Transforming Probability Integral Transforms

Michal Malyska
Alin Morariu

April 15, 2019

Abstract

We begin with an overview of Bayesian Model Comparison methods. Then we introduce a two number summary for the visual diagnostic of the Probability Integral Transform histograms using a Beta distribution. Based on this conclusions are drawn about model calibration while providing a basis for comparing models in a more precise manner.

Contents

1 Introduction

Statisticians are often tasked with placing a mathematical framework on observations in order to explain the real world. To do so, we *simply* model the data. However, before rolling these models out to the world and displaying the findings it is important to verify that the often complex equations and techniques used to create these models actually work. In simpler terms, does our model translate what the data is trying to tell us? Classical methods enjoy a wide array of hypothesis testing based methods for verifying the fit of a model since they are parameterized with point estimates. This naturally extends to model comparison as where two models can be pitted against each other under the null hypothesis scheme (see likelihood ratio tests as an example). Under the Bayesian framework, the fact that a full distribution is used to specify the parameter makes life more difficult.

Throughout this paper, we provide an outline of the current state of Bayesian model comparison in terms of both visual and numerical diagnostics while finally proposing a two-number summary for interpreting the Probability Integral Transform (PIT) histograms to act similarly to a test statistic. The goal is to provide a numerical tool for interpreting these diagrams since we can then quantify how well or poorly our models perform.

2 Current State

We begin with a discussion of the current standards for Bayesian model comparison.

2.1 Numerical Methods

2.1.1 Scoring Rules

Scoring rules are functions that measure predictive accuracy and because of this lend themselves naturally to Bayesian model comparison. With Bayesian statistics, we are ultimately trying to identify the *true* distribution of our parameters by using our prior knowledge (however limited that may be) and the data available. We define scoring rules as functions from the space of a probability measure and some values (our data) into the reals.

$$S : \Omega \times \mathbb{R} \mapsto \mathbb{R} \tag{1}$$

Where Ω is the probability measure. It is worth noting that these rules are negatively oriented. In more mathematical terms, if we are to take two different probability measures, say F and G,

$$S(F, y) \geq S(G, y) \Rightarrow G \text{ is "better" than } F$$

Since the true distribution of the parameter is unknown a lower score indicates a better predictive accuracy which in turn tells us that set distribution is closer to the true distribution.

Two commonly used scoring rules are the Continuous Ranked Probabilty Score (CRPS) and Cross validated predictive distribution (CPO). The CRPS is defined as

$$CRPS(F, y) = \int_{\mathbb{R}} [F(x) - H(x - y)]^2 dx \tag{2}$$

$$H(x - y) = \begin{cases} 1 & x > y \\ 0 & x < y \end{cases} \tag{3}$$

The intuition here is that we are using a step-wise function (namely H) as an approximation to the true cumulative distribution function. The square distance between that and the our estimate F is computed at every point on the $\mathbb{R}$ axis. In other words we are taking the sum of infinitely many difference between our estimated posterior and the true distribution (it can be shown that $H(x - y)$ converges to the true distribution - exercise is left to the reader). The second scoring function is the CPO which is a tool that is adapted from the classical method of leave-one-out-cross-validation (LOOCV). We define it in the following way

$$CPO = -\frac{1}{2}\sum_{i=1}^{n}\log\mathbb{P}(y_i|y_{-i}) \tag{4}$$

$$-\frac{1}{2}\sum_{i=1}^{n}\log\int_{\mathbb{R}}p(y_i,\theta|y_{-i})d\theta \tag{5}$$

$$-\frac{1}{2}\sum_{i=1}^{n}\log\int_{\mathbb{R}}p(y_i|\theta,y_{-i})p(\theta|y_{-i})d\theta \tag{6}$$

Here we compute the posterior distribution by leaving out 1 point at each iteration. Once again, we have a negatively oriented rule so smaller values are indicative of better a better model.

2.1.2 Kullback-Liebler Divergence

Continuing with the idea of comparing probability distributions, we turn our attention to the Kullback-Liebler Divergence (KLD). This is a measure of how much information we gain or lose when using one distribution as opposed to a second. Take two probability distributions F and G, then we define the KLD from F to G as

$$D_{KL}(F \parallel G) = \int_{-\infty}^{\infty} f(x)\log\left(\frac{f(x)}{g(x)}\right)dx \tag{7}$$

Once defined this way, it becomes clear that this is the expected log difference with respect to F between the two distributions and can hence be rewritten as

$$D_{KL}(F \parallel G) = \int_{-\infty}^{\infty} f(x)\left[\log f(x) - \log g(x)\right]dx \tag{8}$$

The underlying message conveyed by the KLD is how much information we are losing when approximating G with F. It is important to note that order *does* matter with this quantity. With this in mind, KLD is limited in it's ability to say model A is better than model B since it can only compare their respective posterior distributions. Unless we know the true distribution, which sadly we never do, the KLD is reduced to a way to compare simplicity between models. When comparing two models with this method we are looking at whether the simpler (definition for simplicity can vary by case but for our purposes we are referring to the number of parameters and interpretability of the model) model captures *almost* all of the information the more complex version does. While useful, this does not give us a mean of comparing models as a whole and we turn to information criterion, building towards Watanabe-Akaike information criterion (WAIC) as our final numerical tool for comparing models.

