Estimating the Sources of Metagenomic Data using Bayesian Statistical Methods

Zeph Turner

Spring Semester 2018

Abstract

In a study of knockout mice, we want to determine whether or not bacteria from the gut are migrating through the gut lining and entering the bloodstream. We have samples of taxa found in the mouse gut and in the blood. How can we tell what proportion of the blood sample consists of taxa originating in the gut, and what proportion comes from other, unknown sources? In general, given a "sink" sample of taxa and several possible "source" samples, how can we determine how the sources are mixed in the sink?

I will introduce two implementations of a Bayesian hierarchical statistical model used to solve this problem: the open-source software SourceTracker2 and my implementation of the model in JAGS. I will discuss details of the model before comparing the two implementations with results from a simulation study. Lastly, I will describe the statistical analysis of the mouse data and our findings.

1 Practical Application: Knockout Mice and Gut Bacteria in the Blood

This investigation of a Bayesian hierarchical model for microbiome mixing was motivated by a practical application: detecting gut bacteria in the blood of knockout mice. A study was performed on 4 cohorts of mice: mice with the ACE2 gene knocked out, diabetic-model mice ("Akita" mice), mice with both the ACE2 knockout and the diabetic variant, and "wild-type", or unmodified, mice. The diabetic and ACE2 knockout mice had health problems that were hypothesized to be caused by gut bacteria migrating through a weakened gut lining into the bloodstream and causing infection or illness. Given samples of the microbiome of the mouse gut and plasma for each group of mice, how can we tell what proportion of the plasma microbiome likely originated in the gut?

The metagenomic data used in this application comes in a non-normalized[1] OTU table.[6] OTU stands for Operational Taxonomic Unit. The data originally consists of a large list of gene sequences. Each sequence is compared to known type of microbe to match it to an OTU, which can be a species of microbe, but may also be a subspecies or conversely a family or genus, depending on how accurately the set of genes can be classified. The OTU table consists of counts of sequences assigned to each observed OTU across each sample. An example OTU table is shown in Table 1.

[1] Taxa found in each sample are treated by this model as discrete events; therefore levels of taxa in each sample must be integers.

OTU	Gut Sample	Blood Sample
Gracilibacteraceae family sp.	103	56
Clostridiales order sp.	0	62
Bacillales order sp.	45	3
⋮	⋮	⋮

Table 1: Head of an example OTU table.

In this application, the gut sample is a "source" genetic sample, and the plasma sample is a "sink" that we believe consists of bacteria mixed from one or many sources. We are particularly interested in microbes from the gut source, but we also have microbiome data from the mouse eye, brain, and bone environments. These can be treated as additional sources. We want to know what the proportions of those sources are in the sink sample. Those proportions will be described using a vector. If we test only for gut bacteria, the vector will have two components:

(p = proportion of gut bacteria in sink, $1 - p$ = proportion of other bacteria in sink)

We may also have more than one "known", or sampled, source to choose from. This would be the case if we concurrently tested for proportions of gut, eye, bone, and brain microbes in the plasma data. We can also introduce more than one distinct unknown source into the model. In general, we want a vector describing the likely proportions of *each* of the n known and j unknown sources in the sample:

(Source 1 = p_1, source 2 = p_2, ... source n = p_n, unknown source 1 = p_{n+1}, ... unknown source j = p_{n+j})

where $\sum p_i = 1$.

I will describe a Bayesian hierarchical model used to estimate this vector. I will begin by discussing advantages of Bayesian statistical methods. Then I'll describe the statistical model used and the two software implementations I looked at in this research. Lastly, I'll review results of simulation studies investigating the model and the programs that implement it and results of this mouse study.

2 Advantages of Bayesian Statistical Methods

There are two main schools of thought in the field of probability: frequentist and Bayesian. Each standpoint leads to a different framework of statistical inference. The Bayesian framework has several advantages over frequentist methods for this application.

Frequentist statistical methods perceive probabilities as limits of relative frequency of an event in a large number of trials. The probability of an event is fixed and unknowable, but we can estimate the number by looking at how often the event happens in some sample of trials. The output of a frequentist data analysis is usually a p-value. To get a p-value, the researcher comes up with a null hypothesis and creates a statistical model describing the data that would be observed *if* that hypothesis were true. They then evaluate the probability of the observed data, or data "more extreme" than what was observed, in that

statistical model. The final output is the probability of the observed data or more extreme data in a hypothetical model matching the researcher's null hypothesis; if that probability is low the null hypothesis is rejected. Such an analysis only rules out one possible model out of the infinite number that could describe reality. For example, in the case of finding the mean of a normal distribution, the researcher might conclude that the mean is not 0, but this gives no information on whether the mean is actually 2, 8, 1,000, -5, or any other number.

A frequentist approach to the mouse gut data might be to propose the null hypothesis that the proportion of gut bacteria in the bloodstream is 0. The observed data would then be compared to the range of possible outcomes expected if the true proportion of gut bacteria in the bloodstream really was 0. If the observed data would be very unlikely in that scenario, we could reject the null hypothesis that the proportion of gut bacteria in the bloodstream is 0, effectively concluding that there is *some* proportion of gut bacteria contributing to the bloodstream sample. However, the true proportion of gut bacteria in the bloodstream is still unknown, and we have not distinguished between the case where the bloodstream includes 10% gut bacteria and the case where the bloodstream contains 40% gut bacteria. Alternate hypothesis tests exist – we could, for example, test to see if the level of gut bacteria is *different* in a knockout mouse blood sample as compared to a genetically normal mouse blood sample – but all such hypothesis tests suffer from similar epistemological problems.[1]

Obtaining a very low p-value also does not conclusively rule out the model that produced it. Perhaps the p-value for the observed data under that model is 0.05, but the p value for every other possible model would be even lower, say 0.001. In that case, although we obtained a low p-value that seems to suggest that our tested model is unlikely, it's actually the *best* fit for the data among all possible models. Frequentist statistical methods usually fail to take this relativity of p-values into account.

An additional problem with the calculation of p-values is the definition of "more extreme" data. p-values can depend on factors outside the data that, intuitively, should not affect the statistical analysis, such as the stopping rule for data collection. Suppose two researchers collect data by flipping coins and analyzing the results to figure out if the coin is fair by testing the hypothesis that the true proportion of heads flips is .5. Researcher 1 decides they will collect data until they encounter 3 heads in a row, whereas Researcher 2 decides to record 10 flips and then stop. If both researchers flip H, T, T, H, T, H, T, H, H, H, they will have the *same* observed data, but *different* sets of data more extreme than what they observed: the observation H, H, H is in Researcher 1's set of more extreme data, but not in Researcher 2's, and likewise, H, H, T, H, H, T, H, H, T, H is in Researcher 2's set and not Researcher 1's. Despite having observed the exact same behavior of the coin, their p-values for testing if the true proportion of heads is over 50% will be different.[5, 1]

The applicability of p-values is limited. Most researchers are not actually interested in the probability of getting their observed data under some model (they already have their data, after all!). What they actually want is an estimation of which *model* best describes the real world. This is where Bayesian statistics come in.

In the Bayesian paradigm, instead of representing the relative frequency of an event over many trials, a probability represents a *degree of belief* that an event will take place or that a statement of fact is true. Someone's degree of belief in a fact is knowable and changes based on new data they encounter. The goal of Bayesian statistical method is to start with

some set of beliefs and update them based on data until they correctly anticipate real-world events (they match the true probability or frequency of events). In scientific applications, the "beliefs" we update are in the form of parameters that describe statistical models. For example, if the model were that values were drawn from a normal distribution, estimated parameters could include the mean or standard deviation. Bayesian methods take as input a *prior* value for that parameter, which is an educated guess as to its value. Prior values can be based on past or preliminary research, or they can be uninformative, such that every possible parameter value is equally likely. The prior value is then updated based on data until it closely matches behavior observed in the real world.[2, 1]

Priors are the most contentious element of Bayesian data analysis because they are specified subjectively. However, although in Bayesian data analysis priors are specified explicitly, prior beliefs affect any scientific endeavor, including those conducted using frequentist methods. The frequentist statistician chooses their null and alternative hypotheses based on prior beliefs. A scientist may choose to undertake a particular investigation because they believe that their results will be statistically significant – that is a prior belief!

