Bayesian Applications in A Probability Programming Language

Matthew Robinson
Department of Mathematical Sciences
United States Military Academy
West Point, New York 10996
Matthew.Robinson@usma.edu

January 24, 2016

ABSTRACT

Over the past decade, interest in Bayesian statistics has increased rapidly. In consequence of this growth, a variety of software packages have been developed in order to facilitate applied statistical analysis. These software packages, however, use numerical techniques implemented in languages such as R, WinBUGS and Stan to produce discrete approximations of posterior distributions. Currently, no program automates the derivation of exact closed form posterior distributions. These derivations must either be accomplished by hand, which is often intractable, or approximated through simulation.

In this paper, we present software capable of automating the derivation of exact posterior distributions. Specifically, we extend A Probability Programming Language (APPL) to include a number of procedures that integrate Bayesian methods into APPL. APPL is a Maple based software package capable of manipulating random variables, creating new distributions, and exploring new probabilistic models. The Bayesian procedures created here allow users to conduct inference on single and multi-parameter distributions, derive posterior predictive distributions and derive Jeffrey's priors. By automating the application of Bayes' rule, APPL both serves as a time saving device and a tool for theoretical exploration.

1 Introduction

Over the past several decades, both theoretical and applied interest in Bayesian statistics has grown dramatically. Although a variety of factors have contributed to this trend, a significant factor is that computer implementations of Bayesian statistics have made it possible for practitioners of statistics to apply Bayesian methods in more complex settings. The growing popularity of Bayesian statistics has led to a proliferation of software packages capable of performing applied statistical analysis within the Bayesian framework.

Bayesian computer applications, however, have largely remained confined to numerical tools written in languages, such as R, Stan and WinBUGS. While ideal for performing one-time computations within the context of applied statistical analysis, these software packages lack the symbolic manipulation capabilities present in computer algebra systems, such as Maple and Mathematica. Due to underutilization of computer algebra systems, a gap exists within the field of computational Bayesian statistics. Specifically, no software package currently has the ability to automate Bayesian inference on random variables, with the goal of producing the entire probability density function as its output, especially in the case of multi-parameter distributions.

This paper presents a remedy for this problem in the form of software that produces posterior PDFs in closed form. In particular, this paper presents APPL Bayes, a Bayesian extension to "A Probability Programming Language" (APPL), a Maple-based conceptual probability software package. Working in conjunction with existing APPL procedures, APPL Bayes automates the application of Bayes' Theorem to random variables. At a practical level, these automated manipulations serve two primary purposes. First, they save time by eliminating the need to undergo mathematically tedious derivations. This both saves students time, and allows them to check their hand-derived results against the results produced by APPL Bayes. Second, APPL Bayes provides a tool for experimentation by allowing students to solve time-consuming problems rather quickly. Moreover, it eliminates the need to rely on conjugate prior relationships. Even within the context of an introductory course, APPL Bayes gives students the ability to experiment with ad-hoc prior distributions, thus allowing the student to develop a deeper appreciation of theoretical concepts.

The procedures in APPL Bayes fall broadly into two categories: Bayesian inference on single and multi-parameter models. This paper will treat each of these categories in turn, and then present

some avenues for further theoretical exploration that APPL Bayes has already opened.

Currently, the literature contains only a few attempts to apply computational methods in order to derive closed form posterior PDFs. Cook and Broehmling (1995) present several Bayesian applications using Mathematica. Their code, however, is not reusable. They make no attempt to automate the ad-hoc Bayesian procedures that they present in their paper. APPL Bayes, on the other hand, consists of automated procedures that can be readily applied to complex models without abstruse coding. Jaakkola and Jordan (2000), on the other hand, present reusable code capable of producing closed form posterior PDFs for logistic regression. Their algorithm, however, depends on producing normal approximations to the likelihood function. As such, they remain limited to the use of normal prior distributions. APPL Bayes allows for greater flexible that Jaakkola and Jordan's approximations because it produces exact closed form PDFs without remaining constrained to conjugate prior relationships. Given these improvements over current systems, APPL Bayes represents an advance in the field of computational Bayesian statistics.

2 Bayesian Inference on Single Parameter Distributions

Bayesian inference on single parameter distributions represents the simplest class of problems within Bayesian statistics. Due to their relative simplicity, introductory Bayesian courses typically use problems of inference on one variable as a means to motivate the theoretical basis for Bayesian statistics. A typical problem that would appear in a Bayesian statistic textbook would generally take the following form: given a likelihood function, $f(x|\theta)$, a prior distribution, $g(\theta)$ and a data set, derive the posterior distribution, $g(\theta|\vec{x})$. APPL Bayes has the capability of automating the solution to these simple problems. Consider the following example.

Example 2.1 Suppose we have conducted a series of experiments, each consisting of twelve trials, which have a binary outcome where 1 represents a successful run and 0 represents an unsuccessful run. Further suppose that the likelihood function takes the form of a Binomial$(12, \theta)$ random variable. Assume that we have placed a Beta$(\frac{1}{2}, \frac{1}{2})$ prior on the parameter θ and have observed a data vector $\vec{x} = [5, 6, 3, 2, 5]$. Given this information, derive the posterior distribution of the parameter θ. APPL Bayes can accomplish this derivation using the following commands:

```
X:=BinomialRV(12,theta);
```

```
Y:=BetaRV(1/2,1/2);

data:=[5,6,3,2,5];

P:=Posterior(X,T,data,theta);
```

This APPL Bayes code produces a Beta$(39, 21)$ random variable as output. Notice that the posterior distribution in this case is in the same class of distributions as the prior distribution. Bayesians call this relationship between the prior and posterior distribution a *conjugate prior* relationship. Problems that use conjugate prior relationships abound in introductory textbooks because of their mathematical simplicity. To understand why, we must simply glance at Bayes' Theorem, which appears as follows:

$$g(\theta|x) = \frac{f(x|\theta) \cdot g(\theta)}{\int_{-\infty}^{\infty} f(x|\theta) \cdot g(\theta)\ d\theta}$$

Notice that the numerator requires little more than simple multiplication. Evaluating the integral in the denominator, however, can become rather complicated. Using conjugate prior relationships eases the burden of deriving posterior distributions because the posterior distribution falls in the same class of distributions as the prior. As such, the normalizing constant for the posterior distribution can be easily found through pattern matching. In this example, for instance, we recognize the normalizing constant of a Beta$(39, 21)$ random variable as the number $\frac{\Gamma(39+21)}{\Gamma(39)\cdot\Gamma(21)}$. Thus, introductory Bayesian statistics students can derive a posterior distribution without ever evaluating a complex integral.

That same student, however, may come to the realization that the choice of a Beta distribution as a prior for the Binomial distribution is rather arbitrary. Any class of distribution, in fact, could classify our prior beliefs about the parameter. This disillusionment with an arbitrary choice of prior distributions may even lead the student to prefer classical methods, without ever giving Bayesian methods a fair chance. APPL Bayes helps by eliminating the need to rely on conjugate priors within a classroom setting. APPL Bayes, in fact, makes deriving posterior distributions with ad-hoc priors as simple as deriving posteriors with conjugate priors. Consider Example 2.1. Instead of a Beta$(\frac{1}{2}, \frac{1}{2})$ prior, suppose that we place a Triangular$(0, \frac{1}{2}, 1)$ prior on the parameter, θ. Such a problem would never appear in an introductory Bayesian text. Not only does the posterior not fall into a known conjugate prior relationship, but obtaining the normalizing constant also requires integration over a piecewise function. Nevertheless, APPL Bayes can solve this problem with the

same amount of effort as in the conjugate prior case. The following APPL Bayes commands can be used to derive the posterior:

```
X:=BinomialRV(12,theta);

Y:=TriangularRV(0,1/2,1);

data:=[5,6,3,2,5];

P:=Posterior(X,T,data,theta);
```

The resulting posterior distribution is a piece-wise distribution, which appears as follows:

$$g(\theta|\vec{x}) = \begin{cases} \frac{-36900864}{1363} \cdot \theta^6 \cdot (\theta - 1)^7 & \text{for } 0 \leq \theta \leq \frac{1}{2} \\ \frac{-36900864}{1363} \cdot \theta^5 \cdot (\theta - 1)^8 & \text{for } \frac{1}{2} \leq \theta \leq 1 \end{cases}$$

The APPL Bayes software thereby allows students to experiment with a wider range of possibilities within Bayesian statistics than is currently possible.

