STAT 440 Final Project

Vincent Le (vt2le), Edward Lu (h62lu), Jason Ong (jojiekha)

2020-04-17

1. Abstract

Many important and challenging problems in the modern age involve classification, such as differentiating between different types of detected objects for autonomous vehicles or identifying diseases based on symptom data. Time is an important factor in both of these examples, motivating the need for efficient modelling approaches. One statistical approach to this task is model-based classification, in which we assume that each of the observed data points follows some distribution and try to cluster the points based on the parameters. However, applying Bayesian inference directly is slow and does not scale with the size of the data. In this project, we implement a Gibbs sampler as a faster approximation and provide it as an R package. We then evaluate the sampler using a real medical dataset of human bronchial epithelial (HBE) data.

2. Introduction

Gibbs sampling as an approximation to Bayesian inference has recently been successfully applied to many areas of study, particularly in the sciences such as biological sequencing (Wang and Wang (2020)) and visual perception (Malem-Shinitski et al. (2020)). This success is owed to the fact that the sampling approach is flexible and more computationally efficient than other algorithms such as EM. Expensive operations such as matrix inversions scale with

the dimensionality of the number of parameters of the chosen distribution instead of the size of the dataset. Gibbs sampling is used within MCMC where random variables can be generated from a marginal distribution without having to calculate its density (Casella and George (1992)), providing the added benefit of allowing us to easily sample each parameter from complex distributions.

To this end, we implement and test a Gibbs sampler and provide it as the package `rgibbs`. This package includes efficient conditional updates for each parameter that leverages parallel computation in C++ provided by the `mniw` library and unit tests for each parameter. In each test, we compare the normalized conditional log posterior and unnormalized complete log posterior to verify that they differ by the same constant for any value of the argument, when only that argument is changed, though there are other ways to do this as discussed in Gandy and Scott (2020). We then use this package to evaluate the performance of a Gibbs sampler on a subset of the HBE dataset created by Hill et al. (2014), which categorizes trajectories of particles in mucosal fluids by their concentrations. These trajectories are modeled using fractional Brownian motion as discussed in Lysy et al. (2016). We show that the model performs reasonably well as it can identify clusters close to the ones given by the data. For convenience, the package also provides a helper function to load the HBE dataset to be used directly by the sampler. The remainder of this paper will discuss the methods used, justify technical decisions and analyze the results.

3. Methodology

The Gibbs sampler iteratively performs conditional updates for each of its parameters by fixing every other parameter and performing a draw from its conditional distribution. These parameters are each described below in detail, given the observed data y and V.

- Θ: The observation means, initialized to the observed data y. Each conditional draw is

sampled from:

$$\Theta_i|(\mu, \Sigma) \sim N(G_i(y_i - \mu_{z_i}) + \mu_{z_i}, G_i V_i), \quad G_i = \Sigma_{z_i}(V_i + \Sigma_{z_i})^{-1}$$

This is done in parallel using the `rRxNorm` function from `mniw`, since each Θ_i is independent.

- μ: The group means, initialized to be the sample mean for each group for a given initial group estimation. Each conditional draw is sampled from:

$$\mu_k|(\Theta, Sigma, z) \sim N(\bar{\theta}_k, \Sigma_k/N_k)$$

Where N_k is the number of points in group k. Each μ_k is independent, so these draws are done simultaneously using `rmNorm`.

- Σ: The group variances, initialized to be the sample variance for each group. Each conditional draw is sampled from:

$$\Sigma_k|(\Theta, \mu, z) \sim InvWish(\Omega_k + \sum_i (\theta_i - \mu_k)(\theta_i - \mu_k)', N_k + v_k)$$

Where $\Omega_k = var(y)$ and $v_k = p + 2$ with p being the observation dimension. Each draw is independently sampled using `riwish`.

- ρ: The group membership probabilities, initialized to the normalized number of points in each group. Each conditional draw is sampled from:

$$\rho|z \sim Dirichlet(\alpha)$$

Where α is one more than the number of points in each group.

- z: The group memberships, initialized using the `kmeans++` algorithm. Each conditional

draw is sampled from:

$$z_i|(\Sigma, \rho, \mu, \Theta) \sim Multinom(K, \lambda_i), \quad \lambda_i = \frac{exp(\kappa_{ik})}{\sum_m exp(\kappa_{im})}$$

In order to avoid divisions by $N_k = 0$ in the conditional update for μ, we reject any update for z that includes an empty group.

