Fast and accurate approximation of the full conditional for gamma shape parameters

Jeffrey W. Miller
Harvard University, Department of Biostatistics

February 5, 2018

Abstract

The gamma distribution arises frequently in Bayesian models, but there is not an easy-to-use conjugate prior for the shape parameter of a gamma. This inconvenience is usually dealt with by using either Metropolis–Hastings moves, rejection sampling methods, or numerical integration. However, in models with a large number of shape parameters, these existing methods are slower or more complicated than one would like, making them burdensome in practice. It turns out that the full conditional distribution of the gamma shape parameter is well approximated by a gamma distribution, even for small sample sizes. This article introduces a quick and easy algorithm for finding a gamma distribution that approximates the full conditional distribution of the shape parameter. We empirically demonstrate the speed and accuracy of the approximation across a wide range of conditions. If exactness is required, the approximation can be used as a proposal distribution for Metropolis–Hastings.

1 Introduction

The lack of a nice conjugate prior for the shape parameter of the gamma distribution is a commonly occurring nuisance in many Bayesian models. Conjugate priors for the shape parameter do exist, but unfortunately, they are not analytically tractable (Damsleth, 1975; Miller, 1980). In Markov chain Monte Carlo (MCMC) algorithms, one can easily use Metropolis–Hastings (MH) moves to update the shape parameter, but mixing can be slow if the proposal distribution is not well-calibrated. Rejection sampling schemes can also be used for sampling the shape parameter (Son and Oh, 2006; Pradhan and Kundu, 2011), however, the more efficient schemes such as adaptive rejection sampling (Gilks and Wild, 1992) are complicated and require log-concavity, which does not always hold under some important choices of prior. Modern applications involve models with a large number of parameters, necessitating fast inference algorithms that work very generally with minimal tuning. Indeed,

our motivation for developing the approach in this article is a gene expression model involving tens of thousands of gamma shape parameters.

In this article, we introduce a fast and simple algorithm for finding a gamma distribution that approximates the full conditional distribution of a gamma shape parameter. This can be used to perform approximate Gibbs updates by sampling from the approximating gamma distribution. Alternatively, the approximation can be used as an MH proposal distribution, to make a move that exactly preserves the full conditional and has high acceptance rate. The article is organized as follows. Section 2 describes the algorithm, and Section 3 contains an empirical assessment of the accuracy and speed of the algorithm. Section 4 discusses previous work, and Section 5 provides mathematical details on the derivation of the algorithm. Additional empirical results are contained in the appendix.

2 Algorithm

Consider the model $X_1, \ldots, X_n | a, \mu \sim \text{Gamma}(\text{shape} = a, \text{rate} = a/\mu)$ i.i.d., where $a, \mu > 0$, and note that this makes $\mu = \mathbb{E}(X_i \mid a, \mu)$. This parametrization is convenient, since a controls the concentration and μ controls the mean. We assume a gamma prior on the shape, say, $a \sim \text{Gamma}(\text{shape} = a_0, \text{rate} = b_0)$ where, if desired, a_0 and b_0 can depend on μ. The following algorithm produces A and B such that

$$p(a \mid x_1, \ldots, x_n, \mu, a_0, b_0) \approx \text{Gamma}(a \mid \text{shape} = A, \text{rate} = B).$$

We use $\psi(x)$ and $\psi'(x)$ to denote the digamma and trigamma functions, respectively, and $\log(x)$ denotes the natural log.

Algorithm 1: Approximating the full conditional of the shape parameter

input : data $x_1, \ldots, x_n > 0$, parameters $\mu, a_0, b_0 > 0$, tolerance $\epsilon > 0$, and maximum number of iterations M.
output: A and B.
begin
  $R \leftarrow \sum_{i=1}^n \log(x_i)$, $S \leftarrow \sum_{i=1}^n x_i$, and $T \leftarrow S/\mu - R + n\log(\mu) - n$
  $A \leftarrow a_0 + n/2$ and $B \leftarrow b_0 + T$
  for $j = 1, \ldots, M$ **do**
    $a \leftarrow A/B$
    $A \leftarrow a_0 - na + na^2\psi'(a)$
    $B \leftarrow b_0 + (A - a_0)/a - n\log(a) + n\psi(a) + T$
    if $|a/(A/B) - 1| < \epsilon$ **then return** A, B
  return A, B

We recommend setting $\epsilon = 10^{-8}$ and $M = 10$. Using $\epsilon = 10^{-8}$, the algorithm terminates in four iterations or less in all of the simulations we have done, so setting

$M = 10$ is conservative. Most languages have routines for computing the digamma and trigamma functions, which are defined by $\psi(x) = \frac{\partial}{\partial x} \log \Gamma(x)$ and $\psi'(x) = \frac{\partial}{\partial x}\psi(x)$, where $\Gamma(x)$ is the gamma function. See Section 5 for the derivation of the algorithm.

To perform an approximate Gibbs update to a in an MCMC sampler, run Algorithm 1 to obtain A, B using the current values of $x_{1:n}, \mu, a_0, b_0$, and then sample $a \sim \text{Gamma}(A, B)$. (For brevity, in the rest of the paper, we write $x_{1:n}$ to denote $(x_1, \ldots, x_n)$, and we write $\text{Gamma}(\alpha, \beta)$ to denote the gamma distribution with shape α and rate β.)

Alternatively, to perform a Metropolis–Hastings move to update a, one can run Algorithm 1 to obtain A, B using the current values of $x_{1:n}, \mu, a_0, b_0$, then sample a proposal $a' \sim \text{Gamma}(A, B)$, and accept with probability

$$\min\left\{1,\ \frac{p(a' \mid x_{1:n}, \mu, a_0, b_0)\text{Gamma}(a|A, B)}{p(a \mid x_{1:n}, \mu, a_0, b_0)\text{Gamma}(a'|A, B)}\right\},$$

where a is the current value of the shape parameter. Note that the normalization constant of $p(a \mid x_{1:n}, \mu, a_0, b_0)$ does not need to be known since it cancels.

To infer the mean μ, note that the inverse-gamma family is a conditionally conjugate prior for μ given a. Therefore, Gibbs updates to μ can easily be performed when using an inverse-gamma prior on μ, independently of a, given a_0 and b_0.

