Gaussian Process Regression using GPyTorch

Lorena Magaña Zertuche
(Dated: September 1, 2022)

GstLAL, one of LIGO's detection pipelines, relies on the use of matched-filtering. This allows us to get an estimate of the initial system parameters, such as masses, and spins. However, there is a significant discrepancy between the true values of the parameters and the recovered values. By using machine learning methods, such as Neural Networks and Gaussian Process Regression, one can decrease the errors between true and recovered values. A better parameter recovery means that we can better understand whether or not the remnant of a binary system will produce an electromagnetically bright event. This, in turn, will improve the reliability of the events sent out to astronomers. In this work, we show the results of using Gaussian Process Regression on the parameters of simulated (or injected) signals.

I. INTRODUCTION

These document will outline the use of Gaussian Process Regression using `GPyTorch` and keep results as a means to keep this process as transparent as possible. `Scikit-learn` was explored earlier during the IPAM long program but its lack of flexibility when it comes to incremental learning and fine-tuning has made us move away from its further usage. Additionally, we looked into the GPR implementation of `TensorFlow` since it gives us the desired flexibility. However, both of these methods lack the speed of `GPyTorch`. For example, training and testing on the `GstLAL` fake data took approximately 4 hours for both `Scikit-learn` and `TensorFlow`. However, the same exact run took less than 4 minutes with `GPyTorch`. This is mainly due to the Lanczos Variance Estimates, or LOVE, method for fast variances and sampling introduced in (https://arxiv.org/abs/1803.06058).

II. CONDITIONING THE DATASET

Thus far, GPR has been tested using a dataset generated from `GstLAL` early warning triggers. This low-latency pipeline uses matched-filtering techniques for the detection of gravitational-wave signals from compact binaries. The template bank of simulated binary neutron stars is generated using the TaylorF2 waveform approximant. The waveforms are for non-spinning systems with masses $m_1 > 0.95\mathrm{M}_\odot$ and $m_2 < 2.4\mathrm{M}_\odot$.

Once the data is read, we prepare it for regression in the following way:

(i) Shuffle the data.

(ii) Constrain the data to ensure positive values in mass predictions. This requires

  (a) Mapping the data to be in the range from $(0, 1)$. This alone does not ensure positive values it is an intermediary step.

  (b) Mapping the data to be in the range from $(-\infty, +\infty)$.

(iii) Shuffle the data.

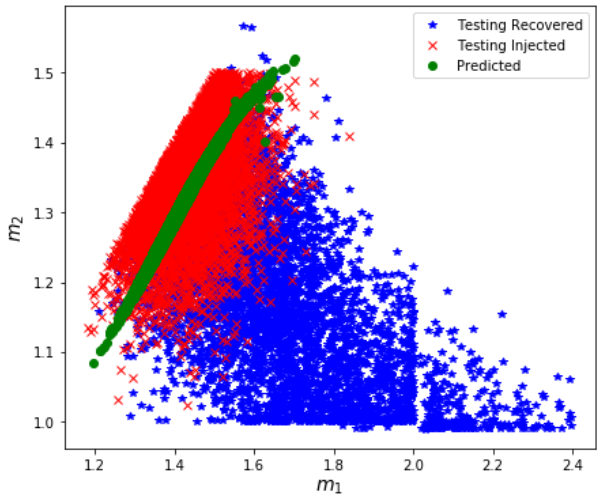

FIG. 1. The m_1-m_2 parameter space of injections is shown in red, the recovered values from matched-filtering are in blue, and the predicted values from regression are in green.

(iv) Standardize the data so the data has a zero mean and unit variance.

Once the algorithm gives us the predicted data, the reverse process is done, i.e., the data is scaled back from standardization, mapped back to $(0, 1)$, and exponentiated.

III. METHODOLOGY

The kernel used for training is the radial basis function (RBF) which is a squared exponential function given by

$$K(X_1, X_2) = e^{-\frac{||X_1 - X_2||^2}{2\sigma^2}}, \tag{1}$$

where X_1 and X_2 are input data points and σ is the variance, or lengthscale parameter. The predictions of the `GstLAL` dataset are independent of kernel choice. However, the RBF kernel, apart from its popularity due to its versatility, seems to perform slightly better than other kernels and without a loss of speed.

The Adams optimizer is used to find the optimal hyperparameters with a learning rate of 0.1 and 50 iterations. This combination of learning rate and iterations allows us to decrease the marginal log likelihood, or loss, at a quick rate without loss of accuracy.

