1 Gaussian Process Regression

Gaussian process regression is a statistical learning technique that permits a computationally tractable description of a "distribution of functions." Specifically, a gaussian process is a non-parametric model that allows do regression without explicitly specifying a form that the fit should take. This makes it especially useful for QCD background estimation, where there is no apriori description for the background, often resigning analyses to pick a random function that seems to fit the data.

1.1 Mathematical Description

Definition 1.1. A Gaussian Process (GP) is a stochastic process for which any finite subset of the random variables is jointly gaussian.

Being a gaussian, a GP $f(x)$ is defined by its mean and its covariance.

$$m(x) = \mathbb{E}\left(f(x)\right) \tag{1}$$
$$k(x, x') = \mathbb{E}\left((f(x) - m(x))(f(x') - m(x'))\right) \tag{2}$$

We write such a process as $\mathcal{GP}(m(x), k(x, x'))$. Note that in this case, the random variables are $f(x)$, meaning there is a separate random variable for each value of x. In common statistics language, x plays the role of the index in a random process.

Note that there is an inherent self-consistency requirement here. If we are looking at some collections of points X_1 and another collection X_2 then the process must behave consistently regardless of whether we look at the join distribution of $X_1 + X_2$ or the sets separately.

Since a multivariate gaussian has this property inherently, everything works out if $k(x, x') = C_{x,x'}$ where C is the multivariate covariance.

1.2 Distribution

The combination of the mean,covariance, and self-consistency requirements means that a Gaussian process described a distribution over a space of functions. Specifically, if we select test points X_* then the

$$P(f_*) \sim \mathcal{N}\left(m(X_*), k(X_*, X_*)\right) \tag{3}$$

1.3 Training

Ultimately we want to perform regression: taking known data and extrapolating in to some unknown region. To accomplish this we need to be able adapt our GP based on some known observations. This is precisely the domain of Baye's theorm: taking a prior distribution and some data, and forming a posterior distribution that describes our updated belief in the model.

Because the normal distribution behaves very nicely under conditioning and marginalization, we can actually do this process with direct appear to Baye's theorem.

In general, we have some noisy data $y = f(x) + \sigma$, where σ gaussian distributed noise. Then we can write

$$cov(y) = K(X, X) + I\sigma_n^2 \tag{4}$$

Then the joint distribution of our noisy training points and our test points – the points where we want to perform regression – is

$$\begin{bmatrix} y \\ f_* \end{bmatrix} = \mathcal{N}\left(\begin{bmatrix} X \\ X_* \end{bmatrix}, \begin{bmatrix} K(X, X) + I\sigma_X^2 & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix}\right) \tag{5}$$

However, in almost all cases, the mean is set to 0, since

Conditioning this Gaussian distribution on the observed values y gives the posterior distribution over the test points X_*.

$$m(X_*) = \mathbb{E}\left[f_*|X, y, X_*\right] = K(X, X_*)\left(K(X, X) + \sigma_X^2 I\right)^{-1} y \tag{6}$$

$$K(X_*, X_*) = cov(f_*) = K(X_*, X_*) - K(X_*, X)\left(K(X, X) + \sigma_X^2 I\right)^{-1} K(X, X_*) \tag{7}$$

This is most tractable represented by considering a single test point x_*:

$$\mathbb{E}\left[f(x_*)\right] = k(X, x_*)^T\left(K(X, X) + \sigma_X^2 I\right)^{-1} y \tag{8}$$

$$Var(x_*) = K(x_*, x_*) - K(x_*, X)^T\left(K(X, X) + \sigma_X^2 I\right)^{-1} K(X, x_*) \tag{9}$$

The mean prediction for the point x_* is a linear combination of the test points y weighted by their covariance with the test point. The variance of the point is simply the prior variance, modified some term involving only the training points (not their values).

1.4 Kernel Rundown

In most problems, the mean is simply taken to be zero, since it can always be added back, and in most cases the posterior mean adapts very well.