A scientist's interpretation of the results of their statistical analysis may also be colored by their priors, whether they were specified explicitly or not. XKCD comic author Randall Munroe posits a hypothetical scenario in his comic shown in Figure 1 that illustrates the importance of priors to statistical analysis.

The frequentist statistician takes the null hypothesis that the sun has not exploded, finds that the machine would output "yes" only $\left(\frac{1}{6}\right)^2 = \frac{1}{36}$ of the time in this scenario, and rejects the null hypothesis because the observed data is unlikely given that the sun has not exploded, concluding that the alternative hypothesis must be true. The Bayesian statistician makes these same statistical observations, but *also* takes into account their belief that *the sun is extremely unlikely to have exploded in the first place.* Taking that into account, it's clear that the both-dice-come-up-6 scenario is far more likely than that sun-exploded scenario, so the Bayesian statistician is willing to bet against it.

As shown by this parable, priors reflect the reality that beliefs build up cumulatively from many sources of data. Priors allow us to build on past scientific achievements by using other scientists' and statisticians' results as a foundation on which to build our own findings. They can adjust our confidence that our data resulted from random chance rather than systematic attributes of the world, as well as allowing us to take other scientists' findings as our priors and refine their results further instead of being forced to statistically start from scratch each time we collect new data.

Historically, there have been serious computational limitations to Bayesian data analysis. Bayes' Theorem can be difficult to solve analytically depending on the form of data and priors used in Bayesian inference. As a result, even moderately-sized datasets can require computationally intensive numerical integration or Monte Carlo-Markov Chain methods to analyze. [4] Although historically this made Bayesian data analysis inaccessible for the majority of researchers, increases in computing power have made it possible to perform Bayesian data analysis on consumer laptops with the help of software such as JAGS.

Bayesian data analysis has several advantages for this application. It has a stronger epistemological foundation and produces a more practical output, the full distribution describing our posterior beliefs about the parameters we are estimating. Performing a Bayesian analysis on this data required only a couple hours of runtime on a consumer laptop. Bayesian statistical methods were, overall, a better and more practical choice for this data set than

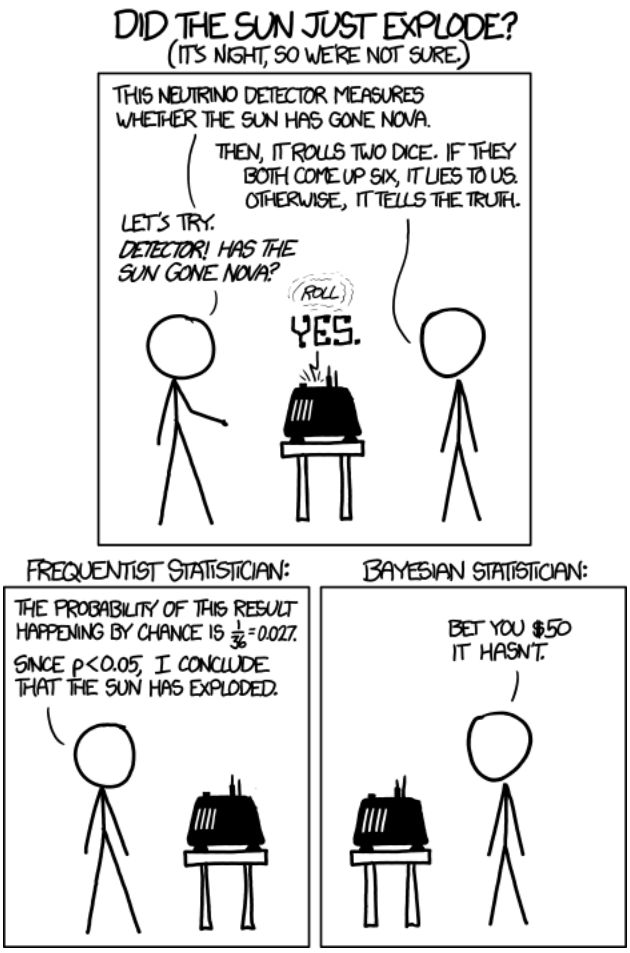


Figure 1: Frequentist vs. Bayesian methods of inference. (https://xkcd.com/1132/)

frequentist methods.

3 SourceTracker2 and JAGS

I investigated two different software implementations of a statistical model used to apply Bayesian data analysis to the problem of inferring the sources of a metagenomic sink sample, as in the mouse study application. These are SourceTracker2 [3], a purpose-built software package designed specifically for this model, and a program I wrote using the software called Just Another Gibbs Sampler (JAGS). First I will review the model of metagenomic mixing used to apply Bayesian data analysis to metagenomic data. Then I'll talk about attributes of each software implementation of the model. Lastly, I'll compare the two software implementations with a simulation study and discuss prior sensitivity in this model.

3.1 Theory

Both SourceTracker2 and the JAGS implementation of this problem use a hierarchical Bayesian statistical model that describes how genetic material mixes in samples to estimate

the likely proportions of a genetic sample that came from each of a number of sources. This statistical calculation could be useful in a variety of applications: the mouse study presented in the abstract, determining whether a genetic sample has been contaminated during processing, or even forensic applications.

Each genetic sample input consists of a list of sequences, each of which is mapped to an OTU. Some of these are "source" samples, samples from known environments. Others are "sink" samples, samples which may contain a mix of DNA from sources represented by the source samples or from other, unknown sources. In the mouse study, the sink sample is the sample of the blood microbiome and the primary source sample we are interested in is the fecal microbiome. Additional source samples coud include brain, bone, and eye data also collected from the studied mice. Microbes found in blood could also have originated from another source not included in the source samples, such as bacteria that entered the bloodstream through another mucous membrane such as the mouth.

Each sink sample is modeled as a set of sequences drawn at random without replacement. For each sequence, first, a source is selected. Then the sequence is drawn from among the taxa represented in the source. The model considers possible combinations of sources and reports which ones are most likely to have resulted in the sink sample actually observed. It also reports how much uncertainty there is in that combination of sources: if multiple different source combinations would be likely to produce the same sink sample, there will be a high level of uncertainty in the prediction.

Two statistical distributions are used in this model. The first is the categorical distribution. A categorical distribution represents the likelihood of drawing one element from a finite number of distinct categories. It is parameterized by a vector representing the relative likelihood of drawing from each category. Consider picking one coin at random from a collection of 3 pennies, 2 nickels, 1 dime, and 8 quarters. The probability of drawing each coin type is represented by the categorical distribution with parameter $\left(\frac{3}{14}, \frac{2}{14}, \frac{1}{14}, \frac{8}{14}\right)$ where the first element of the vector represents the probability of drawing a penny, the second the probability of drawing a nickel, and so on. In general, a categorical distribution describing the probability of one of K mutually exclusive events occurring has parameter $(p_1, p_2, ...p_K)$ with $\sum p_i = 1$, where p_i is the probability of event i. The parameter of the categorical distribution describing the probability that some taxon in the sink sample came from each of the sources is the one we want to infer.

The second distribution used in this model is the Dirichlet distribution. In this case, the Dirichlet distribution is used to describe uncertainty about the true parameter of a categorical distribution. Consider again picking one coin at random from a collection of coins. This time, though, we don't know exactly what coins are in the collection, only that we have already drawn (and replaced) 1 penny, 1 nickel, 1 dime, and 1 quarter. How can we describe the categorical distribution from which we're drawing the coins? The parameter for the categorical distribution is now unknown: our best guess so far is $\left(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}\right)$, but it could also plausibly be $\left(\frac{2}{6}, \frac{1}{6}, \frac{2}{6}, \frac{1}{6}\right)$, $\left(\frac{8}{35}, \frac{6}{35}, \frac{2}{35}, \frac{19}{35}\right)$, or any other combination of coins with at least one coin from each category. Still, the distribution is *more* likely to be fairly even across coin types, rather than 99% quarters plus one of each of the other coins, based on what we have seen so far. What can we say about the parameter of the categorical distribution if we can't describe it exactly?