Both the GstLAL and O2 datasets are divided as 58% training and 42% testing.

IV. GSTLAL RESULTS

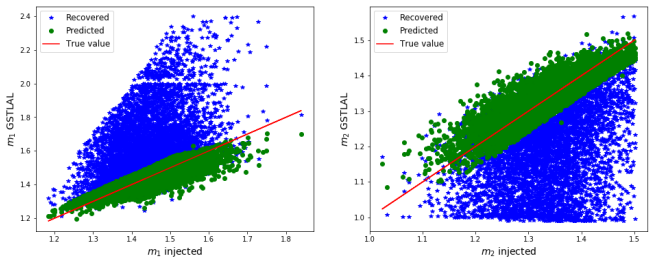

FIG. 2. The left panel show the injected (red line), recovered (blue stars), and predicted (green circles) values for m_1 while the right panel shows the values for m_2. Regression is significantly better at producing the true values.

After running GPR on the datasets, we are able to better predict the masses of the binary systems. Fig. 2 shows the advantages of using regression. The values of the injected masses are depicted by the red line while the recovered masses are shown by the blue stars. The predicted masses, shown in green, are much more closely aligned in parameters space to the true values. Although the results from GPR are not perfect, the comparison between the recovered and predicted values is clear. Moreover, as illustrated in Fig. 3 the error in the recovery of the primary mass increases as the injected m_1 increases and the error in the secondary mass decreases as the injected m_2 value increases. Nevertheless, the error using the predicted values stays around zero regardless of the values of the masses injected.

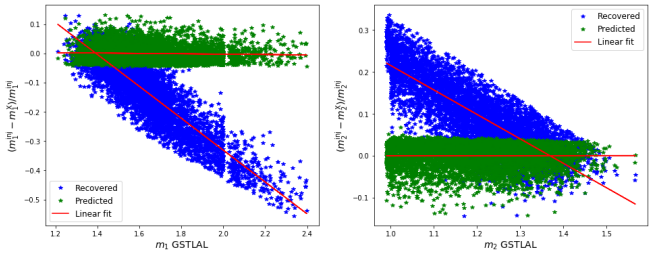

FIG. 3. The left/right panel show the relative errors in the recovered (blue) and predicted (green) values for m_1/m_2. The red line is a fit of these values, which shows a relationship between injected m_1/m_2 values and the recovered and predicted values. GPR has no significant bias that depends on the recovered mass while the GstLAL data does.

A different way to visualize Fig. 3 is by creating a histogram showing how the recovered and predicted values vary from those of the injected values. As seen in Fig. 4 the error in the recovered masses is very spread out with errors as high as 50%. However, using regression allows us to narrow down the errors to no more than about 10%. Overall, we are able to decrease the mean of the error by as much as three orders of magnitudes.

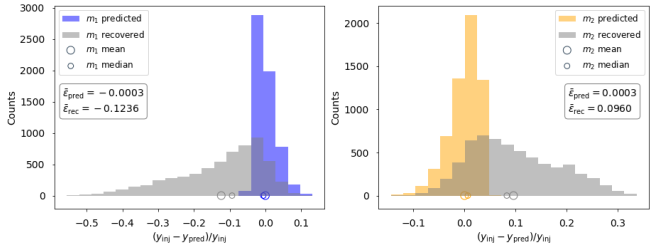

FIG. 4. The histogram on the left show the relative error between the injected m_1 values and both the recovered (gray) and predicted (blue) values. Similarly, the right panel shows the relative error between the injected m_2 values and both the recovered (gray) and predicted (orange) values. The mean of each dataset is shown by circles on the horizontal axis with the values in boxes. GPR does a great job at decreasing the mean error of the masses.

A. Statistical Analysis

Shapiro-Wilk Test. To better understand the dataset and the predictions, we perform some statistical tests. First, we check for the Gaussianity of the data by performing a Shapiro-Wilk test, i.e., by comparing the p-values of the injected, recovered, and predicted values, we are able to determine whether the results are normally distributed. Small p-values correspond to distributions that are not normal. As shown in Fig. 5, the injected masses are relatively normal with p-values of 10^{-5} and10^{-12} for m_1 and m_2, respectively. Instead the p-values for the recovered masses are on the order of 10^{-38} and10^{-25} which indicates that the distribution cannot be considered "as normal" as the injected distribution. Furthermore, we use the same test for the predicted values and notice that the p-values resemble those of the injected masses. This tells us that using regression not only decreases the errors in the masses but also takes us back to a distribution similar to the injected one. Therefore, we can say that the predicted values agree with the null hypothesis of normality.