Of much greater interest is the kernel function. This is really the function that determines the behavior of the process, and is the subject of most study. This is perhaps not surprising – it is the kernel that determines how different datapoints interact, and so it is ultimately the kernel that furnishes predictive power.

Below we provide a rundown of some covariance functions.

1.4.1 RBF

The archetype for kernels is the Radial Basis Function, sometimes called the squared-exponential kernel. It is defined as

$$k(x, x') = e^{-\frac{(x-x')^2}{\ell^2}} \tag{10}$$

In multiple dimensions, such a kernel could instead be

$$k(x, x') = e^{-(x-x')^T M(x-x')} \tag{11}$$

where M is positive semidefinite, to ensure that this is, in fact, a kernel. This is the most frequently used kernel for GPs.

1.4.2 Matern Kernels

A generalization of the RBF kernel is the Matern kernel

$$k(x, x') = \frac{2^{1-\nu}}{\Gamma\nu}\left(\frac{(x-x')\sqrt{2\nu}}{\ell}\right)^{\nu} K_\nu\left(\frac{(x-x')\sqrt{2\nu}}{\ell}\right) \tag{12}$$

1.4.3 Rational Quadratic Kernel

The rational quadratic kernel is an infinite sum of RBF kernels with different length scales

$$k(x, x') = \left(1 + \frac{(x-x')^2}{2\alpha\ell^2}\right)^{-\alpha} \tag{13}$$

1.5 Model Selection

Thus far, all our prediction have assumed an existing form for our kernel, both in type and in choice of hyperparameters. In the real world, however, we need a method of determining these things obervationally.

The general conclusion is: select the model that minimizes the log marginal likelihood.

2 Regression for Combinatorial Backgrounds

In this ever growing section, we present ongoing work on developing background estimation in both 1 and 2 dimension space for combinatorial backgrounds.

The gist of the procedure for estimating the background for a certain mass point is this:

- Blind a window of size and shape X around the mass points
- Using a kernel X optimize the hyperparameters using this exterior region
- Use GP regression to estimate the background within the signal area.

Fundamentally, what is needed is a choice of window blinding area and kernel that allows for an accurate estimation of the background within the blinded in region. In two dimensions, this problem is much more challenging, since there is additional structure in 2 dimensions that disobeys the normal "smoothly falling everywhere" the prevails in 1 dimension.

We want to develop a gaussian process procedure that can accurately estimate the background in a blinded window of various sizes.

2.1 Points of Interest

There is a substantial amount of "parameter" space to cover as we work towards the determination of a good procedure. We want to study how the below interact in isolation and with each other.

Kernel The choice of kernel is the most important aspect of the GP.

Window Size/Shape The size and shape of the blinding window.

Valid Region The region of space we examine the regression, in particular if we exclude zero event bins.

Mean The choice of mean function.

Binning How bin size influences the fit.

We want to not only accurately estimate the background in the blinded region, but also be robust against variations in shape, binning, statistics, etc.

The majority of our efforts currently revolve around trying to determine an optimal kernel for the regression.

3 Kernel Studies

The choice of kernel is the most important aspect of Gaussian process regression. Here we document efforts to determine a kernel that provides both a flexible and accurate estimation of the background in the signal window.

We first do studies on the complete plane. This is used to determine if the fit is flexibile enough to accurately capture the features of the background.

3.1 Unblinded Studies

Of course, our estimation should perform well in the case that all points are available for training.

