

Statistical Techniques Practice: Gibbs Sampler.

1. Use Matlab to implement the Gibbs Sampler presented in lectures (the twins example). To do so you must do the following:
 - (a) (Set up the Sampler)
 - i. Create a vector '*data*', where the *i*th position contains the number of males (0, 1 or 2) in the *i*th pair (you will read the data into this vector).
 - ii. Create a vector '*identical*' of the same length as *data*, where the *i*th position is '1' if the *i*th pair are identical and '0' if they are not.
 - iii. Randomly initialize the vector *identical*. Make sure that any pair assigned to be identical does not contain both a boy and a girl!
 - iv. Create the variables *p* and *q* - the parameters we are interested in.
 - v. Create and assign values to the parameters *a*, *b*, *c* and *d* for our Beta priors for *p* and *q*. Note: this is where you are fixing your prior.
 - vi. Create variables *x* and *s* to hold the counts of the number of identical twin pairs and the number of boys (sort of - see the definition of *s* in your notes) respectively.
 - (b) (Update *p*) Write a routine to sample a new value of *p*, given that we know the vector *identical*:
 - i. Sum up the elements of *identical* and assign this value to *x*.
 - ii. Draw a new value for *p* from a $\text{Beta}(a + x, b + n - x)$ distribution.
 - (c) (Update *q*) Write a routine to sample a new value of *q*, given that we know the vector *identical*:
 - i. For each *i* calculate $\text{data}[i] \times (2 - \text{identical}[i])/2$, and sum these values. Assign the result to *s*.
 - ii. Draw a new value for *q* from a $\text{Beta}(c + s, d + x + 2(n - x) - s)$ distribution.
 - (d) (Update *identical*) For each pair (say the *i*th element of *data*):
 - i. If $\text{data}[i] = 0$: Draw a random number between 0 and 1. Set *identical*[*i*] to 1 if this random number is less than $p(1 - q)/(p(1 - q) + (1 - p)(1 - q)^2)$. Otherwise set it to 0.
 - ii. If $\text{data}[i] = 1$: Set *identical*[*i*] to 0.
 - iii. If $\text{data}[i] = 2$: Draw a random number between 0 and 1. Set *identical*[*i*] to 1 if this random number is less than $pq/(pq + (1 - p)q^2)$. Otherwise set it to 0.
 - (e) (Iterate the sampler) Enclose your routines in a loop which performs 1b, 1c, 1d sequentially. This is where you set the number of iterations to run the sampler for.
 - (f) (Record posterior values) Once the sampler has completed half its allotted iterations, keep a record of the values of *p* and *q* at the end of each iteration (store these values in two vectors - one for each parameter).
 - (g) (Graph Results) For both *p* and *q* plot a histogram of your results. Overlay this with a plot of your prior distribution for each parameter.

Test your program on the dataset *Twins.txt* which is available on website. It contains data for 200 pairs of twins where, for each pair, the number of males is recorded.

Choose parameters for your two beta priors and run the code for 2,000 iterations. Try varying your priors for *p* and *q* (by choosing different values of *a*, *b*, *c* and *d*). What are your best estimates for *p* and *q*? How does varying the priors effect your posterior estimates for *p* and *q*?

2. Real populations often consist of genetically distinct but unobserved subpopulations. This is called population structure. Understanding population structure is vital in many settings - particularly population genetic analyses. In this question and the next you will write a program which uses Gibbs sampling to learn about population structure from genetic data.

A copy of the file `xchromreal.txt` is available in the website. This file contains data from the X-chromosome of sixty male humans - some of them of African descent and the rest of European descent. Recall, for humans, females have two copies of the X-chromosome whereas men have only one X and one Y. For each individual, there is data recorded at 292 sites (loci) on the X-chromosome. At each site there are two possible alleles (called single nucleotide polymorphisms, or SNPs for short), labelled 0 or 1 and each individual has either a '0' or '1' at each site. We are going to use the frequencies of these 0s and 1s at each site to learn about the population structure.

Thus think of the data as a matrix, D , where the i th row lists the alleles of individual i at each locus, and the j th column refers to the alleles at locus j .

- (a) We will formulate a Gibbs sampler to assign individuals to either population. Suppose, for now, that we have sampled N individuals at L loci and we know that there are two populations represented in the sample. The 'missing data' are the whole population allele frequencies, for if we knew these we would be able to assign individuals to populations.

Write:

S_i for the population of origin of individual i , and S for the vector of the S_i .

p_1^j for the frequency of allele 1 in population 1 at locus j , and p_1 for the vector of the p_1^j .

p_2^j for the frequency of allele 1 in population 2 at locus j , and p_2 for the vector of the p_2^j .

