Posterior Calibration of Posterior Predictive P-values

Geert H. van Kollenburg, Joris Mulder, Jeroen K. Vermunt

Tilburg University, Tilburg, Netherlands

Abstract

In order to accurately control the type I error rate (typically .05), a p-value should be uniformly distributed under the null model. The posterior predictive p-value (ppp), which is commonly used in Bayesian data analysis, generally does not satisfy this property. For example there have been reports where the sampling distribution of the ppp under the null model was highly concentrated around .50. In this case, a ppp of .20 would indicate model misfit, but when comparing it with a significance level of .05, which is standard statistical practice, the null model would not be rejected. Therefore, the ppp has very little power to detect model misfit. A solution has been proposed in the literature, which involves calibrating the ppp using the prior distribution of the parameters under the null model. A disadvantage of this "prior-cppp" is that it is very sensitive to the prior of the model parameters. In this paper, an alternative solution is proposed where the ppp is calibrated using the posterior under the null model. This "posterior-cppp" (i) can be used when prior information is absent, (ii) allows one to test any type of misfit by choosing an appropriate discrepancy measure, and (iii) has a uniform distribution under the null model. The methodology is applied in various testing problems such as testing independence of dichotomous variables, checking misfit of linear regression models in the presence of outliers, and assessing misfit in latent class analysis.

Keywords: Goodness-of-fit, posterior predictive check, p-values, calibration, regression, latent class analysis

Posterior Calibration of Posterior Predictive P-values

A crucial step in a statistical analysis is to assess whether the employed statistical model fits the observed data. Different tools are available for this purpose. When one is interested in testing whether one statistical model better fits the data than another statistical model, *model comparison* tools are useful. Commonly used model comparison tools are the AIC (Akaike, 1973), the BIC (Schwarz et al., 1978), or the Bayes factor (Jeffreys, 1961; Kass & Raftery, 1995). These criteria penalize model complexity in the sense that a simple model with few parameters is generally preferred over a more complex model with many parameters if both models fit the data equally well (Mulder, 2014; Myung, 2000). On the other hand, when one is interested in testing whether one specific model fits the observed data, *model checking* tools are useful. Such model checks are often performed using Fisherian p-values in a classical framework and using posterior predictive p-values (ppp's) in a Bayesian framework (Berkhof, Van Mechelen, & Gelman, 2003; Choi, Hui, & Bell, 2010; Meng, 1994). Although these methods come from different paradigms, the p-value and the ppp are applied in a similar fashion to assess misfit of the employed statistical model. If the p-value is smaller than a pre-specified threshold value (typically .05), this indicates model misfit. If this is the case, an extension of the employed model may be necessary to better fit the observed data. Because the p-value and ppp are applied in a similar manner in practice, we shall also refer to this threshold value as the "significance level" of the Bayesian test to avoid additional terminology (despite the fact that the term "significance" is not commonly used in Bayesian statistics). We shall also borrow frequentist terminology when referring to the type I error rate, the type II error rate, and power as the probability of incorrectly rejecting a Bayesian null model that is true, the probability of not rejecting an incorrect null model, and the probability of correctly rejecting an incorrect null model, respectively. Note that such frequentist properties are of interest in objective Bayesian statistics (Berger et al., 2006).

In this paper we shall focus on model checking in the Bayesian framework using the posterior predictive p-value (ppp) (Gelman, Meng, & Stern, 1996; Meng, 1994). The

ppp has three useful properties. Firstly, it can straightforwardly be used for testing any type of model misfit; we only need to formulate a discrepancy measure which is able to detect the type of misfit of interest. Secondly, a ppp can easily be computed from MCMC output because the discrepancy is allowed to depend on the (sampled) unknown model parameters. Thirdly, the ppp can be computed using non-informative (or objective) improper priors. This third property is useful when prior information is unavailable or when a researcher does not want to include external information via the prior distribution when evaluating the model. Because of these useful properties, the ppp has been used in many different types of applications in psychology (Oravecz, Faust, Batchelder, & Levitis, 2015), as well as in other fields such as marketing (Choi et al., 2010), medicine (Chaves, Chakraborty, Benziger, & Tannenbaum, 2014), psychiatry (Berkhof et al., 2003) and sociology (Hoverd & Sibley, 2013).

A potential problem of the ppp, however, lies in its interpretation. In order to reliably interpret a ppp, the type I error rate should be equal to the significance level. This equality only holds when the sampling distribution of the p-value is uniform. For the ppp, however, the sampling distribution under the null model is typically not uniformly distributed, but instead it is concentrated around .5 (Meng, 1994), possibly with a lower bound that is larger than 0 (Robins, van der Vaart, & Ventura, 2000). As a result, the type I error rate for the ppp is generally lower than the significance level. Consequently a badly fitting model may not be rejected as a result of very low statistical power of the ppp.

To resolve this issue and to accurately control the type I error probability, one can calibrate the ppp under the employed null model. For this purpose, Hjort, Dahl, and Steinbakk (2006) proposed calibrating the ppp using a proper informative prior, yielding what we will refer to as a prior-calibrated ppp (prior-cppp). The prior-cppp is uniformly distributed under the null model and the chosen prior, and therefore it potentially resolves the problem associated with the standard ppp.

A key property of the prior-cppp is however that the employed statistical model and the informative prior are simultaneously tested. It is therefore crucial to carefully

formulate informative priors for the unknown model parameters based on one's substantive beliefs before observing the data. When prior information is weak or unavailable the prior-cppp is not recommendable. On the other hand when prior knowledge is available, the specification of the informative prior distribution for all model parameters can be a rather difficult and time consuming exercise (Berger et al., 2006; Hjort et al., 2006), which substantive researchers may prefer to avoid. Furthermore, researchers may only be interested in assessing model misfit of the employed statistical model and not in simultaneously testing the appropriateness of the informative prior. By taking these considerations into account, the applicability of the prior-cppp may be limited.

In this paper, a new type of calibrated ppp is proposed. Instead of calibrating the ppp under an informative prior, as in the prior-cppp, the ppp is calibrated under the posterior distribution of the unknown parameters of the employed null model. The resulting ppp will be referred to as the posterior-calibrated ppp (posterior-cppp). Unlike the prior-cppp, the posterior-cppp can be used when prior information is weak or when one is only interested in testing model misfit. Furthermore, the posterior-cppp has all the useful properties of the original ppp, with the additional advantage that it is uniformly distributed under the null model.

The remainder of the paper is outlined as follows. In the next section we explain how to obtain the ppp, the prior-cppp, and the posterior-cppp. Subsequently, in Application I, the performance of the three posterior predictive checks is assessed for a simple test for independence in contingency tables by looking at type I error probabilities. In Application II, we apply the new methodology in a linear regression analysis to test whether the model adequately explains extreme observations. A simulation experiment is conducted to evaluate type I error rates and the power of this test. The method is applied in an empirical example to predict the quality of life of elderly people. In Application III, the methodology is applied in the context of latent class analysis. We investigate type I error rates and power when testing bivariate residuals and the number of latent classes. An empirical data set is used to illustrate

the practical use and benefits of the posterior-cppp in testing for different sub-types of depression. The paper ends with a discussion of the methods and results.

Posterior Predictive Checks

The posterior predictive check is a flexible and efficient tool to assess misfit of a Bayesian statistical model for the observed data. The general idea of a posterior predictive check is to assess systematic discrepancies between the observed data and (hypothetical) replicated data generated from the fitted model (Gelman, Carlin, Stern, & Rubin, 2004). When there is a small discrepancy between the replicated data and the observed data, this suggests a good fit of the model. When there is a large discrepancy between the replicated data and the observed data, this suggests model misfit.

The procedure works as follows. First we have to specify a prior distribution for the unknown model parameters $\boldsymbol{\theta}$. The prior contains our knowledge or beliefs about the model parameters before observing the data. The prior will be denoted by $p(\boldsymbol{\theta})$. The model can be fitted to the observed data by deriving the posterior distribution of $\boldsymbol{\theta}$. The posterior is a combination of the information in the prior, $p(\boldsymbol{\theta})$, and the information in the observed data, $\mathbf{y}_{\text{obs}}$. The information about $\boldsymbol{\theta}$ in the observed data is formalized in the likelihood function of the model, which is denoted by $p(\mathbf{y}_{\text{obs}}|\boldsymbol{\theta})$. Subsequently the posterior can be obtained using Bayes' theorem,

$$\begin{aligned} p(\boldsymbol{\theta}|\mathbf{y}_{\text{obs}}) &= \frac{p(\mathbf{y}_{\text{obs}}|\boldsymbol{\theta}) \times p(\boldsymbol{\theta})}{p(\mathbf{y}_{\text{obs}})} \\ &\propto p(\mathbf{y}_{\text{obs}}|\boldsymbol{\theta}) \times p(\boldsymbol{\theta}), \end{aligned} \tag{1}$$

where $p(\boldsymbol{\theta}|\mathbf{y}_{\text{obs}})$ denotes the posterior of the unknown parameters $\boldsymbol{\theta}$ given the observed data $\mathbf{y}_{\text{obs}}$. The posterior contains our knowledge about the model parameters after observing the data. In (1), the marginal likelihood $p(\mathbf{y}_{\text{obs}})$ does not depend on $\boldsymbol{\theta}$, and therefore it does not play a role when deriving the posterior. Due to the many possible specifications of likelihood and prior, the posterior does not always belong to a known family of probability distributions. In such cases, the posterior is usually approximated by sampling posterior draws of $\boldsymbol{\theta}$ from $p(\boldsymbol{\theta}|\mathbf{y}_{\text{obs}})$ using an MCMC algorithm (for an

extensive overview of MCMC algorithms, see Liang, Liu, & Carroll, 2011).

The posterior can be used to obtain estimates for the model parameters (such as posterior means, modes, or medians) and credibility intervals (the Bayesian counterpart of classical confidence intervals). Furthermore, the posterior can be used to draw a replicated data set, denoted by $\mathbf{y}_{\text{rep}}$. A replicated data set can be viewed as data that we could see tomorrow if the experiment that produced the observed data, $\mathbf{y}_{\text{obs}}$, were replicated with the same model and with the same value for $\boldsymbol{\theta}$ that produced the observed data today (Gelman et al., 2004). The posterior predictive distribution of $\mathbf{y}_{\text{rep}}$ is given by

$$p(\mathbf{y}_{\text{rep}}|\mathbf{y}_{\text{obs}}) = \int p(\mathbf{y}_{\text{rep}}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathbf{y}_{\text{obs}})d\boldsymbol{\theta}.$$

By looking at specific characteristics of the observed data and a replicated data set we can check whether both data sets were likely to be generated from the employed statistical model, similar as a classical test. If this is (not) the case, this suggests a good (bad) model fit to the observed data. This can be done using a so-called discrepancy measure, denoted by $D(\mathbf{y};\boldsymbol{\theta})$, which is a function of a data set $\mathbf{y}$ (which could be either the observed data set, $\mathbf{y}_{\text{obs}}$, or a replicated data set, $\mathbf{y}_{\text{rep}}$) and the unknown model parameters $\boldsymbol{\theta}$. For example, discrepancies can measure overall fit based on Pearson χ^2-type statistics (van Kollenburg, Mulder, & Vermunt, 2015), or specific aspects of the model such as adequately capturing extreme values (Gelman et al., 2004). Note that discrepancies can depend on both the data and the model parameters (Meng, 1994), while classical fit statistics only depend on the data. Examples of discrepany measures will be provided in the next sections. Through the posterior predictive check we assess the probability – quantified by the ppp – that replicated data under the posterior are more extreme than the observed data (Gelman et al., 2004). The following algorithm describes how to obtain the ppp.

Algorithm 1: Computation of the ppp

Step 1: Specify a prior, $p(\boldsymbol{\theta})$, for the model parameters and choose a discrepancy measure, $D(\mathbf{y};\boldsymbol{\theta})$.

Step 2: Obtain the posterior based on the prior and the likelihood of the observed data using (1).