2.1.3 WAIC

Information criterion refers to measures of predictive accuracy and will be defined in terms of the log predictive density of data for some set point estimate of the fitted model. Typically, we take the maximum likelihood estimator (MLE or mode) as our point estimate but posterior medians are an acceptable and sometimes preferred alternative. The basis of our comparison is the predictive accuracy of a model. In other words, how well can our current model predict what will happen in the future. Evaluating what will come in the future is difficult, especially when you don't have a steady stream of data rolling in to verify your predictions against. In the ideal world we would be computing the expected log posterior density (elpd)

$$\text{elpd}_{\hat{\theta}} = \mathbb{E}(\log p(y|\hat{\theta}(y_{obs}))) \tag{9}$$

however, that is simply not possible. That being said, we can get around this by using the the log posterior density of the data itself as an approximation. Combining this with a correction for overfitting (namely, the number of parameters in the model denoted by k), and some constants for convenience, we get the Akaike Information Criterion (AIC)

$$AIC = \widehat{elpd}_{AIC} = -2\log p(y|\hat{\theta}_{MLE}) + 2k \tag{10}$$

The penalization for number of parameters is great when working with linear models but often fails in more complex settings, particularly when we have multilevel models and more complex hierarchical structures that have this feature built into them. They already reduce the amount of overfitting by construction. AIC is a great tool in the frequentist realm but we need something that is more appropriate for our new Bayesian approach. This is where we introduce the Watanabe-Akaike information criterion which starts by making a change in the adjustment factor (previously k) to reflect the *effective* number of parameters. It computes the sample variance of the log predictive posterior density based on posterior samples of the parameter θ.

$$p_{WAIC} = \sum_{i=1}^{n} V_{s=1}^{S}(\log p(y_i|\theta^s)) \tag{11}$$

Where θ^s is a draw from the posterior density of θ. Explicitly, it can be written as

$$p_{WAIC} = \sum_{i=1}^{n} \frac{1}{S-1} \sum_{s=1}^{S} (\log p(y_i|\theta^s) - \mu_S)^2 \tag{12}$$

$$\text{Where } \mu_s = \frac{1}{S} \sum_{s=1}^{S} \log p(y_i|\theta^s) \tag{13}$$

Based on the same idealogy as the AIC, we combine this correction with the log pointwise predictive density and constants to give us the WAIC defined as

$$\begin{aligned} \text{WAIC} &= -2lppd + 2p_{WAIC} \qquad (14)\\ &= -2\sum_{i=1}^{n} \log\left(\frac{1}{S}\sum_{s=1}^{S} p(y_i|\theta^s)\right) + 2\sum_{i=1}^{n} \frac{1}{S-1}\sum_{s=1}^{S}(\log p(y_i|\theta^s) - \mu_S)^2 \qquad (15)\\ &= -2\left[\log\left(\frac{1}{S}\sum_{s=1}^{S} p(y_i|\theta^s)\right) - \frac{1}{S-1}\sum_{s=1}^{S}(\log p(y_i|\theta^s) - \mu_S)^2\right] \qquad (16)\\ \text{Where } \mu_s &= \frac{1}{S}\sum_{s=1}^{S} \log p(y_i|\theta^s) \qquad (17) \end{aligned}$$

The assumption when using WAIC is that we generate a suffciently large number of posterior samples (denoted S) to accurately capture the distribution of the data. Similar to the scoring rules mentioned earlier, lower scores are regarded as ”better”. When computing the WAIC it is important to be aware of the computation costs associated with sampling and calculating such a large number of individual quantities.

2.2 Visual Diagnostics

2.2.1 Posterior Plots

The most natural of places to start is that of the posterior plots regarding the values our model parameters could take. Once a model is fit, it is fairly simple to take samples from the posterior distribution in order to visualize what the marginal posterior distribution of an individual parameter. While quite informative about the set parameter, it is limited in the amount of information it can tell you regarding the model as a whole. Really, the only thing we are able to do is draw conclusions of reasonability or sensibility. These plots act as major red flags for us where we verify if the outputs of the model make sense. However, this can only be done in the case where a significant amount of knowledge of the subject matter is present (often done best with the help of a expert in the subject matter the model is trying to explain). Evidently, this is a primitive tool that while useful, can make models that pass the sensibility test hard to distinguish between.