As an extra step, I was interested to see how the p-values change with the amount of data included. This is because for more than $5,000$ data points, the p-value test may not be accurate. There are a couple of references to add here, but I will come back to that soon. Meanwhile, Fig. 6 show how the p-value changes as one add more data. Although the injected and predicted p-values seem to diverge slowly with an increasing number of data points, the values are

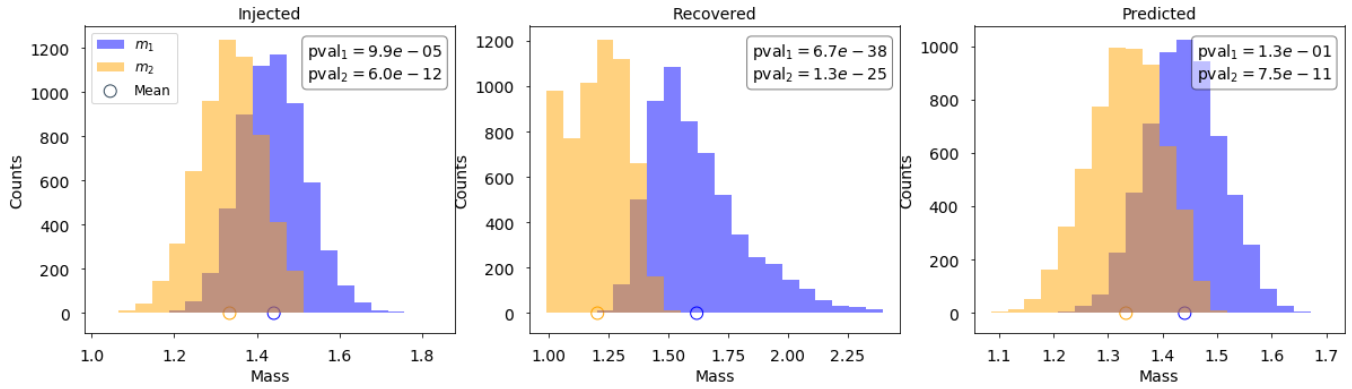

FIG. 5. The histograms show the mass distributions for the injected (left panel), recovered (middle panel), and predicted (right panel) masses. The p-values corresponding to the Shapiro-Wilk test for normality are printed in the plots. Note that the injected and predicted panels have similar p-values, showing that their distributions are similar in Gaussianity. On the other hand, the recovered p-values are extremely small. This means that they reject the null hypothesis of normality. The distributions are for $N = 4,000$

still considered to be similar. Not surprisingly, the p-value of the recovered masses reject the null hypothesis that the data is normally distributed.

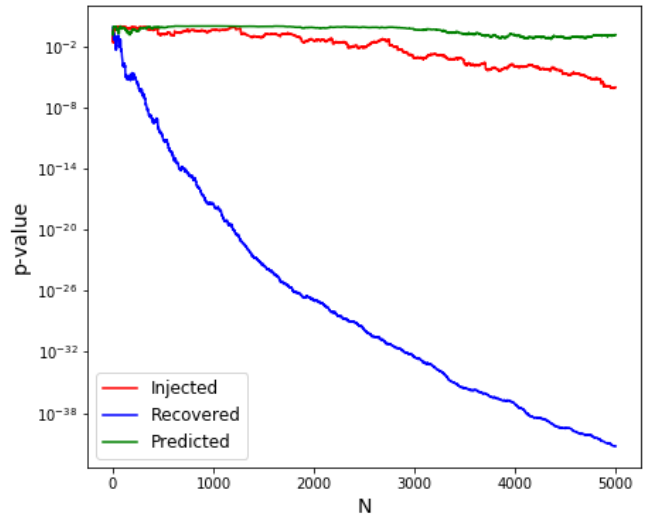

FIG. 6. The p-value as a function of number of data points is shown for the injected, recovered, and predicted data. Note that the p-values between the injected and predicted data sets are always closer together than those of the recovered data.

Epps-Singleton Test. The Epps-Singleton Test allows us to check if two sets of data have the same underlying probability distribution. A p-value of 1 means that two set of data have the same probability distribution while a p-value of 0 tells us that the two datasets are completely different in their distributions. When comparing the injected and recovered values, the p-value is $2.5 * 10^{-22}$. On the other hand, comparing the injected and predicted values gives us a p-value of 0.32.