3.2 Blinded Studies

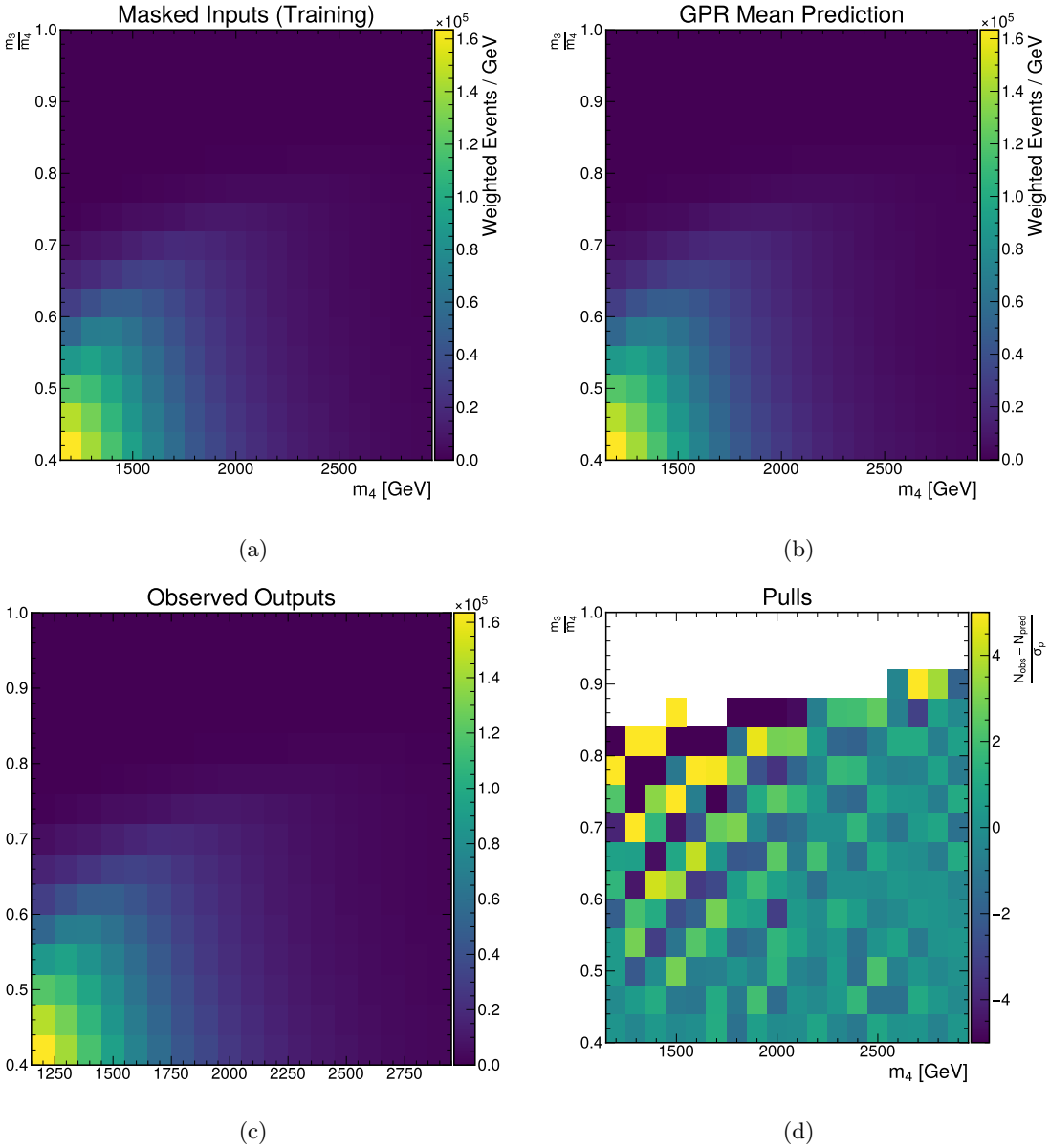