3 PIT histograms

3.1 A conceptual discussion

Probability Integral Transform (PIT) histograms are a model checking tool in Bayesian statistics that branches off of the Fundamental Theorem of Sampling. The motivation behind these histograms is that if you have the posterior predictive distribution $y \sim F_Y$ then $F_Y(y) \sim Unif(0,1)$ and we can say that the model is *calibrated.* When our models involve a random variable, it is straight forward to show that they can be converted to uniform random variables (shown below 3.1).

Proof. Take a random variable y with corresponding CDF F_Y, and define $X = F_Y(Y)$.

$$\begin{aligned} F_X(X) &= \mathbb{P}(X \leq x) \\ &= \mathbb{P}(F_Y(Y) \leq x) \\ &= \mathbb{P}(Y \leq F_Y^{-1}(x)) \\ &= F_Y(F_Y^{-1}(x)) \\ &= x \end{aligned}$$

Since x is just the CDF of $Unif(0,1)$, we get that X is also uniformly distributed on $[0,1]$ □

3.1.1 Smiley face or Frownie Face

In the ideal case, the result histograms displays a perfectly uniform distribution indicating that the model is calibrated and that the underlying random variable follows the proposed distribution. However, this is often not the case and we can use it as a diagnostic. Four major shapes arise and they are the following:

- $\cup$ shaped: predictive distribution is too narrow
- $\cap$ shaped: predictive distribution is too wide
- Asymmetrical (higher on the left): predictive distribution too far left (bias)
- Asymmetrical (higher on the right): predictive distribution too far right (bias)

These diagrams can be used to tell which model is has fewer problems as a mean of comparing them.

3.1.2 One is just like the other

We would like to highlight the duality of PIT histograms and Q-Q plots (frequently used in frequentist statistics to see if empirical CDFs (ECDF) deviate from their theoretical counterpart). Q-Q plots map quantiles of the ECDF against those a theoretical distribution with the $y = x$ line meaning that the two are the same. A histogram can easily be transformed into QQ plot but summing over the bars and dividing by the total to give you the respective quantiles.

3.1.3 When 1 meets 1

As with all visual diagnostics, interpretation is hard. It becomes a subjective exercise and that is why we prefer numbers. They cleaner and can be abused more easily. We attempted to create a two number summary that would act as a measure of bias and dispersion based on the PIT histogram. To do so, we parameterize the the histogram using a Beta distribution. Using the fact that the $Beta(1,1)$ and the $Unif(0,1)$ are equivalent, we can compare PIT histograms on a numerical scale by comparing the α and β parameters to the ideal case. The Beta distribution is incredibly flexible which allows us to detect any shape the histogram could take.

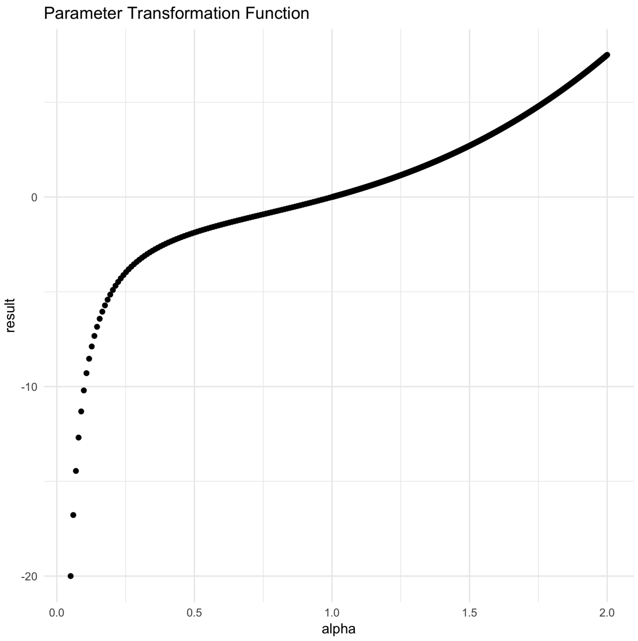

Figure 1: Graph of parameter transform function

3.2 Two number summary for PIT histograms

With the goal of a $Beta(1,1)$ distribution, we compute estimates for the parameters of the beta distribution that fits our PIT histogram via the method of moments estimator (could also use MLE). Now that we have our estimates $\hat{\alpha}$ and $\hat{\beta}$, we scale the parameter using the following function

$$\varphi\begin{pmatrix} \hat{\alpha} \\ \hat{\beta} \end{pmatrix} = \begin{pmatrix} \frac{-1}{\hat{\alpha}} + \hat{\alpha}^3 \\ \frac{-1}{\hat{\beta}} + \hat{\beta}^3 \end{pmatrix} = \begin{pmatrix} \alpha_{test} \\ \beta_{test} \end{pmatrix} \tag{18}$$

And propose the following as the two number summary

$$\boldsymbol{\phi} = \begin{pmatrix} \alpha_{test} - \beta_{test} \\ \alpha_{test} + \beta_{test} \end{pmatrix} \tag{19}$$

The goal was to create a diagnostic which is defined on the full $\mathbb{R}$ line (for both numbers) which inspired our choice for the function φ. The transformation ensures that when the parameter is close to 1 (the good case), transformed version is close to zero. The function penalizes deviations away from the ideal case by diving into the negatives when values are below 1 and shooting up to the positives when above 1. The summary $\boldsymbol{\phi}$ then can be used to determine the direction of bias and over/under-dispersion as detailed in Table 1.