Step 3: Obtain values for the chosen discrepancy measure for the observed data and replicated data sets based on random draws from the posterior:

3a: Draw a random value for the model parameters, denoted by $\boldsymbol{\theta}^{(k)}$, from the posterior:

$$\boldsymbol{\theta}^{(k)} \sim p(\boldsymbol{\theta}|\mathbf{y}_{\text{obs}}). \tag{2}$$

3b: Sample a replicate data set, $\mathbf{y}_{\text{rep}}^{(k)}$, given the posterior draw $\boldsymbol{\theta}^{(k)}$:

$$\mathbf{y}_{\text{rep}}^{(k)} \sim p(\mathbf{y}_{\text{rep}}|\boldsymbol{\theta}^{(k)}) \tag{3}$$

3c: Calculate the observed discrepancy $D(\mathbf{y}_{\text{obs}}; \boldsymbol{\theta}^{(k)})$ and the replicated discrepancy $D(\mathbf{y}_{\text{rep}}^{(k)}; \boldsymbol{\theta}^{(k)})$.

3d: Repeat Steps 3a to 3c for $k = 1, \ldots, K$ (e.g., $K = 1000$).

Step 4: Compute the ppp as the proportion of replicated data sets where $D(\mathbf{y}_{\text{rep}}^{(k)}; \boldsymbol{\theta}^{(k)})$ was greater than or equal to $D(\mathbf{y}_{\text{obs}}; \boldsymbol{\theta}^{(k)})$:

$$\text{ppp} = K^{-1} \sum_{k=1}^{K} I(D(\mathbf{y}_{\text{rep}}^{(k)}; \boldsymbol{\theta}^{(k)}) \geq D(\mathbf{y}_{\text{obs}}; \boldsymbol{\theta}^{(k)})), \tag{4}$$

where the indicator function $I(\cdot)$ equals 1 if the replicated discrepancy $D(\mathbf{y}_{\text{rep}}^{(k)}; \boldsymbol{\theta}^{(k)})$ is larger than or equal to the observed discrepancy $D(\mathbf{y}_{\text{obs}}; \boldsymbol{\theta}^{(k)})$, and 0 otherwise.

Hence the goal of the algorithm is to obtain a large set of K draws from the posterior (Step 3a), generate replicated data sets for all posterior draws (Step 3b), and compute the discrepancy based on the posterior draw for the observed data and the replicated data based on all posterior draws (Step 3c). This results in a set of K pairs of observed discrepancies and replicated discrepancies. The ppp is defined as the proportion of draws where the replicated discrepancy is larger than the observed

discrepancy (Step 4). As an example Figure 1 displays $K = 1,000$ pairs of observed and replicated discrepancies in a posterior predictive check for independence of 4 dichotomous variables with a Pearson χ^2 discrepancy measure (elaborated in the next section). The ppp is equal to the proportion of pairs where replicated discrepancies are at least as large as the observed discrepancies (above the line where Drep = Dobs). In this example, the ppp was equal to .146.

Note that the prior that is specified in Step 1 does not play an important role when computing the ppp because typically the prior is completely dominated by the likelihood. It is even possible to specify a non-informative improper prior as long as the resulting posterior in Step 2 is proper.

[Figure 1 about here]

Prior-calibrated posterior predictive p-values.

It has been shown in previous work that the ppp is generally non-uniform under the null model (Hjort et al., 2006; Meng, 1994; Robins et al., 2000; van Kollenburg et al., 2015). A solution to the non-uniformity of the ppp has been proposed in which the ppp is calibrated with respect to a proper informative prior of the parameters (Hjort et al., 2006). This proper prior is used to construct a reference distribution for the ppp to check how extreme the observed ppp is. We shall refer to the resulting ppp as the prior-calibrated ppp (prior-cppp). The prior-cppp is uniformly distributed under the null model and the chosen proper prior, and therefore results in accurate type I error rates under the null. The prior-cppp can be obtained as follows.

Algorithm 2: Computation of the prior-cppp

Step 1: Specify an informative proper prior, $p(\boldsymbol{\theta})$, based on one's prior knowledge and choose a discrepancy measure, $D(\mathbf{y}; \boldsymbol{\theta})$.

Step 2: Compute the ppp for the observed data using Algorithm 1. The observed ppp will be denoted by ppp_{obs}.

Step 3: Obtain a reference distribution of the ppp using the informative prior in Step 1:

3a: Draw a parameter value, $\boldsymbol{\theta}^{(l)}_{\text{prior}}$, from the informative prior:

$$\boldsymbol{\theta}^{(l)}_{\text{prior}} \sim p(\boldsymbol{\theta}).$$

3b: Sample a data set using the likelihood of the model given the prior draw:

$$\mathbf{y}^{(l)}_{\text{prior}} \sim p(\mathbf{y}_{\text{prior}}|\boldsymbol{\theta}^{(l)}_{\text{prior}}) \tag{5}$$

3c: Compute the corresponding $\text{ppp}^{(l)}$ for data set $\mathbf{y}^{(l)}_{\text{prior}}$ using Algorithm 1.

3d: Repeat Steps 3a to 3c for $l = 1, \ldots, L$ (e.g., $L = 1000$).

Step 4: Calculate the prior-cppp as the proportion of $\text{ppp}^{(l)}$'s that are smaller than or equal than the observed ppp:

$$\text{prior-cppp} = L^{-1} \sum_{l=1}^{L} I(\text{ppp}_{\text{obs}} \geq \text{ppp}^{(l)}), \tag{6}$$

where the indicator function $I(\cdot)$ equals 1 if the observed ppp (ppp_{obs}) is larger than or equal to the prior-based ppp ($\text{ppp}^{(l)}$), and 0 otherwise.

Hence in the posterior predictive check using the prior-cppp, the ppp itself is treated as a test statistic (Step 4) where the informative prior from Step 1 is used to obtain a reference distribution.

Figure 2 shows the reference distribution of prior-based ppp's, $\text{ppp}^{(l)}$ (obtained in Step 3d of Algorithm 2), for the same test and data that resulted in the ppp in Figure 1. Uniform priors were used for the response probabilities of the 4 items. The ppp of the observed data was equal to $\text{ppp}_{\text{obs}} = .146$. The prior-cppp is estimated as the proportion of prior-based ppp's that are smaller than the ppp of the observed data (Step 4 in Algorithm 2; grey area in Figure 2). In this example the prior-cppp was equal to .034.

[Figure 2 about here]

A central property of the posterior predictive check based on the prior-cppp is that it simultaneously tests model fit and prior fit for the observed data. This implies

that the prior-cppp may result in a rejection of the null model either in the case of model misfit (i.e., the model does not fit the observed data) or in the case of prior misfit (i.e., one's prior beliefs about the unknown parameters conflict with the information in the observed data). As a consequence, the prior-cppp may highly depend on the chosen prior. This will be shown in the next section. We argue that the prior-cppp should only be used if a researcher has clear prior information that can be translated to an informative prior for all model parameters, and if the researcher is also interested in testing these prior beliefs simultaneously with the statistical model.

Posterior-calibrated posterior predictive p-values.

In the absence of prior information or when one is only interested in evaluating model misfit with accurate type I error rates, neither the prior-cppp nor the standard ppp can be used. To keep the useful properties of the ppp (i.e., the flexibility to detect any form of model misfit, straightforward computation using standard MCMC algorithms, and its usage with non-informative (improper) priors), while maintaining an accurate type I error rate if the null model is true, we propose to calibrate the ppp under the posterior under the null model. The resulting ppp will be referred to as the posterior-calibrated ppp (posterior-cppp). The exact steps to compute the posterior-cppp are given in Algorithm 3.

Algorithm 3: Computation of the posterior-cppp

Step 1: Specify a prior, $p(\boldsymbol{\theta})$, and choose a discrepancy measure, $D(\mathbf{y}; \boldsymbol{\theta})$.

Step 2: Derive the posterior via (1) and compute the ppp for the observed data using Algorithm 1. The observed ppp will be denoted by ppp_{obs}.

Step 3: Obtain a reference distribution of the ppp using the posterior from Step 2:

3a: Draw a parameter value, $\boldsymbol{\theta}^{(m)}_{\text{post}}$, from the posterior:

$$\boldsymbol{\theta}^{(m)}_{\text{post}} \sim p(\boldsymbol{\theta}|\mathbf{y}_{\text{obs}}).$$

3b: Sample a data set using the likelihood of the model given the posterior draw:

$$\mathbf{y}_{\text{post}}^{(m)} \sim p(\mathbf{y}_{\text{post}}|\boldsymbol{\theta}_{\text{post}}^{(m)}) \tag{7}$$

3c: Compute the corresponding $\text{ppp}^{(m)}$ for data set $\mathbf{y}_{\text{post}}^{(m)}$ using Algorithm 1.

3d: Repeat Steps 3a to 3c for $m = 1, \ldots, M$ (e.g., $M = 1000$).

Step 4: Calculate the posterior-cppp as the proportion of $\text{ppp}^{(m)}$'s that are smaller than or equal to the observed ppp:

$$\text{posterior-cppp} = M^{-1} \sum_{m=1}^{M} I(\text{ppp}_{\text{obs}} \geq \text{ppp}^{(m)}), \tag{8}$$

where the indicator function $I(\cdot)$ equals 1 if the constraint is satisfied, and 0 otherwise.

Note that the only difference between Algorithm 2 for the prior-cppp and Algorithm 3 for the posterior-cppp is in the generation of parameters draws for $\boldsymbol{\theta}$ where either an informative prior is used (Step 3a in Algorithm 2) or the posterior is used (Step 3a in Algorithm 3). When computing the posterior-cppp, we recommend to use diffuse or non-informative priors that are completely dominated by the data. Note that if an informative prior would be used for the computation of the posterior-cppp, the testing criterion would become a hybrid of the prior-cppp and the posterior-cppp. Though one could imagine very specific situations in which this may be useful, researchers typically compute ppp's using diffuse priors (Berkhof et al., 2003; Choi et al., 2010; Hoverd & Sibley, 2013).

Figure 3 shows the reference distribution of posterior-based ppp's, $\text{ppp}^{(m)}$ (obtained in Step 3c of Algorithm 3), for the same test and data that resulted in the ppp and the prior-cppp in Figures 1 and 2, respectively. Uniform priors were used for the response probabilities of the 4 items. The posterior-cppp is estimated as the proportion of posterior-based ppp's that are smaller than the ppp of the observed data (Step 4 in Algorithm 3; grey area in Figure 3). In this example the posterior-cppp was

equal to .018.

[Figure 3 about here]

Next we will investigate the performance of the different posterior predictive checks in various testing problems by looking at frequentist criteria such as type I error rates and power.

Application I: A Monte Carlo Study of a Bayesian Test for Independence

The first application is a simple test of independence of J dichotomous items (outcome 1 or 2). The goal of the application is (i) to illustrate how a posterior predictive check can be conducted using the three different types of ppp's, (ii) to get insight about the type I error rates of the different posterior predictive checks for this simple test, and (iii) to investigate to what degree the prior-cppp depends by the choice of the proper prior. This will be assessed by means of a Monte Carlo study.

The data consists of responses of N individuals to J dichotomous items. We are interested in the following test:

$\mathcal{M}_0$: The J dichotomous variables are independent.

$\mathcal{M}_1$: Not $\mathcal{M}_0$, i.e., there is a dependence between at least two variables.

It is standard practice to specify conjugate priors with independent beta distributions for the response probabilities, denoted by π_{1j}, for items $j = 1, \ldots, J$. The beta prior will be written as $Beta(\pi_{1j}|\alpha_j, \beta_j)$, where α_j and β_j are the hyper parameters discussed in the following subsection. In the case of independent items, as under $\mathcal{M}_0$, the likelihood follows a binomial distribution, resulting in a posterior with independent beta distributions, $Beta(\pi_{1j}|n_{1j} + \alpha_j, N - n_{1j} + \beta_j)$, where n_{1j} is the number of individuals having response 1 to item j (see Appendix A).

We cross-tabulate the items resulting in a contingency table with $S = 2^J$ cells. Thus cell s corresponds to a particular response pattern $\mathbf{y}_s$, for $s = 1, \ldots, S$, e.g., $\mathbf{y}_1$ is a vector of J ones. Under $\mathcal{M}_0$ the probability of pattern $\mathbf{y}_s$, denoted by π_s (with a slight

abuse of notation), is given by

$$\pi_s = \prod_{j=1}^{J} (\pi_{1j})^{d_{js}} (1 - \pi_{1j})^{1-d_{js}}, \tag{9}$$

where the dummy indicator d_{js} equals 1 if the response to variable j in pattern s is 1, and 0 otherwise. For example the response probability of pattern $\mathbf{y}_1$ with J ones equals $\pi_1 = \pi_{11} \times \ldots \times \pi_{1J}$. The number of individuals in the data having response pattern s will be denoted by n_s.

