Course EPIB-621 - Data Analysis for the Health Sciences

Assignment 5 - Solutions

1. Recall again the data set used for assignment 4 called drugfree.txt. To remind you the variables contained in this data set are described in the table below:

Description	Code	Variable Name
Age at Enrollment	Years	age
Beck Depression Score at Admission	0.000-54.000	beck
IV Drug Use History at Admission	1 = Never, 2 = Previous 3 = Recent	ivhx
Number of Prior Drug Treatments	0-40	ndrugtx
Subject's Race	0 = White 1 = Other	race
Treatment Randomization Assignment	0 = Short 1 = Long	treat
Remained Drug Free for 12 Months	1 = Remained Drug Free 0 = Otherwise	drug.free

We previously used this data set to crudely investigate confounding, but now that we have learned about the bic.glm function, we can investigate this use further, as well as see what final model(s) can be used for best predictions.

(a) Run bic.glm on this data set, and report what variables were selected to be in the best model. Recall that you need to load the BMA program first, before running the bic.glm command. Remember also to declare the inhx variable as a factor before you run the regressions. Report the analysis summary, which includes the best five models, and the model probabilities associated with each of these five models.

```
# Read in data set

> drugfree.dat <- read.table(file="g:\\assignments\\drugfree.txt", header=T)

#  Change ivhx to be a factor variable

> drugfree.dat$ivhx <- as.factor(drugfree.dat$ivhx)

#  Run the bic.glm program

> output <- bic.glm(drug.free ~ age + ivhx + race + treat + beck + ndrugtx,
         glm.family=binomial, data=drugfree.dat)

#  Look at summary output

> summary(output)

Call:
bic.glm.formula(f = drug.free ~ age + ivhx + race + treat + beck +
  ndrugtx, data = drugfree.dat, glm.family = binomial)


  14  models were selected
 Best  5  models (cumulative posterior probability =  0.7157 ):

           p!=0   EV       SD       model 1     model 2     model 3     model 4     model 5
Intercept  100   -1.26313  0.66579  -7.678e-01  -9.991e-01  -1.762e+00  -2.041e+00  -8.897e-01
age        35.5   0.01455  0.02255        .           .       3.192e-02   5.080e-02       .
ivhx       22.7
    .2           -0.13457  0.28532        .           .           .      -5.899e-01       .
    .3           -0.19156  0.37547        .           .           .      -8.197e-01       .
race       11.8   0.04534  0.14423        .           .           .           .       3.953e-01
treat      31.5   0.13854  0.23195        .       4.348e-01       .           .           .
beck        0.0   0.00000  0.00000        .           .           .           .           .
ndrugtx    88.4  -0.06596  0.03418  -7.496e-02  -7.392e-02  -8.629e-02  -6.323e-02  -7.146e-02

nVar                                     1           2           2           3           2
BIC                                  -2.999e+03  -2.998e+03  -2.997e+03  -2.996e+03  -2.996e+03
post prob                             0.315       0.162       0.096       0.071       0.071
```

The best models includes just ndrugtx, which is also in all five topp models. Next best model (ndrugtx + treat) has only roughly half the posterior probability.

(b) Output the maximum likelihood estimators from the top 14 models along with the SEs from these models. Looking down the columns for each model, do you see any evidence for confounding? If so, report on which variables may be confounded.

```
> output$names
[1] "age"     "ivhx"    "race"    "treat"   "beck"    "ndrugtx"

> output$mle
            [,1]       [,2]       [,3]       [,4]      [,5]      [,6] [,7]        [,8]
 [1,] -0.7677805 0.00000000  0.0000000  0.0000000 0.0000000 0.0000000    0 -0.07495823
 [2,] -0.9990724 0.00000000  0.0000000  0.0000000 0.0000000 0.4347941    0 -0.07392151
 [3,] -1.7619230 0.03191628  0.0000000  0.0000000 0.0000000 0.0000000    0 -0.08629077
 [4,] -2.0408837 0.05079960 -0.5899204 -0.8196606 0.0000000 0.0000000    0 -0.06322734
 [5,] -0.8896943 0.00000000  0.0000000  0.0000000 0.3953129 0.0000000    0 -0.07146142
 [6,] -2.0598983 0.03370213  0.0000000  0.0000000 0.0000000 0.4551782    0 -0.08575186
 [7,] -2.0331833 0.04540983 -0.6810308 -1.0019480 0.0000000 0.0000000    0  0.00000000
 [8,] -2.3327637 0.05259341 -0.6236554 -0.8056123 0.0000000 0.4513352    0 -0.06375980
 [9,] -1.0996163 0.00000000  0.0000000  0.0000000 0.3651188 0.4136589    0 -0.07099559
[10,] -0.6797242 0.00000000 -0.4810199 -0.7748382 0.0000000 0.0000000    0  0.00000000
[11,] -2.3136538 0.04697727 -0.7096506 -0.9909218 0.0000000 0.4398554    0  0.00000000
[12,] -1.0686906 0.00000000  0.0000000  0.0000000 0.0000000 0.0000000    0  0.00000000
[13,] -0.5474864 0.00000000 -0.3794202 -0.6004013 0.0000000 0.0000000    0 -0.05418847
[14,] -1.8439429 0.03077461  0.0000000  0.0000000 0.3782577 0.0000000    0 -0.08222149
> output$se
            [,1]       [,2]      [,3]      [,4]      [,5]      [,6] [,7]       [,8]
 [1,] 0.13032443 0.00000000 0.0000000 0.0000000 0.0000000 0.0000000    0 0.02467966
 [2,] 0.16907553 0.00000000 0.0000000 0.0000000 0.0000000 0.1947792    0 0.02446905
 [3,] 0.51769154 0.01598013 0.0000000 0.0000000 0.0000000 0.0000000    0 0.02604486
 [4,] 0.52974598 0.01714263 0.2827944 0.2436586 0.0000000 0.0000000    0 0.02575498
 [5,] 0.14759880 0.00000000 0.0000000 0.0000000 0.2131153 0.0000000    0 0.02469015
 [6,] 0.53640112 0.01604022 0.0000000 0.0000000 0.0000000 0.1958251    0 0.02586096
 [7,] 0.52654126 0.01690291 0.2791433 0.2356073 0.0000000 0.0000000    0 0.00000000
 [8,] 0.54838279 0.01721042 0.2847013 0.2445347 0.0000000 0.1985954    0 0.02562626
 [9,] 0.18024939 0.00000000 0.0000000 0.0000000 0.2142740 0.1956386    0 0.02449241
[10,] 0.14173949 0.00000000 0.2657061 0.2165764 0.0000000 0.0000000    0 0.00000000
[11,] 0.54455972 0.01696901 0.2810954 0.2366635 0.0000000 0.1972224    0 0.00000000
[12,] 0.09559892 0.00000000 0.0000000 0.0000000 0.0000000 0.0000000    0 0.00000000
[13,] 0.15342528 0.00000000 0.2698433 0.2280573 0.0000000 0.0000000    0 0.02476883
[14,] 0.52259284 0.01604878 0.0000000 0.0000000 0.2141568 0.0000000    0 0.02602466
```