Kolmogorov-Smirnov Test. This last test compares the underlying continuous distributions of two independent samples. What we find that that the p-value for the injected and recovered distributions is 0, i.e., the distribution of the injected values are not equal to the distribution of the recovered values. Instead for the predicted values, we get a p-value of 0.002 which tells us that although the continuous distributions are not the same, they are not completely different since there is some overlap. Fig. 7 shows the distribution of data as a function of mass. This visual shows us clearly that the recovered values for m_1 tend to overestimate the mass and the recovered m_2 values mostly underestimate the mass. This is corroborated by calculating the p-values. For example, the p-value between the injected and recovered m_1 is 1 when the null hypothesis is that the underlying distribution of the injected parameters is less than the underlying distribution of the recovered parameters. In other words, the recovered masses overestimate the values. The converse gives us a p-value of 0. If instead we turn our attention to the right panel of Fig. 7, we can test for the distribution of the m_2 values. The p-value between the injected and recovered m_2 is 1 when the null hypothesis is that the underlying distribution of the injected parameters is greater than the underlying distribution of the recovered parameters. When comparing the predicted data, however, the p-values are small but not so small that any null hypothesis can be rejected. "Any" meaning that the continuous distribution of the predicted values is similar to the injected values.

V. O2 DATA

The analysis below is done using a training data set consisting of 139,999 points and a testing set containing 59,999 points with 4 features, namely the initial masses and the initial spin magnitudes of the black holes and neutron stars. This gives us a training/testing breakdown of 57/43. Since we are interested in regressing for a large

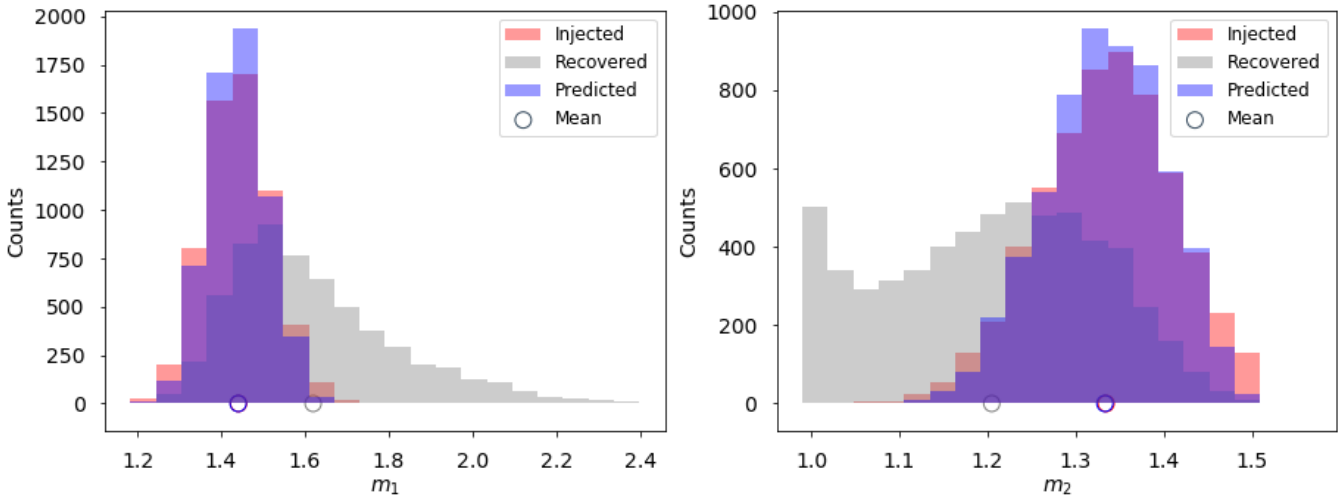

FIG. 7. The panels show the mass distributions for the injected, recovered, and predicted datasets. This hints at the biases in the recovered data.

parameters space, it is essential that our training data covers as much of that parameter space as possible.