3 Performance assessment

To assess the performance of Algorithm 1, we evaluated its accuracy in approximating the true full conditional distribution, and its speed in terms of the number of iterations until convergence, across a wide range of simulated conditions.

Specifically, for each combination of $n \in \{1, 10, 100\}$, $r \in \{0.5, 1, 2\}$, $a_{\text{true}} \in \{10^{-6}, 10^{-5}, \ldots, 10^5, 10^6\}$, $\mu_{\text{true}} \in \{10^{-6}, 10^{-5}, \ldots, 10^5, 10^6\}$, and $a_0 \in \{1, 0.1, 0.01\}$, we generated five independent data sets $x_1, \ldots, x_n \sim \text{Gamma}(a_{\text{true}}, a_{\text{true}}/\mu_{\text{true}})$ i.i.d., ran Algorithm 1 with inputs $x_{1:n}$, $\mu = r\mu_{\text{true}}$, a_0, and $b_0 = a_0$, and then compared the resulting $\text{Gamma}(a|A, B)$ approximation to the true full conditional distribution $p(a \mid x_{1:n}, \mu, a_0, b_0)$. For the prior parameters, we set $b_0 = a_0$ in each case so that the prior mean is 1. In each case, we used $\epsilon = 10^{-8}$ for the convergence tolerance, and $M = 10$ for the maximum number of iterations.

3.1 Approximation accuracy

First, to get a visual sense of the closeness of the approximation, Figure 1 shows the cumulative distribution functions (CDFs) of the true and approximate full conditional distributions for each case with $r = 1$, $a_{\text{true}} = 1$, and $\mu_{\text{true}} = 1$. In each plot, we see that the approximation is close to the true distribution, and in most cases the approximation is visually indistinguishable from the truth. These results are

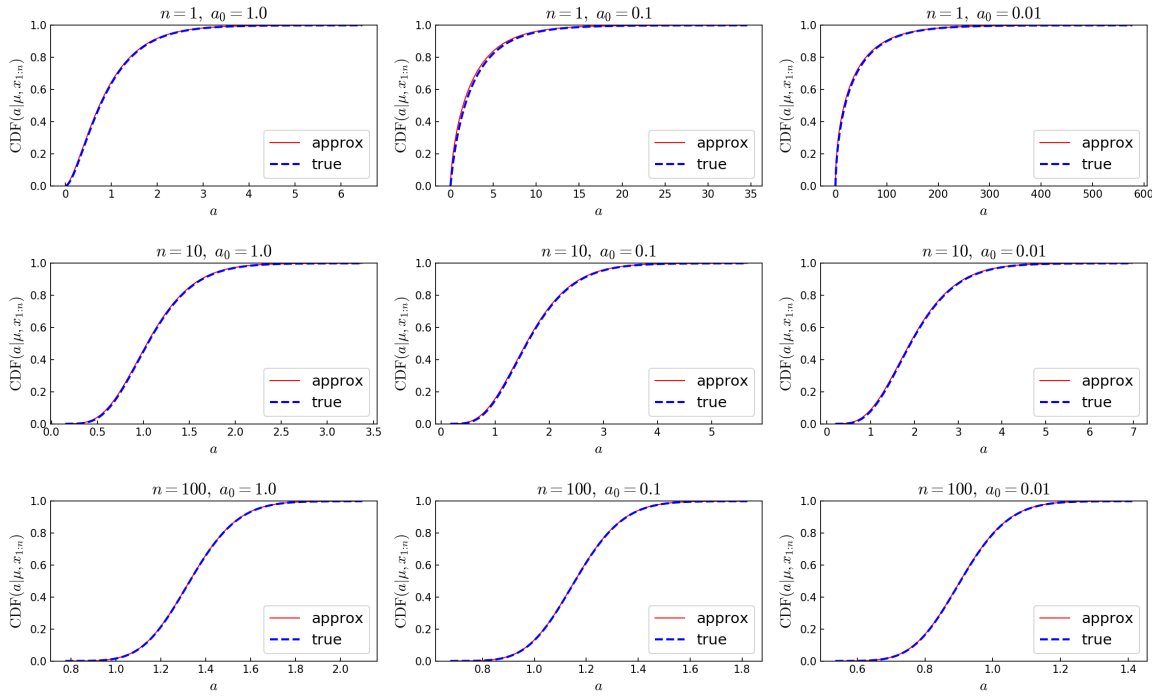

Figure 1: Cumulative distribution function (CDF) of the true full conditional $p(a \mid x_{1:n}, \mu, a_0, b_0)$ and approximate full conditional, on simulated data $x_1, \ldots, x_n \sim \mathrm{Gamma}(1, 1)$, when conditioning on $\mu = 1$ and the prior is $a \sim \mathrm{Gamma}(a_0, a_0)$.

representative of the full range of cases considered — similar plots were observed across the other cases.

To quantify the discrepancy between the truth and the approximation, for each simulation run we computed:

1. the total variation distance, $d_{\mathrm{TV}}(f, g) = \frac{1}{2} \int |f(a) - g(a)| da$,
2. the Kullback–Leibler divergence, $d_{\mathrm{KL}}(f, g) = \int f(a) \log(f(a)/g(a)) da$, and
3. the reverse Kullback–Leibler divergence, $d_{\mathrm{KL}}(g, f) = \int g(a) \log(g(a)/f(a)) da$,

where $f(a) = p(a \mid x_{1:n}, \mu, a_0, b_0)$ is the true density and $g(a) = \mathrm{Gamma}(a|A, B)$ is the approximate density. Numerical integration was performed to compute these integrals, in order to compare with the truth; see Appendix A for details.

Figure 2 shows the largest observed value (that is, the worst-case value) of $d_{\mathrm{TV}}(f, g)$, $d_{\mathrm{KL}}(f, g)$, and $d_{\mathrm{KL}}(g, f)$ across all cases for each a_0 and each n, using the average discrepancy over the five data sets for each case. The approximation accuracy increases as n increases, and the accuracy is decent even when n is small. This shows that the approximation is quite good across a very wide range of cases.

