f(x))$ was added to $\mathcal{D}$'
11. Steps 8-10 were repeated 150 times, resulting in a final dataset $\mathcal{D}$' of 155 points

The above experiment was repeated 20 times with 20 different RNG seeds to create different random initializations for the Branin, LDA, and SVM functions, resulting in a total of 60 experiments. The learning curves of the 60 experiments using only the first 30 new observations for both EI and RS acquisition are shown in the figures below:

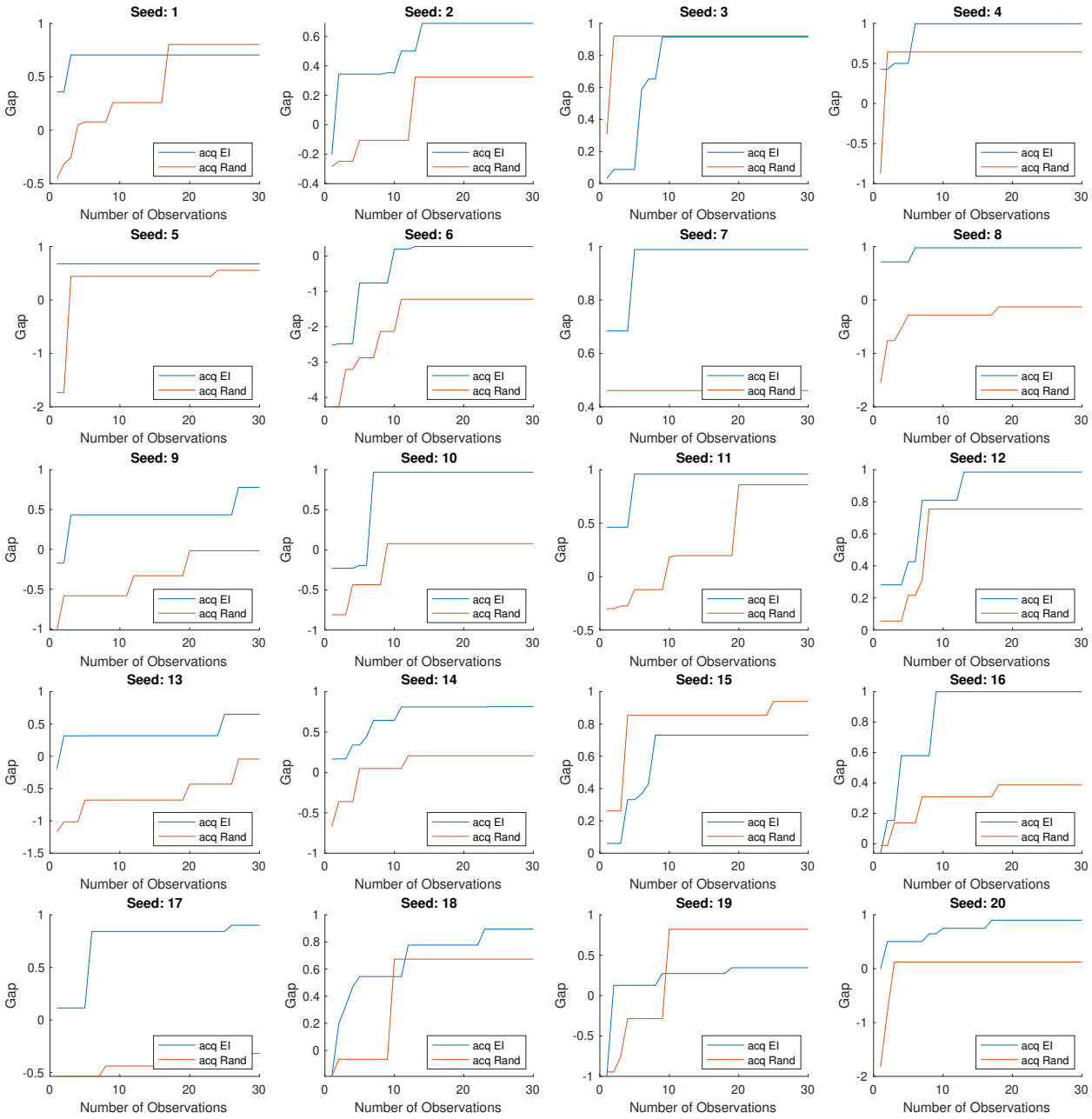

Figure 15: Learning curves for 20 branin function experiments. The EI acquisition function is shown in blue, and the RS acquisition function is shown in orange. Observe that EI consistently outperforms RS in both rate of GAP increase, and final GAP convergence at 30 observations.