Along with the derivation of posterior distributions, additional functionalities provided by APPL Bayes allow students to solve follow on problems related to the posterior distribution. Consider our modification to Example 2.1 presented above. Given a data set, a Binomial likelihood function and a triangular prior, we derived a posterior distribution describing our updated belief about the parameter θ. Another typical problem in a Bayesian statistics textbook may ask a student to conduct inference on the parameter, given this posterior distribution. For instance, suppose we wish to test the hypothese $H_0 : \theta \geq \frac{1}{2}$. The APPL command `SF(X,x)`, which automatically computes the survivor function of a random variable, allows us to calculate the probability that θ is greater than $\frac{1}{2}$. APPL can likewise compute the left-tail probability using an analogous command, `CDF(X,x)`. The APPL Bayes code that accomplishes this appears as follows:

```
P:=Posterior(X,T,data,theta);

prob:=evalf(SF(P,1/2));
```

The result of this code shows that $P(\theta > \frac{1}{2}|\vec{x}) = .013775$, which allows the student to reject or fail to reject the null hypothesis, depending on the confidence level. Similarly, a student can perform a two-tailed hypothesis test with similar ease using APPL's `CredibleSet(X,alpha)` command, which constructs a credible set with confidence level α. Assume we want to test the null hypothesis $H_0 : \theta \neq \frac{1}{2}$. APPL Bayes can construct the credible set using the following code:

```
P:=Posterior(X,T,data,theta);

CS:=CredibleSet(P,.05);
```

The APPL Bayes output produces a credible set $[0.2517, .03149]$, which allows the student to reject the null hypothesis at the 5 percent confidence level. These examples show how APPL Bayes can be used to conduct inference on a parameter using an ad-hoc prior, using only a few simple lines of code. In so doing, APPL Bayes allows students to solve a much broader range of problems than was previously possible.

Another typical problem presented to introductory students in Bayesian statistics involves the derivation of posterior predictive distributions, which shows the distribution of the $(n+1)^{\text{th}}$ observation, given a vector of n observations. Again, the problems within grasp of introductory students remain confined to distributions that demonstrate a conjugate prior relationship. In order to see why, consider the following example:

Example 2.2 Suppose we are observing a process that is described by a Poisson likelihood and have gathered one observation, $x = 5$. Assume we have placed a Gamma$(2, 5)$ prior distribution on the parameter, θ. Derive the posterior predictive distribution of y, the next observation. APPL Bayes can accomplish this derivation using the following commands:

```
X:=PoissonRV(theta);

Y:=GammaRV(2,5);

data:=[5];

PP:=PosteriorPredictive(X,Y,data,theta);
```

The resulting APPL Bayes output shows that the posterior predictive distribution is described by the function:

$$f(y|x) = \frac{729}{4697620490} 4^{-y}(y+1)(y+2)...(y+9) \text{ for } y > 0$$

In the case of posterior predictive distributions, it is less clear that this solution relies on the conjugate prior relationship. In order to clarify the nature of this dependence, it is useful to derive the posterior predictive distribution. In this case, the posterior predictive distribution can

be derived as follows from the definition of the posterior predictive, which states that $f(y|x) = \int_{-\infty}^{\infty} g(\theta|x) \cdot f(y|\theta)\ d\theta$. We start with the likelihood function and the prior distribution:

$$\begin{aligned} f(x=5|\theta) &= \frac{e^{\theta}\theta^5}{5!} \text{ for } \theta > 0 \\ g(\theta) &= \frac{2^5}{\Gamma(5)}\theta^4 e^{-2\theta} \end{aligned}$$

We then derive the posterior:

$$\begin{aligned} f(x=5|\theta)g(\theta) &= \left(\frac{e^{\theta}\theta^5}{5!} \cdot \frac{2^5}{\Gamma(5)}\theta^{5-1}e^{-2\theta}\right) \\ &\propto e^{-3\theta}\theta^{(5+5-1)} \\ g(\theta|x=5) &= \frac{f(x=5|\theta)g(\theta)}{\int_0^{\infty} f(x=5|\theta)g(\theta)d\theta} \\ &= \frac{3^{10}}{\Gamma(10)} \cdot e^{-3\theta}\theta^9 \end{aligned}$$

From here we can determine the posterior predictive distribution:

$$\begin{aligned} f(y|x=5) &= \int_0^{\infty} g(\theta|x=5)f(y|\theta)d\theta \\ &= \int_0^{\infty} \frac{3^{10}}{\Gamma(10)}e^{-3\theta}\theta^9 \frac{e^{-\theta}\theta^y}{y!} \\ &= \frac{3^{10}}{\Gamma(10)y!}\int_0^{\infty} e^{-3\theta}\theta^9 \cdot e^{-\theta}\theta^y \\ &= \frac{3^{10}}{\Gamma(10)y!}\int_0^{\infty} e^{-4\theta}\theta^{9+y} \\ &= \frac{3^{10}}{\Gamma(10)y!} \cdot \frac{\Gamma(10+y)}{4^{10+y}}\int_0^{\infty} \frac{4^{10+y}}{\Gamma(10+y)}e^{-4\theta}\theta^{9+y} \\ f(y|x) &= \frac{3^{10}}{\Gamma(10)y!} \cdot \frac{\Gamma(10+y)}{4^{10+y}} \text{ for } y > 0 \end{aligned}$$

Note that, although the posterior predictive distribution derived above takes a different form than the posterior predictive derived by APPL Bayes, they are in fact the same distribution. A simple check in Maple shows that $\frac{3^{10}}{\Gamma(10)\cdot 4^{10}} = \frac{729}{4697620490}$, the constant produced by APPL Bayes, while the identity $\Gamma(10+y) = (y+1)(y+2)...(y+9)$ proceeds directly from the definition of the Gamma function. Notice that, unlike the posterior distribution, the resulting posterior predictive distribution does not fall in the same class of distributions as the prior. However, the conjugate prior relationship also made the derivation of the posterior predictive easier by eliminating the integral $\int_0^{\infty} e^{-4\theta}\theta^{9+y}$. Since this takes the form of the Gamma distribution, we can avoid evaluating the integral by multiplying the inside of the integral by $\frac{4^{10+y}}{\Gamma(10+y)}$, the constant that normalizes the

Gamma distribution, and the outside of the integral by its inverse. By doing this, the integral simply evaluates to one and the function remains unchanged, because multiplying the function by $\frac{4^{10+y}}{\Gamma(10+y)} \cdot \frac{\Gamma(10+y)}{4^{10+y}}$ is equivalent to multiplying the function by one. As with posterior distributions, introductory Bayesian statistics textbook remain confined to addressing the derivation of posterior predictive distributions where conjugate prior relationships holds. Again, with APPL Bayes we can eliminate the need to use conjugate prior relationships with no additional effort. Considering Example 2.2, assume that, instead of a Gamma prior, we instead wish to place a Normal$(1/2, 1/2)$ prior distribution on the parameter, θ. APPL Bayes can accomplish this derivation using the following code:

```
X:=PoissonRV(theta);

N:=NormalRV(1/2,1/2);

data:=[5];

PP:=PosteriorPredictive(X,N,data,theta);
```

This APPL code shows that the posterior predictive distribution in this case is:

$$f(y|x=5) = \frac{-16}{281} \frac{e^{-\frac{1}{8}} \cdot (4+y) \cdot (y+2) \cdot \sqrt{2} \cdot (-1^y - 1) \cdot 2^{-\frac{3}{2}y}}{\Gamma(\frac{1}{2} \cdot y + \frac{1}{2})} \text{ for } y > 0$$

Deriving this distribution by hand could only be accomplish very tediously, due to the lack of a conjugate prior relationship. Nevertheless APPL Bayes has the ability to produce a closed form posterior predictive distribution, even in this relatively complex case, with no more effort than it requires for the posterior predictive distribution in the simple case using conjugate priors. This presents an opportunity for fruitful theoretical exploration. Specifically, posterior predictive distributions using ad-hoc distributions can be compared to their conjugate prior counterparts in order to determine which makes better predictions regarding the next observation in the data set. If the ad-hoc prior performs better, then it makes little sense to remain constrained by the use of conjugate prior relationships.