To see a more fine-grained breakdown of the discrepancies, case-by-case, Figure 3 shows $d_{\mathrm{TV}}(f, g)$ for all of the cases with $a_0 = 1$. For the corresponding figures with

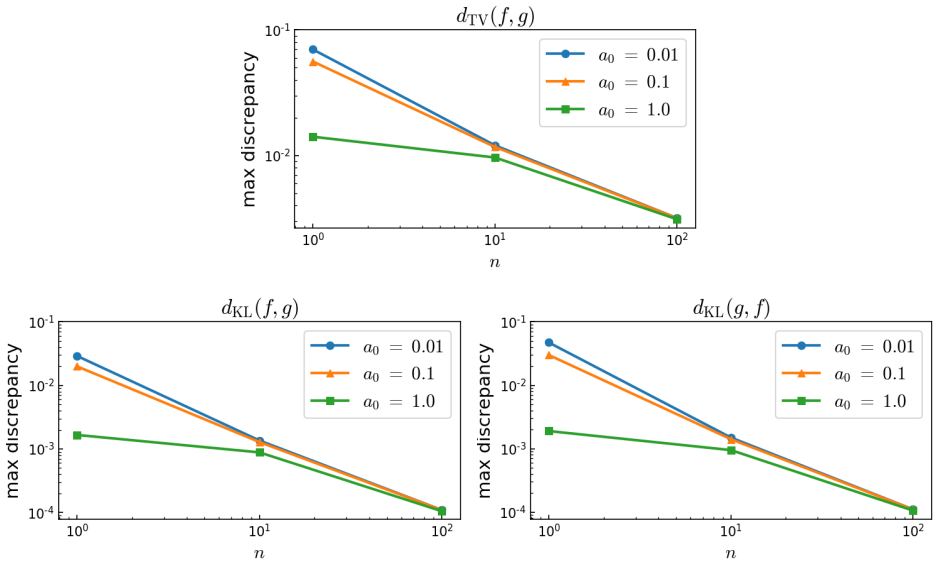

Figure 2: Largest observed discrepancy between the true full conditional f and the approximate full conditional g, for the total variation distance $d_{\mathrm{TV}}(f, g)$ and the Kullback–Leibler divergences $d_{\mathrm{KL}}(f, g)$ and $d_{\mathrm{KL}}(g, f)$, for each a_0 and n.

$a_0 = 0.1$ and $a_0 = 0.01$, see Appendix B. Also, for the corresponding figures showing $d_{\mathrm{KL}}(f, g)$ and $d_{\mathrm{KL}}(g, f)$ case-by-case, see Appendix B.

3.2 Speed of convergence

To evaluate the speed of Algorithm 1 in terms of the number of iterations required, Table 1 shows the number of runs in which the algorithm required k iterations to reach the termination condition, for each k. All runs terminated in four iterations or less. (For each a_0, these results are over all of the 1521 combinations of n, r, a_{true}, μ_{true}, times five replicates each, as described above.) This shows that the algorithm terminates very rapidly, across all cases considered.

Table 1: Number of runs that required k iterations before terminating.

	$a_0 = 1$	$a_0 = 0.1$	$a_0 = 0.01$
1 iteration	0	0	0
2 iterations	0	318	631
3 iterations	5751	4699	4308
4 iterations	1854	2588	2666

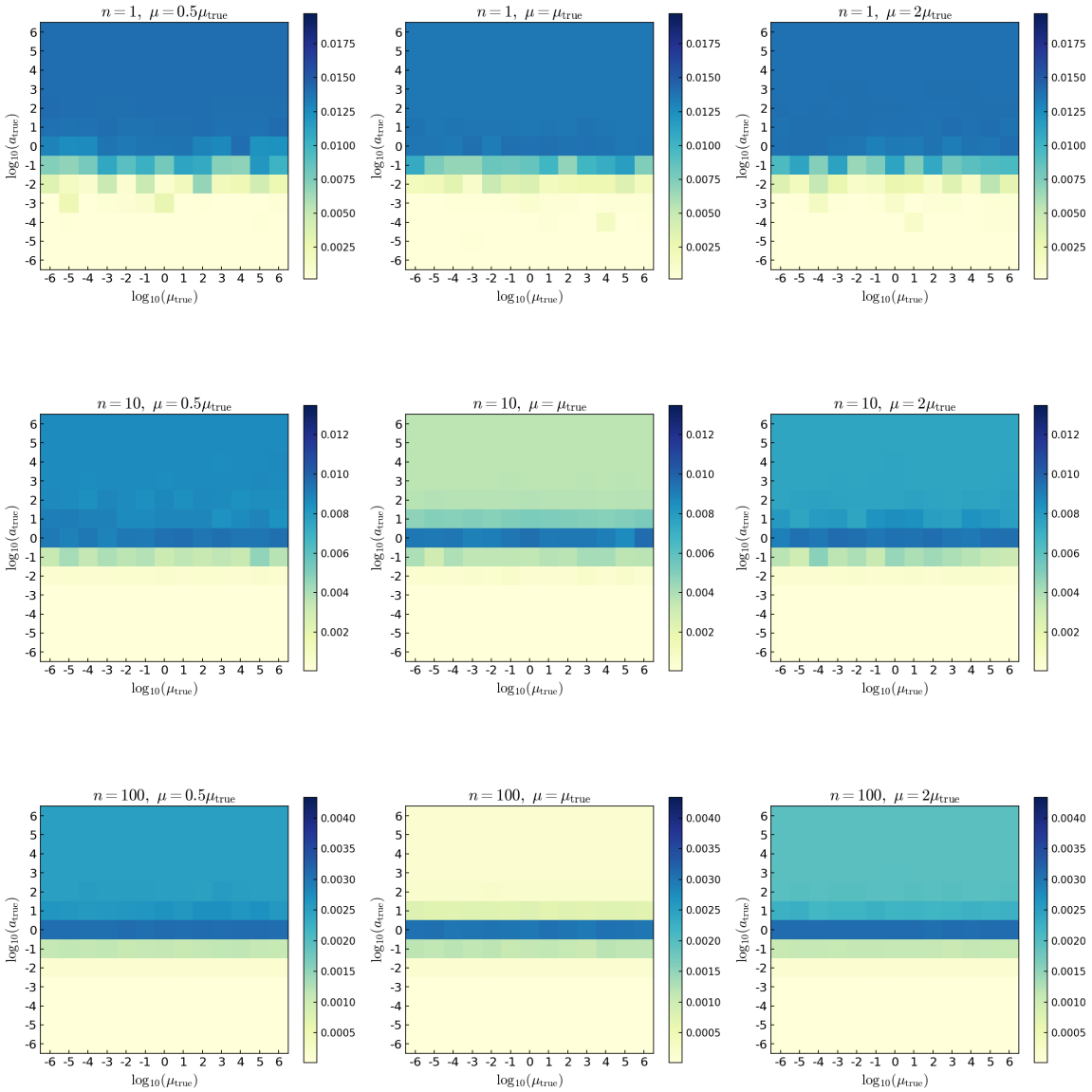

Figure 3: Total variation distance $d_{\text{TV}}(f, g)$ between the true full conditional f and the approximate full conditional g, for every case with $a_0 = 1$. The values shown are the averages over the five independent data sets for each case. Note the scale at the right of each plot.