To assess overall model fit of $\mathcal{M}_0$ we can use a discrepancy measure based on the Pearson χ^2-statistic, given by

$$D_{\chi^2}(\mathbf{n}; \boldsymbol{\pi}) = \sum_{s=1}^{S} \frac{n_s - e_s}{e_s}, \tag{10}$$

where e_s is the expected number of individuals having response probability s based on the pattern probabilities in $\boldsymbol{\pi}$, i.e., $e_s = N\pi_s$.

To obtain the k-th posterior draw for the pattern probabilities $\boldsymbol{\pi}$, first draw the $\pi_{1j}^{(k)}$'s from their beta posteriors, and subsequently, plug these draws in (9) to obtain $\boldsymbol{\pi}^{(k)}$ (Step 3a in Algorithm 1). Given the k-th posterior draw, the k-th replicated data set can be drawn from the Multinomial$(y_{\text{rep},1}, \ldots, y_{\text{rep},S} | \pi_1^{(k)}, \ldots, \pi_S^{(k)})$ distribution (Step 3b in Algorithm 1). Subsequently, the observed and replicated discrepancies (Step 3c of Algorithm 1) are calculated as

$$D_{\chi^2}(\mathbf{n}_{\text{obs}}; \boldsymbol{\pi}^{(k)}) = \sum_{s=1}^{S} \frac{n_{\text{obs},s} - e_s^{(k)}}{e_s^{(k)}} \tag{11}$$

$$D_{\chi^2}(\mathbf{n}_{\text{rep}}^{(k)}; \boldsymbol{\pi}^{(k)}) = \sum_{s=1}^{S} \frac{n_{\text{rep},s}^{(k)} - e_s^{(k)}}{e_s^{(k)}}, \tag{12}$$

in which $n_{\text{obs},s}$ and $n_{\text{rep},s}^{(k)}$ are the frequencies of pattern s in the observed data and the k-th replicated data, respectively.

Simulation set-up.

The type I error rates of the ppp, prior-cppp and posterior-cppp were investigated under conditions with $J = 4$ independent dichotomous variables and where the success probabilities π_{1j} in population A are equal to .2 and in population B follow a $Beta(6, 24)$-distribution, for all variables $j = 1, \ldots, J$. Sample sizes of $N = 100$ and $N = 1000$ were considered. Under each condition, 2000 data sets were generated.

For the ppp, non-informative uniform priors were used for the response probabilities π_{1j} by setting the hyper parameters to $\alpha_j = \beta_j = 1$, for all $j = 1 \ldots, J$ (Step 1 of Algorithm 1), see also Appendix A. To compute the ppp, the number of replicated data sets was set to $K = 1,000$ (Step 3 of Algorithm 1). To compute the posterior-cppp also non-informative uniform priors were used for the response probabilities (Step 1 of Algorithm 3).

Three different prior-cppp's were considered based on three different priors (Step 1 of Algorithm 2).

1. Prior 1: $\pi_{1j} \sim Beta(6, 24)$ for all $J = 4$ variables (Figure 4, dotted line). This prior is *in agreement* with population B, and therefore results in prior-cppp's with a uniform distribution in this case. Because this prior is concentrated around .2, it is expected that the sampling distribution of the prior-cppp's is also close to uniform for data generated from population A where $\pi_{1j} = .2$.

2. Prior 2: $\pi_{1j} \sim Beta(15, 15)$ for all $J = 4$ variables (Figure 4, dashed line). This prior is concentrated around .5 and therefore this prior is *not* in agreement with both populations. It is expected that the sampling distributions of this prior-cppp's are not uniformly distributed under population A and B.

3. Prior 3: $\pi_{1j} \sim Beta(1, 1)$ for all $J = 4$ variables (Figure 4, solid line). This uniform prior assumes that every probability value is equally likely. This is a standard prior choice if no prior information is available.

As noted earlier, the prior-cppp simultaneously tests the employed statistical model and

the informative prior. Thus when using the prior-cppp the test can be formulated as

$\mathcal{M}_0$: The J dichotomous variables are independent, and the response probabilities for variable j follow a $\pi_{1j} \sim Beta(\alpha_j, \beta_j)$-prior, for $j = 1, \ldots, J$.

$\mathcal{M}_1$: Not $\mathcal{M}_0$, i.e., there is a dependency between at least two variables and/or the priors under $\mathcal{M}_0$ are not in accordance with the information in the data.

[Figure 4 about here]

Results of the Monte Carlo study.

The type I error rates, which were based on the common significance level of $\alpha = .05$, can be found in the first row of Table 1. The corresponding Monte Carlo errors were computed as $\sqrt{\frac{\hat{p}(1-\hat{p})}{2000}}$, where $\hat{p}$ corresponds to the estimated type I error rates and the denominator corresponds to 2000 because the estimate is based on 2000 randomly generated data sets. As can be seen, the type I error rates for the ppp are around .002, which is much too low. This can be understood by looking at the sampling distribution of the ppp's which are plotted in Figure 5 for each of the four scenarios (dotted lines). As can be seen the sampling distributions of the ppp's are peaked around .55, which explains the dramatically low type I error rates. As a consequence the ppp test is too conservative.

[Figure 5 about here]

Table 1 (second row, last two columns) shows that the type I error rates of the prior-cppp are accurate when the prior is correctly specified, i.e., when using $Beta(6, 24)$-priors in the case of $Beta(6, 24)$-distributions in the populations. Also the corresponding sampling distributions are approximately uniform (Figure 5; right panels, dash-dotted lines). In all other situations, the prior-cppp neither results in accurate type I error rates nor in approximately uniform sampling distributions. This is even the case when calibrating the prior-cppp under standard uniform priors for the response

probabilities (Table 1; $Beta(1, 1)$ in the fourth row). Furthermore, it is interesting to observe that the rejection rates for the prior-cppp are lower than 5% in most case when the prior does not correspond with the population distribution (except when using the $Beta(15, 15)$-prior and $N = 100$). This is somewhat surprising because one might expect a larger rejection rate than 5% due to the mismatch of the prior.

The results for the posterior-cppp are all acceptable. As can be seen in Figure 5 (thick solid lines), the sampling distributions are approximately uniform in all scenarios. Furthermore, the last row in Table 1 shows that the type I error rates of the posterior-cppp do not differ significantly from .05 in all scenarios.

In sum, this simple example provides the following useful insights about posterior predictive checking. First, the standard ppp, even though it is flexible and simple to compute, does not result in accurate type I error rates, not even in this very simple test for independence. Second, the prior-cppp highly depends on the exact choice of the specified proper prior. For this reason we cannot recommend the prior-cppp for default model checking. Therefore, the prior-cppp will not be considered further in this paper. Third, the posterior-cppp with standard diffuse priors clearly outperforms the ppp and the prior-cppp by providing accurate type I error rates.

Application II: A Bayesian Test for Extreme Observations in Linear Regression Analysis

To illustrate the generality of the proposed approach, we evaluate the type I error rates and power of the posterior-cppp in a regression model. The standard linear regression model assumes that a dependent variable can be explained by a linear combination of certain predictor variables and a normally distributed error. The model can be written as

$$y_i \sim N(\mathbf{x}_i'\boldsymbol{\beta}, \sigma^2), \tag{13}$$

for $i = 1, \ldots, N$, where y_i is the i-th observation of the dependent variable, $\mathbf{x}_i$ is a vector with k predictor variables of the i-th observation, $\boldsymbol{\beta}$ is a vector with k unknown regression coefficients, and σ^2 is the error variance. The standard independence Jeffreys

prior is used for the unknown parameters $(\boldsymbol{\beta}, \sigma^2)$ (Step 1 of Algorithm 1 and 3). The posterior follows a normal-inverse-gamma distribution (Step 2 of Algorithm 1; Step 3 of Algorithm 3). See Appendix B for details.

To illustrate the flexibility of posterior predictive checking we shall test whether the employed linear regression model appropriately captures extreme observations. This can be achieved using the following discrepancy measure (Hjort et al., 2006):

$$D_{\max}(\mathbf{y}, \mathbf{X}; \boldsymbol{\beta}, \sigma^2) = \max_{i \in \{1,\ldots,n\}} |y_i - \mathbf{x}_i' \boldsymbol{\beta}|/\sigma, \tag{14}$$

where $\mathbf{y} = (y_1, \ldots, y_n)'$ is the vector containing the N observations of the dependent variable, and $\mathbf{X}$ is a $N \times k$ matrix where the i-th row contains the k predictor variables, denoted by $\mathbf{x}_i$. The discrepancy measure computes the largest standardized error between the observations, y_i, and their predictions according to the model, i.e., $\mathbf{x}_i' \boldsymbol{\beta}$. Note that we are not interested in determining which observations are extreme (i.e., we are not doing outlier detection); we are only interested in checking whether the employed linear regression model appropriately captures extreme observations.

For a given posterior draw $(\boldsymbol{\beta}^{(k)}, \sigma^{2,(k)})$, a replicated data set $\mathbf{y}_{\text{rep}}^{(k)}$ can be obtained via (13) using the matrix of covariates from the observed data, $\mathbf{X}_{\text{obs}}$. Note that is also standard practice to assume the covariates to be fixed in Bayesian linear regression. The observed and replicated discrepancies are then given by

$$D_{\max}(\mathbf{y}_{\text{obs}}, \mathbf{X}_{\text{obs}}; \boldsymbol{\beta}^{(k)}, \sigma^{2,(k)}) = \max_{i \in \{1,\ldots,n\}} |y_{\text{obs},i} - \mathbf{x}_{\text{obs},i}' \boldsymbol{\beta}^{(k)}|/\sigma^{(k)}, \tag{15}$$

$$D_{\max}(\mathbf{y}_{\text{rep}}^{(k)}, \mathbf{X}_{\text{obs}}; \boldsymbol{\beta}^{(k)}, \sigma^{2,(k)}) = \max_{i \in \{1,\ldots,n\}} |y_{\text{rep},i}^{(k)} - \mathbf{x}_{\text{obs},i}' \boldsymbol{\beta}^{(k)}|/\sigma^{(k)}. \tag{16}$$

Monte Carlo study when testing extreme observations.

A Monte Carlo simulation was conducted to investigate whether the ppp and the posterior-cppp are able to pick up extreme observations using the discrepancy measure in Equations (15) and (16). Two different forms of misfit were considered. First, instead of a normal distribution, errors were generated using a Student t distribution

with 1, 2, 5, 20, and 50 degrees of freedom. Note that when the degrees of freedom equals ∞ the errors are normally distributed, while 1 degree of freedom corresponds to a Cauchy distribution resulting in much more extreme observations than normally distributed errors. Second, the residual standard deviation was set to be a monotonic function of the sum of the explanatory variables, which results in heteroskedastic errors. The residual standard deviation for the i-th observation was set to $\sigma_i = 1 + c \times \frac{w_i - w_{\min}}{w_{\max} - w_{\min}}$, where $w_i = \mathbf{x}_i'\mathbf{1}$ is the sum of the explanatory variables, and $w_{\min} = \min_i\ w_i$ and $w_{\max} = \max_i\ w_i$. Larger values for c imply larger error variances for large values of the explanatory variables. Note that $c = 0$ implies homoskedastic errors of $N(0, 1)$. For the simulation we choose the following values $c = 1,\ 2,\ 3,\ 5$, and 10. Sample sizes were set to $n = 50$, 100, and 250. For every condition, 1000 datasets were generated using 3 explanatory variables from independent standard normal distributions and regression coefficients equal to $\boldsymbol{\beta} = (.3, .3, .3)$.

The results on the type I error rates and the power can be found in Table 2. Again these results show that the ppp is too conservative with error rates that are too small for all sample sizes. The posterior-cppp on the other hand results in reasonable type I error rates. Furthermore, the posterior-cppp consistently has higher power than the ppp in the case of model misfit.

[Table 2 about here]

An empirical analysis of quality-of-life in elderly.

A posterior predictive check was performed to detect model misfit for a regression model testing the effects of physical, psychological and social frailty on the quality of life with respect to social relationships of elderly people (Age, mean±SD 84.8±9.7, range 55–101)(Gobbens, Krans, & van Assen, 2015). The sample consisted of $n = 156$ observations. The regression model consisted of the three explanatory variables of frailty, presence of disease, and 8 control variables (such as marital status and income), as well as an intercept.