Difference ($\boldsymbol{\phi}_1$)	**Sum ($\boldsymbol{\phi}_2$)**	**Bias**	**Dispersion**
Highly Positive	Highly Positive	Shifted *Right*	*Under* Dispersion
Highly Positive	Highly Negative	Shifted *Right*	*Over* Dispersion
Highly Negative	Highly Positive	Shifted *Left*	*Under* Dispersion
Highly Negative	Highly Negative	Shifted *Left*	*Over* Dispersion

Table 1: Interpretation of the $\boldsymbol{\phi}$ two number summary for PIT histograms. Numbers close to zero are considered ”good” and hence omitted from the table.

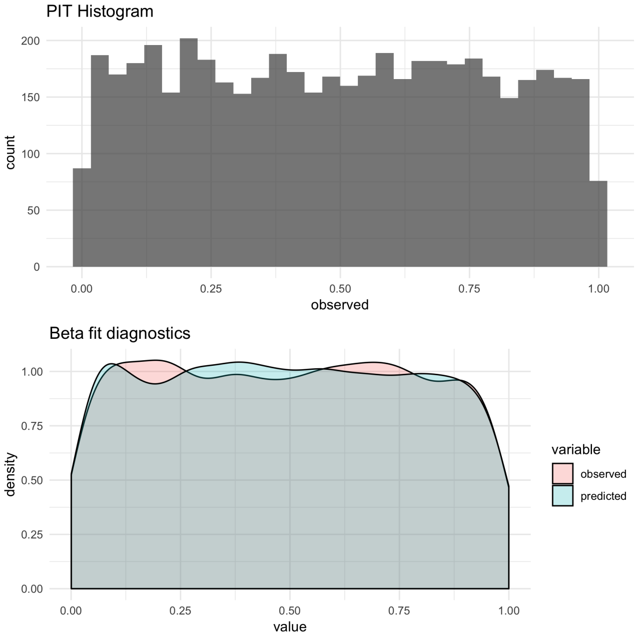

Figure 2: PIT Histogram from a uniform distribution. Two number summary $\phi = (0.02293776, -0.1835408)$

The remaining section provide examples of the cases which arise when using this tool as well as applications on real examples.

3.3 Let's give these bad boys a try

3.3.1 In a perfect world

We created an R function that takes in an INLA results object, makes sure CPO calculations were conducted in the fitting, and then outputs the two number summary, the PIT histogram plot, as well as a diagnostics plot for the beta distribution that was fitted from data. it can be found in the appendix.

To showcase the behaviour of the two number summaries first, we generated 5 idealistic scenarios where the PIT data comes from an exact beta distribution. Then we have 3 examples from real models to showcase the beta distribution diagnostics and real case scenarios.
By cross referencing Table 1 to the captions of each figure we see that the two number diagnostic performs as expected in each of these case.

3.3.2 And in the Real World

Further a number of experiments was conducted using the inbuilt R data sets for Bayesian statistics. And results are presented below.

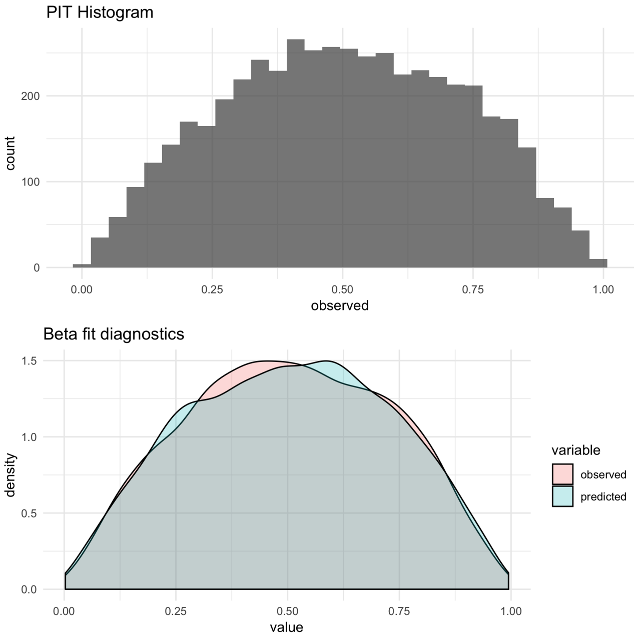

Figure 3: PIT Histogram from a Beta(2,2) distribution. Two number summary $\phi = (0.3577983, 12.48185)$