A. Conditioning the Dataset

The training data set has mass ranges $m_1 = [1.02, 116]$ and $m_2 = [0.81, 81]$ and spin ranges $\chi_{1,2} = [-0.99, 0.99]$, as shown in Fig. 8

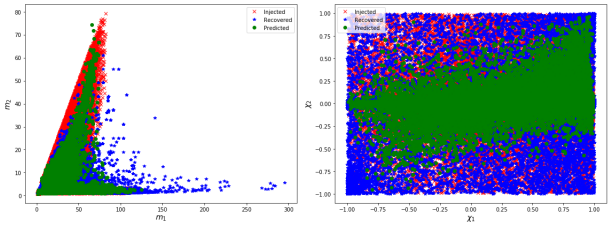

FIG. 8. The m_1-m_2 (left) and χ_1-χ_2 (right) parameter space of injections is shown in red, the recovered values from matched-filtering are in blue, and the predicted values from regression are in green.

Once the data is read, we prepare it for regression in the following way:

(i) Constrain the data to ensure positive values in mass and spin predictions. This requires

  (a) Mapping the data to be in the range from $(0, 1)$. This alone does not ensure positive values; it is an intermediary step. This is done separately for the masses and spins since they span a different range of values.

  (b) Mapping the data to be in the range from $(-\infty, +\infty)$.

(ii) Standardize the data so the data has a zero mean and unit variance.

Once the algorithm gives us the predicted data, the reverse process is done, i.e., the data is scaled back from standardization, mapped back to $(0, 1)$, and exponentiated.

B. Memory and Speed

Running the code in the standard way leads to out-of-memory (OOM) issues. They usually manifest themselves during training when the loss function is assigned and when the algorithm takes a step backwards. Other OOM problems arise during prediction. There are several ways one can decrease memory usage and increase speed – some fixes working better than others. In this work, we ran the algorithm in both CPUs and GPUs. By employing the following methods in our algorithm we decreased the running time from over 1 month to just under an hour.

(i) *Automatic Mixed Precision (AMR).* Most ML algorithms operate using 32-bit floating-point data, but this high level of precision requires more memory storage than a lower level. In AMR, half precision is used in calculations that do not require single precision without a loss in accuracy. For example, calculations starting with half-precision values end with single precision values. This allows for a speed-up of up to 25x in calculations and a decrease in memory usage and power consumption.

(ii) *Kernel Operations (KeOps).* KeOps is a library that allows for efficient calculations of kernel matrix-vector products that may otherwise require large memory usage and heavy data transfer. It also contains a conjugate gradient solver that is used

in GPR. Its memory footprint is linear instead of quadratic for many types of computations. By taking advantage of the structure of CUDA, speed-ups of 10x-100x are provided for `PyTorch` GPU programs.

(iii) *Garbage Collection (GC).* GC is a memory management tool to free up memory allocated by the program that is no longer needed.

(iv) *zero_grad(set_to_none=True).* `PyTorch` accumulates gradients during training which can be useful in certain situations, but in training with GPR, we want the parameters to be updated for each iteration. Otherwise, the gradient will point the parameters to be in a direction other than the minimum of the loss function. By resetting the gradients to None instead of a tensor of zeroes, memory usage decreases.

VI. RESULTS

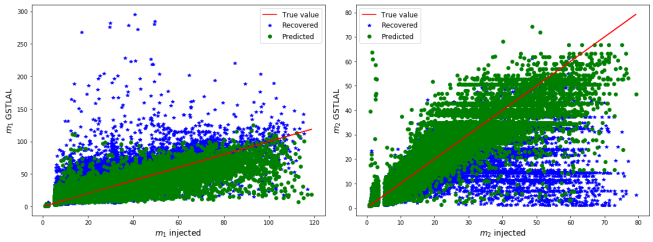

FIG. 9. The left panel show the injected (red line), recovered (blue stars), and predicted (green circles) values for m_1 while the right panel shows the values for m_2.

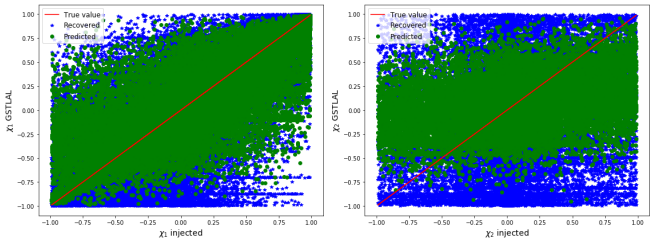

FIG. 10. The left panel show the injected (red line), recovered (blue stars), and predicted (green circles) values for χ_1 while the right panel shows the values for χ_2.