In each of the M iterations of updating each parameter in turn, we store each updated value (excluding Θ and z unelss the flags `Theta_out` and `z_out` are set to `TRUE`) and cumulatively sum up the posterior probability matrices $\Lambda^{(m)}$, which are computed using the λ_{ik} formula in the conditional update for z. Finally, we compute the posterior probabilities $\Lambda = \frac{1}{M} \sum_m \Lambda^{(m)}$.

To unit test each parameter, we generate random values for it while fixing every other parameter. We then use the corresponding PDF and compare it to the overall data likelihood, ensuring that it differs by the same constant in each case within a very small tolerance. Since these tests pass and we use the corresponding distribution to sample points to conditionally update each parameter, we can be reasonably sure that the updates are correct.

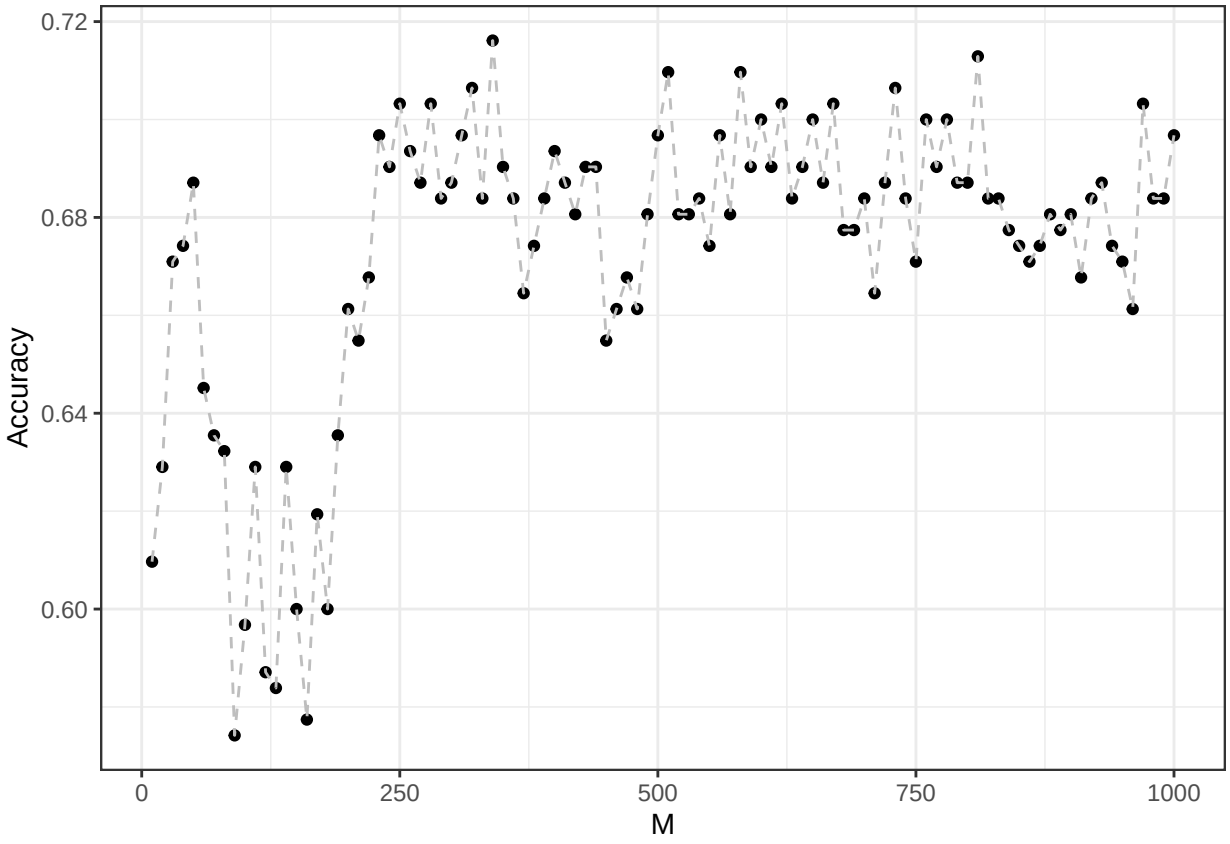
The HBE dataset uses a fractional Brownian motion model with $p = 6$ parameters. In order to reduce the amount of computation required, we use a smaller subset of the dataset by considering only points with concentration percentage `1p5`, `2`, `2p5` or `3`. Therefore, we only consider $N = 310$ observations and $K = 4$ classes. We run the Gibbs sampler for $M = 1000$ iterations with an additional 1000 steps of burn-in.