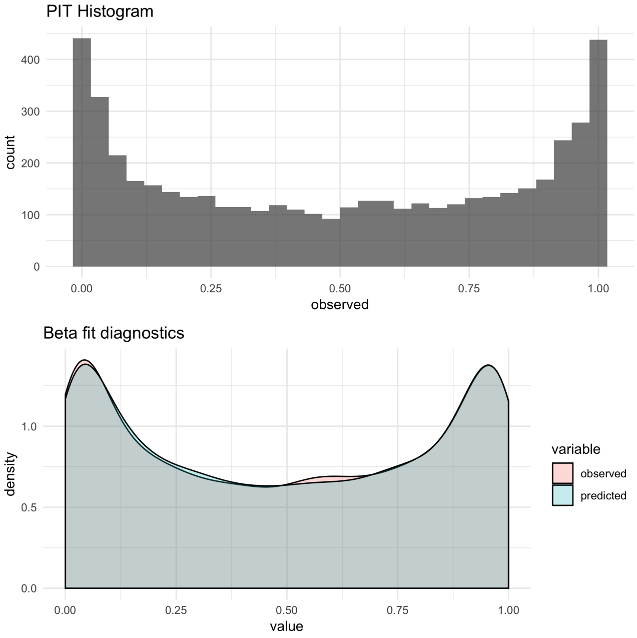

Figure 4: PIT Histogram from a Beta(0.5,0.5) distribution. Two number summary $\phi = (0.03588651, -3.898585)$

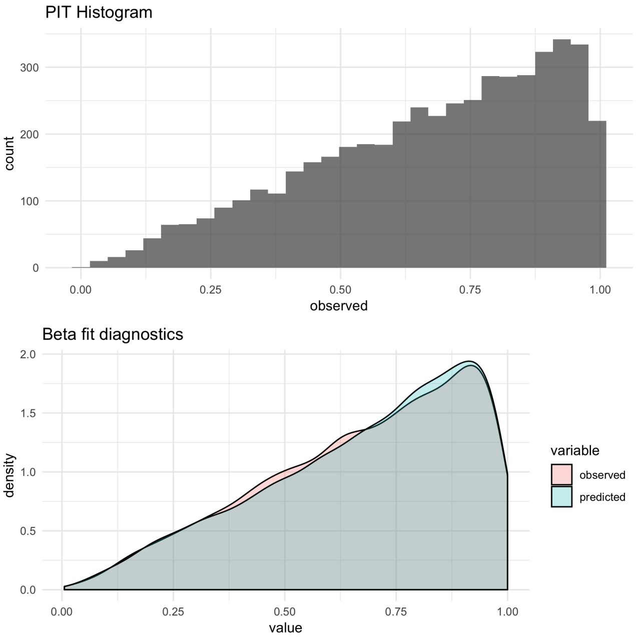

Figure 5: PIT Histogram from a Beta(2,0.5) distribution. Two number summary $\phi = (6.986783, 6.764088)$

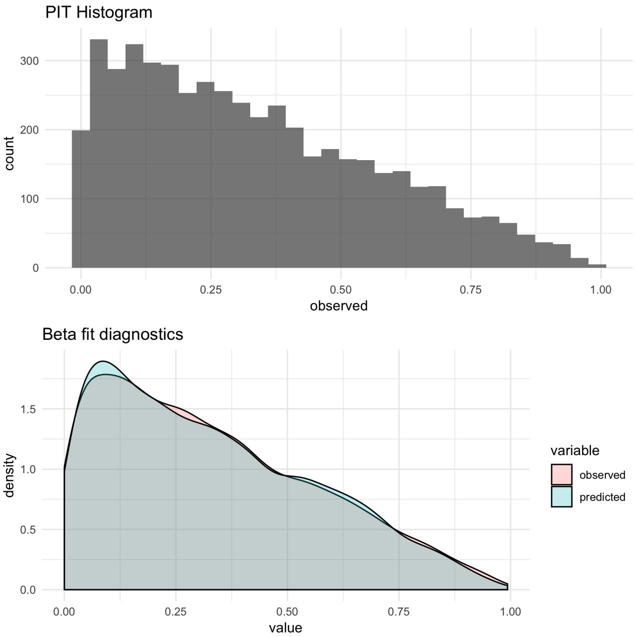

Figure 6: PIT Histogram from a Beta(0.5,2) distribution. Two number summary $\phi = (7.956036, 7.956036)$