After running GPR on the dataset we are able to better predict the masses of the binary systems. Fig. 9 shows the advantages of using regression. The values of the injected masses are depicted by the red line while the recovered masses are shown by the blue stars. By eye, the predicted masses for the primary black hole, shown in green, look more closely aligned in parameters space to the true values. Although the results from GPR are not perfect, the difference between the recovered and predicted values is clear especially in cases when the recovered masses are very high. In Fig. 9, the positive effect of regression on the secondary mass is even more clear since the predicted masses are more tightly bound to the true masses with the exception of a handful of predictions in the neutron star mass range. Similarly, in Fig. 10 the primary spins are closer to the true value using regression. Although both the recovered and the predicted values have seemingly large errors, the predictions improve the overall error significantly. This will be quantitatively shown in the text that follows.

To get a better idea of the errors and how they correlate with the value of the masses and spins, we have Fig. 11 and Fig. 12. A linear fit for the recovered and predicted values is depicted by the magenta and red lines, respectively. One can see that for larger recovered masses the relative error increases but the same is not true for predicted masses. The differences of the relative errors for the recovered and predicted values are much smaller for the secondary masses, but overall, the fit of the predicted values is closer to the zero-error line. For the spins, we instead calculate the difference between the injected and the recovered/predicted values to avoid an infinity for zero-spin injections. Based on the fits of the errors for highly positive and highly negative spins, the recovered values are underestimated and overestimated, respectively, by a large amount. The predicted values, although not in a narrow band, are found to be constantly around a difference of zero. This effectively gets rid of the high difference in the highly negative and positive spin cases.

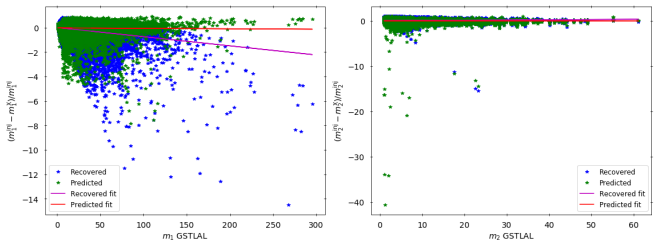

FIG. 11. The left/right panel show the relative errors in the recovered (blue) and predicted (green) values for m_1/m_2. The red line is a fit of these values, which shows a relationship between injected m_1/m_2 values and the recovered and predicted values. GPR has no significant bias that depends on the recovered mass while the `GstLAL` data does.

A different way to visualize errors is by creating a histogram showing a distribution of how the recovered and predicted values vary from those of the injected values. As seen in Fig. 13 the error in the recovered masses is very spread out with a handful of outliers in the prediction of the secondary mass. These outliers are neutron star mass injections that were predicted to be an order of magnitude larger. However, using regression allows us to narrow down the errors of each parameter significantly. More specifically, the mean recovered errors for m_1 and m_2 are 34.8% and 25.6%, respectively, while the mean

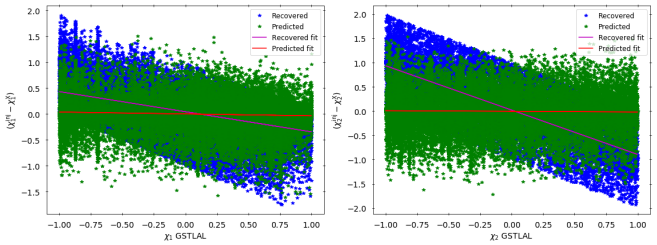

FIG. 12. The left/right panel show the relative errors in the recovered (blue) and predicted (green) values for m_1/m_2. The red line is a fit of these values, which shows a relationship between injected m_1/m_2 values and the recovered and predicted values. GPR has no significant bias that depends on the recovered mass while the `GstLAL` data does.

predicted errors are 12.7% and 11.3%. Therefore, we can conclude that regression decreases mean errors for the masses by more than half. The spins also show an improvement. However, since we cannot calculate the mean error due to zero spin values, we calculate the differences. By using GPR the tails of the differences become smaller. The mean recovered differences for χ_1 and χ_2 are -0.0019 and 0.066 while the predicted values are closer to zero with mean differences of -0.0019 and -0.0031. Note that the secondary spin gains an order of magnitude improvement with prediction. For both the masses and spins, the mean errors are illustrated using the circle on the zero horizontal line.