The ppp and the posterior-cppp were computed with the discrepancy measure in

Equation (14) using the standard independence Jeffreys prior. The ppp calculated with $K = 500$ replications was equal to .064, which would generally not be considered to indicate significant misfit. The posterior-cppp with $M = 500$, on the other hand, was equal to .012, which would in most situations be considered to indicate significant misfit. For this reason, it is recommendable to reconsider the employed regression model before making inferences about the quality of life of elderly people.

Application III: Bayesian Tests for Latent Class Analysis

To get more insights about the performance of posterior predictive checks in more complex situations, we shall test model misfit of latent class models. Latent class models are commonly used to create typologies or classifications of observations, based on their response patterns (Goodman, 1974). It has been shown that p-values based on asymptotic sampling distributions, p-values based on the parametric bootstrap, and posterior predictive p-values based on test statistics may not result in accurate type I error rates for this type of model (van Kollenburg et al., 2015). For this reason, the latent class model is a good test case to check the performance of the posterior-cppp.

Again note that in order to apply the prior-cppp, informative priors need to be formulated for all the unknown model parameters in the latent class model, such as the latent class proportions and the response probabilities for a given latent class. This is not feasible from a practical point of view. Based on our experience researchers are mainly interested in evaluating the fit of a latent class model, and not in evaluating the prior beliefs about the unknown model parameters. These considerations again exemplify the limited usability of the prior-cppp in more complex models. Therefore the prior-cppp will not be considered in this application.

We shall consider a latent class model for a given data set of N individuals who responded to J dichotomous items, when assuming the individuals can be divided into C latent classes. Appendix D contains all the technical details of the latent class model. Two Monte Carlo studies will be conducted: (i) testing the assumption of local independence, and (ii) testing for the number of latent classes. Additionally, the

posterior-cppp will be used for an empirical latent class analysis on sub-types of depression in males.

Monte Carlo study on bivariate residuals

Let us first focus on the key assumption of the latent class model that the observations of each pair of items are independent given the (unknown) latent class memberships of the individuals. To test whether this property is violated for item pair (j, j'), we use the bivariate residual (BVR) (Vermunt & Magidson, 2016):

$$D_{BVR_{jj'}}(\mathbf{n}; \boldsymbol{\rho}, \boldsymbol{\pi}) = \sum_{s=1}^{4} \frac{(n_s - e_s)^2}{e_s}, \tag{17}$$

where the sum is over the $S = 2^2 = 4$ cells of the contingency table, n_s is the observed frequency of response pattern s, and $e_s = N\pi_s$ denotes the expected frequency of response pattern s of the variable pair (j, j') given the latent class proportions $\boldsymbol{\rho}$ and the response probabilities $\boldsymbol{\pi}$ of the latent class model (see Appendix D for the technical details).

For a given posterior draw of the latent class proportions, $\boldsymbol{\rho}^{(k)}$, and response probabilities, $\boldsymbol{\pi}^{(k)}$, the observed and replicated discrepancies are calculated as

$$D_{BVR_{jj'}}(\mathbf{n}_{\text{obs}}; \boldsymbol{\rho}^{(k)}, \boldsymbol{\pi}^{(k)}) = \sum_{s=1}^{4} \frac{(n_{\text{obs},s} - e_s^{(k)})^2}{e_s^{(k)}}, \tag{18}$$

$$D_{BVR_{jj'}}(\mathbf{n}_{\text{rep}}^{(k)}; \boldsymbol{\rho}^{(k)}, \boldsymbol{\pi}^{(k)}) = \sum_{s=1}^{4} \frac{(n_{\text{rep},s}^{(k)} - e_s^{(k)})^2}{e_s^{(k)}}, \tag{19}$$

where $\mathbf{n}_{\text{rep}}^{(k)}$ denotes the frequencies of the response patterns of a replicated data set generated using $(\boldsymbol{\rho}^{(k)}, \boldsymbol{\pi}^{(k)})$ (see Appendix D).

A Monte Carlo simulation was conducted to evaluate the power and type I error rates of the posterior-cppp when testing conditional independence for pairs of variables using the BVR. We generated $J = 6$ dichotomous variables from a latent class model with $C = 2$ equally sized classes (i.e., $\rho_1 = \rho_2 = .5$). Sample sizes were either $N = 100$, 500 or 1000. The conditional probabilities for a 1-response were $\pi_{1jc} = .8$ in class 1 and

$\pi_{1jc} = .2$ in class 2 for variables $j = 1$ to 5. For class 1, the probability of 1-response to variable 6 conditional on having a 1-response to variable 5 was set to $(\pi_{161}|y_5 = 1) = .8 + \delta$, where δ was used to add conditional association/local dependence between the last two variables. For observations with a 2-response on variable 5, we set $(\pi_{161}|y_5 = 2) = .8$. In class 2, all response probabilities were set to the complement of the response probability of class 1, i.e., $\pi_{rj2} = 1 - \pi_{rj1}$. Thus the 2-class model fits when $\delta = 0$, while local independence is violated between variables 5 and 6 in the 2-class model when $\delta \neq 0$. Table 3 summarizes the response probabilities for all variables.

Under each condition 500 datasets were generated. The original ppp and the posterior-cppp were computed using vague uniform $Beta(1, 1)$-priors for all model parameters (i.e., class proportion ρ_c, and the probabilities of a 1-response in each class, π_{1jc}, for $j = 1, \ldots, 6$, and $c = 1$, or 2). To compute the original ppp, $K = 500$ replications were generated. To compute the posterior-cppp, a reference distribution was obtained using $M = 500$ posterior-based data sets. To compute each ppp for the reference distribution $K = 501$ replications were used, so that there are no ties with the observed ppp.

The results for the study on bivariate residuals are displayed in Table 4. The condition in which $\delta = 0$ confirms that the posterior-cppp has accurate type I errors which is not the case for the standard ppp. Moreover, the results show that the power to detect misfit (in the current example being local dependency between pairs of variables) is greatly improved by calibrating the original ppp with respect to the posterior distribution, as is done in the posterior-cppp.

[Table 4 about here]

Monte Carlo study on the number of latent classes.

The second major testing problems involves checking whether enough latent classes are specified. To do this we can assess the overall misfit of the latent class model. This can be done using the Pearson χ^2 or the likelihood ratio G^2 as discrepancy

measure, which are given by

$$D_{\chi^2}(\mathbf{n}; \boldsymbol{\rho}, \boldsymbol{\pi}) = \sum_{s=1}^{S} \frac{(n_s - e_s)^2}{e_s}, \tag{20}$$

and

$$D_{G^2}(\mathbf{n}; \boldsymbol{\rho}, \boldsymbol{\pi}) = 2 \sum_{s=1}^{S} n_s \ln(n_s/e_s), \tag{21}$$

where the sum is over all S possible response patterns, n_s denotes the observed frequency of response pattern s, and $e_s = N\pi_s$ denotes the expected frequency of response pattern s given the model parameters (Appendix D).

Type I error rates of the ppp and the posterior-cppp for the discrepancy measures in (20) and (21) were investigated under a null model with two latent classes, by means of a Monte Carlo simulation. The population under the null model had $C = 2$ equally sized classes, with conditional response probabilities for a 1-response of $\pi_{1jc} = .8$ in class 1 and $\pi_{1jc} = .2$ in class 2 for all $J = 6$ dichotomous variables. The power of the tests was investigated by assuming a population with $C = 3$ equally sized classes with the same conditional probabilities for classes 1 and 2. For class 3, the probability of 1-response was $\pi_{1jc} = .8$ for the first half of the variables and $\pi_{1jc} = .2$ for the last half of the variables. Table 5 shows the parameter values for the conditions in which data was generated from a three-class model with six variables. Sample sizes were either $N = 100$, or 500.

[Table 5 about here]

For these conditions we ran 500 Monte Carlo simulations per condition. The ppp was calculated with $K = 500$ replications, and calibrated using $M = 500$ posterior-based data sets on which we performed a posterior predictive check with $K = 501$ replications. The ppp and the posterior-cppp were computed using default uniform $Beta(1, 1)$-priors.

The results for the study on the number of classes can be found in Table 6. The conditions in which the fitted model holds (when the true number of classes equals 2) confirm that the posterior-cppp has accurate type I error rates, unlike the ppp.

Moreover, the posterior-cppp clearly has more power than the ppp to detect model misfit.

[Table 6 about here]

An empirical analysis of sub-types of depression in males.

The posterior-cppp was used to analyse the depression scale data for white male respondents from the problems of everyday life study (Pearlin & Johnson, 1977; Schaeffer, 1988). Persons who reported to have a symptom in the previous week were coded 1, all others were coded 0. Five symptoms were measured, namely, lack of enthusiasm, low energy, sleeping problems, poor appetite, and feeling hopeless. The data set consisted of 748 males. Research has shown that a 2-class model does not adequately fit these data while a 3-class model does (Magidson & Vermunt, 2001). Here, we will check whether the same result is obtained using the new posterior-cppp.

First a latent class model was fitted with 2 latent classes. Model misfit was assessed using the Pearson χ^2-statistic to test if conditional independence was violated over all variables, as well as with the BVR to test if conditional independence was violated between pairs of variables. This was done using the standard ppp and the posterior-cppp. The results can be found in Table 7. As can be seen, the ppp for the χ^2 test is not significant using a significance level of .05 while the posterior-cppp is significant with a value of .001. Furthermore, none of the BVR-tests are significant using the ppp while the posterior-cppp suggests there is model misfit for variable pairs (1,2), (2,5), (3,4), (3,5), and (4,5) using a significance level of .05. Furthermore the BVR for the variable pair (2,5) is still significant after a Bonferroni correction of the significance level to $.05/10 = .005$. These results show that the standard ppp is unable to detect model misfit and would thus result in incorrect conclusions about the true number of latent classes. The posterior-cppp on the other hand does not have this problem.

Because of the misfit of the 2-class model as indicated by the posterior-cppp, a 3-class model was fitted as well. The ppp's and posterior-cppp's can be found in Table 7. In this case, the posterior-cppp's do not indicate any serious form of model misfit.

This analysis confirms that the depression scale data for males can be adequately described using a 3-class model.

Discussion

Posterior predictive checking is a very flexible methodology to evaluate various forms of model misfit without relying on large sample theory. The most commonly used posterior predictive check is based on the posterior predictive p-value (ppp), which can efficiently be computed using MCMC output (Gelman et al., 1996; Meng, 1994). A problem with this approach however is that the ppp is generally too conservative, which results in tests with very low statistical power.

The prior-calibrated posterior predictive p-value (prior-cppp) resolves this issue by calibrating the ppp under a proper prior distribution for all model parameters. The prior-cppp simultaneously checks model misfit and prior misfit. As a result, the choice of the prior that is used for the calibration has a serious effect on the outcome of the test. Therefore the prior-cppp should only be used when clear prior information is available for all model parameters and one is interested in simultaneously testing the model and the prior. In our experience however this is hardly ever the case. Either clear prior information is not available for all model parameters, or researchers are only interested in testing whether the employed statistical model fits the observed data.

As an alternative the posterior-calibrated posterior predictive p-value (posterior-cppp) was proposed. The posterior-cppp is obtained by calibrating the ppp under the posterior given the observed data under the null model. The posterior-cppp has all the advantages of the original ppp, i.e., it can be used to detect any form of misfit, it can be computed from MCMC output, and it can be computed using non-informative improper priors. In addition, the posterior-cppp also results in accurate type I error rates, which is not the case for the original ppp. Moreover, the posterior-cppp results in more statistical power than the ppp. The usefulness of the posterior-cppp was illustrated in different testing problems, such as testing independence of dichotomous variables, assessing misfit of regression models in the case

of extreme observations, and testing misfit in latent class models.

A potential drawback of the posterior-cppp (and the prior-cppp) is that it requires more computational time than the standard ppp because an additional calibration step is needed. In order to compute the posterior-cppp, say, 500 ppp's need to be calculated which takes maximally 500 times longer than computing the standard ppp. These 500 ppp's however can be computed in parallel and therefore computation time can be drastically reduced. In the empirical regression application, for example, the computation of the standard ppp was .05 seconds and the computation of the posterior-cppp was 7.1 seconds. For this reason we believe the additional computational time of the posterior-cppp hardly limits its applicability.

References

Akaike, H. (1998). Information theory and an extension of the maximum likelihood principle. In *Selected papers of hirotugu akaike* (pp. 199–213). Springer.

Berger, J., et al. (2006). The case for objective bayesian analysis. *Bayesian analysis*, *1*(3), 385–402.