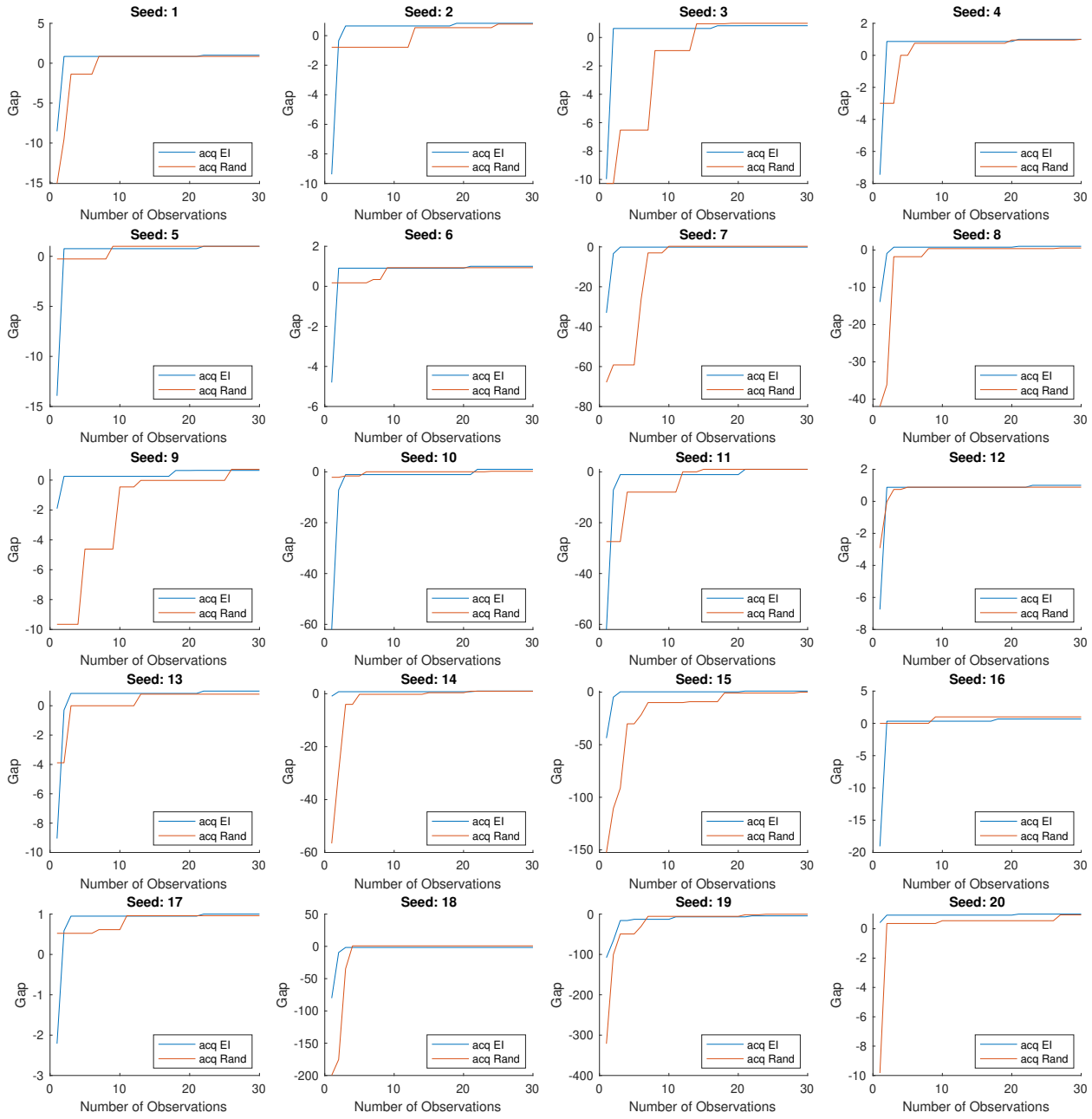

Figure 16: Learning curves for 20 LDA function experiments. The EI acquisition function is shown in blue, and the RS acquisition function is shown in orange. Note that EI and RS both consistently converge towards the same GAP value at 30 observations