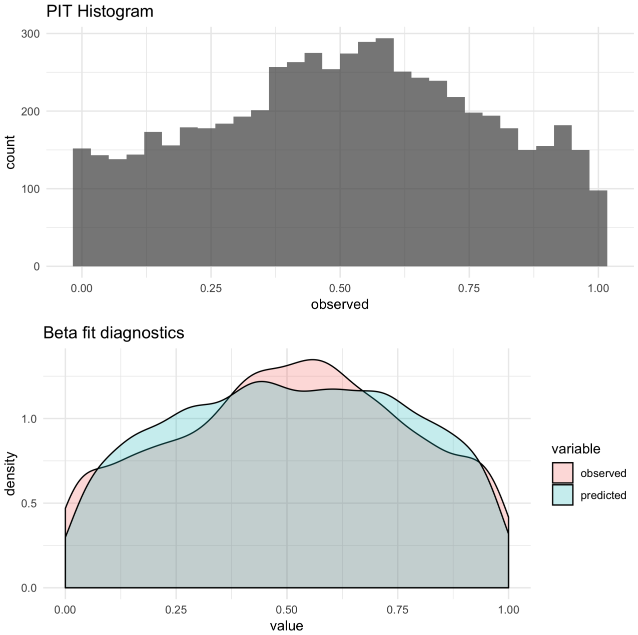

Figure 7: Our function output for a dataset and full model from our Spatial Assignment 1. Two number summary $\phi = (0.1582978, 2.428908)$

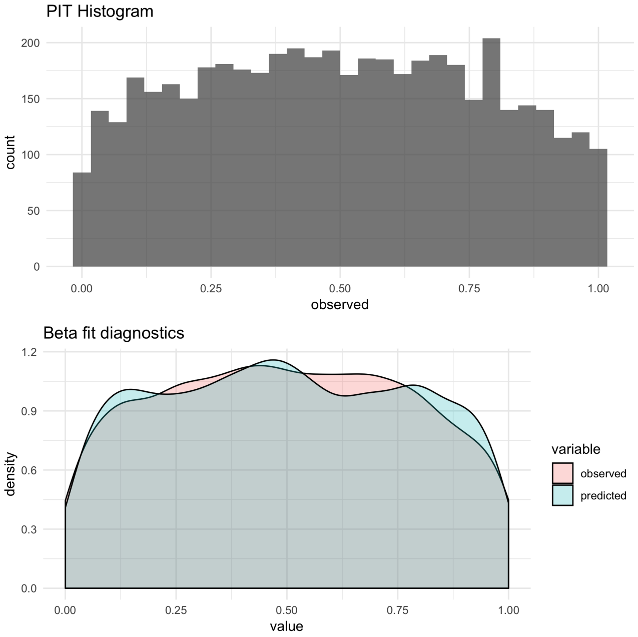

Figure 8: Our function output for a dataset and model1 from Martino and Rue (2010) on INLA website. Two number summary $\phi = (-0.08217218, 1.178578)$

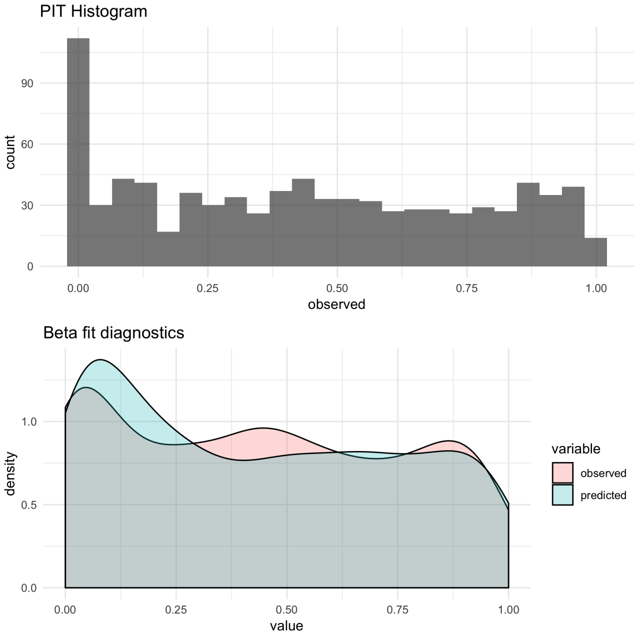

Figure 9: Our function output for a dataset and model from Van Niekerk, Bakka and Rue (2019) on INLA website. Two number summary $\boldsymbol{\phi} = (-0.6227838, -2.007153)$