Berkhof, J., Van Mechelen, I., & Gelman, A. (2003). A bayesian approach to the selection and testing of mixture models. *Statistica Sinica*, *13*(2), 423–442.

Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. B. (2014). Julia: A fresh approach to numerical computing. *CoRR*, *abs/1411.1607*. Retrieved from `http://arxiv.org/abs/1411.1607`

Chaves, R. L., Chakraborty, A., Benziger, D., & Tannenbaum, S. (2014). Clinical and pharmacokinetic considerations for the use of daptomycin in patients with staphylococcus aureus bacteraemia and severe renal impairment. *Journal of Antimicrobial Chemotherapy*, *69*(1), 200–210.

Choi, J., Hui, S. K., & Bell, D. R. (2010). Spatiotemporal analysis of imitation behavior across new buyers at an online grocery retailer. *Journal of Marketing Research*, *47*(1), 75–89.

Gelman, A., Carlin, J. B., Stern, H. S., & Rubin, D. B. (2004). *Bayesian data analysis*. Chapman & Hall/CRC.

Gelman, A., Meng, X. L., & Stern, H. (1996). Posterior predictive assessment of model fitness via realized discrepancies. *Statistica Sinica*, *6*, 733–759.

Gobbens, R. J., Krans, A., & van Assen, M. A. (2015). Validation of an integral conceptual model of frailty in older residents of assisted living facilities. *Archives of gerontology and geriatrics*, *61*(3), 400–410.

Goodman, L. A. (1974). Exploratory latent structure analysis using both identifiable and unidentifiable models. *Biometrika*, *61*(2), 215–231.

Hjort, N., Dahl, F., & Steinbakk, G. (2006). Post-processing posterior predictive p values. *Journal of the American Statistical Association*, *101*(475), 1157–1174.

Hoverd, W. J., & Sibley, C. G. (2013). Religion, deprivation and subjective wellbeing:

Testing a religious buffering hypothesis. *International Journal of Wellbeing*, *3*(2).

Jeffreys, H. (1961). Theory of probability, harold jeffreys. *International series of monographs on physics.*.

Kass, R., & Raftery, A. (1995). Bayes factors. *Journal of the american statistical association*, *90*(430), 773–795.

Liang, F., Liu, C., & Carroll, R. (2011). *Advanced markov chain monte carlo methods: learning from past samples* (Vol. 714). John Wiley & Sons.

Magidson, J., & Vermunt, J. K. (2001). Latent class factor and cluster models, bi-plots, and related graphical displays. *Sociological methodology*, *31*(1), 223–264.

Meng, X. L. (1994). Posterior predictive p-values. *The Annals of Statistics*, *22*(3), 1142–1160.

Mulder, J. (2014). Bayes factors for testing inequality constrained hypotheses: Issues with prior specification. *British Journal of Mathematical and Statistical Psychology*, *67*(1), 153–171.

Myung, I. J. (2000). The importance of complexity in model selection. *Journal of Mathematical Psychology*, *44*(1), 190–204.

Oravecz, Z., Faust, K., Batchelder, W. H., & Levitis, D. A. (2015). Studying the existence and attributes of consensus on psychological concepts by a cognitive psychometric model. *The American Journal of Psychology*, *128*(1), 61–75.

Pearlin, L. I., & Johnson, J. S. (1977). Marital status, life-strains and depression. *American sociological review*, *42*, 704–715.

Robins, J. M., van der Vaart, A., & Ventura, V. (2000). Asymptotic distribution of p values in composite null models. *Journal of the American Statistical Association*, *95*(452), 1143–1156.

Schaeffer, N. C. (1988). An application of item response theory to the measurement of depression. *Sociological methodology*, *18*, 271–307.

Schwarz, G., et al. (1978). Estimating the dimension of a model. *The annals of statistics*, *6*(2), 461–464.

Tanner, M. A., & Wong, W. H. (1987). The calculation of posterior distributions by

data augmentation. *Journal of the American statistical Association*, *82*(398), 528–540.

van Kollenburg, G. H., Mulder, J., & Vermunt, J. K. (2015). Assessing model fit in latent class analysis when asymptotics do not hold. *Methodology: European Journal of Research Methods for the Behavioral and Social Sciences*, *11*(2), 65–79.

Vermunt, J. K., & Magidson, J. (2016). Technical guide for Latent GOLD 5.1: Basic, advanced and syntax. *Belmont Massachusetts: Statistical Innovations Inc.*

Table 1
Type I error rates with Monte Carlo errors for the ppp, three prior-cppp's based on a $Beta(6, 24)$-prior, a $Beta(15, 15)$-prior, and a $Beta(1, 1)$-prior, and the posterior-cppp.

Population	$\pi_{1j} = .2$		$\pi_{1j} \sim Beta(6, 24)$	
Sample size	$N = 100$	$N = 1000$	$N = 100$	$N = 1000$
ppp	$.002 \pm .001$	$.002 \pm .001$	$.002 \pm .001$	$.000 \pm .000$
prior-cppp with $Beta(6, 24)$-prior	$.030 \pm .004$	$.043 \pm .005$	$.042 \pm .004$	$.059 \pm .005$
prior-cppp with $Beta(15, 15)$-prior	$.643 \pm .011$	$.043 \pm .005$	$.646 \pm .011$	$.037 \pm .004$
prior-cppp with $Beta(1, 1)$-prior	$.023 \pm .003$	$.030 \pm .004$	$.029 \pm .004$	$.030 \pm .004$
posterior-cppp	$.043 \pm .005$	$.048 \pm .005$	$.047 \pm .005$	$.047 \pm .005$

Table 2
Estimated type I error rates (first row), power in the case of Student $t(\nu)$ distributed errors with degrees of freedom ν (second to sixth row), and power in the case of heterogeneous normally distributed errors with $\sigma_i = 1 + c \times \frac{w_i - w_{min}}{w_{max} - w_{min}}$.

	Sample Size					
	50		100		250	
$\epsilon \sim$	ppp	post-cppp	ppp	post-cppp	ppp	post-cppp
$N(0,1)$	$.004 \pm .002$	$.045 \pm .007$	$.015 \pm .004$	$.041 \pm .006$	$.034 \pm .006$	$.058 \pm .007$
t(50)	$.007 \pm .003$	$.073 \pm .008$	$.028 \pm .005$	$.073 \pm .008$	$.053 \pm .007$	$.079 \pm .009$
t(20)	$.016 \pm .004$	$.078 \pm .008$	$.059 \pm .007$	$.141 \pm .011$	$.127 \pm .011$	$.161 \pm .012$
t(5)	$.168 \pm .012$	$.345 \pm .015$	$.388 \pm .015$	$.553 \pm .016$	$.710 \pm .014$	$.779 \pm .013$
t(2)	$.609 \pm .015$	$.798 \pm .013$	$.906 \pm .009$	$.947 \pm .007$	$.998 \pm .001$	$.998 \pm .001$
t(1)	$.928 \pm .008$	$.977 \pm .005$	$.998 \pm .001$	$.998 \pm .001$	$1.000 \pm .000$	$1.000 \pm .000$
$c = 1$	$.045 \pm .007$	$.184 \pm .012$	$.103 \pm .010$	$.226 \pm .013$	$.204 \pm .013$	$.283 \pm .014$
$c = 2$	$.086 \pm .009$	$.316 \pm .015$	$.244 \pm .014$	$.390 \pm .015$	$.413 \pm .016$	$.488 \pm .016$
$c = 3$	$.144 \pm .011$	$.408 \pm .016$	$.319 \pm .015$	$.480 \pm .016$	$.548 \pm .016$	$.621 \pm .015$
$c = 5$	$.196 \pm .013$	$.459 \pm .016$	$.422 \pm .016$	$.617 \pm .015$	$.640 \pm .015$	$.727 \pm .014$
$c = 10$	$.270 \pm .014$	$.569 \pm .016$	$.528 \pm .016$	$.687 \pm .015$	$.778 \pm .013$	$.852 \pm .011$

Table 3
Class proportions ρ_c and response probabilities π_{rjc} for response r on dichotomous variable j under class c, for $r = 1$ or 2, $j = 1, \ldots, 6$, and class $c = 1$ or 2. Note that $\pi_{2jc} = 1 - \pi_{1jc}$

class	$c = 1$	$c = 2$
ρ_c	.5	.5
π_{11c}	.8	.2
π_{12c}	.8	.2
π_{13c}	.8	.2
π_{14c}	.8	.2
π_{15c}	.8	.2
$\pi_{16c}\|y_5 = 1$	$.8 + \delta$	$.2 - \delta$
$\pi_{16c}\|y_5 = 2$	.8	.2

Table 4
Estimated type-I error rates (row where $\delta = 0$) and power (rows where $\delta \neq 0$) when testing local independence in a 2-class model using a significance level of .05.

	Sample size					
	100		500		1000	
δ	ppp	post-cppp	ppp	post-cppp	ppp	post-cppp
$-.2$	$.012 \pm .005$	$.186 \pm .017$	$.246 \pm .019$	$.912 \pm .013$	$.764 \pm .019$	$1.000 \pm .000$
$-.1$	$.000 \pm .000$	$.032 \pm .008$	$.000 \pm .000$	$.322 \pm .021$	$.016 \pm .006$	$.646 \pm .021$
$-.05$	$.000 \pm .000$	$.042 \pm .009$	$.000 \pm .000$	$.064 \pm .011$	$.000 \pm .000$	$.180 \pm .017$
.00	$.000 \pm .000$	$.058 \pm .010$	$.000 \pm .000$	$.048 \pm .010$	$.000 \pm .000$	$.040 \pm .009$
.05	$.000 \pm .000$	$.108 \pm .014$	$.000 \pm .000$	$.232 \pm .019$	$.000 \pm .000$	$.362 \pm .021$
.1	$.000 \pm .000$	$.208 \pm .018$	$.000 \pm .000$	$.548 \pm .022$	$.000 \pm .000$	$.840 \pm .016$
.2	$.000 \pm .000$	$.426 \pm .022$	$.000 \pm .000$	$.768 \pm .019$	$.000 \pm .000$	$.820 \pm .017$

Table 5
Class proportions ρ_c and response probabilities π_{rjc} for response 1 on dichotomous variable j under class c, for $r = 1$ or 2, $j = 1, \ldots, 6$, and class $c = 1$ or 2. Note that $\pi_{2jc} = 1 - \pi_{1jc}$

class	$c = 1$	$c = 2$	$c = 3$
ρ_c	1/3	1/3	1/3
π_{11c}	.8	.2	.8
π_{12c}	.8	.2	.8
π_{13c}	.8	.2	.8
π_{14c}	.8	.2	.2
π_{15c}	.8	.2	.2
π_{16c}	.8	.2	.2

Table 6
Estimated type-I error rates (rows where $C = 2$) and power (rows where $C = 3$) when testing the global fit of a two-class model when using a significance level of .05.