4 Previous work

The original inspiration for Algorithm 1 was the generalized Newton algorithm of Minka (2002) for maximum likelihood estimation of gamma parameters. That said, our algorithm is different since we are considering the full conditional distribution of the shape, rather than finding the unconditional MLE of the shape. Further, we iteratively set a to the mean rather than the mode of the approximating density. This is an important difference since the mode can be at 0 when $a_0 < 1$, but the derivatives blow up at 0, making it incoherent to match the derivatives at 0. Meanwhile, our algorithm handles $a_0 < 1$ without difficulties. Also, preliminary evidence suggests that even when the mode is valid, the mean yields a more accurate approximation.

Damsleth (1975) found conjugate priors for the shape a in the case of known rate b, as well as the case of unknown b. However, it seems that the distributions in these conjugate families are not easy to work with, analytically. Consequently, Damsleth (1975) resorts to numerical integration over a, for each value of b, in order to obtain $p(b \mid x_{1:n}) = \int p(a, b \mid x_{1:n})\, da$, the posterior density of b.

Similarly, Miller (1980) numerically integrates over a to obtain the posterior of b (or of the mean μ), but instead of doing so for each value of b, he numerically computes the mean, variance, skewness, and kurtosis of $p(b \mid x_{1:n})$, and then constructs an approximation to $p(b \mid x_{1:n})$ by using the method of moments to fit a member of the Pearson family to it. Miller (1980) uses improper priors as well as the conjugate priors of Damsleth (1975).

These numerical integration approaches would be fine for a small number of shape parameters with fixed data $x_{1:n}$ and fixed hyperparameters a_0, b_0. However, modern applications involve hierarchical models with many shape parameters, with latent x's and unknown hyperparameters. In such cases, the numerical integration approaches of Damsleth (1975) and Miller (1980) would be very computationally intensive.

Son and Oh (2006) propose to use the adaptive rejection sampling (ARS) method of Gilks and Wild (1992) to sample from the full conditional for the shape a, in a Gibbs sampler for a and b. There are a few disadvantages to this approach. ARS is quite a bit more complicated to implement than our procedure, and requires additional bookkeeping that is burdensome when dealing with a large number of shape parameters. Further, ARS requires the target distribution to be log-concave, but log-concavity of the full conditional of a is not always guaranteed. For instance, when using a Gamma(a_0, b_0) prior on a, if $0 < a_0 < 1$ then the prior is not log-concave, which can cause the full conditional given $\mu, x_{1:n}$ to fail to be log-concave when a_0 is sufficiently small and $n = 1$. In practice, it is important to allow a_0 to be small in order to obtain a less informative prior while holding the prior mean a_0/b_0 fixed.

Rejection sampling approaches have also been used by other authors, for instance, Pradhan and Kundu (2011) use the log-concave rejection sampling method of Devroye (1984), and Tsionas (2001) uses rejection sampling with a calibrated Gaussian proposal distribution.

5 Derivation of the algorithm

Let $f(a) = p(a \mid x_{1:n}, \mu, a_0, b_0)$ denote the true full conditional density, and let $g(a) = \text{Gamma}(a|A, B)$ for some A, B to be determined. The basic idea of the algorithm is to choose A and B so that the first and second derivatives of $\log g$ match those of $\log f$ at a point a near the mean of f. To find a point near the mean of f, the algorithm is initialized with a choice of A and B based on an approximation of the gamma function, and then iteratively, A and B are refined by matching the first and second derivatives of $\log f$ at the mean A/B from the previous iteration.

It might seem like a better idea to use a point near the mode of f rather than the mean, by analogy with the Laplace approximation. However, on this problem, empirical evidence suggests that using the mean tends to provide better accuracy than the mode. Using the mean also simplifies the algorithm since the mode of g is sometimes at 0, but the derivatives of $\log f$ and $\log g$ blow up at 0.

5.1 Matching derivatives

First, to find the formulas for matching the first and second derivatives of $\log g$ to those of $\log f$ at some point $a > 0$, note that

$$\begin{aligned}
\log g(a) &= A\log(B) - \log\Gamma(A) + (A-1)\log(a) - Ba \\
\frac{\partial}{\partial a}\log g(a) &= \frac{A-1}{a} - B \\
\frac{\partial^2}{\partial a^2}\log g(a) &= -\frac{A-1}{a^2}.
\end{aligned}$$

Meanwhile, $f(a) \propto p(x_{1:n} \mid a, \mu, a_0, b_0)\, p(a \mid \mu, a_0, b_0)$ and

$$\begin{aligned}
p(x_{1:n} \mid a, \mu, a_0, b_0) &= \prod_{i=1}^{n} \frac{(a/\mu)^a}{\Gamma(a)} x_i^{a-1} \exp\big(-(a/\mu)x_i\big) \\
&= \frac{(a/\mu)^{na}}{\Gamma(a)^n} \exp\big((a-1)R\big) \exp\big(-(a/\mu)S\big) \\
&= \frac{a^{na}}{\Gamma(a)^n} \exp\big(-R - (T+n)a\big) \qquad (5.1)
\end{aligned}$$

where $R = \sum_{i=1}^n \log(x_i)$, $S = \sum_{i=1}^n x_i$, and $T = S/\mu - R + n\log(\mu) - n$. Thus,

$$\begin{aligned}
\log f(a) &= \text{const} + \log p(x_{1:n} \mid a, \mu, a_0, b_0) + \log p(a \mid \mu, a_0, b_0) \\
&= \text{const} + na\log(a) - n\log\Gamma(a) - (T+n)a + (a_0 - 1)\log(a) - b_0 a \\
\frac{\partial}{\partial a}\log f(a) &= n\log(a) + n - n\psi(a) - (T+n) + \frac{a_0 - 1}{a} - b_0 \\
\frac{\partial^2}{\partial a^2}\log f(a) &= n/a - n\psi'(a) - \frac{a_0 - 1}{a^2}
\end{aligned}$$