4. Results

To evaluate the model performance on the HBE dataset, we can compare the predicted group memberships at the final timestep $z^{(1000)}$ to the true group memberships z provided in the data. Note that this is difficult to do numerically, since `init_z` does not use any consistent order to assign the groups, so we have to manually match each of the values $(1, 2, 3, 4)$ to a

concentration label. Since the initialization for z changes every run, we re-run the sampler several times and see an average classification accuracy of around 70%. Interestingly, we notice that the majority of points with the lowest and highest concentrations `1p5` and `3` are correctly classified, while the concentrations in the middle are often misclassified as one of the two extremes. This suggests that the distinctions between similar concentrations are not particularly clear, and that the 0.5% difference that separates each label may be too small.

As a point of interest, we run the sampler with `z_out = TRUE` and use intermediate values $z^{(m)}$ to plot the accuracy over time:



While this graph does not show convergence, it does indicate significantly higher stability past 250 iterations, which suggests that $M = 1000$ is sufficient.

5. Discussion

In this project, we create the R package `rgibbs`, which includes a Gibbs sampler implementation with conditional updates and unit tests for each parameter and a helper function to load the HBE dataset. We then evaluate the sampler on this dataset and attain an accuracy of 70%.

Further extensions of this work include:

- Considering additional evaluation metrics such as precision and recall using Λ.
- Facilitate variable selection techniques to pick smaller p.
- Evaluating the sampler on other datasets.

References

- Casella, G. and George, E.I. (1992). "Explaining the Gibbs Sampler." *The American Statistician*, 46(3): 167-174.
- Gandy, A. and Scott, J. (2020). "Unit Testing for MCMC and other Monte Carlo Methods." *arXiv*, https://arxiv.org/abs/2001.06465.
- Hill, D.B., Vasquez, P.A., Mellnik, J., McKinley, S.A., Vose, A., Mu, F., Henderson, A.G.,Donaldson, S.H., Alexis, N.E., Boucher, R.C., and Forest, M.G. (2014). "A biophysical basis for mucus solids concentration as a candidate biomarker for airways disease." *PLoS ONE*, 9(2): e87681.
- Lysy, M., Pillai, N.S., Hill, D.B., Forest, M.G., Mellnik, J.W., Vasquez, P.A., and McKinley, S.A. (2016). "Model comparison and assessment for single particle tracking in biological fluids." *Journal of the American Statistical Association*, 111(516): 1413–1426.
- Malem-Shinitski, N., Opper, M., Reich, S., Schwetlick, L., Seelig, S.A., Engbert, R. (2020). "A Mathematical Model of Exploration and Exploitation in Natural Scene Viewing." *arXiv*, https://arxiv.org/abs/2004.04649.
- Wang, S. and Wang, L. (2020). "Particle Gibbs Sampling for Bayesian Phylogenetic inference." *arXiv*, https://arxiv.org/abs/2004.06363.

Apendix

HBE analysis and graph code.

```
require(ggplot2)
source("gibbs-hbe.R")


data <- hbe.load()
N <- data$N
p <- data$p
K <- data$K
y <- data$y
V <- data$V
z <- data$z


# Run the sampler.
M <- 1000
out <- gibbs.fit(M, K, y, V, z_out = TRUE)
save(out, file = "out.RData")


# Evaluate sampler accuracy.
load(file = "out.RData")
# This (hardcoded) mapping changes based on the initialization!
mapping <- c("3", "2", "1p5", "2p5")
labels <- match(z, mapping)
sum(out$z[,1000] == labels) / N


# Plot accuracy over iterations.
```

```r
outputs <- sapply(1:100, function(i) {
  sum(out$z[,i*10] == labels) / N
})
df <- data.frame(x = 1:length(outputs), y = outputs)


ggplot(df, aes(x = 10*(1:100), y = outputs)) +
  geom_point() +
  geom_line(linetype = "dashed", color = "grey") +
  theme_bw() +
  labs(x = "M", y ="Accuracy")
```