			χ^2		G^2	
C	N	π_{1j1}	ppp	$\text{cppp}_{\text{post}}$	ppp	$\text{cppp}_{\text{post}}$
2	100	.8	$.004 \pm .003$	$.054 \pm .010$	$.036 \pm .008$	$.062 \pm .011$
		.9	$.008 \pm .004$	$.028 \pm .007$	$.018 \pm .006$	$.052 \pm .010$
	500	.8	$.000 \pm .000$	$.038 \pm .009$	$.032 \pm .008$	$.050 \pm .010$
		.9	$.002 \pm .002$	$.046 \pm .009$	$.032 \pm .008$	$.046 \pm .009$
3	100	.8	$.402 \pm .022$	$.632 \pm .022$	$.372 \pm .022$	$.478 \pm .022$
		.9	$.416 \pm .022$	$.660 \pm .021$	$.372 \pm .022$	$.470 \pm .022$
	500	.8	$1.000 \pm .000$	$1.000 \pm .000$	$1.000 \pm .000$	$1.000 \pm .000$
		.9	$1.000 \pm .000$	$1.000 \pm .000$	$1.000 \pm .000$	$1.000 \pm .000$

Table 7
Model fit results for the depression data

	2-class model		3-class model	
Discrepancy	ppp	posterior-cppp	ppp	posterior-cppp
χ^2	.140	.001	.489	.597
BVR_{12}	.424	.013	.415	.011
BVR_{13}	.549	.790	.545	.915
BVR_{14}	.498	.341	.523	.768
BVR_{15}	.500	.345	.557	.972
BVR_{23}	.512	.558	.545	.934
BVR_{24}	.502	.429	.509	.524
BVR_{25}	.258	.002	.389	.031
BVR_{34}	.143	.008	.314	.017
BVR_{35}	.159	.018	.521	.719
BVR_{45}	.239	.041	.527	.805

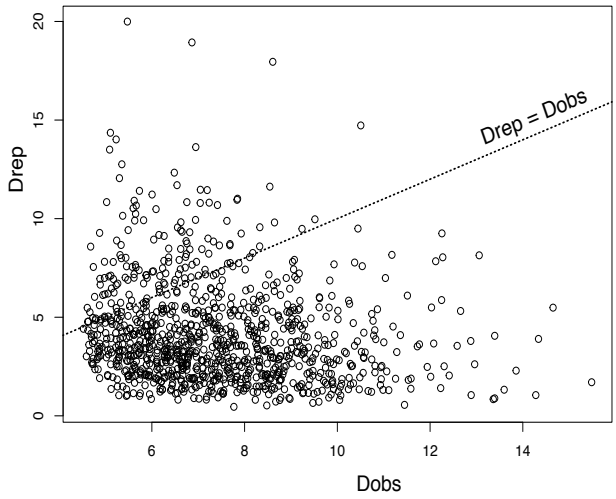

Figure 1. Plot of pairs of observed and replicated discrepancies, $D(\mathbf{y}_{\text{obs}}; \boldsymbol{\theta}^{(k)})$ (Dobs) and $D(\mathbf{y}_{\text{rep}}^{(k)}; \boldsymbol{\theta}^{(k)})$ (Drep), for $k = 1, \ldots, 1000$, when testing independence of 4 dichotomous variables using a Pearson χ^2 discrepancy measure. The pairs were obtained using Algorithm 1. The ppp is defined as the proportion of pairs where the replicated discrepancies are at least as large as the observed discrepancies, which was equal to .146.

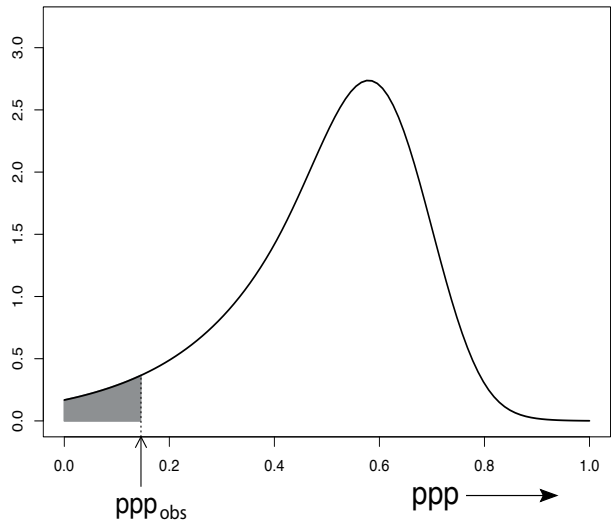

Figure 2. Reference distribution of prior-based ppp's, $\text{ppp}^{(l)}$, for a test of independence of 4 dichotomous variables using uniform priors for the response probabilities, for $l = 1, \ldots, 1000$ (Algorithm 2). The prior-cppp is estimated as the proportion of prior-based ppp's that are smaller than the observed ppp (grey area). The prior-cppp was equal to .034.

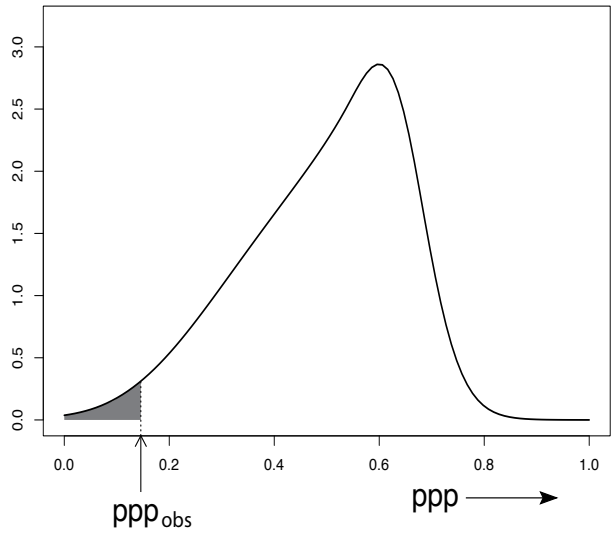

Figure 3. Reference distribution of posterior-based ppp's, $\text{ppp}^{(m)}$, for a test of independence of 4 dichotomous variables using uniform priors for the response probabilities, for $m = 1, \ldots, 1000$ (Algorithm 3). The posterior-cppp is estimated as the proportion of posterior-based ppp's that are smaller than the observed ppp (grey area). The posterior-cppp was equal to .018.

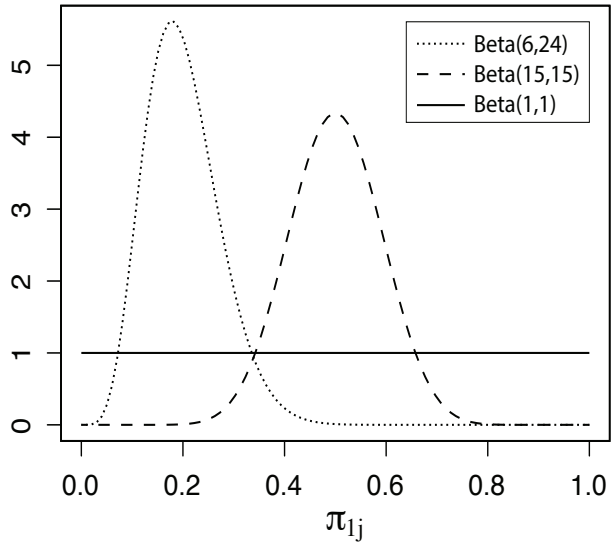

Figure 4. Three beta-priors for the probability of a success to variable j.

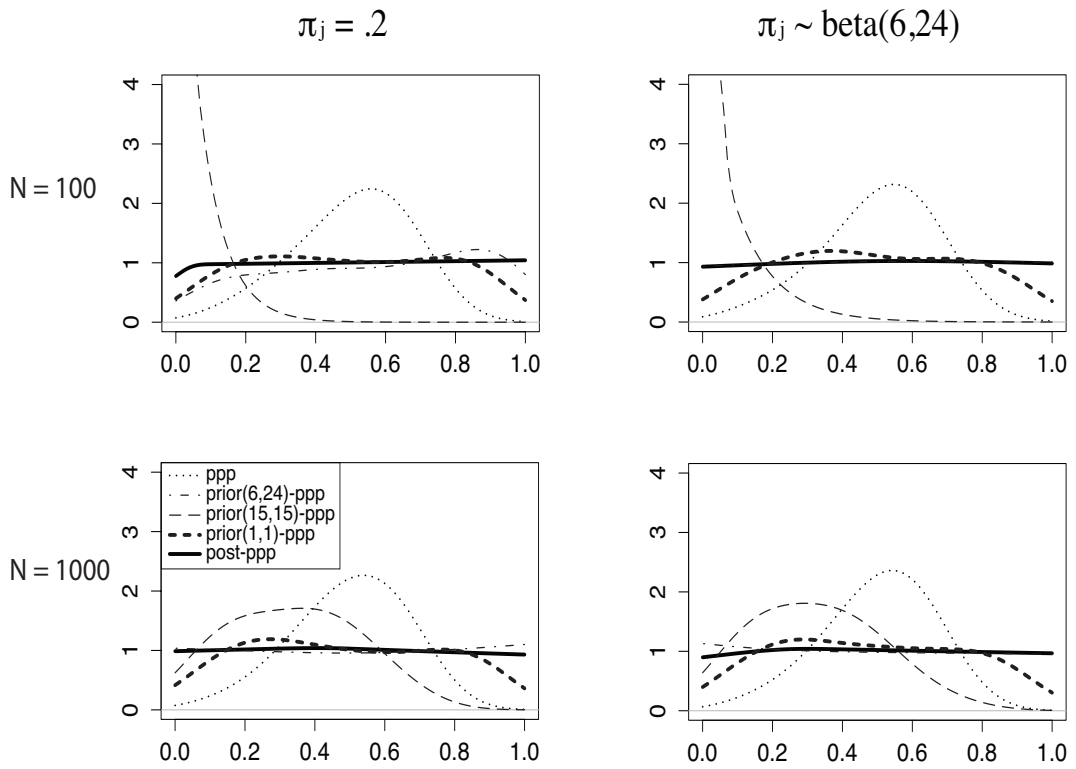

Figure 5. The sampling distribution of different posterior predictive p-values (ppp's) in the case of $J = 4$ dichotomous variables, $N = 100$ observations (upper panels) and $N = 1000$ (lower panels), and a true population where $\pi_{1j} = .2$ and $\pi_{1j} \sim Beta(6, 24)$, $j = 1, \ldots, 4$. Distributions are displayed for the standard ppp using a uniform prior for π_{1j} (dotted line), three different prior-cppp's based on a $Beta(6, 24)$-prior (dash-dotted line), a $Beta(15, 15)$-prior (thin dashed line), and a $Beta(1, 1)$-prior (thick dashed line), and the posterior-cppp using a uniform prior (thick solid line).

Appendix A

Deriving the Posterior when Testing for Independence

For a data set with sample size N, let n_{1j} and $N - n_{1j}$ be the observed number of persons with scores 1 and 0 on the j-th variable, and let π_{1j} and $1 - \pi_{1j}$ be the corresponding probabilities, for $j = 1, \ldots, J$. Thus, the vector of model parameters consists of the unknown probabilities $(\pi_{11}, \ldots, \pi_{1J})$. Under the assumption that the J variables are independent, the likelihood is obtained as a product of J independent binomial distributions, i.e.,

$$p(n_{11}, \ldots, n_{1J}|\pi_{11}, \ldots, \pi_{1J}) = \prod_{j=1}^{J} p(n_{1j}|\pi_{1j}) \tag{22}$$

where the term for the j-th variable is proportional to

$$p(n_{1j}|\pi_{1j}) \propto (\pi_{1j})^{n_{1j}}(1 - \pi_{1j})^{N-n_{1j}}. \tag{23}$$

The beta distribution is the conjugate prior for the binomial model, i.e.,

$$\begin{aligned} p(\pi_{1j}) &= Beta(\pi_{1j}|\alpha_j, \beta_j) \\ &\propto (\pi_{1j})^{\alpha_j - 1}(1 - \pi_{1j})^{\beta_j - 1}. \end{aligned}$$

The hyper-parameters α_j and β_j can be specified in accordance with our prior knowledge (or lack thereof) regarding the distribution of the response probability of variable j, for $j = 1, \ldots, J$. Note that when $\alpha_j = \beta_j = 1$, a uniform prior is obtained for π_{1j}. Multiplying the prior and the likelihood according to (1), yields the posterior for π_{1j} which is given by

$$\begin{aligned} p(\pi_{1j}|n_{1j}) &\propto p(n_{1j}|\pi_{1j}) \times p(\pi_{1j}) \\ &\propto (\pi_{1j})^{n_{1j}+\alpha_j - 1}(1 - \pi_{1j})^{N-n_{1j}+\beta_j - 1} \\ &\propto Beta(\pi_{1j}|n_{1j} + \alpha_j, N - n_{1j} + \beta). \end{aligned}$$

Appendix B

Posterior Distributions for the Regression Model Parameters

We use the standard independence Jeffreys prior $\pi(\boldsymbol{\beta}, \sigma^2) \propto \sigma^{-2}$ throughout our regression analyses (e.g., Kass & Wasserman, 1996). The (conditional) posteriors used in Step 2 of Algorithm 1 are then given by

$$\begin{aligned} \sigma^2|\mathbf{y}, \mathbf{X} &\sim IG(\tfrac{n-J}{2}, \tfrac{s^2_{\mathbf{y}}}{2}) \\ \boldsymbol{\beta}|\sigma^2, \mathbf{y}, \mathbf{X} &\sim N(\hat{\boldsymbol{\beta}}, \sigma^2(\mathbf{X}'\mathbf{X})^{-1}), \end{aligned}$$

where the ML estimate is given by $\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$, the sum of squares equals $s^2_{\mathbf{y}} = (\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}})'(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}})$, and $IG(\alpha, \gamma)$ denotes an inverse gamma distribution with shape parameter α and scale parameter γ. The annotated Julia Bezanson, Edelman, Karpinski, and Shah (2014) code used for the computation of the posterior-cppp can be found in Appendix C.