Instead of specifying it as we did above with a particular vector, we can describe the possible parameter vectors with a Dirichlet distribution. Like the categorical distribution,

a Dirichlet distribution describes discrete events or categories. It is parameterized by a vector with the same number of components as there are events. Unlike the categorical distribution, the components of the vector need not be probabilities; they can be any positive real number. The Dirichlet distribution is used here to describe uncertainty about the true parameter of a categorical distribution. The relative magnitudes of each component of the Dirichlet distribution's parameter indicate the mean categorical distribution parameter we would expect to see. This is shown in Figure 2, which displays Dirichlet distributions with 3 categories, in the bottom row. From left to right, these are Dirichlet distributions parameterized by (6, 2, 6), (14, 9, 5), and (2, 6, 11). As the relative magnitudes of each component change, the probability mass favors categorical distribution parameters matching those relative magnitudes. In the first case, the peak of the probability mass would be at $\left(\frac{3}{7}, \frac{1}{7}, \frac{3}{7}\right)$.

The higher the *absolute* magnitude of the components, the more *concentrated* the probability density is on that guess. This is shown in the top row of Figure 2. All of these Dirichlet distributions are symmetric, with all components of their parameters equal, but the magnitude of the components grows larger from left to right, from 1.3, to 3, to 7. The distribution parameterized by (7, 7, 7) represents higher confidence that each category or event is about equally likely than the distribution parameterized by (1.3, 1.3, 1.3). The latter places more likelihood on the true categorical parameter strongly favoring one event (represented by the corners of the simplex), though its peak is still at the categorical parameter assigning equal likelihood to each event.

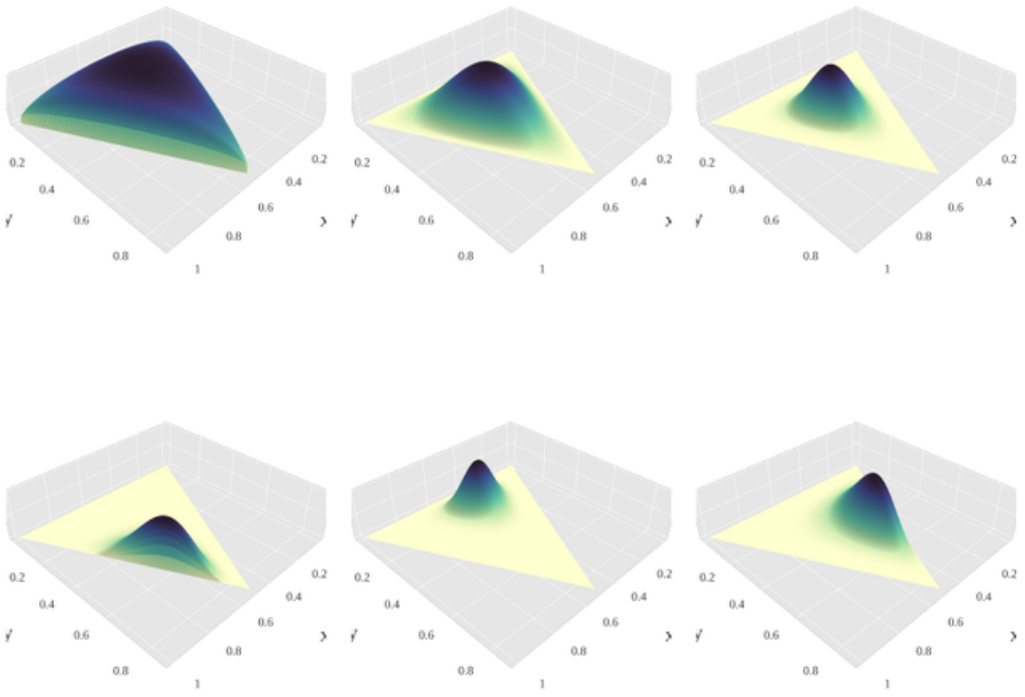


Figure 2: Dirichlet distributions parameterized by, clockwise from upper left, (1.3, 1.3, 1.3), (3, 3, 3), (7, 7, 7), (2, 6, 11), (14, 9, 5), and (6, 2, 6). By Empetrisor under CC BY-SA 4.0, from Wikimedia Commons.

Returning to our coin example, the Dirichlet distribution parameterized by $10^6, 10^6, 10^6, 10^6$) would

represent virtual certainty that the true parameter of the categorical distribution is $\left(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}\right)$ and each coin type is equally abundant in the bag. By contrast, the Dirichlet distribution parameterized by $(1, 1, 1, 1)$ is *uniform* over every possible categorical parameter. It says only that there is a nonzero probability of drawing each type of coin, but places equal probability on categorical distributions parameterized by $\left(\frac{13}{16}, \frac{1}{16}, \frac{1}{16}, \frac{1}{16}\right)$ and $\left(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}\right)$. To finish introducing Dirichlet parameters, what would the parameter $(\frac{1}{10}, \frac{1}{10}, \frac{1}{10}, \frac{1}{10})$ indicate? This parameter has low concentration; all of its components are small. Parameters with low concentration favor sparse categorical distributions, made up of mostly one event. These would have high densities of probability mass in the corners of the triangles shown in Figure 2, where most of the categorical distribution is made up of one event, and low density of probability mass in the middle of the triangle where each event has similar probability of occurring. Parameters with high concentration favor dense categorical distributions, where many events are all likely; they concentrate probability mass in the middle of the triangles in Figure 2.

In this case we have a categorical distribution whose vector is drawn from the Dirichlet distribution parameterized by $(1, 1, 1, 1)$. This Dirichlet distribution says that our best guess so far is that one is equally likely to draw each of the coin types from the coin collection, but the true distribution of coin types could be something different. It will place higher probability on parameters for the categorical distribution like $\left(\frac{1}{5}, \frac{2}{5}, \frac{1}{5}, \frac{1}{5}\right)$ that are similar to the Dirichlet parameter and lower probability on parameters like $\left(\frac{800}{803}, \frac{1}{803}, \frac{1}{803}, \frac{1}{803}\right)$ that are very different to what we have observed so far. If the categorical distribution is Categorical(A), then we would say $A \sim$ Dirichlet$(1, 1, 1, 1)$: the parameter A (which is a vector with four components) is drawn from a Dirichlet distribution which itself has parameter $(1, 1, 1, 1)$.

The Bayesian hierarchical model describing the relationship between the source and sink samples is constructed by describing the unknown true parameters of categorical distributions with Dirichlet distributions. [8] Consider drawing one sink sample from some sources. We model drawing each taxon in the sink sample as two sequential draws from categorical distributions. The first randomly selects a source from among all possible sources, including the unknown source. Each source has a different probability of being selected, which represents the proportion of gene samples it contributes to the sink sample. This is modeled using a categorical distribution. The second draws a taxon from the selected source, again modeled by a categorical distribution. The particular categorical distribution depends on the source: each source has a unique microbiome with different taxa in varying proportions.

This process is shown in Figure 3. Each box represents a source, with the grey, empty box representing the unknown source from which we have no data. Each colored ball represents a taxon. This model of creating the sink sample by drawing from different sources is what we will reverse engineer to get $P(\text{Data}|\text{Parameters})$: given the parameter describing the proportion of each source in the sink sample and the distribution of taxa in each source, the probability of generating the observed list of taxa via this process can be calculated.

The formal specification of the model as it is used in both software packages is as follows. The sample contains n sequences, each mapped to a taxon. These sequences are $x_1, x_2, ..., x_n$. The set of taxa is modeled as having been drawn from sources in the set V. The sources are $v_1, v_2, ..., v_{|V|}$.