In the case of single parameter distributions, APPL Bayes also has a function that can automatically derive Jeffrey's priors, which are uninformed priors that are invariate under reparameterizations of the distribution. For distributions with one parameter, Jeffrey's priors can be found using the following equation:

$$g(\theta) = \sqrt{E\left[\frac{d\ ln(f(x|\theta))}{d\theta}\right]}$$

This function is useful because it allows users to check posterior distributions produced using informed priors against a result using a non-informed prior. APPL Bayes has the ability to produce Jeffrey's priors both for likelihood functions with well known Jeffrey's priors, as well as ad-hoc likelihood functions. An an example, consider a Poisson likelihood function. The Jeffrey's prior for this likelihood function can be found by using the following APPL Bayes commands:

```
X:=PoissonRV(lambda);

J:=Jeffreys(X,0,infinity,lambda);
```

The output of these APPL Bayes commands shows that the Jeffrey's prior for the Poisson distribution is $g(\lambda) = \frac{1}{\sqrt{\lambda}}$ for $\lambda > 0$. This matches the Jeffrey's prior for the Poisson Distribution, which is a well known result.

Using these Bayesian procedures, APPL Bayes provides greater flexibility in using ad-hoc prior distributions for inference on a single parameter. This has the potential not only to aid students in the exploration of Bayesian concepts, but also to open the door to theoretical experimentation.

3 Theoretical Explorations Using APPL Bayes

So far, it may appear as though APPL Bayes is only useful in solving simple, introductory level problems in the single parameter case. However, the computer algebra capabilities inherent in APPL Bayes have already facilitated important theoretical exploration.

APPL Bayes, for instance, can be used to compare the effect of using different prior distributions. Using previous software, such comparisons would have required an extraordinary amount of coding. The reusable APPL Bayes code makes such comparisons simple to implement. For example, suppose we have observed a data vector $\vec{x} = [2, 3, 2, 5, 6]$. We posit that this data has been drawn from a Normal$(\mu, 2)$ distribution, and would like to estimate the unknown parameter μ. Under the typical Bayesian paradigm, the choice of prior distributions would be limited to either conjugate or uninformed priors. Here we will evaluate the effect of using two ad-hoc, but nevertheless reasonable prior distributions, the LaPlace and triangular distributions. Of these two distributions, the triangular is particularly interesting because it provides a compromise between an uninformed and a well informed prior. In spite of its utility, the use of the triangular distribution as a prior is notably absent from the literature.

We can set up our experiment using the following APPL Bayes commands for each prior distribution:

```
X:=NormalRV(mu,2);

data:=[2,3,2,5,6];

Y1:=NormalRV(2,2);

B1:=BayesUpdate(X,Y1,data,mu);
```

These commands produce the following posterior distributions as output:

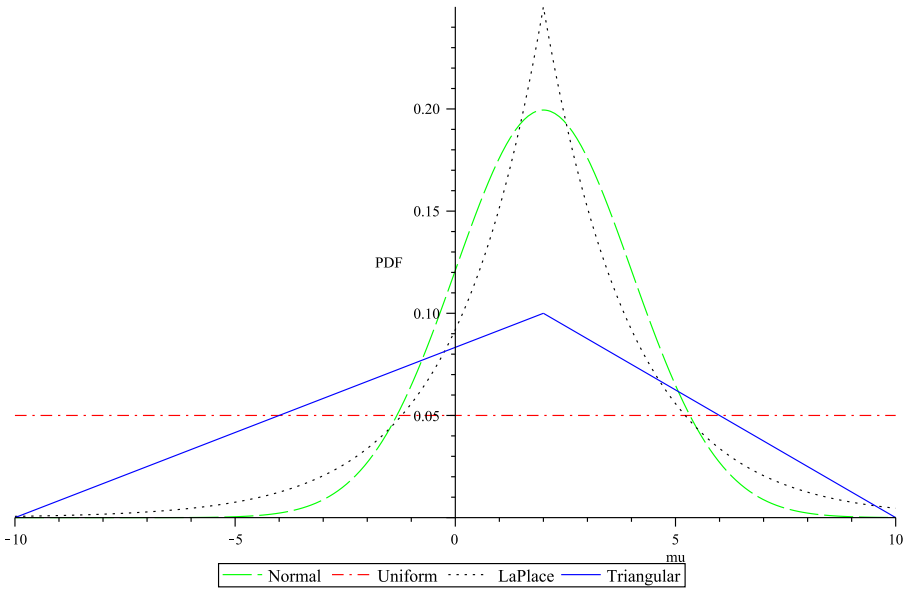
$$\text{Normal:} \quad g(\mu|\vec{x}) = \frac{1}{2}\frac{e^{-\frac{3}{4}\cdot\mu^2+5\cdot\mu-\frac{47}{5}}\sqrt{3}}{\sqrt{\pi}\cdot e^{-\frac{16}{15}}} \text{ for all } \mu$$

$$\text{Uniform:} \quad g(\mu|\vec{x}) = \frac{e^{-\frac{5}{8}\cdot\mu^2+\frac{9}{2}\cdot\mu-9}\cdot\sqrt{5}\sqrt{2}}{\sqrt{\pi}\cdot e^{-\frac{9}{10}}\left(\operatorname{erf}\left(\frac{17}{5}\cdot\sqrt{2}\sqrt{5}\right)+\operatorname{erf}\left(\frac{8}{5}\cdot\sqrt{2}\sqrt{5}\right)\right)} \text{ for } -10 \le \mu \le 10$$

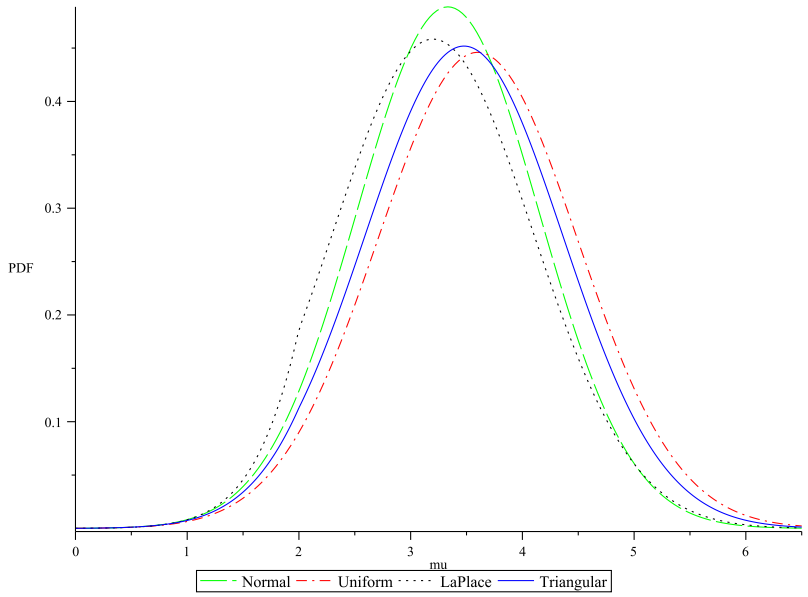
$$\text{LaPlace:} \quad g(\mu|\vec{x}) = \frac{e^{-\frac{1}{2}\cdot|\mu-2|-\frac{5}{8}\cdot\mu^2+\frac{9}{2}\cdot\mu-\frac{163}{16}}\cdot\sqrt{5}\sqrt{2}}{\sqrt{\pi}\left(e^{-\frac{19}{16}}+e^{-\frac{223}{80}}-\operatorname{erf}\left(\frac{1}{2}\sqrt{5}\sqrt{2}\right)e^{-\frac{19}{16}}+\operatorname{erf}\left(\frac{3}{10}\sqrt{2}\sqrt{5}\right)e^{-\frac{223}{80}}\right)} \text{ for all } \mu$$

$$\text{Triangular:} \quad g(\mu|\vec{x}) = \begin{cases} 14.83\,(.0164\cdot\mu+.164)\cdot e^{-0.625\cdot\mu^2+4.5\cdot\mu-9.75} & \text{for } -10 \le \mu \le 2 \\ 14.83\,(2.456-0.02456\cdot\mu)\cdot e^{-0.625\cdot\mu^2+4.5\cdot\mu-9.75}s & \text{for } 2 \le \mu \le 10 \end{cases}$$

Note that posterior distribution for the triangular prior uses floating point rather than exact numbers. The PDF using exact values covers several pages. With these functions in hand, we can plot the posterior distributions in order to visualize how the choice of prior distribution affects the posterior. A plot of the various prior distributions appears as follows:



and a plot of the corresponding posterior distributions appears as follows:



Using APPL Bayes, it is not only possible to use ad-hoc priors such as the LaPlace and triangular distributions. It is also possible to show visually how the posterior distribution changes with the choice of priors. Thus, APPL Bayes serve as an excellent tool for investigating the impact of using various prior distributions.