Appendix C

Julia Code for the Regression Analysis

```
#Installing required Julia libraries
Pkg.add("Distributions")
Pkg.add("Iterators")
#Updating any package if necessary, which may take some time.
# Pkg.update()
#Loading the packages
using Distributions
using Iterators

#This Julia code computes a posterior predictive p-value based on a discrep
#input:
# y = vector of N observations on the dependent variable.
# X = N*J matrix of N observations on J predictor variables. To include an
# K = number of replicate data sets to calulate a ppp.
# M = number of posterior-based ppps to calibrate the ppp with.

#########################################################################
# Algorithm 1: Calculation of a posterior predictive p-value for the Regre
#########################################################################


function   pppREG(y, X, K)

    # Sample size N = number of observations in y.
    N = length(y)
    if N != size(X, 1)
        throw("Number of observations in dependent and independent variable
```

```
end
# Check whether first column is a constant 1. If so, the number of vari
J = ifelse(all(X[:, 1] == 1), size(X, 2) - 1, size(X, 2))
# the precision matrix PreMat (inverse of covariance matrix) of the pre
PreMat = inv(X'X)

##########################################################################
# Algorithm 1, STEP 1: Specify a prior and choose discrepancy
##########################################################################
# Jeffreys' prior was used.
# discrepancy testing for outliers in the observed errors.

function Dmax(Dependent, PredMat, effects, sigma2)
    maximum(abs(Dependent - PredMat * effects))/sqrt(sigma2)
end
# different discrepancies could be specified of course.
# E.g. if one is only interested in the absolute range of the dependent
# function Drange(Dependent)
#     max(y) - min(y)
# end

# Initialisation of the ppp object.
ppp = 0.


##########################################################################
# Algorithm 1, STEP 2: Obtain the posterior distribution
##########################################################################
# Given dependent variable y and the predictor variables in matrix X, v
```

```
## Posterior modes/maximum likelihood estimates for regression effects
bhat = PreMat*X'y
# The hyperparameter for the posterior distribution of the variance par
s2 = (1/(N - J)) * ((y - X * bhat)' * (y - X * bhat))[1]


# Start of the actual PPC.

########################
# Algorithm 1: STEP 3
########################
for k in 1:K

    ##########################################################################
    # Algorithm 1, STEP 3a: Obtain draws from posterior
    ##########################################################################

    # We first take a draw from the marginal posterior of the variance
    # Note. Re-parameterised, the inverse-gamma is equivalent to a inve
    sigma2draw = rand(InverseGamma((N - J)/2, (N - J)s2/2), 1)[1]
    # We calculate the variance in the posterior of b, given the drawn
    varb = PreMat * sigma2draw
    # And then draw values for b from its condtional multivariate norm
    bdraw = rand(MvNormal(bhat, varb))

    ##########################################################################
    # Algorithm 1, STEP 3b: Generate replicated data
    ##########################################################################
    # The model assumes normally distributed error terms, which have as
```

```
        # So we generate those as randomly drawn from a Normal distribution
        erep = rand(Normal(0, sqrt(sigma2draw)), N)
        # Generating replicate data yrep as specified by the regression mo
        yrep = X * bdraw + erep

            ##########################################################################
        # Algorithm 1, STEP 3c: Calculate the realised and replicated discr
        ##########################################################################
        ####
        # Realised
        Dreal = Dmax(y, X, bdraw, sigma2draw)
        # Replicated discrepancy for the replicated errors, which were gene
        Drep = Dmax(yrep, X, bdraw, sigma2draw)

        ##########################################################################
        # Algorithm 1, STEP 4: Compute the ppp
        ##########################################################################
        # Here we increment the ppp each time the value of the Replicated d
        # This provides the same result as taking the sum described in Algo
        # for example, if the replicated Discrepencies are never greater th
        ppp += (Drep > Dreal)/K

    end

    ppp # Returns ppp as output.
end

##############################################################################
```

```
# Algorithm 3: Posterior Calibration of the ppp in the regression example.
# Algorithm 3 Step 1 and Step 2 are already done above, but used again here
#########################################################################


function   cpppREG(y, X, ppp, K, M) #note that Step 1 of the algorithm can
    N = length(y)
    if N != size(X, 1)
        throw("Number of observations in dependent and independent variable
    end
    # Sample size N = number of observations in y.
    N = length(y)
    # Check whether first column is a constant 1. If so, the number of vari
    J = ifelse(all(Xmat[:, 1] == 1), size(Xmat,2)-1, size(Xmat,2))
    # the precision matrix PreMat (inverse of covariance matrix) of the pre
    PreMat = inv(X'X)

    # Given dependent variable y and the predictor variables in matrix X, v
    ## Posterior modes/maximum likelihood estimates for regression effects
    bhat = PreMat*X'y
    # The hyperparameter for the posterior distribution of the variance par
    s2 = (1/(N - J)) * ((y - X * bhat)' * (y - X * bhat))[1]

    # Initialisation of the posterior-cppp object.
    cppp = 0.

    ########################################################################
    # Algorithm 3, STEP 3: Obtain a reference distribution of the ppp
    ########################################################################
```

```
# Next, we generate M posterior-based data sets for which we calculate
for i = 1:M
    ##########################################################################
    # Algorithm 3, STEP 3a: Obtain draws from the posterior
    ##########################################################################
    # Draw value from the marginal posterior for the variance:
    sigma2drawpost = rand(InverseGamma((N - J)/2, (N - J)s2/2), 1)[1]
    # Calculate the variance in the posterior of b, given the drawn val
    varbpost = PreMat * sigma2drawpost
    # And then draw values for b from its condtional multivariate norma
    bdrawpost = rand(MvNormal(bhat, varbpost))

    ##########################################################################
    # Algorithm 3, STEP 3b: Generate posterior-based data sets
    ##########################################################################
    #errors for the posterior-based data.
    epost = rand(Normal(0,sqrt(sigma2drawpost)), N)
    # generating data ypost as specified by the regression model.
    ypost = X*bdrawpost + epost

    ##########################################################################
    # Algorithm 3, STEP 3c: Calculate a ppp for the posterior-based dat
    ##########################################################################
    #Calling PPC. We use K+1 such that the ppp and postppp are never ex
    postppp = pppREG(ypost, X, K + 1)

    ##########################################################################
    # Algorithm 3, STEP 4: Calculate the posterior-cppp.
    ##########################################################################
```

```
        # Here we increment the cppp each time the value of the ppp is grea
        # This provides the same result as taking the sum described in Algo
        # For example, if the ppp is never greater than the postppp, the p
        cppp += (ppp > postppp)/M

    end

    return [ppp cppp] #returns a 1x2 matrix with elements the ppp and cppp.
end

######## Running a dummy example #############
# Set the number of replications used to calculate a ppp.
K = 500
# Set number of posterior-based ppps to calibrate the ppp with.
M = 500

# generating a dummy matrix of N standard normal i.i.d. observations on exp
J = 3
N = 200
Xvars = randn(N, J)
#adding constant vector so that an intercept will be included in the analys
Xmat = [ones(N) Xvars]
#generating a dummy dependent variable of N standard normal  i.i.d. observa
yobs = randn(N)

#calling the PPC function to calculate a ppp for the given discrepancy (see
#Note. With real data, there is no need to specify N or J
ppp = pppREG(yobs, Xmat, K)
```

```
#calling the calibration function to calibrate the ppp. This outputs both t
cppp = cpppREG(yobs, Xmat, ppp, K, M)

#We can, of course, for any particular ppp calculate its calibrated value.
cppp = cpppREG(yobs, Xmat, .361, K, M)
```

Appendix D

Latent Class Analysis Technical Details

For a data set with sample size N, suppose there are C latent classes. Let ν_c be the (unobserved) class size of the latent class c and let n_{1jc} and $\nu_c - n_{1jc}$ be the observed number of persons with scores 1 and 2 on the j-th variable within class c. Then let ρ_c be the class proportions, and let π_{1jc} and $1 - \pi_{1jc}$ be the corresponding conditional response probabilities, for variables $j = 1, \ldots, J$ and classes $c = 1, \ldots, C$. Thus, the vector of model parameters $\boldsymbol{\theta}$ consists of the class proportions $(\rho_1, \ldots, \rho_C)$ and the unknown conditional response probabilities $(\pi_{11c}, \ldots, \pi_{1Jc})$. Under the main latent class model assumption that the J variables are *conditionally* independent, the likelihood is obtained as a weighted sum of the product of J conditionally independent binomial distributions, i.e.,

$$p(n_{11c}, \ldots, n_{1Jc}|\pi_{11c}, \ldots, \pi_{1Jc}) = \sum_{c=1}^{C} \rho_c \prod_{j=1}^{J} p(n_{1jc}|\pi_{1jc}) \tag{24}$$

where the term for the j-th variable is proportional to

$$p(n_{1jc}|\pi_{1jc}) \propto (\pi_{1jc})^{n_{1jc}}(1 - \pi_{1jc})^{\nu_c - n_{1jc}}. \tag{25}$$

Again, it is standard practice to use the conjugate $Beta(\pi_{1jc}|\alpha_j c, \beta_j c)$ priors for the conditional response probabilities. Since the number of classes can be greater than two, we assume a multinomial (rather than a binomial) likelihood for the class proportions. The conjugate prior for the multinomial likelihood is the multivariate generalisation of the Beta distribution given as $Dirichlet(\rho_c|\alpha_1, \ldots, \alpha_C)$.

By combining the likelihood and the prior using (1), the posterior distribution for π_{1jc} used in Step 2 of Algorithm 1 has a beta distribution given by $p(\pi_{1jc}|y_j) = Beta(\pi_{1jc}|n_{1jc} + \alpha_{jc} - 1, \nu_c - n_{1jc} + \beta_{jc} - 1)$ and the posterior for the vector of class proportions is given by $p(\rho_1, \ldots \rho_C|\mathbf{y}) = \text{Dirichlet}(\rho_1, \ldots, \rho_C|\nu_1 + \alpha_1 - 1, \ldots, \nu_C + \beta_C - 1)$. These posteriors assume that it is known which observations belong to which class, in order to obtain the

observed frequencies ν_c and n_{rjc}. In order to do this, the data needs to be augmented with starting values for the latent class memberships Tanner and Wong (1987), after which a Gibbs sampler is run to iteratively draw values from the posteriors and then updating the latent class memberships. When the Gibbs sampler has converged (usually after thousands of iterations), values for the parameters are obtained by retaining every 50th, or so, draw.

Calculating the bivariate residual.

Cross-tabulating two dichotomous variables, results in a contingency table with $2^2 = 4$ cells. Each cell corresponds to one of the four response patterns, denoted by $\mathbf{y}_s$, for $s = 1, \ldots, 4.$ with pattern probability denoted by π_s and given by

$$\pi_s = \sum_{c=1}^{C} \rho_c (\pi_{1jc})^{d_{js}} (1 - \pi_{1jc})^{1-d_{js}} \times (\pi_{1j'c})^{d_{j's}} (1 - \pi_{1j'c})^{1-d_{j's}}, \tag{26}$$

where the dummy indicator d_{js} and $d_{j's}$ equal 1 if the response to variable j and j' in pattern s are 1, and 0 otherwise. For example the response probability for the pattern (1,1) equals $\sum_{c=1}^{C} \rho_c \pi_{11c} \pi_{12c}$. For example, the expectations used to calculate in the D_{BVR} for pattern (1,1), given the model parameters at iteration k would equals $e_s^{(k)} = N \sum_{c=1}^{C} \rho_c^{(k)} \pi_{11c}^{(k)} \pi_{12c}^{(k)}$. Annotated Julia code for calculating a posterior-cppp for the BVRs can be found in Appendix E.

Calculating the Pearson χ^2 and the likelihood ratio.

Cross-tabulating all J dichotomous variables, results in a contingency table with $S = 2^J$ cells. Each cell corresponds to one of the possible response patterns (which now comprise J responses), again denoted by $\mathbf{y}_s$, for $s = 1, \ldots, S$. The pattern probability is denoted by π_s and given by

$$\pi_s = \sum_{c=1}^{C} \rho_c \prod_{j=1}^{J} (\pi_{1jc})^{d_{js}} (1 - \pi_{1jc})^{1-d_{js}}, \tag{27}$$

where the dummy indicator d_{js} equals 1 if the response to variable j in pattern s is 1, and 0 otherwise. For example the response probability for only ones on all J variables equals $\sum_{c=1}^{C} \rho_c \pi_{11c} \times \ldots \times \pi_{1Jc}$ (which may differ across iterations of the Gibbs sampler).