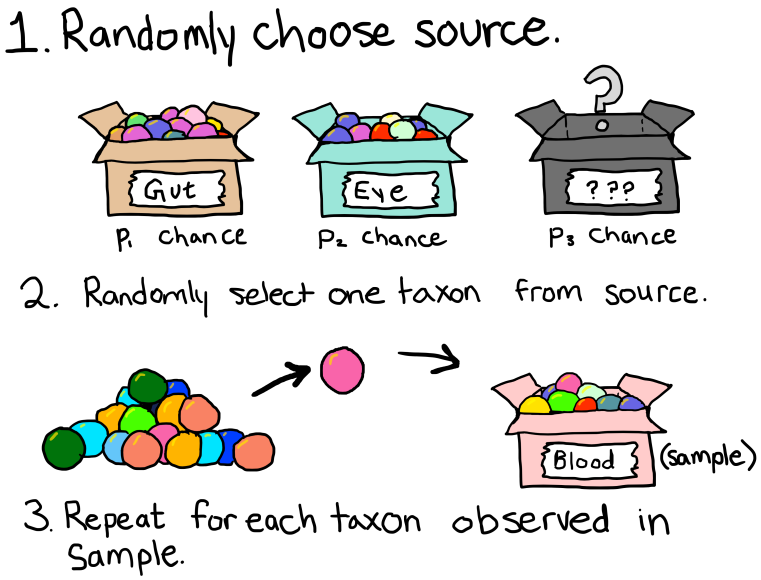


Figure 3: Drawing taxa from sources into a sink sample.

First, the parameters describing the distribution of taxa in each source are updated. There are $|V|$ such parameters (one for each source), A_1 through $A_{|V|}$. This is a "training" step where we teach the model the taxa distributions for each source, which are later used to estimate the sources in the sample. We do not pass each source directly as a categorical distribution because we do not know exactly what taxa are in each source; we have only samples from each source, rather than population data. Instead, we model them as in the above coin example: they are unknown collections of taxa, and we have a random sample of which taxa are in each of them.

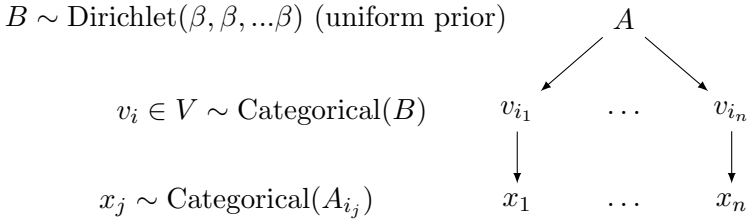
Each source starts with a small prior count in each taxon, α, and is updated based on the observed taxa in the source sample. α is adjustable depending on the level of uncertainty about the source taxa. A very high α will result in a more uniform categorical distribution representing each source; setting α to zero would result in the source categorical distributions matching the source samples exactly (so drawing from each source would be equivalent to drawing without replacement from the observed taxa). In summary, for each source v_i, the parameter of a Dirichlet distribution representing possible taxa distributions in the source, A_i, starts out as a uniform prior, $(\alpha, \alpha, ...\alpha)$, and is updated based on the source samples to match the taxa found in those sources plus some uncertainty reflecting the organisms not sampled.

$$
\begin{array}{lccc}
A_i \sim \text{Dirichlet}(\alpha, \alpha, ...\alpha) \text{ (uniform prior)} & A_1 & \dots & A_{|V|} \\
 & \downarrow & & \downarrow \\
\text{taxa in } v_i \sim \text{Categorical}(A_i) & \text{all taxa in } v_1 & \dots & \text{all taxa in } v_{|V|}
\end{array}
$$

Once the A_i describing each source are established, we can model drawing the sink sample from those sources. First, for each taxon in the sink sample (taxa 1 through n), we choose a source from V, the set of all sources. The source is drawn from a categorical distribution of all sources, which is parameterized by B, the true proportions of taxa from

each source in the sink. B is the parameter we are interested in; an estimate of B is output by the model. As with the distributions representing the taxa in each source, the parameter B is described by a Dirichlet distribution. The prior for that distribution is again uniform: $(\beta, \beta, ..., \beta)$. ($\alpha$ and β need not be equal and can be chosen independently depending on the statistician's preferences.)

Once the source v_{i_j} is chosen, one taxon from that source is drawn into the sink sample using the categorical distribution parameterized by A_{i_j}, which was updated earlier from the taxa found in that sink sample. This simulates drawing one taxon from the population represented by the source sample into the sink sample.



In summary, each sequence x_i is modeled as the outcome of two random draws. The first draws a source z_i from a categorical distribution describing the proportion of the sample drawn from each source. The second draws a taxon from within the selected source from a categorical distribution representing the taxa in the training data for that source.

The two Dirichlet distributions in the model represent uncertainty about the exact categorical distributions parameterized by A and B: B represents the exact distribution of taxa in each source v, and A is the parameter we are trying to estimate, the proportions of each source from which this sample was drawn. The A_i parameters are updated based on training data for each source. Lastly, an additional source is added into the model: the unknown source, which represents the proportion of the sink sample that was not drawn from one of the sources represented in the training data. Because we have no training data for the unknown source, the A_i representing it is not updated and remains a uniform vector, so any taxa could be drawn from the unknown source. (It is also possible to specify multiple unknown sources.) After the sources are established, B is updated based on the likelihood that the sequences actually found in the sink sample would have been drawn from some combination of sources.

This Bayesian hierarchical model is used to determine what proportion of sources best fits the observed sequences for each sample. Proportions of sources that would be likely to produce the observed sample are output. The result is a distribution representing probable source proportions, incorporating both our "best guess" of the source proportions and uncertainty about the true proportions. This is our *posterior*, or updated, distribution estimating B.

3.2 Bayes' Theorem and Markov Chain Monte-Carlo Methods

In order to come up with the probability distribution describing our beliefs about the source proportion parameter, we will apply Bayes' Theorem to the OTU data using the source mixing model described above:

$$P(\text{Parameters}|\text{Data}) = \frac{P(\text{Data}|\text{Parameters}) * P(\text{Parameters})}{P(\text{Data})} \tag{1}$$

Bayes' Theorem takes three inputs, P(Data|Parameters), P(Parameters), and P(Data), and outputs P(Parameters|Data), our posterior beliefs about the parameters given the data. Two of the inputs are relatively easy to obtain. P(Parameters) is specified by the investigator. This represents prior beliefs about the parameters being estimated, as described in section 2. P(Data|Parameters) can be obtained relatively easily using the model specification described in subsection 3.1. (As a toy example, consider finding the probability of flipping a coin three times and getting heads each time (data) given that the coin is fair (parameter): this is simply 0.5^3. The calculation in this case is very similar, but instead of having only heads and tails as outcomes, we have thousands of OTUs.)

The challenging part of this analysis, then, is actually P(Data), which represents the probability of seeing the observed data given *no* information about the parameters of our model whatsoever. This is calculated by integrating over all *possible* model parameters. This can be very computationally intensive: in this model, the parameter we are estimating, A, has as many dimensions as there are events in the categorical distribution. In this case the events are OTUs. In the mouse data set, there are 3,269 OTUs. Analytically calculating an integral of this dimension would be a very difficult task. Most computers perform integrals of this sort using quadrature, by dividing the integral into many tiny boxes. But consider using quadrature on a 3,269-dimensional integral. Even using as few as 100 "slices" on each axis, there are $100^{3,269}$ (or $10^{6,538}$) boxes for which we must calculate the volume. Even with efficient algorithms for numerical integration, this problem size is intractable. It also would not distinguish between the probability of the blood containing 0.001% gut bacteria and 1% gut bacteria. The actual proportions of contamination in genetic data can be very small, and this level of distinction can be important to the analysis: in the mouse study we detected proportions of a source on the order of 10^{-7}, which would correspond to 10,000,000 slices on each axis, worsening the situation by thousands of orders of magnitude.

Because of this difficulty, most Bayesian data analysis on real data uses Markov Chain Monte Carlo (MCMC) methods that avoid calculating this complex integral directly or using numerical analysis. Instead, MCMC methods use a random walk through possible parameter values to generate a random sample of parameter values. This random sample converges to the posterior distribution of the parameter that we are trying to estimate. Each random walk is one "chain". Multiple chains are generated and sampled to get an unbiased estimate of the posterior parameter distribution.

