# SDS 394 Project Update

Carlos Gillett (ctg576) & Kayla Schaefer (khs426)

Spring 2015

## 1 Topic Recap

We are making our project on Approximate Bayesian Computation, specifically looking at what has been deemed "Tiny Data". We are focusing on a concrete example for our project: estimating the number of socks a person has given a random sample of size $< 15$.

The ABC algorithm works as follows [1]:

1. Construct a generative model that produces the same type of data as you are trying to model. Assume prior probability distributions over all the parameters that you want to estimate.
2. Sample tentative parameters values from the prior distributions, plug these into the generative model and simulate a dataset.
3. Check if the simulated dataset matches the actual data you are trying to model. If yes, add the tentative parameter values to a list of retained probable parameter values, if no, throw them away.
4. Repeat step 2 and 3 to build up the list of probable parameter values.
5. Finally, the distribution of the probable parameter values represents the posterior information regarding the parameters.

## 2 Project Goal

Our goal was to let the user can input how many individual and paired socks were in the sample, what prior parameters you want to use, and how many processors over which to parallel. Then we run the ABC algorithm and return estimates for how many socks you have as singles, as pairs, and overall, along with the proportion of single socks. We also will create histograms for the probability distributions of those estimates.

# 3 Project Status

Currently, we have implemented the I/O for the program, allowing the user to either run a demonstrative example or set as many of the inputs as desired. An -h flag is also implemented to output a helpful description of all accepted inputs.

```
> ./example -h
options:
-u: count of unique socks
-p: count of paired socks (e.g. 1 pair = 2 socks)
-m: estimated amount of total socks
-s: error for total socks estimate (default is m / 2)
-n: ’small’ or ’large’ for proportion of paired socks
```

For the prior parameters, an estimated mean and standard deviation are accepted from the command line, and used to calculate the $p$ and $r$ parameters necessary for the negative binomial distribution over the total number of socks. These were derived from

$$\mu = \frac{r(1-p)}{p}$$

$$\sigma^2 = \frac{r(1-p)}{p^2}$$

Simple rearrangement and substitution gives us

$$p = \frac{\mu}{\sigma^2}$$

$$r = \frac{\mu p}{1-p}$$

For the prior on the proportion of paired socks, the user simply inputs either “small” or “large”, to set the starting median (either .9 or .95), as we believed that was more intuitive for the average user.

While we are still working on adapting the acceptance-rejection algorithm for parallel, we have constructed the simulation function. This function returns the desired number of samples from the prior distributions. It takes in the prior parameters, observed information, number of samples, and the pointer to a counter variable to keep track of the number of samples that match the observed data. The samples are generated using the GSL library’s random distribution functions as follows, where r represents a seed:

$$nsocks = gsl\_ran\_negative\_binomial(r, p, n);$$

$$pairProp = gsl\_ran\_beta(r, alpha, beta);$$

$$npairs = floor(nsocks * pairProp);$$

$$nodd = nsocks - 2 * npairs;$$

In addition, we have set aside code for writing all the data to files and to calculate and return the median values for each of the desired outputs.

# 4 Still to Complete

As referenced in the previous section, there are three main points we have yet to complete.

1. Finalize the parallelized simulation code
2. Code up plotting the four posterior distributions
3. Optimize the code using a profiler

We believe that we are well on our way to completing these three tasks before the final presentation, and especially before the final submission date. To address each point specifically:

1. The parallelization of the simulation will involve dividing the total number of samples evenly (or as evenly as possible) among processors. The array of sample results will be concatenated using the MPI_Gather function.
2. The figures will be generated using R. After writing the posterior samples to their own separate files, they will be read into R and plotted with histograms using ggplot2.
3. Profile using Tau. We have discussed some alternative looping methods for the sampling and acceptance-rejection sections of the code, which are likely to be the most expensive components.

# References

[1] Baath, R. (2014). Tiny Data, Approximate Bayesian Computation and the Socks of Karl Broman. Retrieved from http://www.sumsar.net/blog/2014/10/tiny-data-and-the-socks-of-karl-broman/.