since $\psi(a) = \frac{\partial}{\partial a} \log \Gamma(a)$ and $\psi'(a) = \frac{\partial^2}{\partial a^2} \log \Gamma(a)$. Setting $\frac{\partial^2}{\partial a^2} \log g(a) = \frac{\partial^2}{\partial a^2} \log f(a)$ yields

$$A = a_0 - na + na^2\psi'(a), \tag{5.2}$$

and setting $\frac{\partial}{\partial a} \log g(a) = \frac{\partial}{\partial a} \log f(a)$ yields

$$B = b_0 + \frac{A - a_0}{a} - n \log(a) + n\psi(a) + T. \tag{5.3}$$

The updates to A and B in Algorithm 1 are defined by Equations 5.2 and 5.3.

5.2 Updating the matching point a

Given A and B such that Gamma(A, B) approximates f, we can choose a to approximate the mean of f by setting it to the mean of Gamma(A, B), namely, $a = A/B$. By alternating between updating $a \leftarrow A/B$, and updating A, B via Equations 5.2 and 5.3, we iteratively refine the choice of a. This yields the loop in Algorithm 1. Convergence is assessed by checking if the relative change in a is smaller than the tolerance ϵ, that is, if $|a_{\text{old}} - a_{\text{new}}| < \epsilon a_{\text{new}}$, or equivalently, $|a_{\text{old}}/a_{\text{new}} - 1| < \epsilon$.

5.3 Initializing via approximation of $\Gamma(a)$

To find good starting values of A and B, we consider approximations to the gamma function $\Gamma(a)$ in two regimes: (i) when a is large, and (ii) when a is small.

By Stirling's approximation to $\Gamma(a)$, we have that $\Gamma(a)/a^a \sim \sqrt{2\pi}\, a^{-1/2} e^{-a}$ as $a \to \infty$. (Here, $h_1(a) \sim h_2(a)$ as $a \to a^*$ means that $h_1(a)/h_2(a) \to 1$ as $a \to a^*$.) Plugging this into Equation 5.1, we get

$$p(x_{1:n} \mid a, \mu, a_0, b_0) \approx (2\pi)^{-n/2} a^{n/2} e^{na} \exp\big(-R - (T+n)a\big) \propto a^{n/2} \exp(-Ta),$$

and therefore, $f(a)$ is approximately proportional to Gamma$(a \mid a_0 + n/2,\, b_0 + T)$ when a is large. This suggests initializing with $A = a_0 + n/2$ and $B = b_0 + T$, as in Algorithm 1.

As $a \to 0$, we have $\Gamma(a)/a^a \sim a^{-1}$, since $a^a \to 1$ and $\Gamma(a)/a^{-1} = a\Gamma(a) = \Gamma(a+1) \to \Gamma(1) = 1$ as $a \to 0$. Plugging this into Equation 5.1, we see that

$$p(x_{1:n} \mid a, \mu, a_0, b_0) \approx a^n \exp\big(-R - (T+n)a\big) \propto a^n \exp\big(-(T+n)a\big),$$

and therefore, $f(a)$ is approximately proportional to Gamma$(a \mid a_0 + n,\, b_0 + T + n)$ when a is small. This suggests initializing with $A = a_0 + n$ and $B = b_0 + T + n$. While similar to the large a version, empirically we find that this initialization tends to converge slightly slower, by 1 iteration or so.

6 Acknowledgments

I would like to thank Will Townes for helpful conversations.

A Numerical integration for truth comparison

Since the true full conditional distribution is not analytically tractable, it is necessary to approximate the integrals for the total variation distance and the Kullback–Leibler divergences between the true and approximate distributions. To do this, we use a simple adaptive numerical integration technique using an importance distribution to choose points (basically, a deterministic importance sampler).

Specifically, suppose we want to approximate the integral $\int h(a)g(a)da$ where $g(a)$ is a probability density on $\mathbb{R}$, and $h(a)$ is some function on $\mathbb{R}$. Let $G(a)$ be the CDF of g, and let $a_i = G^{-1}(u_i)$ where $u_i = (i - 0.5)/N$ for $i = 1, \ldots, N$. (If the inverse of G does not exist, use the generalized inverse distribution function.) Then $\int h(a)g(a)da \approx \frac{1}{N}\sum_{i=1}^{N} h(a_i)$ when N is large. This is similar to Monte Carlo, but is much more accurate due to using equally-spaced points u_i rather than random draws $u_i \sim \text{Uniform}(0, 1)$.

We apply this approximation with $g(a) = \text{Gamma}(a|A, B)$, where $A, B > 0$ are the output from Algorithm 1. Let $f(a) = p(a \mid x_{1:n}, \mu, a_0, b_0)$ be the true full conditional. Since the normalizing constant of f is unknown, write $f(a) = \tilde{f}(a)/Z$ where $\tilde{f}(a)$ can be easily computed, and observe that

$$Z = \int_{-\infty}^{\infty} \tilde{f}(a)da = \int_{0}^{\infty} \frac{\tilde{f}(a)}{g(a)} g(a)da = \int_{-\infty}^{\infty} w(a)g(a)da$$

where $w(a) = \tilde{f}(a)/g(a)$ for $a > 0$ and $w(a) = 0$ for $a \leq 0$, since $g(a) > 0$ when $a > 0$, and $f(a) = g(a) = 0$ when $a \leq 0$. Thus, $Z \approx \frac{1}{N}\sum_{i=1}^{N} w(a_i)$.