Along with comparisons, APPL Bayes also simplifies theoretical explorations. For example, we can use APPL Bayes to confirm that the posterior distribution of the μ parameter in a normal distribution with a uniform prior follows a truncated normal distribution. Using the following APPL commands, we can derive the PDF of a Normal($\frac{18}{5}, \frac{2}{5}\sqrt{5}$) distribution truncated between -10 and 10:

```
N:=NormalRV(18/5,(2/5)*sqrt(5));

N2:=Truncate(N,-10,10);
```

The output of this code shows that this truncated normal is distributed $C \cdot e^{-\frac{1}{40}(5x-18)^2}$ where C is a normalizing constant. Completing the square shows that $e^{-\frac{1}{40}(5x-18)^2} = e^{-\frac{5}{8}x^2+\frac{9}{2}x-9}$, or the same distribution that we found as the posterior for μ using the uniform prior. As such, we can use APPL Bayes to show that the posterior distribution for μ with a uniform prior follows a truncated normal distribution.

Leaving the parameters of the LaPlace distribution unspecified, we can use APPL Bayes to derive a closed from solution to the posterior distribution for μ with a LaPlace prior. This derivation, which otherwise is quite complicated, can be accomplished using three simple lines of code in APPL Bayes:

```
X:=NormalRV(mu,sigma);

Y:=LaPlaceRV(a,b);

B:=BayesUpdate(X,Y,[x],mu);
```

Our APPL Bayes output shows that if we have a likelihood function that is distributed Normal(μ, σ), a prior distribution that is distributed LaPlace(a, b) and a data observation x, then the posterior distribution takes the form:

$$g(\mu|x) = \frac{2 \cdot e^{-\frac{|-\mu+b|\cdot\sigma^2+a\cdot x^2-2\cdot a\cdot x\cdot\mu+a\cdot\mu^2}{a\cdot\sigma^2}}}{\sqrt{\pi}\cdot\sigma\cdot e^{\frac{1}{4}\frac{-\sigma^2+4\cdot ab+4\cdot xa}{a^2}}\left(-e^{\frac{2x}{a}}-e^{\frac{2b}{a}}-\operatorname{erf}\left(\frac{1}{2}\frac{2ab-2xa-\sigma^2}{a\sigma}\right)e^{\frac{2x}{a}}+\operatorname{erf}\left(\frac{1}{2}\frac{2ab-2xa-\sigma^2}{a\sigma}\right)e^{\frac{2b}{a}}\right)}$$

for $\sigma > 0$, $b > 0$ and all μ

This closed form solution, which does not appear to exist anywhere in the literature despite the potential usefulness of the LaPlace as a prior distribution for μ, can be derived in APPL Bayes with minimal effort. Similar code shows if we have a likelihood function that is distributed Binomial(n, p)–where p is the unknown parameter and n is a positive integer–and a prior distributed Triangular$(0, b, 1)$ such that $0 < b < 1$, then the posterior distribution for p appears as follows:

$$g(p|x) = \begin{cases} \frac{2\cdot p\cdot(n!)(p^x)(1-p)^{n-x}}{b(n-x)!x!\left(\frac{b(n!)p^x(1-p)^{n-x}}{(n-x)!x!}+\frac{p^x(1-p)^{n-x}n!(1-b^2)}{(b-1)(n-x)!x!}-\frac{2\cdot p^x(1-p)^{n-x}n!(1-b)}{(b-1)(n-x)!x!}\right)} & \text{for } 0 \le p \le b \\ \frac{2(p-1)p^x(1-p)^{n-x}}{b(n-x)!x!\left(\frac{b(n!)p^x(1-p)^{n-x}}{(n-x)!x!}+\frac{p^x(1-p)^{n-x}n!(1-b^2)}{(b-1)(n-x)!x!}-\frac{2\cdot p^x(1-p)^{n-x}n!(1-b)}{(b-1)(n-x)!x!}\right)} & \text{for } b \le p \le 1 \end{cases}$$

The ability of APPL Bayes to produce generalized closed form posterior distributions with ad-hoc priors highlights the program's theoretical usefulness. The examples shown above represent just a small portion of what has been accomplished so far using APPL Bayes. As these examples demonstrate, leveraging the computer algebra capabilities of APPL Bayes makes fruitful theoretical exploration a relatively simple matter.

4 Bayesian Inference on Multiple Parameter Distributions

As demonstrated in the previous section, deriving posterior distributions using Bayesian methods quickly becomes intractable when we deviate from the use of conjugate prior distributions, even

in the single parameter case. This problem compounds rapidly when we enter the realm of distributions with multiple parameters. To understand why, we can begin by discussing two parameter distributions. For a distribution with two parameters, Bayes' Theorem becomes:

$$g(\theta_1, \theta_2|\vec{x}) = \frac{f(x|\theta_1, \theta_2) \cdot g(\theta_1, \theta_2)}{\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x|\theta_1, \theta_2) \cdot g(\theta_1, \theta_2)\ d\theta_1\ d\theta_2}$$

This relationship differs from Bayes' Theorem for single parameter distributions in two important ways. First, the multiple parameter case requires a joint prior distribution, $g(\theta_1, \theta_2)$. Second, it requires the evaluation of a double integral. The multiple parameter case, in fact, requires one iterated integration for each parameter in the distribution. As we saw in the previous sections, these integrals are typically rather difficult to evaluate. Thus, for Bayesians seeking a closed form posterior distribution, the dependence on conjugate priors becomes even stronger in the multiple parameter case.

Bayesians have several methods for dealing with this complication. When conducting applied statistical analysis, Bayesians depend on Markov Chain Monte Carlo (MCMC) methods. Without delving into too much detail, MCMC methods essentially simulate the prior distribution and the likelihood function, and use the data points gathered from this simulation in order to construct a discrete approximation to the posterior distribution. While ideal for one time use in statistical analysis, since the simulations depend on pseudo-random number generators, the posterior distributions that result from MCMC are different after each run. As such, it is difficult to use these results for theoretical experimentation. Moreover, MCMC simulations fail to produce closed for posterior distributions; the output of MCMC is essentially a matrix of probabilities.

Several attempts have been made to devise methods capable of producing closed form posterior distributions in the multiple parameter case. Jaakkola and Jordan (1998) present a method capable of producing approximate closed form distributions of the posterior for logistic regression. In order to accomplishes this, Jaakkola and Jordan utilize variational methods. Variational methods attempt to find a normal approximation to the posterior distribution using computational optimization techniques to minimize the difference between the true posterior distribution and the gaussian approximation. This method, however, has limitations. First, in some cases, normal distributions provide a poor approximation to the logit model. Second, the method only applies to logistic regression, whereas a more general method would be preferable. Finally, because of the reliance on

a normal approximation to the logit curve, closed form posteriors can only be found when normal prior distributions are used.