Appendix E

Julia Code for the Bivariate Residuals in Latent Class Analysis

```
#Installing required Julia libraries
Pkg.add("Distributions")
Pkg.add("Iterators")
Pkg.add("DataFrames")
#Updating any package if necessary
# Pkg.update()
#Loading the packages
using Distributions
using Iterators
using DataFrames


#This Julia code computes a posterior predictive p-value for a testing prob
#based on a discrepancy which test the remaining associations between pairs
#input:
# FreqObs = vector with lenght equal to the number of possible response pat
# J = the number of dichotomous variables.
# C = number of classes to use in the latent class analysis.
# AllPatterns = a 2^J x J matrix with all possible response patterns on a r
# K = number of replicate data sets to calulate a ppp.
# M = number of posterior-based ppps to calibrate the ppp with.
#
# We used an ad hoc C program for our latent class model Gibbs sampler,
# but other tools that yield posterior draws for latent class models may al
# The code leading up to Step 3b in Algorithm 1 and Algorithm 3 will have t
# When using this code, make sure that the MCMC sampler is in the working d
# to initialise the function pppBVR, select all code belonging to it and ru
```

```
##########################################################################
# Algorithm 1: Calculation of a posterior predictive p-value
# for the bivariate residuals in a latent class analysis
##########################################################################

function pppBVR(DataPPC, J, C, K)
    #PPC for LCA using BVRs as Discrepancy

    N=sum(DataPPC)
    NPat = length(DataPPC) #number of patterns
    GibbsDataPPC = [AllPatterns DataPPC]  #add column DataObs to file the r

    open("DataPPC.dat", "w") do fd
        writedlm(fd, GibbsDataPPC, ' ')
    end


    ##########################################################################
    # Algorithm 1, STEP 1: Obtain a posterior distribution
    # We used a Gibbs sampler program, written in C called LCAgibbsCount
    # Uniform priors were specified in the line below as "pri 1 1"
    # As discrepancy, we chose the BVR (which is calculated in-line later)
    ##########################################################################

    open("gibbsPPC.inp", "w") do inp
        write(inp, [string("tab ", 2^J, " ", J, "\n"),
                string("cla ", C, "\n"),
                string("ite ", 1000, " ", 10*K," ", 10, "\n"),
                string("rco ", "\n"),
```

```
            string("dat DataPPC.dat \n"),
            string("pri ", 1, " ", 1, "\n")])
end

#initialising the ppp object for the 10 BVR-based discrepancies
ppp = rep(0., int(J*(J-1)/2))
##########################################################################
# Algorithm 1, STEP 2: Obtain the posterior distribution
##########################################################################
# the following calls the Gibbs sampler program called LCAgibbsCount, w
# the .inp file written above as input
# the file to which it writes the aggregated Latent Class model results
# a file in which individual draws for the parameter sets are stored.

Foo = readall(`LCAgibbsCount "gibbsPPC.inp" "gibbsoutPPC.txt" "paramsPP

# We read in the file with the draws from the posterior
ParaMatPPC = array(readtable("paramsPPC.txt", header=false, separator =

########################
# Algorithm 1: STEP 3
########################
for i in 1:K
        ##########################################################################
    # Algorithm 1, STEP 3a: Obtain draws from posterior
    ##########################################################################
    # One row in ParaMatPPC corresponds to a single draw for the param
    ParamsPPC = ParaMatPPC[i, :]
```

```
# the first C values of the vector provide the draw for the class s
PClassPPC = ParamsPPC[1:C]

# The rest of the vector gives the pi_rjc values in the ordering [p
#Creating a 'Profile matrix' in which each column has the condition
PRespPPC = Array(Float64, 2J, C)
if C == 1
    PRespPPC = ParamsPPC[2:(2*J+1)]
else
    for a in 1:C
        PRespPPC[:,a] = ParamsPPC[C+sort([(1+2*(a-1)):(2*C):(4J-1)
    end
end

# We generate replicate data (pattern frequencies) by a draw from a
# To do that we creating a matrix for selecting the correct probab
    MatSelect = Array(Int64, NPat, J)
for f in 1:J
    MatSelect[:,f] = AllPatterns[:,f] + (f-1)*2
end

# Then we calculate the conditional pattern probabilities
PPatternCondPPC = Array(Float64, NPat, C)
if C==1
    PPatternPPC = Array(Float64, NPat)
    for p in 1:NPat
        PPatternPPC[p] = prod(PRespPPC[MatSelect[p,:][1:J],:])
    end
else
```

```
    for cc in 1:C
        for p in 1:NPat
            PPatternCondPPC[p,cc] = prod(PRespPPC[MatSelect[p,:][1:
        end
        PPatternCondPPC[:,cc] = PPatternCondPPC[:,cc] / sum(PPatter
    end
# And finally calculate the (unconditional) pattern probabilities g
PPatternPPC = PPatternCondPPC * PClassPPC #cf. Equation 27
end

##########################################################################
# Algorithm 1, STEP 3b: Generate replicated data
##########################################################################
# generating replicated data
DataRep = rand(Multinomial(N, PPatternPPC))

    ##########################################################################
# Algorithm 1, STEP 3c: Calculate the discrepancies
##########################################################################
# We make a matrix with in the first columns the realised and in th
BVRTable = Array(Float64, int(J * (J - 1)/2), 2)
# We make a matrix with all the combination of variable
AllPairs = Array(Int64, 0, 2)
for y in 1:J
    for x in (y+1):J
        if y!=x
            AllPairs = [AllPairs, [y x]]
        end
    end
```

```
        end
        # Then for each pair of variables, we determine the observed freque

        for ii in int(1:(J * (J - 1)/2))
            itemA = AllPairs[ii,1]
            itemB = AllPairs[ii,2]
            if(C == 1) # PRespPPC is a vector
                probsA = PRespPPC[(itemA*2-(2-1)):(itemA*2)]
                probsB = PRespPPC[(itemB*2-(2-1)):(itemB*2)]
            else       # PRespPPC is a Matrix
                probsA = PRespPPC[(itemA*2-(2-1)):(itemA*2),:]
                probsB = PRespPPC[(itemB*2-(2-1)):(itemB*2),:]
            end

            TabObs = Array(Float64, 2, 2)
            TabPred = Array(Float64, 2, 2)
            TabRep = Array(Float64, 2, 2)
            for a in 1:2
                for b in 1:2
                    # For each pattern we calculate the observed frequencie
                            # first for the observed data
                            TabObs[a,b] = sum(DataPPC[(AllPatterns[:,itemA]
& (AllPatterns[:,itemB].==b)])
                    # then for the replicate data
                    TabRep[a,b] = sum(DataRep[(AllPatterns[:,itemA].==a) &
                    # Calculate the expectations under the model parameters
                            if C == 1
                        TabPred[a,b] = N* (probsA[a]*probsB[b]')
                    end
```

```
                    if C == 2
                        TabPred[a,b] = N* (probsA[a,1]*probsB[b,1] * PClass
                    end
                    if C == 3
                        TabPred[a,b] = N* (probsA[a,1]*probsB[b,1] * PClass
                    end
                end
            end
                    #Calculation of the BVR (as a X2 statistic for a 2x2 ta
            BVRReal = sum((TabObs-TabPred).^2./(TabPred))
            BVRRep = sum((TabRep-TabPred).^2./(TabPred))
            BVRTable[ii,:] = [BVRReal BVRRep]
        end
        ##########################################################################
        # Algorithm 1, STEP 4: Compute the ppp
        ##########################################################################
            # this line increments the corresponding ppp every time the rep
        ppp .+= (BVRTable[:, 1] .< BVRTable[:, 2]) ./ K
    end
return(ppp)
end


##########################################################################
# Algorithm 3: Posterior Calibration of the ppp in the regression example.
# Algorithm 3 Step 1 and Step 2 are already done above, but used again here
##########################################################################

function cpppBVR(DataObs, ppp,  J , C, K, M)
```

```
N=sum(DataObs)
open("gibbsObs.inp", "w") do inp
    write(inp, [string("tab ", 2^J, " ",  J, "\n"),
        string("cla ", C, "\n"),
        string("ite ", 1000, " ", 10*K," ", 10, "\n"),
        string("rco ", "\n"),
        string("dat DataObs.dat \n"),
        string("pri ", 1, " ", 1, "\n")])
end

NPat = length(DataObs)
DataRecords = [AllPatterns DataObs]  #add column DataObs to file "AllPa
writedlm("DataObs.dat", DataRecords, header = false)

foo = readall(`LCAgibbsCount "gibbsObs.inp" "gibbsoutObs.txt" "paramsOb

ParaMat = array(readtable("paramsObs.txt", header=false, separator = '

cppp = rep(0., int(J*(J-1)/2)) #initialising the vector of posterior-cp
##########################################################################
# Algorithm 3, STEP 3: Obtain a reference distribution of the ppp
##########################################################################
for nn = 1:M
    ##########################################################################
# Algorithm 3, STEP 3a: Obtain draws from the posterior
##########################################################################
    # One row in ParamsPost corresponds to a single draw for the param
    ParamsPost = ParaMat[nn, :]
```

```
    # the first C values of the vector provide the draw for the class s
    PClassPost = ParamsPost[1:C]

    # The rest of the vector gives the pi_rjc values in the ordering [p
    #Creating a 'Profile matrix' in which each column has the condition
if C == 1
    PRespPost = ParamsPost[2:(2*J+1)]
else
    PRespPost = Array(Float64, 2J, C)
    for a in 1:C
        PRespPost[:,a] = ParamsPost[C+sort([(1+2*(a-1)):(2*C):(4J-1),2-
        end
end


    # matrix for selecting correct Probabilities
    MatSelect = Array(Int64, NPat, J)
    for f = 1:J
        MatSelect[:,f] = AllPatterns[:,f] + (f-1)*2 #Need to implement
    end

    #Calculate conditional pattern probabilities
    PPatternCondPost = Array(Float64, NPat, C)
    if C==1
        PPatternPost = Array(Float64, NPat)
        for p in 1:NPat
            PPatternPost[p] = prod(PRespPost[MatSelect[p,:][1:J],:])
        end
```

```
else
    for cc in 1:C
        for p in 1:NPat
            PPatternCondPost[p,cc] = prod(PRespPost[MatSelect[p,:][
        end
        PPatternCondPost[:,cc] = PPatternCondPost[:,cc] / sum(PPat
    end
    #Calculate Pattern probabilities given Parameter values.
    PPatternPost = PPatternCondPost * PClassPost
end

    ##########################################################################
# Algorithm 3, STEP 3b: Generate posterior-based data sets
##########################################################################
DataPost = rand(Multinomial(N, PPatternPost))

    ##########################################################################
# Algorithm 3, STEP 3c: Calculate a ppp for the posterior-based dat
##########################################################################
    #Calling PPC. We use K+1 such that the ppp and postppp are neve
postppp = pppBVR(DataPost, J, C, K+1)

##########################################################################
# Algorithm 3, STEP 4: Calculate the posterior-cppp.
##########################################################################
# Here we increment the cppp each time the value of the ppp is grea
# This provides the same result as taking the sum described in Algo
# For example, if the ppp is never greater than the postppp, the p
cppp .+= (ppp .> postppp) ./ M
```

```
    end

  return [ppp cppp] # returns a 10x2 matrix, with in the first column the p
end




# Running dummy example
# data set generated from a 2-class population, with conditional probabilti
FreqObs = [19, 5, 4, 2, 4, 3, 3, 8, 5, 2,  3, 6, 3, 6, 6, 21]

C = 1 # or C = 2
J = 4
K = 100
M = 100

#required matrix with all patterns
AllPatterns =
    [[1 1 1 1],
     [1 1 1 2],
     [1 1 2 1],
     [1 1 2 2],
     [1 2 1 1],
     [1 2 1 2],
     [1 2 2 1],
     [1 2 2 2],
     [2 1 1 1],
     [2 1 1 2],
```

```
      [2  1  2  1],
      [2  1  2  2],
      [2  2  1  1],
      [2  2  1  2],
      [2  2  2  1],
      [2  2  2  2]]

cd("D:\\workingDir\\") #this is where the .exe of the MCMC sampler should b

pppObs = pppBVR(FreqObs, J, C, K)
postcppp = cpppBVR(FreqObs, ppp, J, C, K, M)
```