Thus, our parameter of interest is $(S_1, \dots, S_N, p_1^1, \dots, p_1^L, p_2^1, \dots, p_2^L)$.

- i. Suppose we only have data at one locus i.e. $L = 1$. For one individual.
 - Formulate the Gibbs sampler in this case. You will first need to specify an appropriate prior distribution for p_1^1 and p_2^1 . (Why not use a Beta(a, b) distribution with $a = b = 1$.)
 - Assume you know the population frequencies of allele 1 for each population (i.e. p_1^1 and p_2^1).
 - If the individual has allele 1 at the locus, what is the conditional probability that he is from the first subpopulation?
 - What is this probability if the individual has allele 0?
 - Hence, what is the conditional distribution of S_1 given D ? (The data D in this case is just a 1×1 matrix, aka a number.)
 - What is the conditional distribution of p_1^1 given S_1 .
- ii. Still consider the case of a single locus ($L = 1$) but now with N individuals.
 - Assuming S_1, \dots, S_N are known (i.e. we know which individuals are in which population), what is the conditional distribution of p_1^1 and p_2^1 .
- iii. Generalize the results obtained above for one locus to all L loci. (Adopt prior distributions for the different p 's which are independent, and assume that the population frequencies of the alleles at locus j are *independent* of the population frequencies of the alleles at locus k if $j \neq k$.)
 - Assume you know the population frequencies of allele 1 for each population (p_1^j and p_2^j , for $j = 1, \dots, L$).
 - What is the conditional distribution of S_i , the population assignment of individual i .
 - Now, assume you know the population assignments of all N sampled individuals. What are the conditional distributions of p_1^j and p_2^j , for $j = 1, \dots, L$.
- iv. *When you have finished, see one of the tutors to get the questions for the next part of the class.*

- (b) In Matlab, write code to run the Gibbs sampler for the model you have considered in the previous questions, and you will apply it to the male X-chromosome data provided in `xchromreal.txt`.

You will need to have a vector S to store the population assignment of each individual. You will need vectors p_1 and p_2 to store the population allele frequencies. You will need a matrix with N rows and L columns to store the data. Finally, you will need a vector `pop1count` of length N to store the number of times individual i is in population 1. You will also need two vectors of length L , `onecount1` and `onecount2`, in which you store the number of allele 1 at site j in each population respectively. Finally, you will need two variables, N_1 and N_2 , which store the number of individuals assigned to each population.

Use a Beta(1,1) prior for the population allele frequencies (recall this is a uniform distribution on (0,1) so we are assuming no prior knowledge about the p_s^j).

To initialize the population assignments, randomly assign the individuals to either population (use a draw from a Bernoulli random variable with parameter 0.5 to do this - just draw from a uniform to do this in the usual way). Store these assignments in S . Now update N_1 and N_2 and `onecount1` and `onecount2`.

To update (and initialize) the population allele frequencies: for each locus j and population 1 say, draw from a Beta($1 + \text{onecount1}(j)$, $1 + N_1 - \text{onecount1}(j)$). The method is similar for population 2.

To update the assignment of individuals (the vector S): Say we are updating individual i . To calculate the unnormalized probability that individual i comes from population 1 we just take the product over all loci of the ‘relevant’ probabilities at each locus, where by ‘relevant’ we mean:

- If individual i has a 1 allele at locus j use p_1^j .
- If individual i has a 0 allele at locus j use $1 - p_1^j$.

Note: you *may* need to take the sum of the logs of the probabilities, then exponentiate to avoid underflow error.

The method is similar for the unnormalized probability that individual i comes from population 2.

To normalize these probabilities just divide by their sum. Then use a draw from a uniform(0,1) distribution to assign the individual to a population.

Do this for each individual, then update N_1 and N_2 and `onecount1` and `onecount2`.

Run this for lots of iterations (say 2,000 to start with, then possibly 20,000). Once you have iterated the procedure for half the iterations you should start keeping a record of how often each individual is in population 1, using the vector `pop1count`, where the i th position of `pop1count` is the number of times that individual i is assigned to population 1 once we start keeping records.

When you have finished the loop, use `pop1count` to assign individuals to populations. If individual i has spent the majority of iterations (from when we have kept records) in a population assign it to that population.

Keep a record of your final population assignments.

Now, consider individual 1. How ‘sure’ are you of the population assignment you have made for him? What do you base this on?

Finally, can you suggest a reason for not keeping records from the start?