2011 FALL BIOSTAT 615/815 Homework #6

- Due is Tuesday December 13th, 08:40AM (before the class starts).
- You need to both (1) hand-in your hard copy of the source code (using smaller font is fine), and (2) submit your source codes (ready to compile and run) in a zip or tar.gz compressed format by email.

Problem 1. Gibbs Sampler for Gaussian Mixture Models

Using the "ModelFittingData.txt" available at the class web page,

1. Implement the Gibbs sampling methods in C++ and fit a mixture of one, two, and three normal distributions to the data.
2. Using burn in period of 50,000 and thin interval of 5,000, compute the maximum likelihood, mixing proportions, mean, and standard deviation of each of the parameters using 10 million iterations (you will have approximately 2,000 thinned observations to compute the parameters)

   Below is an example output (using a different input data) when $k = 3$.

```
MLE   = 19135.8
pis   =  0.292729   0.207121 0.50015
means = -0.608487   4.98484  0.350046
sds   =  0.867151   1.03409  0.876139
```

3. Describe whether you observe reasonable estimates of parameters or not across three possible components. Add a small random noise (e.g. $\epsilon \in (-10^6, 10^6)$) to your observation and describe whether you observe difference or not. Briefly discuss the difference between the obervations from the two runs.

Problem 2. Importance Sampling

Implement `probitNormTest` function in lecture 19 into a C++ program. A example run is as follows.

```
$ probitNormTest
Usage: probitNormTest [# rounds] [# of samples per round] [mu] [sigma]

$ probitNormTest 100 1000 -8 1
METHOD  MEAN            STDEV           REL_STDEV
UNIF_MC 7.6439e-09      1.5799e-09      0.20669
NORM_MC 5.2059e-09      1.2395e-08      1.99722
SHFT_IS 7.7020e-09      1.1099e-09      0.01324
```

Run the method with the following sets of parameters, and compare which method produces smallest standard deviation. Briefly discuss why your observe such results.

- $r = 100$, $n = 1,000$, $\mu = -8$, $\sigma = 1$
- $r = 100$, $n = 10,000$, $\mu = -8$, $\sigma = 1$
- $r = 100$, $n = 10,000$, $\mu = 0$, $\sigma = 1$
- $r = 100$, $n = 10,000$, $\mu = -8$, $\sigma = 10$ ($\sigma^2 = 100$)
- $r = 100$, $n = 10,000$, $\mu = -8$, $\sigma = 0.1$ ($\sigma^2 = 0.01$)

You may use the following code snipplets for computing pdf/cdf or random sampling.

```
#include <boost/math/distributions/normal.hpp>
boost::math::normal norm(mu,sigma);
double p = boost::math::cdf(norm,x)
double c = boost::math::pdf(norm,x)
```

```
#include <boost/random/uniform_real.hpp>
#include <boost/random/normal_distribution.hpp>
#include <boost/random/variate_generator.hpp>
#include <boost/random/mersenne_twister.hpp>
boost::uniform_real<> uni_dist(0,1);
boost::normal_distribution<> norm_dist(0,1);

boost::variate_generator<prgType&, boost::uniform_real<> > runi(rng,uni_dist);
boost::variate_generator<prgType&, boost::normal_distribution<> > rnorm(rng,norm_dist);

double r1 = runi();
double r2 = rnorm();
```

Problem 3. Principal component regression

You're given two sets of input files, one representing a label indicating 0 or 1, and the other containing vector of size 256 for each label. The vector contains 16 * 16 grayscale images (-1 to 1). The input file is the same as the one used in lecture 22.

Write a C++ program that computes top k principal components of X (in decreasing order of singular values), and performs linear regression between the label and top k principal components. Report the β for each principal compoents with standard errors, and compute t statistics for the given input.

In each major step of matrix operation or input/outpu routinet, measure the elapsed time and identify which routine takes the most of the time.

Below is an example output of your program.

```
$ ./pcaRegress
Usage: ./pcaRegress [Y] [X] [k]

$ ./pcaRegress zip.01.lab.txt zip.01.val.txt 10
-----------------------------------
ELAPSED TIME
-----------------------------------
Reading matrix ... Elapsed time is *** seconds
Matrix decomposition ... *** seconds
Solving Linear Regression... *** seconds
-----------------------------------
RESULTS
-----------------------------------
SE(beta) = 0.0895216 for all betas
----- beta -----
[-28.19561]
[12.491015]
[-2.8502743]
[-3.2443976]
[-3.7145989]
[0.1323581]
[1.198437]
[-0.87992984]
[-0.8538835]
[1.0166741]
----- t-statistics -----
[-314.95881]
[139.53077]
[-31.838963]
[-36.241513]
[-41.493893]
[1.4785049]
[13.38713]
```

```
[-9.8292483]
[-9.5382979]
[11.356748]
-----------------------------------
```