APPL Bayes, on the other hand, contains an algorithm capable of producing approximate closed form marginal distributions for each parameter in a multiple parameter distribution without depending on conjugate priors. The algorithm works by holding all but one parameter constant, conducting Bayesian inference on that parameter, and then doing the same for the remaining parameter. We then iterate this process until the posterior distributions converge to a final solution. This process results in increasingly better approximations of conditional approximations of the marginal distribution for this parameter. Under the assumption of independence, these approximations converge to the true marginal posterior distribution for each parameter. This section will describe this algorithm, the cases under which the algorithm produces good approximations to the marginal distributions for each parameter, potential issues with the algorithm, as well as its some interest avenues of theoretical exploration that it has already opened.

4.1 Assumptions

In order for the algorithm to produce marginal posterior distributions for the parameters in a multiple parameter likelihood function, we must make several assumptions regarding the parameters, their interdependence and the shape of the posterior distribution. In order to simplify the notation, we will present theses assumptions here in the two-parameter case. First, we assume that, in the joint prior distribution, $g(\theta_1, \theta_2)$, the parameters θ_1 and θ_2 are independent. By the definition of independence, this allows us to write the prior distribution as $g(\theta_1, \theta_2) = g_1(\theta_1) \cdot g_2(\theta_2)$. This assumption serves two purposes. First, APPL's data structure currently only treats single variable distributions. As such, including joint prior distributions would require significant modifications to the underlying structure of APPL. Second, since the posterior distributions must be independent for the conditional distributions that the algorithm produces to equal the marginal distributions for each parameter, the prior distributions must be independent as well. Second, we must assume that the parameters are independent *a posteriori*. That is, the value of each parameter cannot have any effect on the value of the other parameters. This allows us to use our method to compute marginal distributions, rather than simply computing conditional approximations to the marginal distributions. Under independence, the conditional distribution $g(\theta_1|x, \hat{\theta_2})$ is the same as the marginal distribution $g(\theta_1|x)$. This follows directly from the definition of statistical independence.

Along with these assumptions regarding the independence of the parameters, we must also assume that the resulting posterior distributions are convex. This is necessary in order for the iterating mechanism of the algorithm to converge on the correct conditional distribution. This assumption should hold in the vast majority of cases, especially given a large data vectors. Well known results in Bayesian theory show that the posterior distribution is asymptotically normal under a relatively broad class of conditions (see Ghosal, 1999). This is a stricter condition than convexity, which tends to hold even for small data vectors.

Before proceeding to describe the algorithm, it is useful to see how these assumptions allow us to produce an exact marginal distribution. We begin with Bayes' Theorem in the two-parameter case, which, once more, appears as follows:

$$g(\theta_1, \theta_2|\vec{x}) = \frac{f(x|\theta_1, \theta_2) \cdot g(\theta_1, \theta_2)}{\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x|\theta_1, \theta_2) \cdot g(\theta_1, \theta_2) \; d\theta_1 \; d\theta_2}$$

Assuming independence of the parameters in the prior distributions, we can rewrite Bayes' Theorem as:

$$g(\theta_1, \theta_2|\vec{x}) = \frac{f(x|\theta_1, \theta_2) \cdot g_1(\theta_1) \cdot g_2(\theta_2)}{\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x|\theta_1, \theta_2) \cdot g_1(\theta_1) \cdot g_2(\theta_2) \; d\theta_1 \; d\theta_2}$$

Now, convexity guarantees that the algorithm will find the most likely conditional distribution, given a likelihood function, a data vector, and a set of prior distributions. Suppose that we have used our algorithm to find such a distribution for the first parameter. The relationship now becomes:

$$g(\theta_1|\vec{x}, \hat{\theta_2}) = \frac{f(x|\theta_1, \hat{\theta_2}) \cdot g_1(\theta_1)}{\int_{-\infty}^{\infty} f(x|\theta_1, \hat{\theta_2}) \cdot g_1(\theta_1) \; d\theta_1}$$

The integral in the denominator in this case reduces to a single integral. This can either be evaluated symbolically or numerically. In either case, finding the conditional distribution $g(\theta_1|\vec{x}, \hat{\theta_2})$ now reduces to a computational problem. Now, under the assumption of independence in the parameters in the posterior distribution, it follows from the definition of independence that $g(\theta_1|\vec{x}, \hat{\theta_2}) = g(\theta_1|\vec{x})$, the marginal distribution of the parameter θ_1. In this manner, the algorithm is capable of producing marginal posterior distributions for each parameter in a multiple parameter likelihood function.

4.2 The Algorithm

With these assumptions in mind, we can now proceed to describe the steps that the algorithm uses to produces marginal distributions for each parameter. First, the algorithm requires as arguments the likelihood function, a list of marginal prior distributions, a list of unknown parameters, a data

vector, a list of variables in the likelihood function, a list of initial guess values for the parameters and a level of precisions. Written in APPL code, the call sequence for the algorithm appears as:

```
PosteriorMV(X,data,PriorList,paramList,variables,guess,precision);
```

The likelihood function is input as a random variable, using APPL's random variable data structure (described in the appendix). Data is entered as a list-of-sublists. One data vector is required for each variable in the likelihood function. While typically a likelihood function may only contain one variable, for instance an x variable, this becomes important when the algorithm is used for regression. In this cases, the likelihood function may contain severable variable, such as y, x_1, x_2, etc. The list of maginal priors is entered as a list of random variables in APPL random variable format. The list of parameters, variables and the initial guess for each parameter are entered as a simply list. The number of elements in the initial guess list, however, must equal the number of elements in the unknown parameter list. In other words, we must be sure to enter a single guess for each parameter. Precision is a floating point number $\in [0, 1]$.

The Likelihood Function The `PosteriorMV` algorithm begins by deriving the likelihood function using the input data set. Here we introduce the following notation: $f(\vec{x}|\vec{\theta})$ refers to the likelihood function of the data set, given a vector of parameters, $\vec{x}$ refers to the data set, and $f(x|\vec{\theta})$ refers to the input likelihood function for one piece of data. This procedure requires simple multiplication, using Maple's built-in symbolic manipulator. Mathematically, it appears as:

$$f(\vec{x}|\vec{\theta}) \mapsto \prod_{i=0}^{n} f(x_i|\vec{\theta})$$

where n is equal to the number of observations in the data set.

Initial Approximations After deriving the likelihood function, `PosteriorMV` produces initial approximations for the marginal distribution of each parameter, using the initial guess values provided in the call sequence. In order to accomplish this, the algorithm substitutes the initial guess for all but one parameter, and then conducts Bayesian updating on the resulting approximate marginal distributions. Mathematically, this step of the algorithm appears as $f_{\text{approx}} \mapsto g(\hat{\theta}_i|\vec{x}, \vec{\theta}\backslash\theta_i)$ for each θ_i in the parameter vector $\vec{\theta}$. These approximations to the marginal distributions serve as the initial guesses supplied to the algorithm's iteration procedure. Note that, at this point in the algorithm, the approximations remain un-normalized. Since normalizing the marginal distributions is computationally expense, it occurs only once, at the end of the algorithm.

Iteration After finding an initial set of approximations, the algorithm proceeds to find an increasingly better set of approximations. It accomplishes this by repeating the fix-then-update process until the resulting distributions cease to change. In order to determine the degree to which the approximate distributions change after each iteration, however, we require a measure that allows us to track how the distribution changes. The ideal measure would be $\left(\int f^{n+1}_{\text{approx}}\right)^2 - \left(\int f^{n}_{\text{approx}}\right)^2$ which would measure the area between the approximation on step $n+1$ and step n. However, integrals are computationally expensive, making this approach impractical.

An alternative approach may consider using the mean of the distribution. The mean of the approximate marginal can be computed as $E(\hat{\theta}_i) = \int_{-\infty}^{\infty} f^{n+1}_{\text{approx}}(\hat{\theta}_i|\vec{x}, \vec{\theta}\backslash\theta_i)\ d\theta_i$. Again, since the computing the mean also requires integration, using the mean to measure the difference between successive approximations is not ideal.