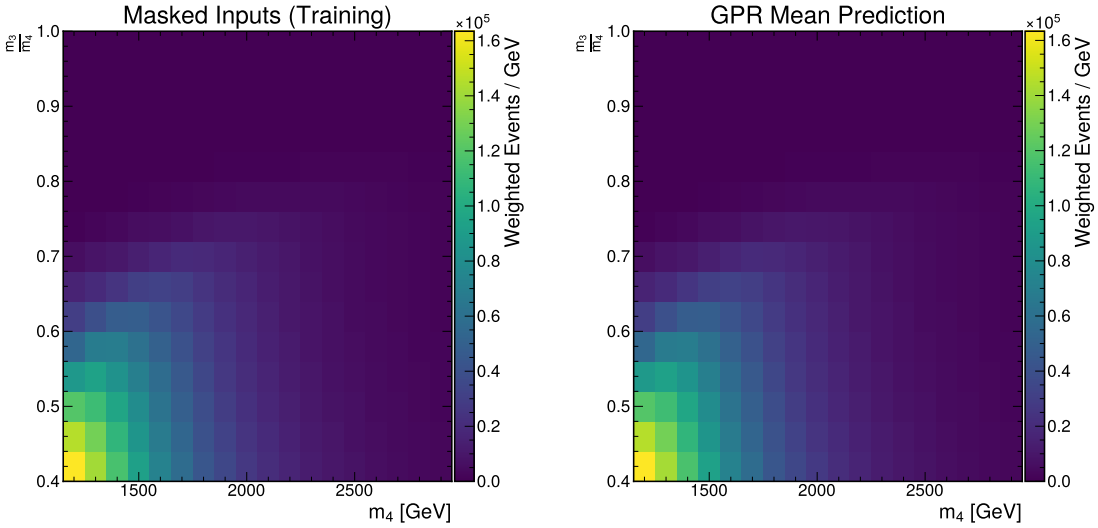
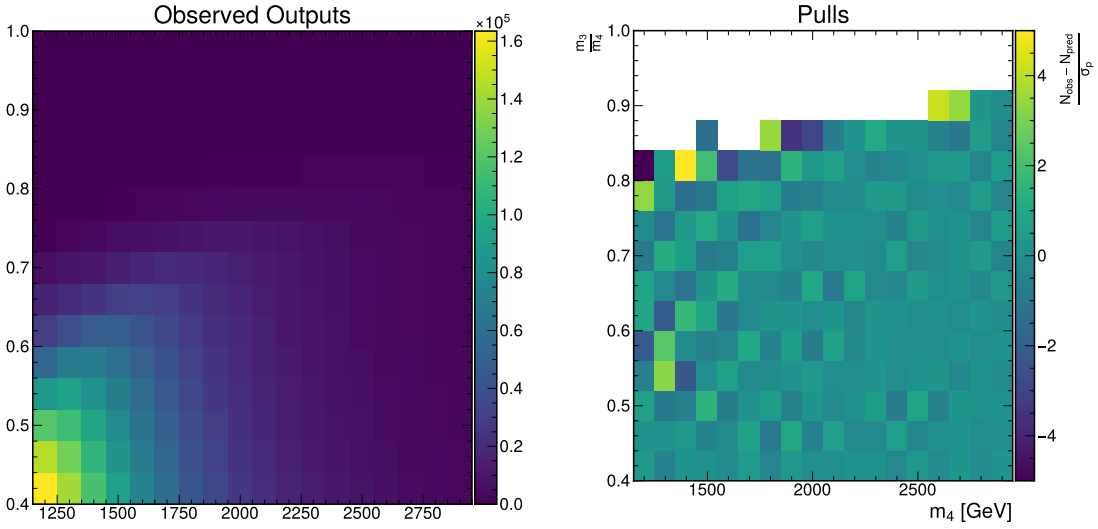