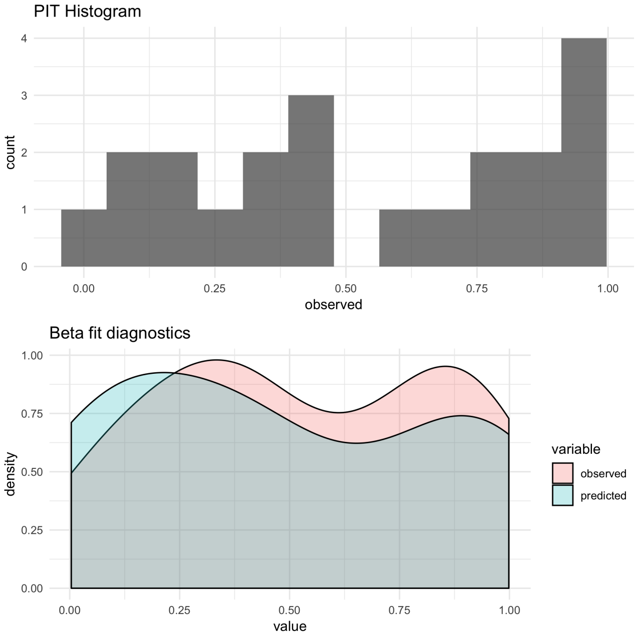

Figure 10: Our function output for a dataset and model from Breslow and Clayton (1993), the Overdispersion study 1. Two number summary $\boldsymbol{\phi} = (0.3148246, -2.228811)$

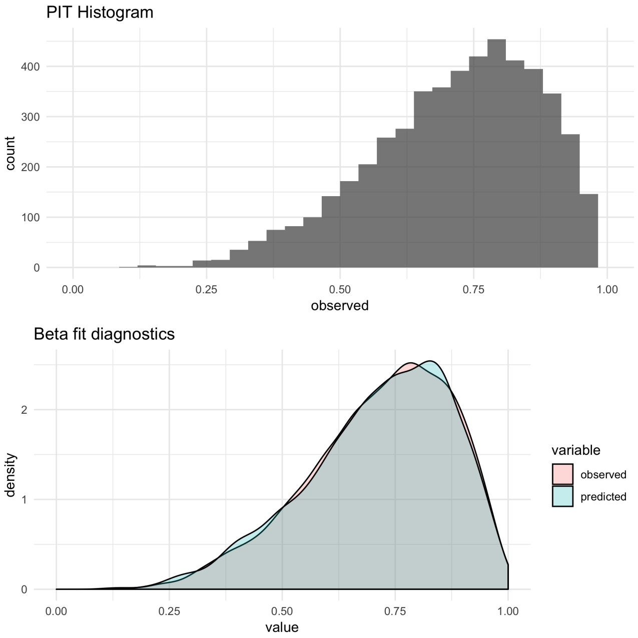

Figure 11: Extreme right bias (left case is identical) showing failure points of the two number diagnostic. Two number summary $\phi = (124.3165, 139.755)$

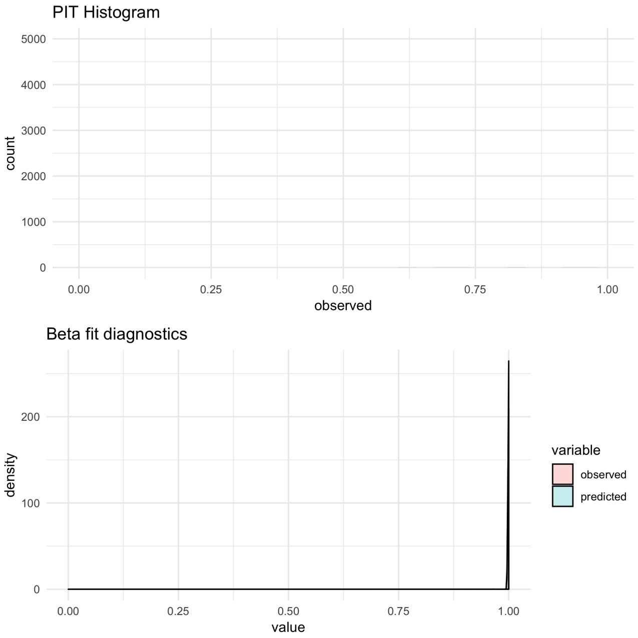

Figure 12: Extreme right bias (left case is identical) showing failure point of the two number diagnostic - this indicates under-dispersion. Two number summary $\phi = (1208.225, -1167.197)$

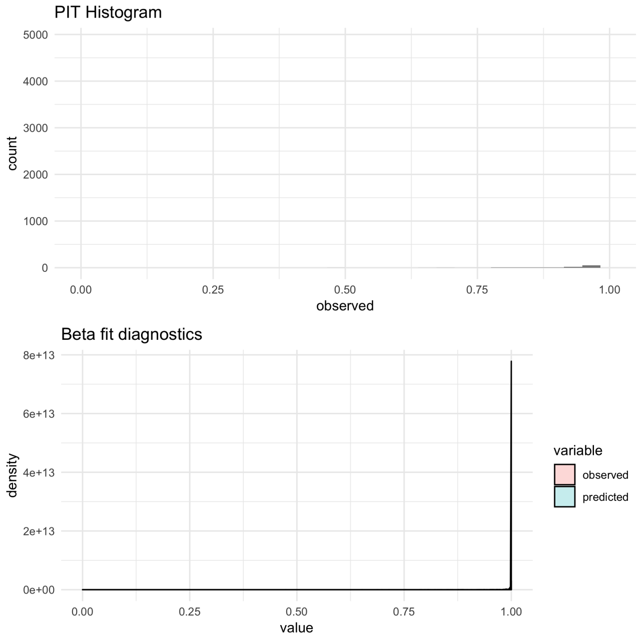

Figure 13: Extreme right bias (left case is identical) showing failure point of the two number diagnostic - this indicates over-dispersion. Two number summary $\phi = (266.4535, 81.5491)$