There are several MCMC methods used for Bayesian data analysis. For this application we use Gibbs sampling, which is a variant of the Metropolis-Hastings method that can efficiently generate this posterior distribution in a reasonable amount of time. Both of the software implementations of this model that I looked at are built on Gibbs samplers. To learn more about Gibbs sampling and MCMC methods used for Bayesian data analysis, see [4].

3.3 Implementation in SourceTracker2

SourceTracker2 uses Gibbs sampling to estimate the posterior distribution of A, the proportion of sequences from each source in the sink sample. The Gibbs sampler is written from scratch. It first assigns each sequence in a sink to a random source environment: z, the vector listing the source of each sequence in the sink sample, is randomly drawn from a uniform distribution. Then, each sequence x_i from the sink sample is reassigned to a new source environment (z_i is updated) via a random draw depending on the likelihood

that that taxa came from each source and the current proportion of the sample drawn from each source. This is done in a random order. The likelihood that a sequence came from a particular source v is

$$P(z_i = v|z^{\neg i}, x) \propto P(x_i|v) \times P(v|z^{\neg i}) \quad . \tag{2}$$

In other words, the probability that sequence x_i was drawn from source z_i, given the current vector of source assignments for all other sequences and the vector of sequences, is proportional to the probability that a draw from source v would produce taxon x_i *and* that the source v would be selected in the first place. $P(x_i|v)$ can be calculated from A_v, the categorical distribution parameter representing the taxa in source v. $P(v|x^{\neg i})$ can be calculated from B, the categorical distribution parameter representing the proportion of each source in the sequence.

Each time this process cycles completely through z – in other words, each time every source in the vector is probabilistically reassigned – z is saved as one random "draw" from the posterior distribution for z.The final output incorporates all of these draws from z to estimate the true proportions of each source in the sample. z is only saved after a burn-in period to allow the process to get away from the randomly assigned prior z. This entire process is restarted several times to provide an accurate estimation of the posterior distribution for z.

There are several particulars worth mentioning about this implementation of the model. In the SourceTracker2 implementation, the "training data" for the unknown sources are updated whenever a sequence is assigned to the unknown source. The A_i paramters for the unknown sources evolve over time, building a picture of what taxa are likely in that source. The sources with training data available are not similarly updated. The unknown source is also initialized by with higher prior counts in each taxon than the sources with training data available in order to avoid overfitting, if the default settings are used.

SourceTracker2 takes two inputs. The first is an OTU table in Biological Observation Matrix (BIOM) format. [6] The BIOM format is not human-readable, but the OTU table represented is formatted like Table 1. The second input is a metadata map. In metagenomic studies, a metadata map records information about each genetic sample, such as from which environment it was sampled. In this case, the map specifies whether each sample is considered to be a source or a sink. If environments are specified in the metadata map, samples coming from the same environment (for example, 3 different samples all representing bacteria swabbed from a reception desk in a hospital) can be automatically collapsed into one averaged sample, reducing the problem size and allowing conclusions to be drawn about that environment as a whole rather than about individual samples from that environment.

SourceTracker2 outputs estimated proportions of each source in the sample and standard deviations for those estimates. SourceTracker2 is written in Python 3 and can be installed and run in an Anaconda environment described on their GitHub[2] page. It can be run in parallel to improve computation time.

3.4 Implementation using JAGS

I wrote a different implementation of this same statistical model using Just Another Gibbs Sampler (JAGS). JAGS takes the specification of a Bayesian hierarchical model and uses it

[2]As of April 2018, https://github.com/biota/sourcetracker2.

to create a Gibbs sampler that can update the parameters of the model based on observed data. I wrote a model specification for this model of genetic source mixing that is compatible with JAGS, as well as a set of functions that take an OTU table (*not* in BIOM format) and a metadata map as input like SourceTracker, then run the data through JAGS and report results.

Unlike SourceTracker2, JAGS outputs the actual samples from the posterior distribution of the parameter being estimated. For this application, the output is a list of possible parameters for the categorical distribution of sources in the sink sample. This represents the proportion of each source in the sink. These can be graphed to visualize uncertainty in the estimate of sources and to evaluate the MCMC process. The output can also be used for data analysis such as confidence intervals and hypothesis testing; the samples can be used to estimate, for example, the degree to which we believe that the proportion of gut bacteria in the blood is over 5%.

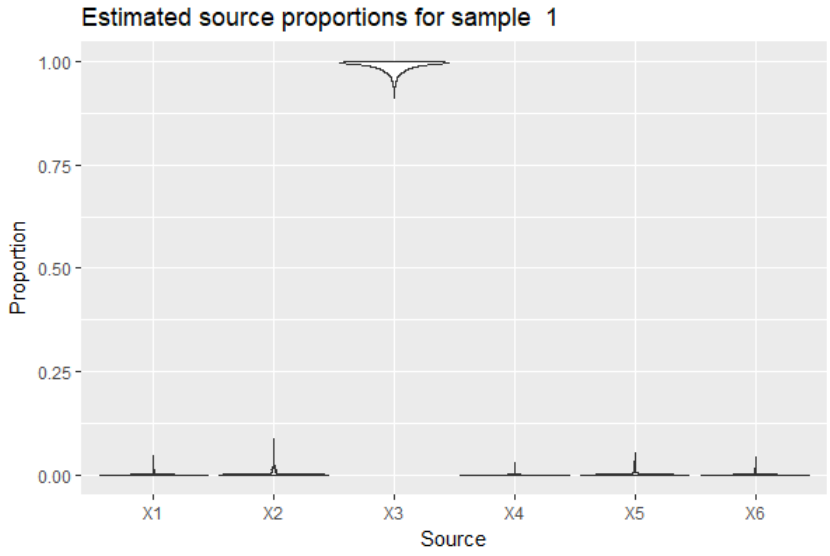


Figure 4: Example violin plot showing distribution of source proportions in sampled parameters. The model has determined that sample 1 was most likely drawn almost entirely from source 3. The identities of each source from the map (the source IDs corresponding to $X1$, $X2$, etc. in the graph) are output to the console.

High variability in source proportions among sampled parameters could indicate that the output is unreliable. Additionally, autocorrelation between sampled parameters can indicate problems with the Gibbs sampling process. These can both be evaluated using the full distribution of sampled parameters.

One feature included in the JAGS implementation of this model that does not exist in SourceTracker2 is the ability to run a model for an "adaptation" phase. [7] This runs a JAGS model for an initial sampling phase during which the software updates model settings to maximize the efficiency of the model. The model might change its step size or other internal parameters during this phase. This reduces the chances of getting a bad chain during the MCMC sampling process, which could be an inefficient random walk through the parameter space with high autocorrelation, or a chain that gets "stuck" in a non-representative region of the parameter space, etc. I used the adaptation phase when comparing the JAGS

implementation of the model and the SourceTracker2 implementation and did not notice any differences in performance, but it's possible that in some scenarios the adaptation phase would confer an advantage to the JAGS implementation over SourceTracker2.

The JAGS implementation currently lacks a command-line interface. Some data manipulation that SourceTracker2 performs automatically (such as converting BIOM files to OTU tables and collapsing multiple samples from the same environment into one averaged sample) must be handled by the user.

4 Simulation Study Results

I evaluated and compared the performance of the JAGS program and SourceTracker2 by generating sources, simulating mixing them into the sink sample, and comparing the true proportions of each source mixed into the sink to those estimated by Bayesian inference.

4.1 Data Generation

I generated small OTU tables for the simulation study. I first generated OTU counts for the source samples, then drew the samples from the simulated sources or from an "unknown" source.

To generate OTU counts for source samples, I set one unique OTU in each source to a high count (for example 500). Then I randomly generated OTU counts for the rest of that sample using the `rmultinom` method in R, with an average count for each OTU lower than the high count (for example 400). How easy it is to distinguish the vectors from each other depends on the difference between the high and low count. If the high and low count are very close, the vectors are difficult to distinguish; if the low count is very low compared to the high count, it's easy to pick out the sources of each sample just by looking at the OTU table.

After sources are generated, they are then "sparsified" by randomly setting some proportion of the OTUs to 0. This mimics real metagenomic data, which tends to be sparse.