A more fruitful option is the mode of the distribution. Using the mode is useful because, owing to the convexity of the marginal approximations, all possible marginal approximations have a unique mode. The mode has the added advantage that it avoids the need to evaluate an integral on each step of the iteration. Several issues nevertheless arose in the initial attempts to use the mode as the basis for measuring changes in the approximate marginal distribution. In particular, usual method used to compute the mode involves taking the first derivative of the function, and then solving for its critical point. This requires us to solve $\frac{df^{n+1}_{\text{approx}}(\hat{\theta}_i)}{d\theta_i} = 0$ for the critical $\hat{\theta}_i$ value Maple's symbolic solve function. This approach proved unsatisfactory, because the marginal distributions are only defined within the limits of the supports of the parameter. Consider, for instance, a approximate marginal for a parameter θ that is defined for $\theta \geq 0$. In this case, the function describing the approximate marginal may contain critical values for $\theta < 0$. Using the traditional differentiation approach, Maple sometimes returns these incorrect critical values. Because of this, numerical methods that take an initial guess as an input provide a better alternative. Again, the most natural choice among involves differentiation, and then using a numerical method, perhaps a bisection search, to solve for $\hat{\theta}_i$ in $\frac{df^{n+1}_{\text{approx}}(\hat{\theta}_i)}{d\theta_i} = 0$. The limitations of Maple's floating point number evaluator, however, caused this approach to fail as well. Specifically, in the tails of these approximate marginal distributions, solving for $\hat{\theta}_i$ in $\frac{df^{n+1}_{\text{approx}}(\hat{\theta}_i)}{d\theta_i} = 0$ evaluates to a number very close to 0. Even adjusting Maple to track floating point numbers to 50 digits, Maple has difficulty distinguishing these numbers from 0, and as a result frequently returns values in the tail as critical values. As such, an ideal method computes the mode without having to differentiate the approximate marginal distribution. This can be accomplished

using a Golden Section Search, which works by evaluating the function itself until it converges to the functions highest value. Since the approximate marginal distribution are convex functions, the Golden Section Search is guaranteed to accurately compute the mode of the distribution, to a pre-specified level of precision. In consequence of these advantage, the `PosteriorMV` algorithms employs a Maple implementation of the Golden Section Search to compute the mode of the approximate marginal distributions. Using the Golden Section Search has the added advantage that it requires no calculus. This leaves open the possibility that the algorithm could be implemented in a lower level language, such as C, which would substantially reduce the algorithm's run-time.

Using the mode as a number that uniquely characterizes each approximate marginal distribution, the algorithm produces increasingly accurate approximations to the marginal distribution using the following steps, beginning with a set of guesses, $\vec{\hat{\theta}}$:

1. Fix all parameters except for θ_i
2. Perform Bayesian updating to find the approximate marginal distribution, $g(\theta_i|\vec{x}, \vec{\hat{\theta}})$
3. Replace $\hat{\theta}_i \in \vec{\hat{\theta}}$ with $\text{argmax}_{\theta_i} g(\theta_i|\vec{x}, \vec{\hat{\theta}})$
4. Repeat for all $\theta_i \in \vec{\theta}$

The algorithm continues this iteration process until the guesses for each parameter cease to change, within a pre-set level of precision. In order to determine when to stop, the algorithm tracks the percent change in the guesses for each parameter on each step of the iteration. The iteration process then stops after the condition $\left|\frac{\hat{\theta_{i2}} - \hat{\theta_{i1}}}{\hat{\theta i1}}\right| < \rho$ for all $\theta_i \in \vec{\theta}$ is met, where ρ is the precision variable described in the call sequence.

The set of marginal distributions that the algorithm converges to does not depend on the initial conditions provided in the algorithm's call sequence. However, the iteration process does end considerably sooner if good guesses are supplied by the user. So far, the maximum likelihood estimates for the parameters have provided the best performance with regard to the initial guesses. With independent or weakly dependent parameters the algorithm tends to converge after only a handful of iterations when maximum likelihood estimates are used as initial guesses.

Normalizing the Marginals After completing the iteration process, the algorithm proceeds to normalize the resulting marginal distributions. Due to the assumption that the parameters are

independent *a posteriori*, this can be accomplished with a single integral. Experience has show this integral to be problematic, even given Maple's symbolic manipulation capabilities, due to the complexity of the functions that result when ad-hoc, non-conjugate prior distributions are used for the parameters. As such, numerical methods are used to obtain the normalizing constant. This is disadvantageous insofar as it eliminates the possibility of producing exact closed form marginal distributions. Nevertheless, given sufficiently accurate numerical techniques, the resulting approximate marginal will be very close to the true distribution.

Problems with this approach arise because it is difficult to perform numerical integration on functions that have infinite support. This problem can be remedied using one of two methods. First, truncated prior distributions can be used. The use of truncated prior distributions is convenient because, in the majority of cases, the probability density contained in the tails of the marginal posterior distributions is negligible. As such, little information is lost when a truncated prior is used, as long as the truncation occurs in the tail. Actually, this highlights the usefulness of APPL Bayes in this respect. Only with the aid of software such as APPL Bayes does the use of ad-hoc, truncated priors become feasible. Second, the limits of the numerical integration can be modified when the support reaches out to infinity. Presently, in cases when infinite support poses a problem, the algorithm changes the support of the numerical integration to a point where Maple's floating point evaluator fails to distinguish the value of the function $g(\hat{\theta}_i|\vec{x},\vec{\hat{\theta}})$ from zero. Since only a negligible amount of the area beneath the curve falls beyond this point, omitting it from the integration has little effect on the result. In cases where the numerical integrator fails to normalize the area under the posterior distribution to 1, the algorithm repeats the normalization process until the area under the curve falls between .999999 and 1.000001. As a practical matter, experience has shown that using gaussian quadrature with a sufficiently high precision setting succeeds in producing a normalizing constant in most cases.

Output The output of the algorithm is a list of random variables, describing a closed form approximation to the marginal posterior distribution for each unknown parameter in the likelihood function. Standard APPL procedures can be applied to these random variables in order to construct percentiles, confidence intervals and other constructs of statistical interests. The algorithm is capable of producing closed form approximations to the marginal distributions for each parameter in all cases where the problem can be re-stated as performing inference on a distribution with n unknown parameters. This not only includes inference on distributions with unknown parameters,

but also linear regression. As such, the range of problems to which the algorithm applies is much broader than the set of APPL procedures that operates on single variable distributions.

4.3 Issues with the Algorithm and Theoretical Possibilities

A variety of issues may limit the usefulness of the `PosteriorMV` algorithm under certain conditions. These issues fall broadly into two categories, computational and theoretical.

Computational Issues A number of computational issues have risen during experimentation with the algorithm. These issues relate strongly to the manner in which Maple handles floating point numbers. The first issue results when an insufficiently high level of precision is used in the implementation of the Golden Section Search for finding the mode. In this case, the Golden Section Search finds a mode that closely approximates, but is not exactly equal to, the mode of the intermediate approximation to the marginal distribution. When the Golden Section Search does not find the mode to a sufficiently high degree of precisions, the algorithm has in some cases failed to converge. Instead, the algorithm bifurcates between two sets of approximations, each very close to the set of true marginal distributions. Under ideal circumstances, the algorithm ought to converge to somewhere between these two sets of approximations. This issues can be addressed by increasing the precision setting on the Golden Section Search algorithm. Increasing the precision setting on the mode finding procedure appears to solve this problem in the majority of cases. An alternative approach may be to modify the algorithm so that it recognizes when these oscillations take place. Then algorithm could then produce a final approximation that lies in between the two sets of looser approximations. This method, however, may fail if the algorithm instead oscillates between three or more intermediate approximations.

Second, when working with posterior distributions whose support extends to infinity, our method of modifying the limits of integration sometimes fails to produce an area of one under the approximate marginal distributions. As outlined above, this problem has been addressed by feeding the resulting posteriors into a loop that re-normalizes them until the area under the distributions becomes one. In practice, using this looped re-normalization technique, this particular computational problem has not proven to be particularly inhibiting.