4 Conclusions and Limitations

In conclusion, we see that the two number diagnostic is consistent with the visual diagnostic of the PIT histograms while providing a numerical framework for interpreting how things go astray. Working off of the parameters of the Beta distribution gives us the advantage of knowing which knobs we must turn to properly calibrate our models. More importantly, in the context of model comparison, pitting two models against each other becomes a more natural exercise. You have a clear indicator of the location and size of the bias and dispersion patterns. Once combined with context, it becomes much clearer which model is more promising and how to adjust it.

While all of this is peachy, we *must* highlight some of the limitations of using this approach (call it a cautionary tale). The function (φ) used to transform the parameter estimates (which themselves can be mediocre) stands to be improved. The powers on the two terms were chosen via a guess and check method on simulated values based on intuition; thus could be subject of a more analytically critique and optimization. Furthermore, the proposed diagnostic shows signs of struggle in extreme left or right bias (i.e. magnitude of the difference is very far from zero, see Figure 12). The struggle is that you can no longer make any statements about the dispersion. However, this is something you would struggle with in the visual case as well so no harm no foul. Lastly, a fundamental failure is observed when the number of data points because the parameter estimation of the underlying Beta distribution is very noisy (see. Figure 11)

All in all, the two number summary provides a certain comfort in interpreting PIT histograms and gives us a higher degree of *confidence*, and *not* accuracy, in our interpretations. That being said it should be used with caution and only as an additional check as opposed to the be all and end all diagnostic for model comparison.

5 Appendix

R code for the function:

```
library(tidyverse)
library(reshape2)

# Convert Parameters to the new values
convert_param <- function(alpha) {
        new_alpha <- (-1)/(alpha) + (alpha)^(3)
        return(new_alpha)
}

# Estimate the beta parameters from data
estBetaParams <- function(x) {
        mu = mean(x)
        var = var(x)
        alpha <- ((1 - mu) / var - 1 / mu) * mu ^ 2
        beta <- alpha * (1 / mu - 1)
        return(params = list(alpha = alpha, beta = beta))
}

# Create Ouput
monkey_hammer <- function(inla_result){
        # Checks that the PIT's were computed:
        if (length(inla_result$cpo$pit) == 0) {
                stop("You␣need␣to␣enable␣computing␣the␣CPO's␣in␣your␣INLA␣
                    computation.␣\n
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣Add␣control.compute␣=␣list(cpo␣=␣TRUE)␣to␣the␣end.
    ")
        }
        ## Compute the number summary:
        # Estimate the parameters
        x <- inla_result$cpo$pit
        mu <- mean(x)
        var <- var(x)
        alpha <- ((1 - mu) / var - 1 / mu) * mu ^ 2
        beta <- alpha * (1 / mu - 1)
        # Transform the parameters
        new_alpha <- convert_param(alpha)
        new_beta <- convert_param(beta)
        # Return the summary:
        param_diff <- new_alpha - new_beta
        param_sum <- new_alpha + new_beta

        plotting_data <- data.frame(observed = x,
                                                                            predicted =
                                                                                rbeta(
                                                                               length(x
                                                                               ), alpha
                                                                               , beta))

        plotting_data <- melt(plotting_data)
        beta_diagn <- ggplot(data = plotting_data, mapping = aes(x = value, fill = variable)) +
                geom_density(alpha = 0.25) +
                theme_minimal() +
                scale_x_continuous(limits = c(0,1)) +
                labs(title = "Beta fit diagnostics")

        PIT_hist <- ggplot(data = tibble(observed = x)) +
                aes(x = observed) +
                geom_histogram(alpha = 0.75, bins = ceiling(1 + 3.322 * log(length(x)))) +
                theme_minimal() +
                scale_x_continuous(limits = c(0,1)) +
                labs(title = "PIT Histogram")

        if (param_diff < 0) {
                print("There might be some right bias")
        }
        else if (param_diff > 0) {
                print("There might be some left bias")
        }

        if (param_sum < 0) {
                print("There might be some under-dispersion")
        }
        else if (param_sum > 0) {
                print("There might be some over-dispersion")
        }

        return(list(difference = param_diff, sum = param_sum, PIT = PIT_hist, beta_diagnostic_plot = beta_diagn))
}
```