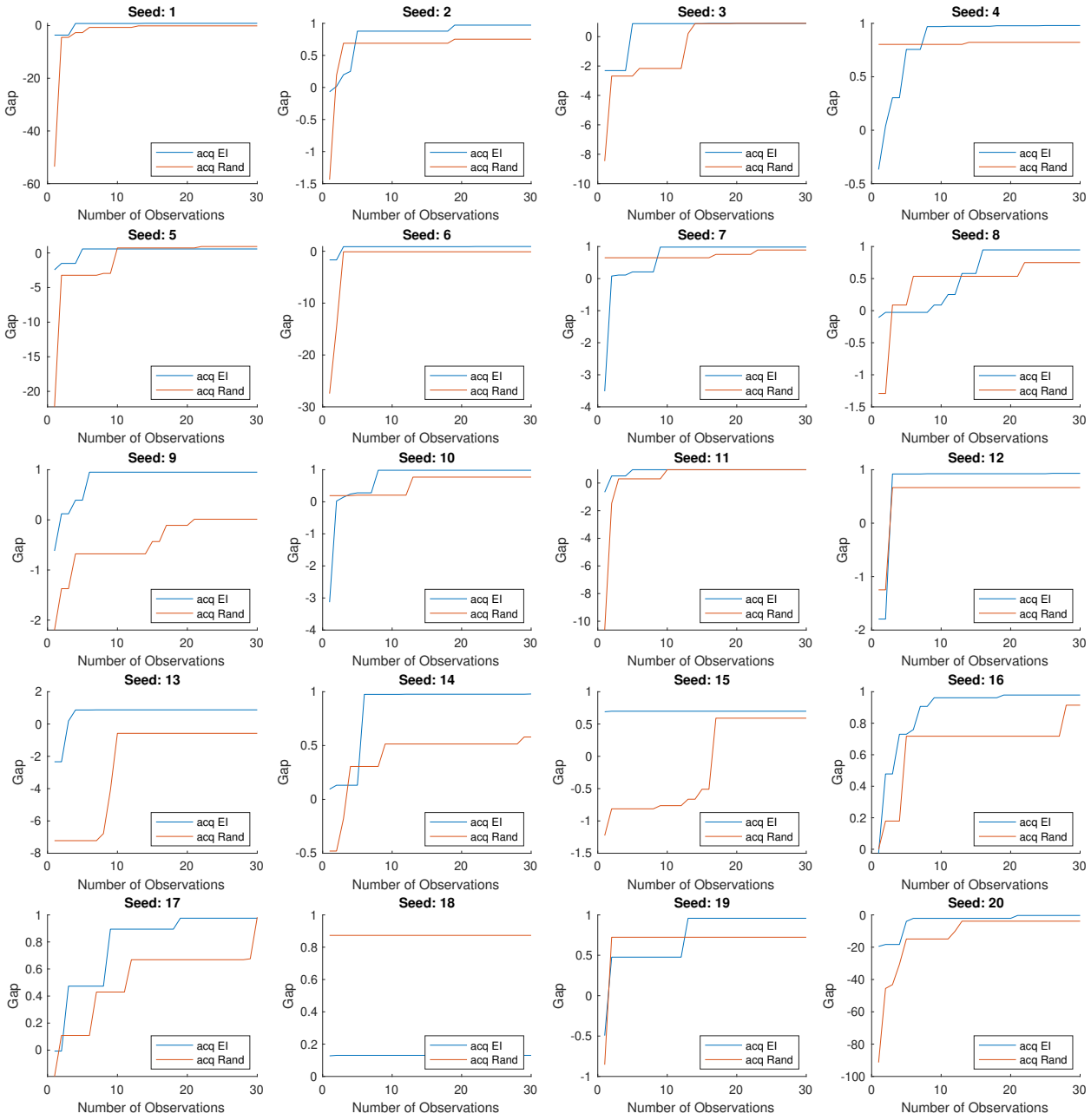

Figure 17: Learning curves for 20 SVM function experiments. The EI acquisition function is shown in blue, and the RS acquisition function is shown in orange. It can be seen that EI performs at least as well as RS in the majority of experiments, but can find and get stuck in a local minima, as seen in Seed 18.