Sink samples are then drawn from the generated sources or from the unknown source. For each sink, a source is selected randomly and 200 OTUs are randomly drawn from a multinomial distribution parameterized by the source vector into the sample vector. This can be repeated and the new OTUs added to the sample vector to simulate a sample that has multiple sources. A sink sample can also draw multiple times from the same source, so the proportions of sources contributing to the sink are not necessarily uniform.

Alternatively, if the user selects this option, sink samples can be drawn from the unknown source. In this case, OTUs are drawn at random from a multinomial distribution parameterized by a random draw from a uniform distribution. The multinomial distribution created this way is randomly high in some OTUs and low in others, making it possible to distinguish it from the existing source samples.

The sinks inherit some sparsity from the sources from which they are drawn, although if a sink is drawn from more than one source it will usually be less sparse than its sources.

	Median	Mean	SD
JAGS	0.99969	0.99606	0.03589
ST2	0.99980	0.99511	0.05755

Table 2: Median, mean, and standard deviation of r^2 in simulation study testing the performance of SourceTracker2 and JAGS.

4.2 Results

I tested both software implementations of the model to evaluate their performance on simulated datasets and to compare their performance with each other. The simulated datasets each had 5 source samples and one sink sample that was drawn from up to 5 of the sources. Each source OTU had a 20% chance to be set to 0 (sparsified). There were 20 OTUs generated. The sources were designed to be challenging to distinguish, with the high OTU count at 50 and the low OTU count at 45. Each program was run with 4 chains or restarts, 500 burn-in iterations, and 500 posterior parameter samples taken per chain. α, the prior count of each taxon in each source, was set to 1, and β, the prior count of taxa from each source in the sink sample, to 0.1.

1,000 simulated datasets were generated and the source parameter estimated for each of them. Then the estimated source parameters were compared with the true source parameters for each dataset by calculating r^2 between the two vectors. Higher r^2 values indicate a better estimation of the source vector.

Both programs had very high r^2 levels, close to the theoretical maximum of 1. These are summarized in Table 2. Performance was nearly indistinguishable between the two programs. r^2 values on these simulated datasets are summarized in Figure 5.

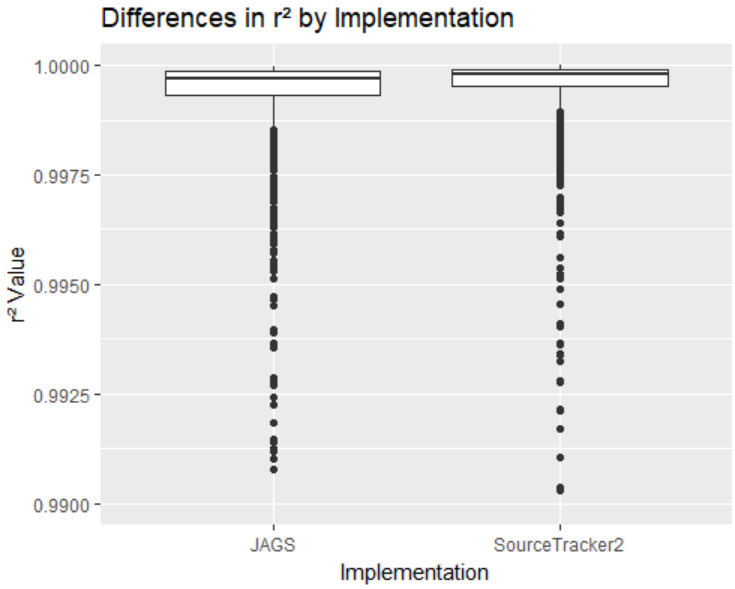


Figure 5: Boxplots comparing r^2 between JAGS and SourceTracker2.

I also investigated prior sensitivity for both α and β. α is the value of each component of the prior vector corresponding to the distribution of taxa in each source. β is the value

of each component for the prior corresponding to the distribution of sources in the sink sample. Changing the values of α and β change prior beliefs about how *concentrated* these distributions are. A high α indicates a smoother distribution over all taxa; we believe that a source will probably contain most of the taxa in the list, instead of being comprised of only one or two OTUs. Similarly, a high β indicates a smoother distribution over sources in the sink sample. Higher β corresponds to a prediction that the sink will be a mix of many source samples. A lower β corresponds to a prediction that the sink will be composed of just one or two samples.

Results are summarized in Figure 6.

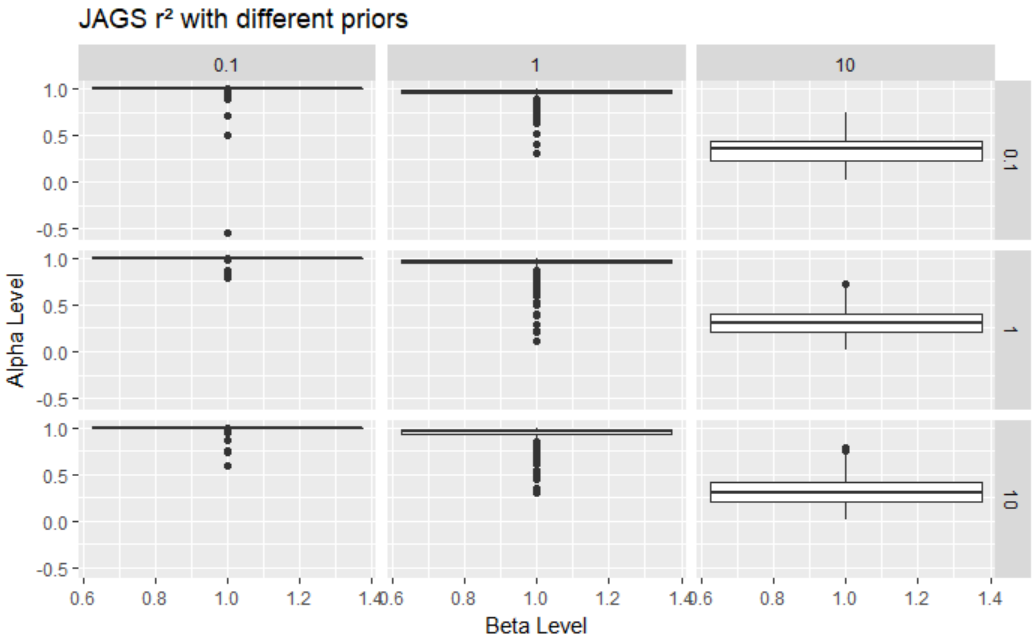


Figure 6: Boxplots comparing r^2 with different levels of α and β.

Although the model is robust to changes in α in the range I investigated, changes in β drastically effect r^2. The mean r^2 with $\alpha = 1, \beta = 0.1$ is 0.997, compared to 0.312 when β is increased to 10 and α is held constant.

The default setting for β in SourceTracker2 is 10, which is problematic given these results. Higher β levels tend to output too smooth of a distribution of source probabilities; with higher β levels, the model is underconfident. This hurts the accuracy of the results. At lower β levels, the model gives more specific predictions that are also more accurate. When this model is used to detect contamination of a sink sample – for example, small amounts of gut bacteria that migrate into an environment dominated by blood bacteria, or contamination from researchers' hands or surfaces that enters a genetic sample during processing – it makes sense to set $\beta < 1$; we expect most of the sample to be from one source with a relatively small amount of contamination from another source.

It is worth noting, however, that the lowest β level in this simulation study, 0.1, had the highest mean and median r^2 values but also the lowest minimum. In general, minimum r^2 increased from highest to lowest β: $\beta = 10$ had minimum r^2 of around 0.01, and $\beta = 1$ had minimum r^2 around 0.2. However, for the different levels of α, $\beta = 0.1$ had minimum

r^2 of 0.58, 0.79, and -0.54. This negative r^2 value[3] indicates that the predicted source parameter for this particular dataset was a worse fit for the true source parameter than a constant vector would have been. This value was an outlier, as indicated by the boxplots; the first quartile of r^2 for this combination of parameters started at 0.999. But this extreme outlier is still concerning. Although decreasing β improves median and mean r^2 of the predictions and also improved the minimum in most cases, it may simultaneously increase the possibility of a dramatically bad prediction like this one.