Define $\widehat{f}(a) = \tilde{f}(a)/\big(\frac{1}{N}\sum_{i=1}^{N} w(a_i)\big)$, so that $\widehat{f}(a) \approx f(a)$. The integral for total variation distance can be written in the form $\int h(a)g(a)da$ as follows:

$$d_{\text{TV}}(f, g) = \frac{1}{2}\int_{-\infty}^{\infty} |f(a) - g(a)|da = \frac{1}{2}\int_{0}^{\infty} \Big|\frac{f(a)}{g(a)} - 1\Big| g(a)da = \int_{-\infty}^{\infty} h(a)g(a)da$$

where $h(a) = \frac{1}{2}|f(a)/g(a) - 1|$ for $a > 0$ and $h(a) = 0$ for $a \leq 0$. Thus, since $a_i > 0$,

$$d_{\text{TV}}(f, g) \approx \frac{1}{N}\sum_{i=1}^{N} \frac{1}{2}\Big|\frac{\widehat{f}(a_i)}{g(a_i)} - 1\Big|.$$

Similarly,

$$d_{\text{KL}}(f, g) \approx \frac{1}{N}\sum_{i=1}^{N} \frac{\widehat{f}(a_i)}{g(a_i)} \log \frac{\widehat{f}(a_i)}{g(a_i)}$$

and

$$d_{\text{KL}}(g, f) \approx \frac{1}{N}\sum_{i=1}^{N} \log \frac{g(a_i)}{\widehat{f}(a_i)}.$$

B Additional empirical results

Figures 4 and 5 show the total variation distance for the cases with $a_0 = 0.1$ and $a_0 = 0.01$, respectively. Figures 6 and 7 show the Kullback–Leibler divergences for the cases with $a_0 = 1$.

References

E. Damsleth. Conjugate classes for gamma distributions. *Scandinavian Journal of Statistics*, pages 80–84, 1975.

L. Devroye. A simple algorithm for generating random variates with a log-concave density. *Computing*, 33(3-4):247–257, 1984.

W. R. Gilks and P. Wild. Adaptive rejection sampling for Gibbs sampling. *Applied Statistics*, 41(2):337–348, 1992.

R. B. Miller. Bayesian analysis of the two-parameter gamma distribution. *Technometrics*, 22(1):65–69, 1980.

T. P. Minka. Estimating a gamma distribution. Technical report, Microsoft Research, 2002.

B. Pradhan and D. Kundu. Bayes estimation and prediction of the two-parameter gamma distribution. *Journal of Statistical Computation and Simulation*, 81(9): 1187–1198, 2011.

Y. S. Son and M. Oh. Bayesian estimation of the two-parameter Gamma distribution. *Communications in Statistics—Simulation and Computation*, 35(2):285–293, 2006.

E. G. Tsionas. Exact inference in four-parameter generalized gamma distributions. *Communications in Statistics—Theory and Methods*, 30(4):747–756, 2001.

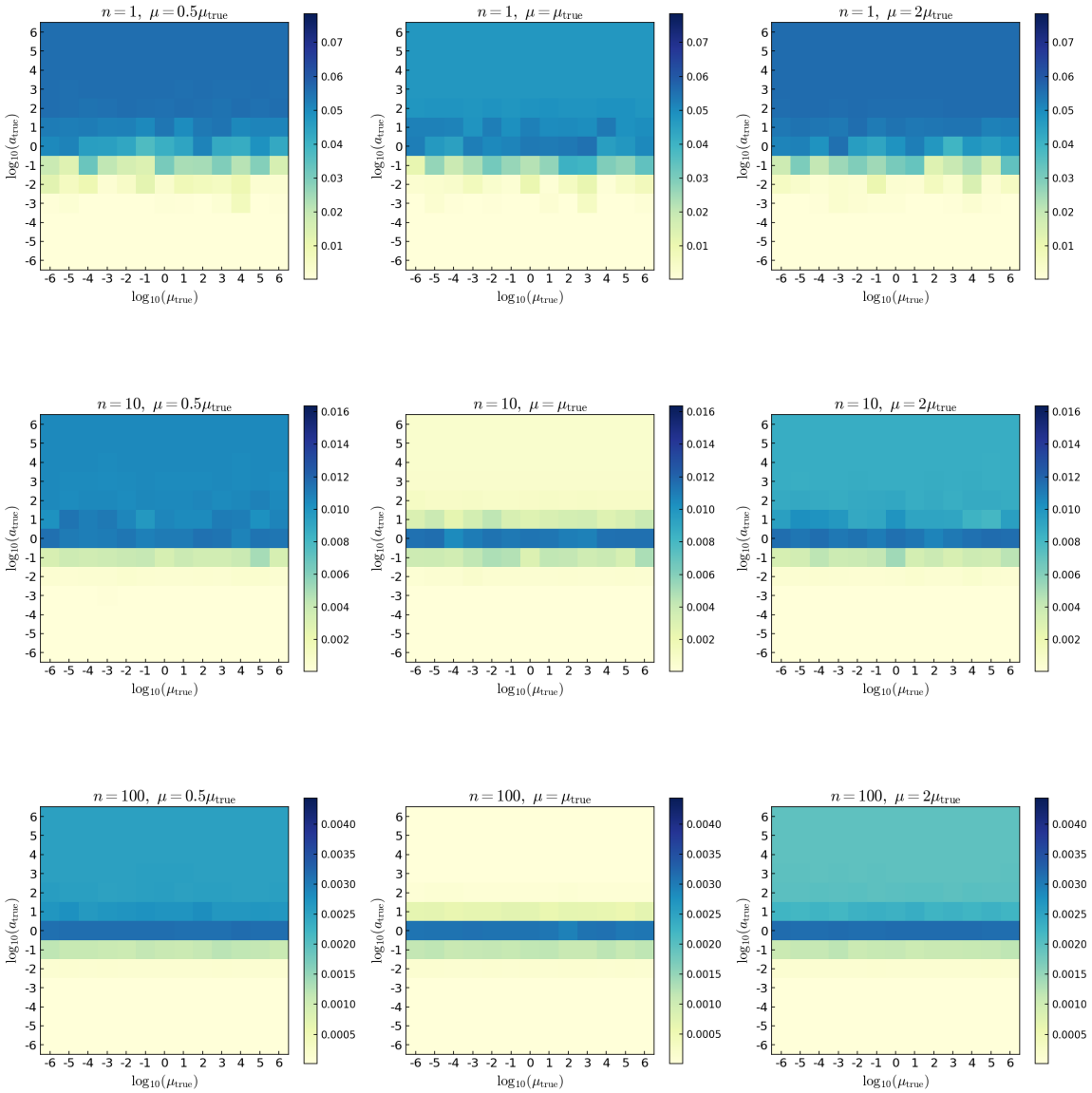

Figure 4: Total variation distance $d_{\text{TV}}(f, g)$ between the true full conditional f and the approximate full conditional g, for every case with $a_0 = 0.1$. The values shown are the averages over the five independent data sets for each case. Note the scale at the right of each plot.