Figure 1: Performing regression on control region data using the plain RBF kernel with no blinding. Figure (a) shows the training data Figure (b) shows the posterior mean found using GP regression. Figure (c) shows the true outputs on the entire domain. Figure (d) shows the pulls for each bin.



(a) (b)



(c) (d)

Figure 2: Performing regression on control region data using the Matern kernel with no blinding. Figure (a) shows the training data Figure (b) shows the posterior mean found using GP regression. Figure (c) shows the true outputs on the entire domain. Figure (d) shows the pulls for each bin.

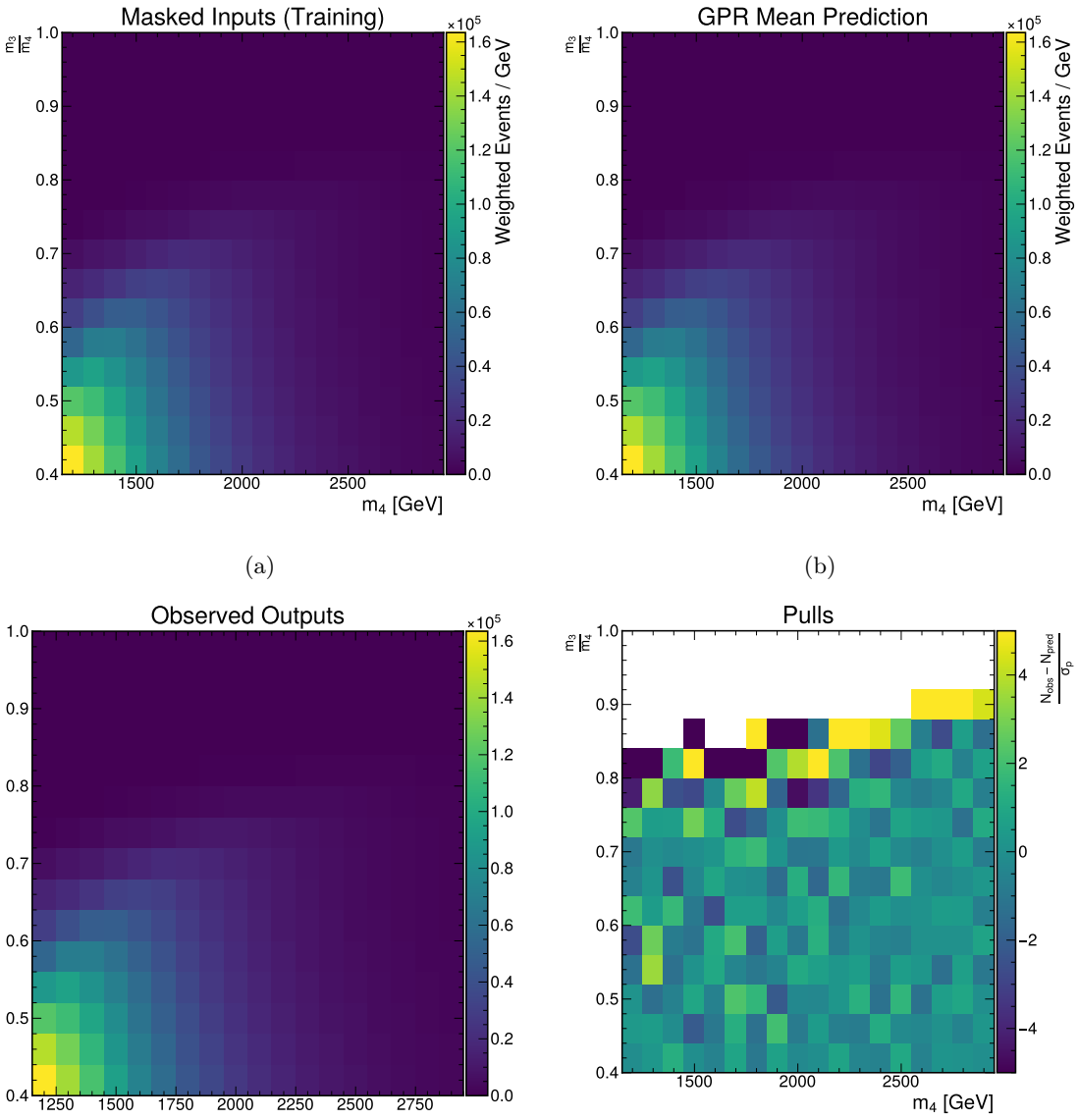
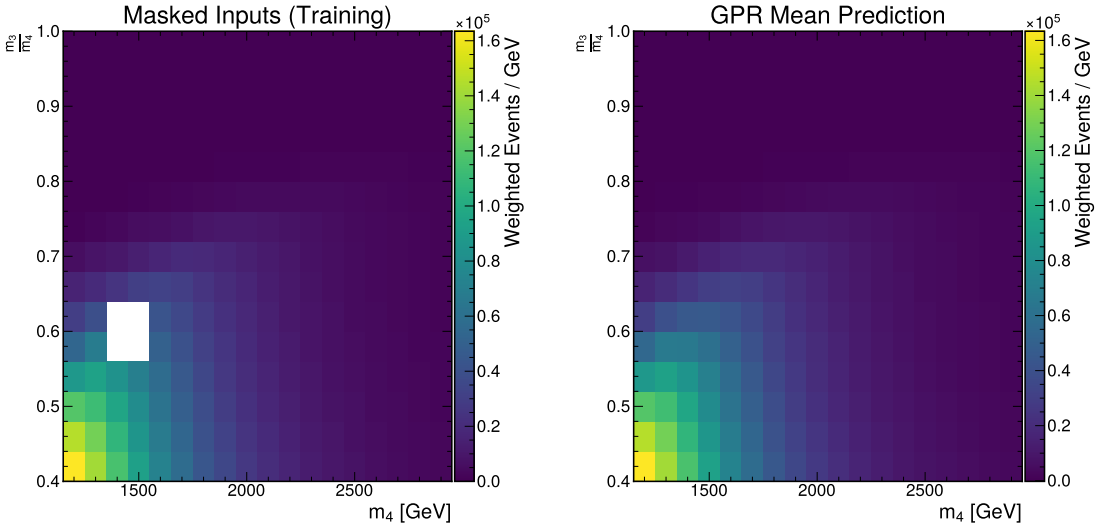
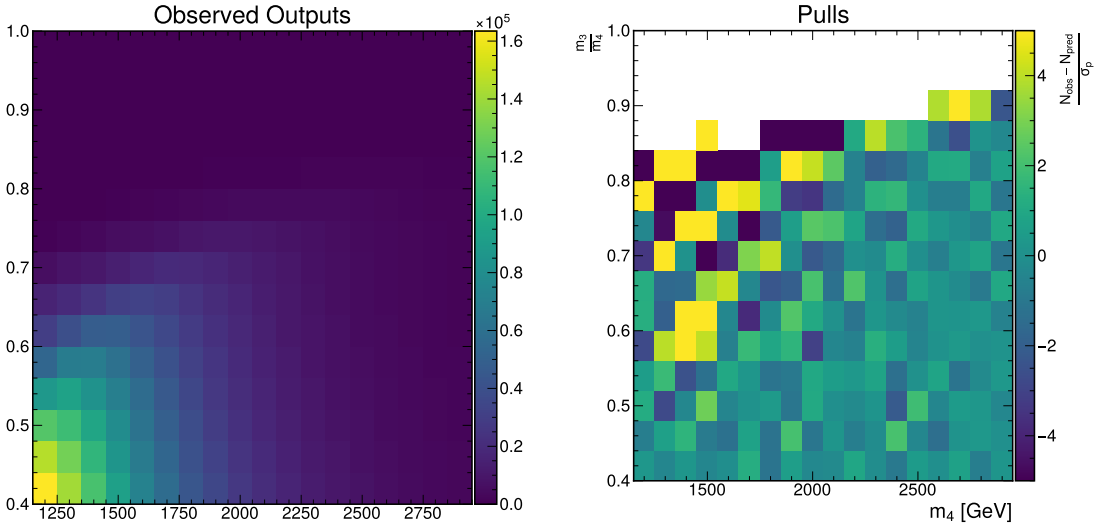