This bad prediction was probably influenced by the difficulty of distinguishing sources in this dataset. I designed the simulated datasets to challenge the program by making all of the sources similar to each other and close to a uniform distribution. This dataset also contained only 20 OTUs, whereas real datasets would contain thousands of OTUs. The higher number of OTUs means the software has more information to use when predicting the true sources of a sample.

For this simulation study, I tested SourceTracker2 and JAGS on datasets where sink samples were drawn from known, rather than unknown, sources; performance may be less accurate when the sink is composed primarily of unknown sources. In addition, any source present in the sink comprises 20% or more of the sink sample. In real datasets, a source may contribute a very tiny amount of OTUs to the sink sample. Further simulation studies should evaluate performance of the model when samples are drawn from unknown sources and when sources are present in the sink sample in very small proportions to see if the model's performance suffers under these more realistic conditions.

5 Mouse Study Results

To analyze the mouse OTU data, I used the JAGS implementation of the metagenomic mixing model with the following settings:

- 3 MCMC chains
- 500 adaptation iterations
- 500 burn-in iterations
- 500 samples of A taken per chain
- $\alpha = 0.1$
- $\beta = 0.1$

I input samples of mouse gut bacteria as the source for each mouse genotype and samples of mouse plasma bacteria as the sink. For each genotype, sample OTUs were averaged across a number of individual mice using integer division.

There are a few ways in which the model of gene mixing may not match the reality of the samples in this study. The most significant is that in the model, taxa are selected from each source into the sample proportionally to their abundance in the source. In the

[3]The r^2 used here is calculated as $1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}}$. This is r^2 measured between two vectors, rather than the r^2 used in linear regression. This can be negative (despite the name, and unlike in the case of linear regression, where r^2 is always positive) when the residual sum of squares is higher than the total sum of squares.

Mouse Genotype	Median Gut Bacteria Proportion
Akita (Diabetic)	0.0004
Akita + Knockout	0.0008
ACE2-KO (Knockout)	0.00000006
Wild Type	0.0000001

Table 3: Median gut bacteria detected in blood plasma microbiome samples from different mouse variants.

mouse study, taxa are hypothesized to be crossing through a mucous membrane inside the mouse. Some taxa (for example, smaller microbes) may be more easily able to cross the barrier than others. A filtering effect of the gut membrane may cause this statistical model to underestimate the proportion of the blood sample that came from the gut, because even taxa that really did migrate to the blood from the gut may not be proportional to their abundance in the gut sample. Proportions of gut taxa in the blood could change further due to the environment of the bloodstream; some bacteria from the gut may die in the plasma environment, or be killed by the mouse's immune system. These factors should be taken into account when considering the results of this study. The true level of uncertainty about the proportion of gut bacteria in the blood could be higher than that represented by the statistical output.

Using only the gut and blood data, we found evidence of higher gut contribution to the blood plasma microbiome in mice with the ACE2 gene knocked out *and* the diabetic Akita variant, as well as in the mice with only the Akita variant. Interestingly, the mice with only the ACE2 gene knocked out had a lower median proportion of fecal bacteria in the blood detected than the wild type (unmodified) mice, although levels in all variants were low (less than 1/10 of a percent). Median gut proportions are shown in Table 3.

Probability distributions for the true proportion of gut bacteria in the plasma samples are shown as violin plots in Figure 7. (Only gut proportions are shown because the remainder of the sample was assumed to come from the unknown source.) The shape of these distributions show why it's inappropriate to summarize the results with a point estimate for B and standard deviations for each component: these distributions, particularly for the ACE2-knockout-only and wild type mice, range from skew right to very skew right and are not normal distributions.

I then re-ran the analysis including bone, brain, and eye data from each mouse type as possible sources for the plasma samples along with the fecal data. Results are shown in Figure 8 and Table 4. These results are summarized by median; means and standard deviations are available in Table 5. Interestingly, with the addition of the rest of the metagenomic data, the source trends changed regarding the proportion of fecal bacteria in plasma for each mouse type: wild type mice now have the highest median and mean estimated fecal proportion in the blood plasma, followed by Akita, Akita-knockout, and finally ACE2-knockout-only mice. This is the opposite of what we expected to find in this data: the wild-type mice were expected to have the *lowest* amount of fecal contamination, with the modified mice more heavily contaminated due to a weakened gut lining.

We did not expect the addition of the bone, brain, and eye data to change trends in fecal contamination. From a biological perspective, the bone, brain, and eye microbiomes should not interact with the blood microbiome in the mice. The gut membrane is known

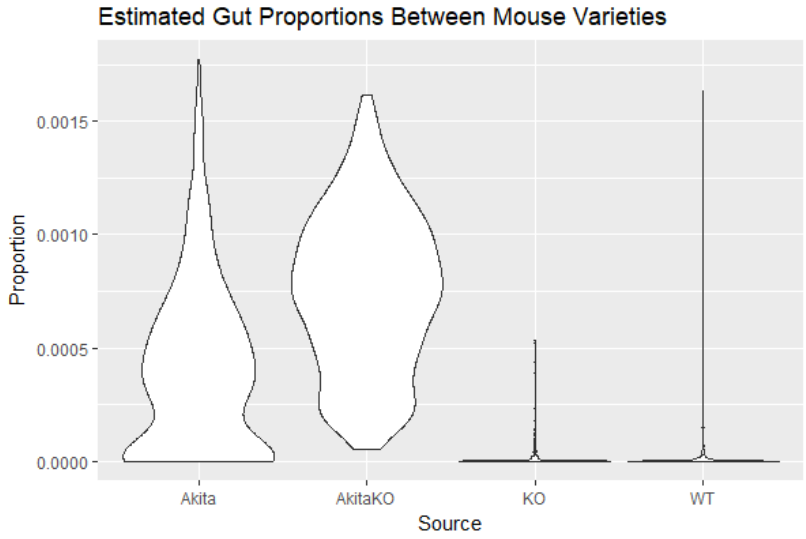


Figure 7: Gut Proportion Distributions for each Type

to be "leaky" and allow some microbes to pass through, but barriers between these other organs and the bloodstream are thought to be more robust. These reversed results may be due to cross-contamination between different organs during dissection, or contamination from lab surfaces and equipment. In future work, we will examine the microbiomes of each organ to determine if any of them have elevated OTU counts of likely contaminant OTUs. We may also re-run this analysis including data from possible contaminant sources, such as swabs from scalpels used for dissection, to rule out the possibility that these results are due to contamination. Lastly, we could re-run the analysis with priors that favor the unknown and gut sources in the blood, rather than priors that are symmetric between all sources, to reflect our prior beliefs about mouse physiology.

Source:	Fecal	Bone	Brain	Eye	Unknown
Akita	0.0005744871	0.0000033140	0.00000001210	0.0000011525	0.9992613
AkitaKO	0.0000026910	0.0000461146	0.00000000812	0.0000305988	0.9996856
KO	0.0000002138	0.0000001799	0.00000002323	0.0000010764	0.9999710
WT	0.0014323011	0.0000002483	0.00000001632	0.0000750365	0.9983532

Table 4: Source proportions in each averaged sample, including bone, brain, and eye data as well as fecal data. Akita = diabetic mouse model; KO = ACE2 gene knockout, AkitaKO = diabetic model + gene knockout, and WT = wild type (unmodified mouse).