Theoretical Issues A host of additional theoretical issues also accompany the algorithm, especially when the assumption of parameter independence is violated. In particular, when the

parameters are dependent, even weakly dependent, the algorithm does not truly produce marginal distributions. Rather, it provides a heuristic technique that produces the conditional distribution for each parameter that most closely matches the marginal distribution. When parameter dependence is weak, the difference between the distributions that the algorithm produces and the true marginal distributions is likely to be extremely small. It is extenuated, however, when the parameters are strongly dependent, for instance between the parameters in a logistic regression. In the case of logistic regression, the algorithm produces credible sets that center around the same values found using MCMC methods, as well as classical maximum likelihood estimation. However, the credible sets in these cases tend to be somewhat narrower. In other words, the variance of the distributions produced by the algorithm may tend to underestimate the variance of the true marginal distributions. This is difficult to establish, however, because the exact marginal distributions are not available, making comparison impossible. Efforts are underway to analyze the nature of this discrepancy. It may be possible to transform the closed form approximations produced by the algorithm in such a way that it better captures the variance of the true marginal distribution.

In addition to this, an analytical proof that the algorithm converges to the marginal distribution in cases where parameter independence holds has yet to be written. This area of theoretical endeavor is currently under exploration. In spite of the lack of an analytical proof of correctness, however, in practice the algorithm appears to produce solid results for independent parameters.

Theoretical Possibilities Regardless of these difficulties, the `PosteriorMV` represents an advance in the field of computational Bayesian statistics, for several reason. First, it has produced the first set of closed form posterior distributions for several outstanding problems in Bayesian statistics. This includes a logistic regression on the failure data in the O-rings on the challenger space shuttle. The results of this regression appear in the appendix. Second, because the algorithm is completely deterministic, as opposed to MCMC methods, it serves as a convenient tool for theoretical experimentation, especially on distributions with independent parameters. For instance, using the algorithm, it is possible to repeatedly update using the same data set. After each run of the algorithm, the resulting posteriors can be used as the prior distributions for the next run. Repeating this process a sufficient number of times results in a set of degenerate distributions equal to the mean of each of the initial posterior distributions. While it is unclear whether this is of any theoretical interest, such experimentation would not be possible outside of the frame work of the APPL Bayes software.

5 Conclusion and Further Research

The applications demonstrated in this paper show how APPL Bayes can expand the current state of understanding in Bayesian statistics. Concerning single parameter, it may appear that APPL Bayes only has the ability to solve elementary problems. In a sense this is true: in this case APPL Bayes primarily serves as a tool to automate the application of Bayes' Theorem and ease mathematical tedium. Nevertheless, APPL Bayes allows users to experiment with non-standard prior distributions using Maple's symbolic manipulation capability and offer a mechanism whereby Bayesians can test the efficacy of different prior distributions. In the realm of multiple variable distributions, APPL Bayes has proven capable of solving problems that Bayesians would have previously considered intractable. On the one hand, having closed form solutions to these problems may proven little more useful than discrete approximations produced by MCMC methods. On the other hand, closed form solutions may provide an unparalleled opportunity for new theoretical insights. Regardless of their usefulness, however, without APPL's unique random variable data structure, deriving these closed form distributions would be tedious. As such, APPL Bayes has opened up an entirely new avenue of computational exploration for Bayesian statisticians.

Appendix: An Application of the PosteriorMV Algorithm

In this section, we present an application of the PosteriorMV algorithm. First, we will generate a data set by drawing randomly from a Normal$(4, 2)$ distribution. The following APPL Bayes code accomplishes this task:

```
xdata:=[];

for i from 1 to 20 do

  xdata:=[op(xdata),NormalVariate(4,2)];

end do;
```

This produced the data vector $\vec{x} = [4.952372179, 5.709441586, 5.43308551, 6.661711285,$
$3.0593391937, 3.82564198, 0.632558631, 4.163596364, 2.544106259, 3.24794389,$
$5.12768509, 2.084339059, 2.820134333, 7.045769745, 2.49379872, 2.6576674, 4.71412016,$
$3.481198931, 2.581350952, 3.971881718]$. First, we assume that the likelihood function takes the

form of a normal distribution, which appears as follows:

$$f(x|\mu,\sigma) = \frac{e^{-\frac{1}{2}\cdot\frac{(x-\mu)^2}{\sigma^2}}}{\sigma} \text{ for } -\infty \leq x \leq \infty, \ -\infty \leq \mu \leq \infty, \ \sigma > 0$$

Note that we omit the normalizing constant, since multiplying the likelihood by a constant has no effect on the posterior distribution. Now, in order to apply the algorithm, we must verify that our assumption of independent parameters is valid. An easy way to accomplish this is to produce a contour plot of the two parameters in the likelihood function. This can be accomplished using the following APPL Bayes code:

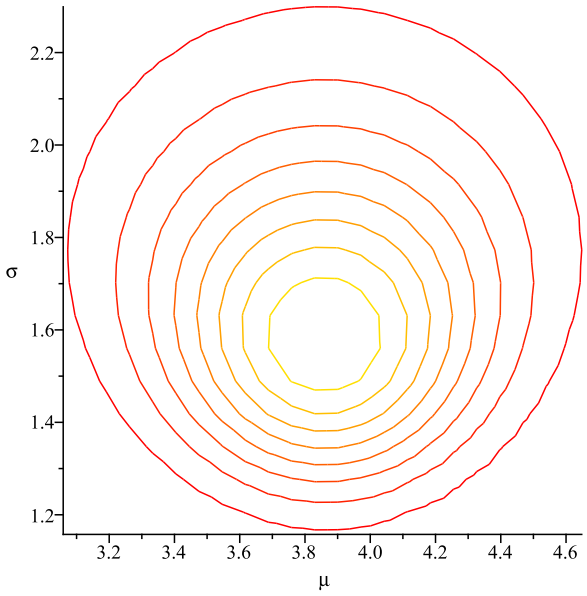
```
X:=NormalRV(mu,sigma);

likeFunc:=simplify(Likelihood(X,[xdata],[x]);

with(plots);

likeContour:=contourplot(likefunc,mu=0..10,sigma=0..10,numpoints=20000);
```

This code produces the following contour plot:



The flat slope of the contour plot confirms that we are justified in assuming independence of the parameters. Since it appears that our assumptions hold, we can now proceed to apply prior distributions to the two unknown parameters, μ and σ. In our example, we will use an Exponential($\frac{2}{7}$)

distribution truncated between 0 and 20 as the prior for μ and a Normal$(2.3, 1)$ distribution truncated between 0 and 12 as the prior for σ. These distributions appear as:

$$\begin{aligned} g(\mu) &= .2867 \cdot e^{-.2857 \cdot \mu} \text{ for } 0 \leq \mu \leq 20 \\ g(\sigma) &= .4033 \cdot e^{-.0050 \cdot (10 \cdot \sigma - 23)^2} \text{ for } 0 \leq \sigma \leq 12 \end{aligned}$$

In APPL Bayes, we set up the priors using the following code:

```
Y1:=Truncate(ExponentialRV(2/7),0,20);

Y2:=Truncate(NormalRV(2.3,1),0,12);

PriorList:=[Y1,Y2];
```

We must now produce sensible initial guesses for each parameter. Typically, using the maximum likelihood estimators for the likelihood function works quites effectively. In this example, we can use $\hat{\mu} = \bar{x}$ and $\hat{\sigma} = \bar{s}$ as sensible initial guesses for the parameters. Note that the output of the algorithm is independent of the initial guesses. Good initial guesses simply serve to limit the amount of time the algorithm requires to converge. We can establish our initial guesses using the following APPL commands:

```
B:=BootstrapRV(xdata);

muHat:=Mean(B);

sigHat:=sqrt(Variance(B));

guess:=[muHat,sigHat];
```

We are now ready to apply the algorithm:

```
paramList:=[mu,sigma];

variables:=[x];

precision:=.001;

Params:=PosteriorMV(X,data,PriorList,paramList,variables,guess,precision);
```

The algorithm converges after three iterations, and produces the following two marginal posterior distributions:

$$g(\mu|\vec{x}) \approx 74494.74 \cdot e^{30.5864\cdot\mu - 3.9986\cdot\mu^2 - 69.5891\cdot\mu^3} \text{ for } 0 \leq \mu \leq 20$$