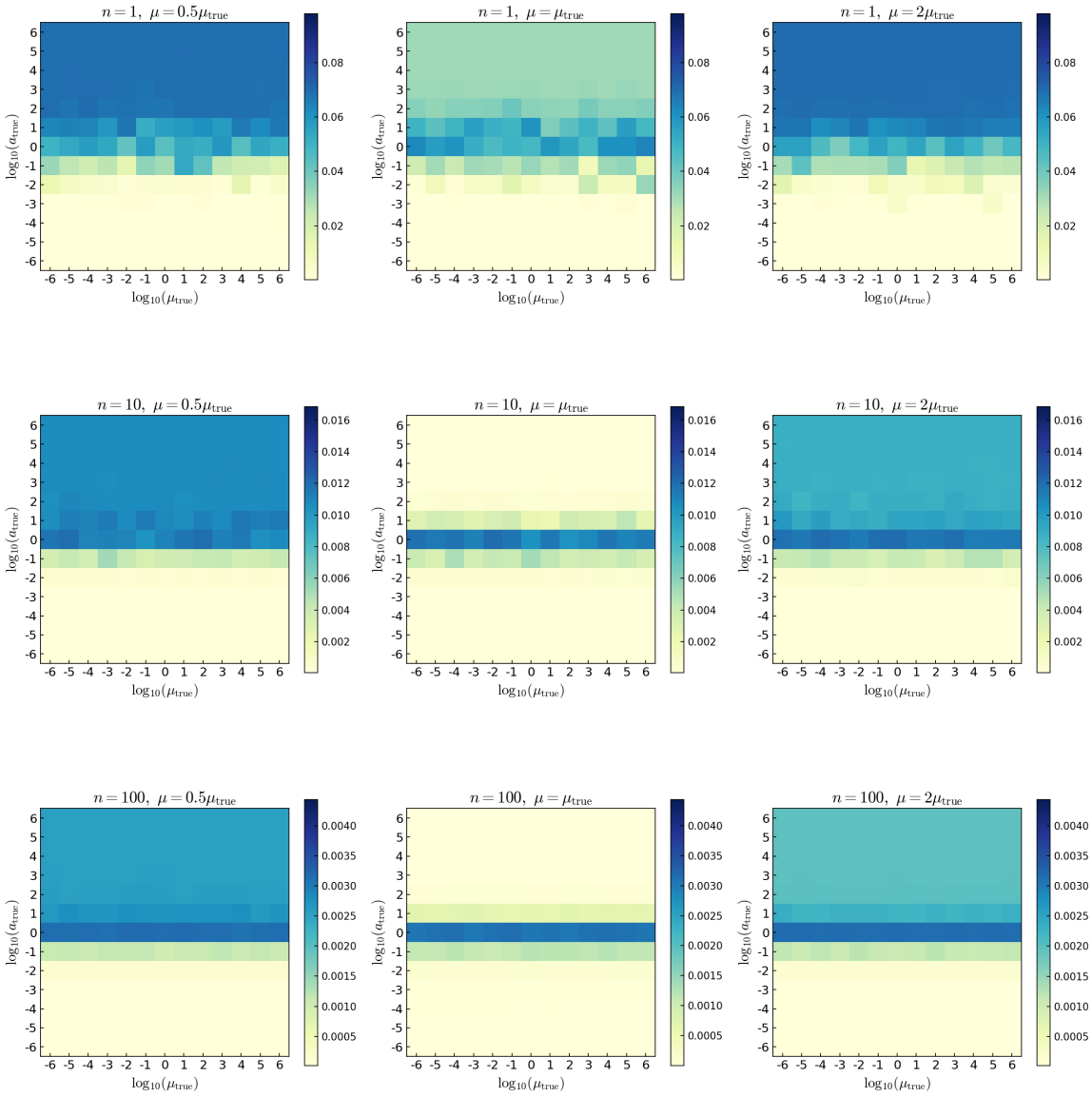

Figure 5: Total variation distance $d_{\text{TV}}(f, g)$ between the true full conditional f and the approximate full conditional g, for every case with $a_0 = 0.01$. The values shown are the averages over the five independent data sets for each case. Note the scale at the right of each plot.