From figures 15, and 17 above, it can be seen that EI significantly outperforms RS on both the Branin Function and the SVM Function. Note that in these experiments, EI often converges to its final GAP value in significantly fewer observations than RS does, and that its final GAP values at 30 observations are generally higher than those of RS. This behavior was expected, as EI combines both exploitation and exploration together to identify locations with higher expected benefit, whereas RS only applies (random) exploration. The addition of exploitation and non-random exploration in EI allows it to actually navigate towards optima, whereas RS can only approach

optima through luck. However, there are cases where EI and RS have similar performance: from figure 16, it can be seen that the two acqusition functions perform similarly to each other in the majority of experiments at 30 observations. We believe that EI's lack of performance on the LDA function can be attributed to a large amount of local minima; EI can often become stuck in local minima, thereby decreasing its performance, whereas RS is purely exploratory and therefore unhindered by the presence of any local optima.

The difference in performance between EI and RS can be further seen when one considers their mean gaps at 30, 60, 90, 120 and 150 observations, as shown in the table below:

# Observations	EI (Branin)	RS (Branin)	EI (LDA)	RS (LDA)	EI (SVM)	RS (SVM)
30	0.8072	0.3409	0.4877	0.7122	0.8106	0.3728
60	0.8532	0.6086	0.8611	0.7944	0.8415	0.5676
90	0.8532	0.7124	0.8753	0.8174	0.8592	0.8190
120	0.8532	0.8096	0.8753	0.8633	0.9018	0.8716
150	0.8532	0.8202	0.9473	0.9016	0.9018	0.8737

Table 2: Mean Gaps of EI and RS on the Branin, LDA, and SVM Functions at 30, 60, 90, 120 and 150 observations. Note that EI reaches a higher Gap at 150 observations in all functions when compared to RS. Furthermore, for the Branin and SVM functions, EI achieves these higher values significantly faster than RS.

In all cases, it is evident that EI outperforms RS on all three functions, including LDA, at 150 observations. Additionally, EI requires less observations to achieve higher gap values, implying a significantly faster learning rate than RS. We also note that EI likely spends considerable time navigating local minima in the LDA function, as evident by its noticeably worse performance at 30 observations when compared to that of RS. However, EI for LDA rapidly improves following this local-minima exploration and quickly outperforms RS. It is also important to note that on average, no method was able to find the global minima. This result was expected for RS, which would need to be extraordinarily lucky to find the global minima, but was somewhat surprising for EI. It appears that in the limit, EI may become entrenched in local minima and thus converges towards a suboptimal result.

We more rigorously compare the performance of EI and RS on all three functions using a paired t-test and evaluate the null hypothesis that the gap values attained for EI and RS come from the same distribution (i.e. the performance of EI and RS are equivalent or similar). We begin the test using only 30 observations for both EI and RS and calculate the corresponding p-values, as seen in the table below:

Function	P-Value
Branin	5.39E-04
LDA	0.3774
SVM	0.0387

Table 3:

We then begin RS at one observation and incremement the number of observations for RS while holding EI at 30 observations until the P-value exceeeds 0.05. The resultant number of observations before RS attains possibly similar performance as EI is shown below:

Function	Observations Required	P-Value
Branin	59	0.0624
LDA	10	0.0599
SVM	3	0.0951

Table 4:

Look at tables <><> and <><>, one can see that in the branin function, EI significantly outperforms RS. It isn't until 59 additional points have been observed before RS and EI show possibly similar performance. However, it is also evident that RS likely performs as well as EI on the LDA function. This was expected, given their similar learning curves above in figure 16. Perhaps more interestingly, the performance of RS varies significantly relative to that of EI; when there are less observations, the two acqusition methods demonstrate similar performance. However, as the number of observations increase, the two methods begin to differ. Another similarly iterative paired t-test experiment was performed, and it was determined that 44 new observations are required before the performance of EI and RS remain consistently comparable.

Bonus

For the bonus section, we implemented two more acquisition functions: max variance and lower confidence bound (LCB). We chose to implement max variance as one of our acquisition functions because it seemed like an intuitive way to explore a function's values and shape

$$\alpha_{\max \mathrm{Var}}(x) = K(x, x) \tag{0.6}$$

We then compared their performances with EI when used on their own and with two heuristics.