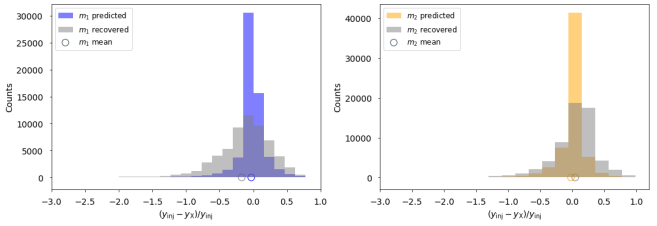

FIG. 13. The histogram on the left show the relative error between the injected m_1 values and both the recovered (gray) and predicted (blue) values. Similarly, the right panel shows the relative error between the injected m_2 values and both the recovered (gray) and predicted (orange) values. The mean of each dataset is shown by circles on the horizontal axis with the values in boxes. GPR does a great job at decreasing the mean error of the masses.

A. Statistical Analysis

Shapiro-Wilk Test. To better understand the dataset and the predictions, we perform some statistical tests. First, we check for the Gaussianity of the data by performing a Shapiro-Wilk test. That is, by comparing the p-values of the injected, recovered, and predicted values, we are able to determine whether the results are normally distributed. Small p-values correspond to distributions

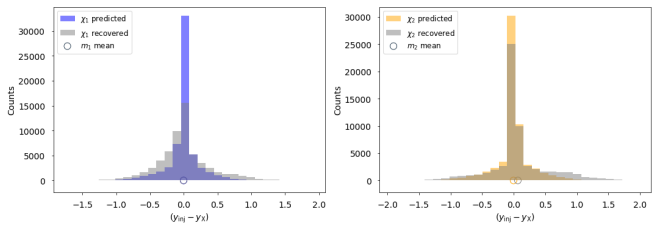

FIG. 14. The histogram on the left show the relative error between the injected m_1 values and both the recovered (gray) and predicted (blue) values. Similarly, the right panel shows the relative error between the injected m_2 values and both the recovered (gray) and predicted (orange) values. The mean of each dataset is shown by circles on the horizontal axis with the values in boxes. GPR does a great job at decreasing the mean error of the masses.

that are not normal. As seen in Fig. 15, the parameter space of the injections do not resemble a Gaussian distribution. In fact, the p-values for the injected, recovered, and predicted m_1, m_2, and χ_2 are zero. For the primary spin, the p-values have orders of magnitude 10^{-28}, 10^{-34}, and 10^{-26}, respectively. Note how the predicted data recovers the distribution of the injected data more closely than does the recovered data. This hints that regression not only decreases the error in the mass and spin values but also takes us back to more similar distribution.

As an extra step, I was interested to see how the p-values change with the amount of data included. This is because for more than $5,000$ data points, the p-value test may not be accurate. There are a couple of references to add here, but I will come back to that soon. Meanwhile, Fig. 16 shows how the p-value changes as one add more data. Although the injected and predicted p-values seem to diverge slowly with an increasing number of data points, the values lies more closely to each other than to the recovered data.

Epps-Singleton Test. The Epps-Singleton Test allows us to check if two sets of data have the same underlying probability distribution. A p-value of 1 means that two set of data have the same probability distribution while a p-value of 0 tells us that the two datasets are completely different in their distributions. When comparing the injected and recovered values of all four parameters, the p-values are $\{m_1 = 10^{-152}, m_2 = 0, \chi_1 = 0, \chi_2 = 10^{-54}\}$. On the other hand, comparing the injected and predicted values gives us p-values $\{m_1 = 10^{-77}, m_2 = 10^{-21}, \chi_1 = 10^{-123}, \chi_2 = 10^{-187}\}$. In conclusion, only the probability distribution of the predicted primary spins has any resemblance to the injected distribution.

Kolmogorov-Smirnov Test. This last test compares the underlying continuous distributions of two independent samples. We find that the p-values for the injected and recovered continuous distributions are zero. For the injected and predicted values, the continuous distributions resemble each other more except in the case of the secondary spin where the p-values is zero.