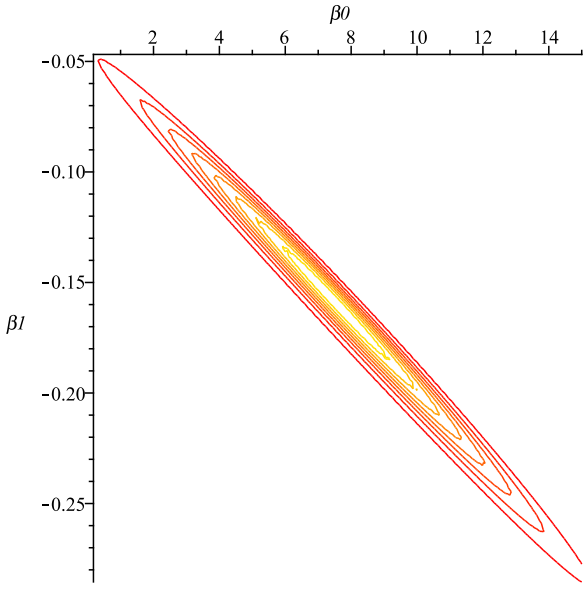
$$g(\sigma|\vec{x}) \approx 3.9847 \cdot 10^8 \frac{\cdot e^{\frac{-1\cdot 10^{-28}(2.5001\cdot 10^{29} + 5.000\cdot 10^{27}\cdot\sigma^4 - 2.300\cdot 10^{28}\cdot\sigma^3 + 2.645\cdot 10^{28}\cdot\sigma^2)}{\sigma^2}}}{\sigma^{20}} \text{ for } 0 \leq \sigma \leq 12$$

These distributions show that $E(\mu|\vec{x}) = 3.825$ and $E(\sigma|\vec{x}) = 1.000$. Additionally APPL commands can be used in order to conduct further inference on these posterior distributions.

As mentioned above, the algorithm produces credible sets that are too narrow when the assumption of parameter independence does not hold. In order to demonstrate this, we will use the algorithm to conduct a logistic regression, using the failure data for the O-rings in the Challenger rocket. Since the code used to generate this example is similar to that presented above, it will be omitted for this example. We begin with the likelihood function for the logistic regression, which appears as follows:

$$f(x|\beta_0, \beta_1) = \left(\frac{e^{\beta_0+\beta_1\cdot x}}{1+e^{\beta_0+\beta_1\cdot x}}\right)^y \cdot \left(1 - \frac{e^{\beta_0+\beta_1\cdot x}}{1+e^{\beta_0+\beta_1\cdot x}}\right)^{1-y} \text{ for } 0 \leq x \leq 1$$

Again, we can construct a contour plot of the two parameters in the likelihood function in order to determine whether or our assumption of independence is valid. Using the O-ring data set, this likelihood function appears as follows:



The distinct slope of the contour plot indicates that the two parameters are dependent. Because of this slope, the output of the algorithm will be a conditional slice of the multi-parameter that

is narrower than the true marginal distribution. In order to determine an approximation to the marginal distribution, we will begin by applying prior distributions to each parameter. In this case, we will use an Exponential(1/8.9) distribution as a prior for the parameter β_0 and a Uniform$(-1, 1)$ distribution as a prior for the parameter β_1. These distributions appear as follows:

$$\begin{aligned} g(\beta_0) &= 0.1124 \cdot e^{-.1124 \cdot \beta_0} \text{ for } \beta_0 > 0 \\ g(\beta_1) &= \frac{1}{2} \text{ for } -1 \leq \beta_1 \leq 1 \end{aligned}$$

Applying these prior distributions to the likelihood function, the algorithm converges after five iterations and produces closed form approximations to the marginal distributions for each parameter. The resulting approximate marginal distributions take the following form:

$$\begin{aligned} g(\beta_0|\vec{x}) \approx & \left(1.02 \cdot 10^{60} \cdot e^{6.89 \cdot \beta_0 - 76.64}\right) / (1 + e^{\beta_0 - 11.93})^6 \cdot (1 + e^{\beta_0 - 12.65})^{24} \cdot (1 + e^{\beta_0 - 12.47})^6 \\ & (1 + e^{\beta_0 - 14.46})^6 \cdot (1 + e^{\beta_0 - 12.29})^6 \cdot (1 + e^{\beta_0 - 12.11})^{18} \cdot (1 + e^{\beta_0 - 13.01})^6 \\ & (1 + e^{\beta_0 - 13.20})^6 \cdot (1 + e^{\beta_0 - 10.30})^6 \cdot (1 + e^{\beta_0 - 11.39})^6 \cdot (1 + e^{\beta_0 - 14.09})^6 \\ & (1 + e^{\beta_0 - 9.58})^6 \cdot (1 + e^{\beta_0 - 13.56})^{12} \cdot (1 + e^{\beta_0 - 14.64})^6 \cdot (1 + e^{\beta_0 - 13.74})^{12} \\ & (1 + e^{\beta_0 - 14.28})^6 \cdot (1 + e^{\beta_0 - 10.48})^6 \text{ for } \beta_0 > 0 \\ g(\beta_1|\vec{x}) \approx & \ 2.39 \cdot 10^{11} \cdot e^{62.23 + 424 \cdot \beta_1} / (1 + e^{8.89 + 66 \cdot \beta_1})^6 \cdot (1 + e^{8.89 + 70 \cdot \beta_1})^{24} \cdot (1 + e^{8.89 + 69 \cdot \beta_1})^6 \\ & (1 + e^{8.89 + 80 \cdot \beta_1})^6 \cdot (1 + e^{8.89 + 68 \cdot \beta_1})^6 \cdot (1 + e^{8.89 + 67 \cdot \beta_1})^{18} \cdot (1 + e^{8.89 + 72 \cdot \beta_1})^6 \\ & (1 + e^{8.89 + 73 \cdot \beta_1})^6 \cdot (1 + e^{8.89 + 57 \cdot \beta_1})^6 \cdot (1 + e^{8.89 + 63 \cdot \beta_1})^6 \cdot (1 + e^{8.89 + 78 \cdot \beta_1})^6 \\ & (1 + e^{8.89 + 53 \cdot \beta_1})^6 \cdot (1 + e^{8.89 + 75 \cdot \beta_1})^{12} \cdot (1 + e^{8.89 + 81 \cdot \beta_1})^6 \cdot (1 + e^{8.89 + 76 \cdot \beta_1})^{12} \\ & (1 + e^{8.89 + 79 \cdot \beta_1})^6 \cdot (1 + e^{8.89 + 58 \cdot \beta_1})^6 \text{ for } -1 \leq \beta_1 \leq 1 \end{aligned}$$

From these distributions, we can construct 95 percent credible. For the intercept parameter, β_0, APPL produces a credible set of $[8.379, 9.238]$ and for the slope parameter, β_1, APPL produces a credible set of $[-.1888, -.1750]$.

References

Basu, S., and Mukhopadhyay, S. (2000). *Sankhya: The Indian Journal of Statistics, Series B*, "Bayesian Analysis of Binary Regression Using Symmetric and Asymmetric Links," Vol. 62, No. 3.

Cook, P., Broemeling, D., (1995), *The American Statistician*, "Bayesian Statistics Using Mathematica," Vol. 49, No. 1.

Drew, J., Evans, D., Glen, A., and Leemis, L., (2008) *Computational Probability: Algorithms and Applications in the Mathematical Sciences*, Springer.

Gelman, A., Carling, J., Stern, H., and Rubin, D. (2004). *Bayesian Data Analysis, Second Edition*, Chapman & Hall/CRC, Boca Raton, Fl.

Jaakkola, T., and Jordan, M. (2000). *Statistics and Computing*, "Bayesian parameter estimation via variational methods," Vol. 10, No. 1.

Siddhartha, D., Fowlkes, E., and Hoadley, B. (1989). *Journal of the American Statistical Association*,"Risk Analysis of the Space Shuttle: Pre-Challenger Prediction of Failure", Vol.84, No. 408.

Tanis, E. (1999). *The American Statistician*, "A Review of Maple V ©Release 5", Vol. 53.