Figure 3: Performing regression on control region data using the Matrix RBF kernel with no blinding. Figure (a) shows the training data Figure (b) shows the posterior mean found using GP regression. Figure (c) shows the true outputs on the entire domain. Figure (d) shows the pulls for each bin.

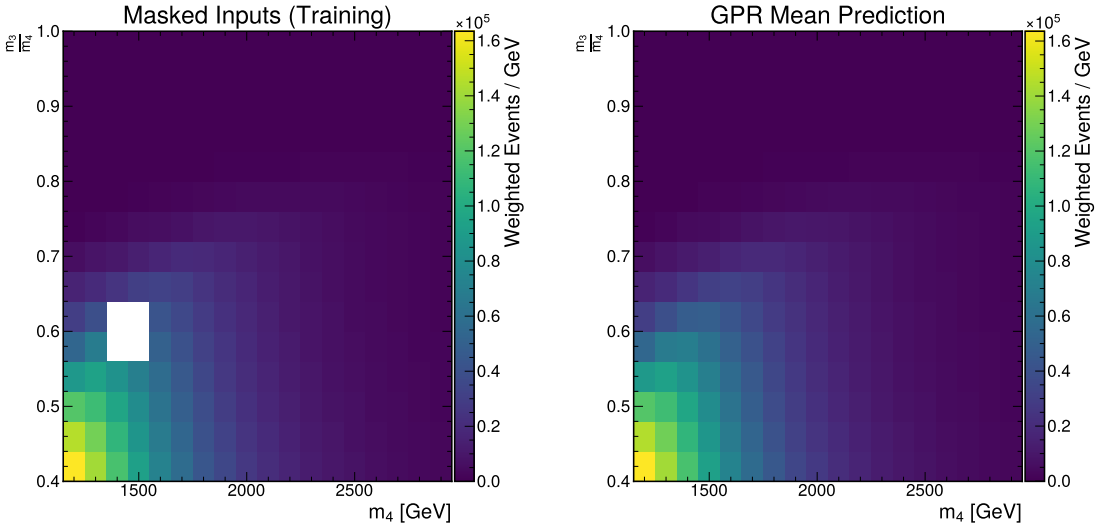


(a) (b)

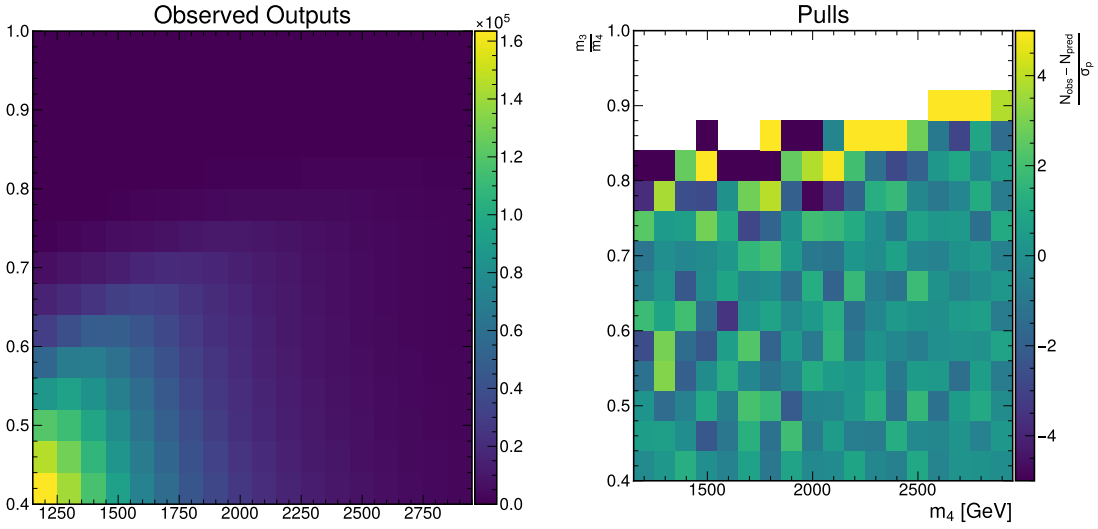


(c) (d)

Figure 4: Performing regression on control region data using the plain RBF kernel with blinding around a signal of 1500,400. Figure (a) shows the training data Figure (b) shows the posterior mean found using GP regression. Figure (c) shows the true outputs on the entire domain. Figure (d) shows the pulls for each bin.



(a) (b)



(c) (d)

Figure 5: Performing regression on control region data using the Matrix RBF kernel with blinding around a signal of 1500,400. Figure (a) shows the training data Figure (b) shows the posterior mean found using GP regression. Figure (c) shows the true outputs on the entire domain. Figure (d) shows the pulls for each bin.