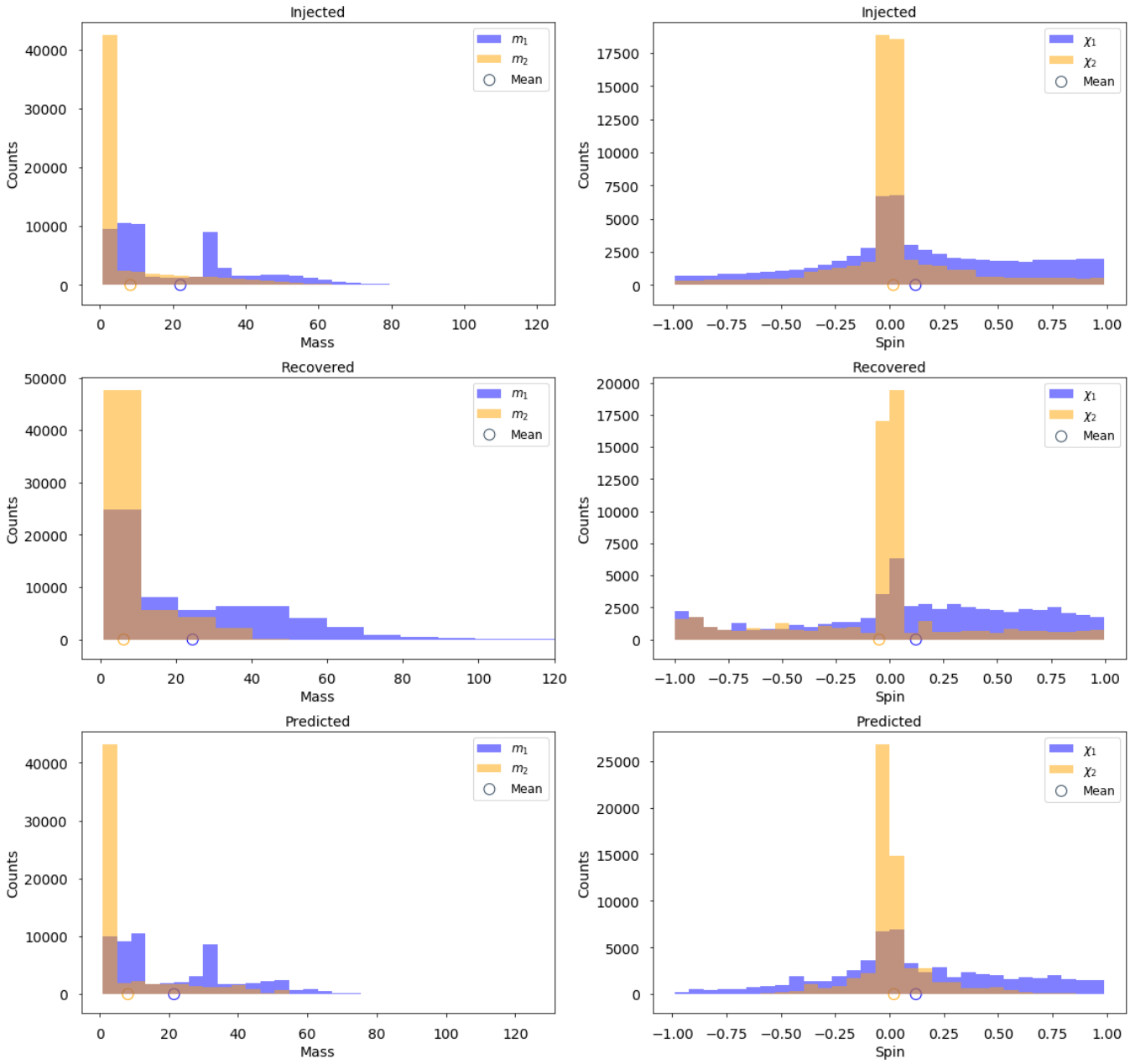

FIG. 15. The histograms show the mass distributions for the injected (top panels), recovered (middle panels), and predicted (bottom panels) masses.

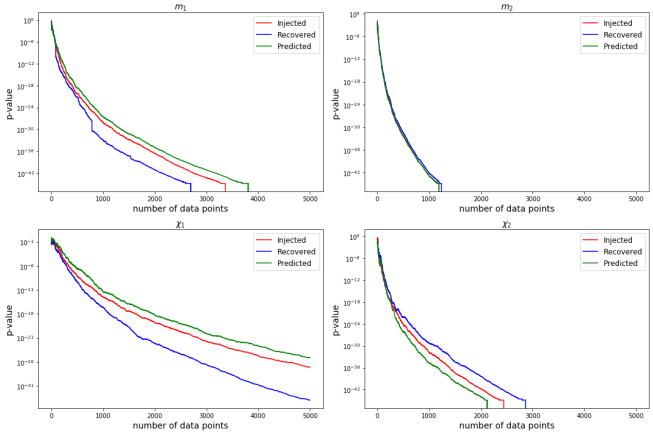

FIG. 16. The p-value as a function of number of data points is shown for the injected, recovered, and predicted data. Note that the p-values between the injected and predicted data sets are always closer together than those of the recovered data.