6 Conclusion

In this paper I've shown the application of a Bayesian hierarchical model of metagenomic mixing to the task of distinguishing the sources of a genetic sample. In our motivating example, we were able to show our actual beliefs about the proportions of microbes from different sources found in plasma samples from genetically modified mice to gain a better understanding of the effects of the ACE2 gene. Although different subsets of the data

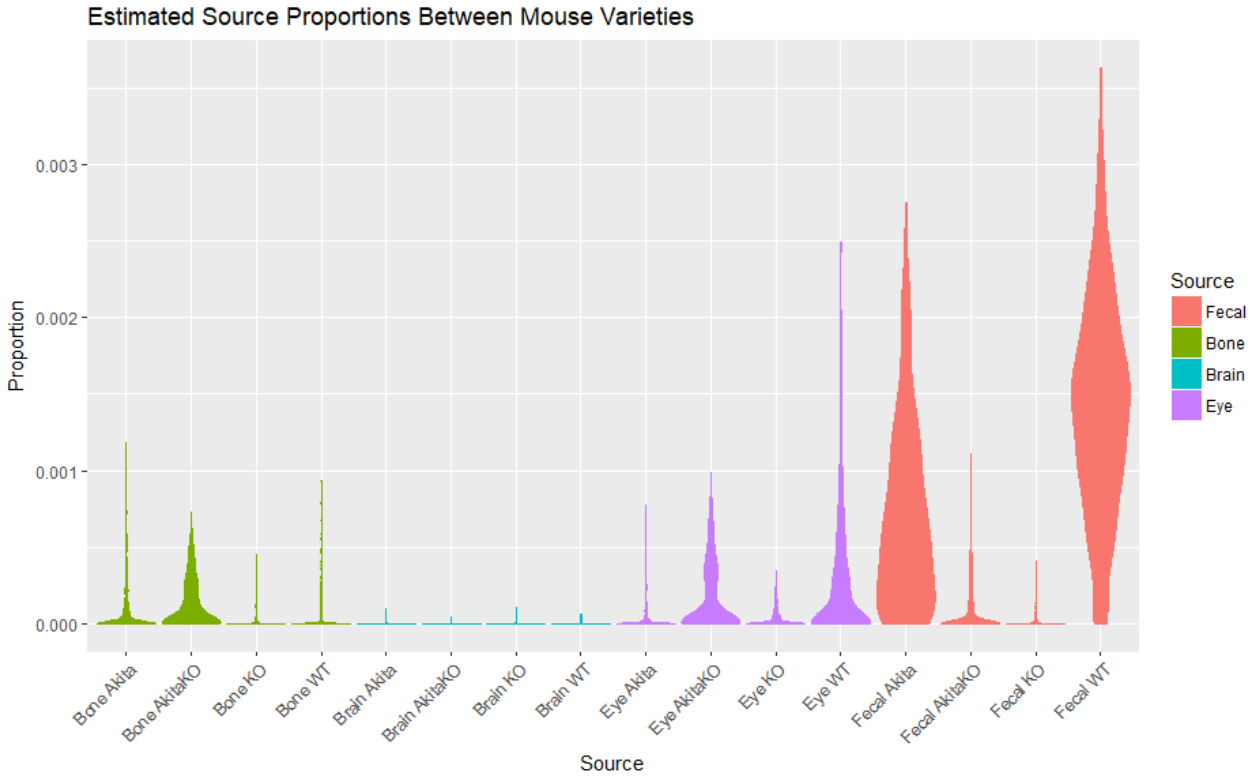


Figure 8: Source proportion distributions for each mouse type, including fecal, bone, brain, and eye data.

produced different results, the data do suggest some impact of the ACE2 and Akita mouse variations on what microbes end up in the mouse bloodstream. More information on the physiology of mice with these variations is needed to determine why these effects occurred and whether the reversal of the results with the full dataset was due to contamination.

I compared two software implementations of this statistical model, SourceTracker2 and the implementation I wrote using JAGS. The JAGS implementation has several advantages over SourceTracker2. The most significant is that by default it outputs the full distribution of the posterior proportion estimates for the different samples, rather than just a point estimate and standard deviations for each component of that vector. I've shown using simulated data that both pieces of software are extremely accurate in detecting the sources of a metagenomic sample even under difficult conditions where many very similar sources are mixed in the same sample. I've also shown that performance is prior-sensitive. The prior count of taxa from each source in the sink sample, β, should be set much lower than SourceTracker2's default of 10 to achieve optimal performance; I tested levels 0.1, 1, and 10 and found that 0.1 performed the best by far.

Bayesian statistics have important advantages over frequentist statistics. Chief among them is the fact that Bayesian statistics output descriptions of the real world, which are much more practically applicable than the p-values output by frequentist statistics. Historically, Bayesian statistical analyses have been far more computationally intensive and complex to implement than frequentist analyses, but advancements in computing power and the proliferation of free, open-source software designed to simplify Bayesian analysis, such as SourceTracker2 and JAGS, have made Bayesian statistics far more accessible to the average statistician and enable models like this to be run easily even on a consumer laptop. The advantage of being able to output distributions and graphics that describe one's actual beliefs about model parameters, rather than hypothetical statements under a null hypothesis *or* descriptions of sample data, make Bayesian statistical analysis the best choice whenever possible for understanding complex data such as metagenomic datasets.

References

[1] Zoltan Dienes. Bayesian Versus Orthodox Statistics : Which Side Are You On? *Perspectives on Psychological Science*, 6(3):274–290, 2011.

[2] Alexander Etz, Quentin F. Gronau, Fabian Dablander, Peter A. Edelsbrunner, and Beth Baribault. How to become a Bayesian in eight easy steps: An annotated reading list. Technical report.

[3] Dan Knights, Justin Kuczynski, Emily S. Charlson, Jesse Zaneveld, Michael C. Mozer, Ronald G Collman, Frederic D Bushman, Rob Knight, and Scott T Kelley. Bayesian community-wide culture-independent microbial source tracking. *Nat Methods*, 8(9):761–763, 2013.

[4] John K. Kruschke. *Doing Bayesian Data Analysis*. Elsevier, London, 2 edition, 2015.

[5] Dennis V Lindley. The Analysis of Experimental Data: The application of tea and wine. *Teaching Statistics*, 15(1):22–25, 1993.

[6] Daniel McDonald, Jose C. Clemente, Justin Kuczynski, Jai Ram Rideout, Jesse Stombaugh, Doug Wendel, Andreas Wilke, Susan Huse, John Hufnagle, Folker Meyer, Rob Knight, and J. Gregory Caporaso. The Biological Observation Matrix (BIOM) Format, 2012.

[7] Martyn Plummer, Alexey Stukalov, and Matt Denwood. rJAGS: Bayesian Graphical Models using MCMC, 2016.

[8] Stephen Tu. The Dirichlet-Multinomial and Dirichlet-Categorical models for Bayesian inference - Lecture Notes. pages 1–6, 2014.

	Source	Group	Median	Mean	SD
1	Fecal	Akita	0.0005745	0.0007075	0.0006002142
2	Fecal	AkitaKO	0.0000027	0.0000941	0.0001804391
3	Fecal	KO	0.0000002	0.0000187	0.0000487800
4	Fecal	WT	0.0014323	0.0014159	0.0007138426
5	Bone	Akita	0.0000033	0.0000754	0.0001559187
6	Bone	AkitaKO	0.0000461	0.0001253	0.0001604179
7	Bone	KO	0.0000002	0.0000228	0.0000594606
8	Bone	WT	0.0000002	0.0000390	0.0001059446
9	Brain	Akita	0.0000000	0.0000027	0.0000092190
10	Brain	AkitaKO	0.0000000	0.0000015	0.0000042509
11	Brain	KO	0.0000000	0.0000030	0.0000091548
12	Brain	WT	0.0000000	0.0000031	0.0000088401
13	Eye	Akita	0.0000012	0.0000561	0.0001296189
14	Eye	AkitaKO	0.0000306	0.0001626	0.0002193625
15	Eye	KO	0.0000011	0.0000340	0.0000588063
16	Eye	WT	0.0000750	0.0002541	0.0003701199
17	Unknown	Akita	0.9992613	0.9991583	0.0005761020
18	Unknown	AkitaKO	0.9996856	0.9996164	0.0003122462
19	Unknown	KO	0.9999710	0.9999215	0.0001014867
20	Unknown	WT	0.9983532	0.9982879	0.0006686837

Table 5: Source proportions in each averaged sample, including bone, brain, and eye data as well as fecal data. Akita = diabetic mouse model; KO = ACE2 gene knockout, AkitaKO = diabetic model + gene knockout, and WT = wild type (unmodified mouse).