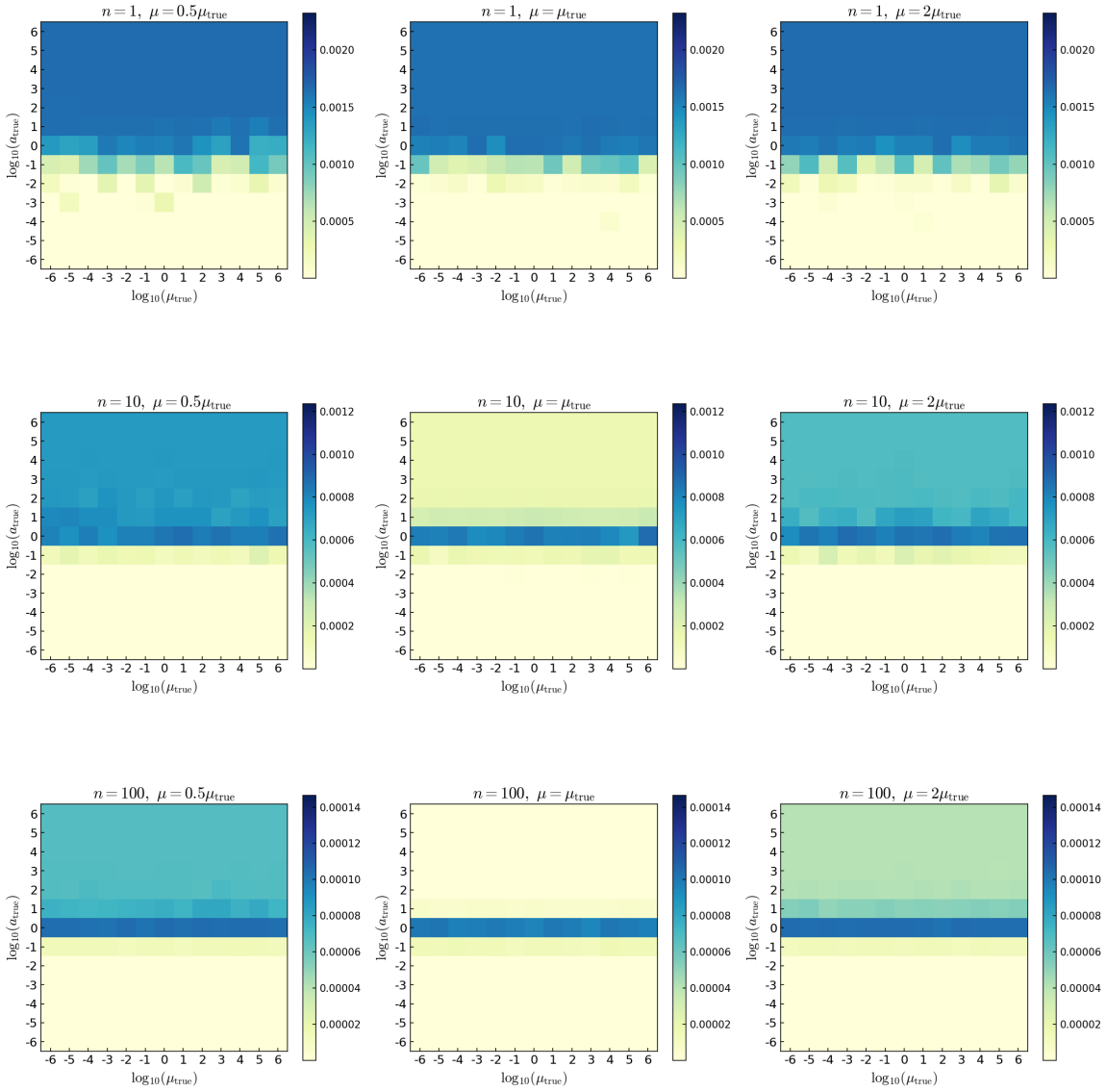

Figure 6: Kullback–Leibler divergence $d_{\text{KL}}(f, g)$ between the true full conditional f and the approximate full conditional g, for every case with $a_0 = 1$. The values shown are the averages over the five independent data sets for each case. Note the scale at the right of each plot.

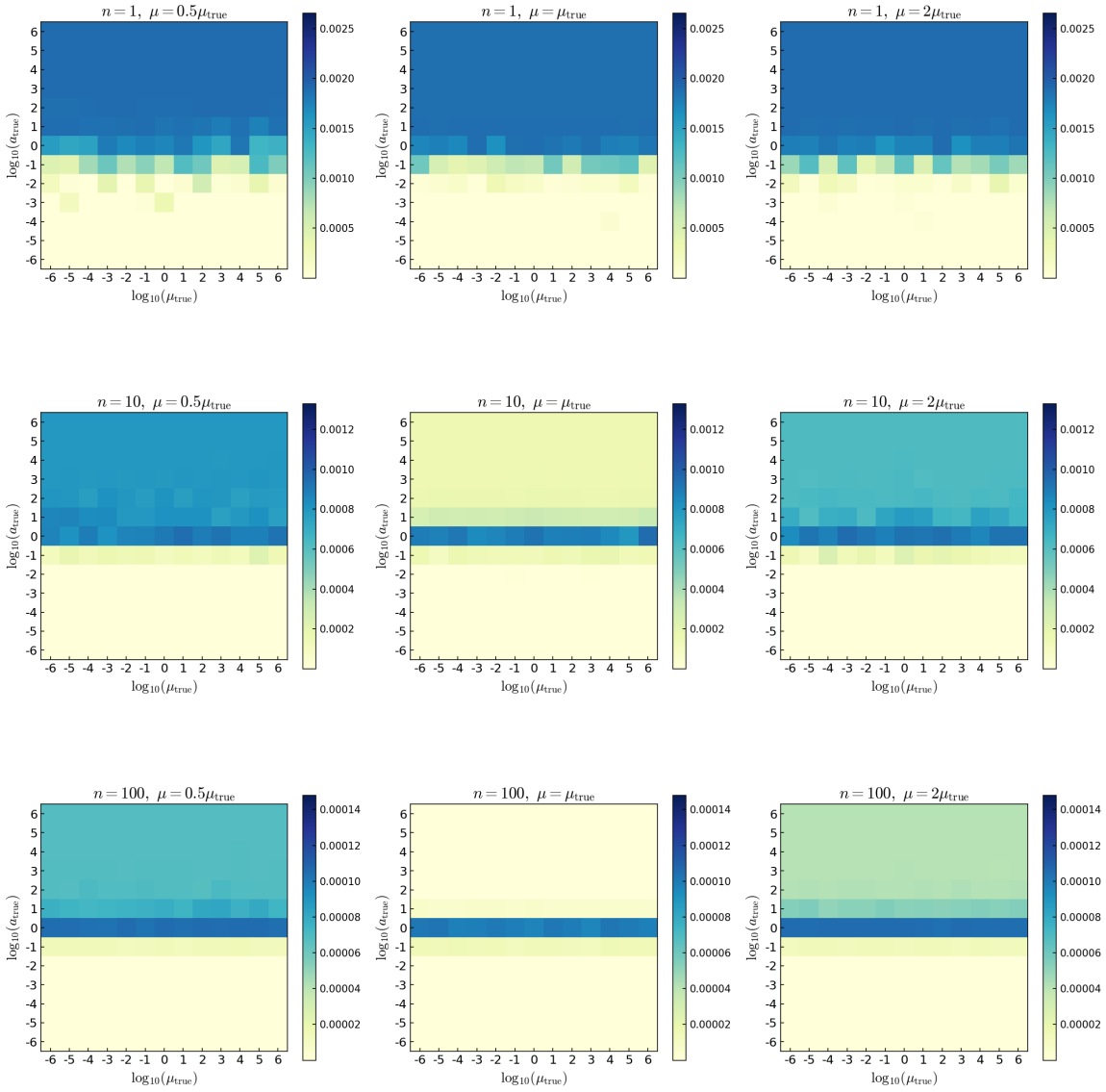

Figure 7: Kullback–Leibler divergence $d_{\text{KL}}(g, f)$ between the true full conditional f and the approximate full conditional g, for every case with $a_0 = 1$. The values shown are the averages over the five independent data sets for each case. Note the scale at the right of each plot.