First, we implemented a wrapper for aquisition functions that selects a point at random with probability $p = 0.1$; we dubbed this heuristic "random restarts" (RR). We also tried each aquisition function while minimizing the model's hyperparameters after each iteration; we called this "online optimization" (OO).

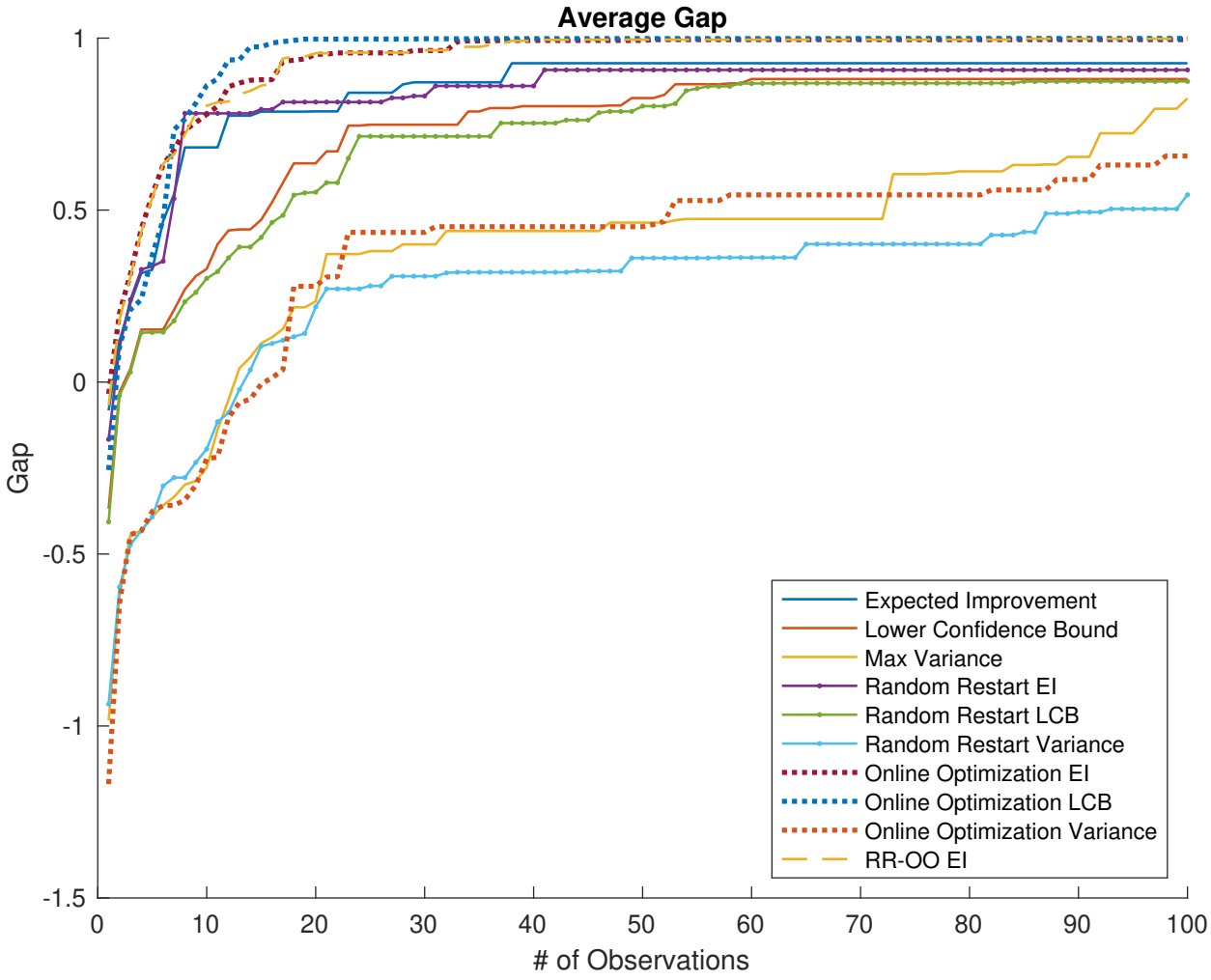

Figure 18: A busy plot showing gap as a function of the number of observations made.