Age seems to be confounded with ivhx, as every time ivhx enters the model, the coefficient for age jumps up (from 0.03 to 0.050). Also appears to be some confounding between ndrugtx and ivhx. Note that the answer here is somewhat different from that given in assignment 4, because here we are able to focus on best predictive models, and not just the univariate and full models. The BIC programs are very useful for investigating confounding for these reasons.

(c) Report your "best" (i.e., model averaged) prediction for someone who is aged 30, has a beck depression score of 10, has a recent history of IV drug use, is white, no prior drug treatment, and had a short treatment scheme.

from the above results, we see that the model averaged coefficients are:

```
> output$postmean
[1] -1.26313126  0.01454645 -0.13456751 -0.19155720  0.04533731
     0.13854044  0.00000000 -0.06595955

> output$names
[1] "age"     "ivhx"    "race"    "treat"   "beck"    "ndrugtx"
```

So, for someone who is aged 30, has a beck depression score of 10, has a recent history of IV drug use, is white, no prior drug treatment, and had a short treatment scheme, we calculate:

```
#  Need to calculate this expression:

#  exp(alpha + b.age*30 + b.beck*10 + b.ivhx3 )/(1+ exp(alpha +
#  b.age*30 + b.beck*10 + b.ivhx3))

#  So plugging in the above values where needed, we have:

> exp(-1.26313126 + 0.01454645*30 + 0*10 -0.19155720 )/(1+ exp(-1.26313126 +
+                   0.01454645*30 + 0*10 -0.19155720))
[1] 0.2653597

#  So we predict 0.265 as an event rate, quite different (probably better)
#  from the prediction from the last assignment using the full model.
```

2. The risk of a heart attack increases with age, but varies from country to country depending on local habits. The data set heart.txt provides the age and heart attack data (heart = yes/no = 1/0). The data are structured such that the first 100 subjects are from country 1, the second 100 are from country 2, and so on, so that the last 100 are from country 5. Therefore, no country variable needs to be defined. Use WinBUGS to create a hierarchical logistic regression model. The first level (individual patient level) regression should depend on a random intercept coefficient which is different for each country, and a fixed age coefficient, which is considered as constant across countries. All prior distributions should be normal, including the hierarchial distribution on the intercepts.

(a) Run this model in WinBUGS, and report the results from all coefficients, including the random effect (hierarchical) intercepts.

(b) Create lines such as

```
diff12 <- step(alpha[1]- alpha[2])
diff13 <- step(alpha[1]- alpha[3])
```

and so on to estimate the probability that one countries rate (after adjusting for the effect of age) is different from another country's rates. Since we have five countries, you will need to add

10 such lines. Run the program, and report these probabilities. [To save running time, you can run parts (a) and (b) as one program.]

Below are the program and results.

```
\begin{verbatim}

model                          #  Usual model statement in WinBUGS
{
 for (j in 1:5)                #  Loop over 5 countries
{

 for (i in (100*(j-1)+1):(100*j))    # Index for jth country
     {
     logit(p[i]) <- alpha[j] + beta*age[i] #  Logit for individual probability
     heart[i] ~ dbern(p[i])               #  Likelihood function for ith individual
     }                                 #
     alpha[j] ~ dnorm(mu, tau)         #  Hierarchical component:  coountry rates
   }                                   #  "tied together" through normal distribution
 mu ~  dnorm(0,0.001)                  #  Prior on hierarchical mean
 tau <- 1/(sigma*sigma)                #  Needed for WinBUGS
sigma ~ dunif(0,20)                    #  Prior for hierarchical Sd
 beta  ~  dnorm(0, 0.001)              # Prior for beta

#  Difference probabilities

diff12 <- step(alpha[1]- alpha[2])
diff13 <- step(alpha[1]- alpha[3])
diff14 <- step(alpha[1]- alpha[4])
diff15 <- step(alpha[1]- alpha[5])
diff23 <- step(alpha[2]- alpha[3])
diff24 <- step(alpha[2]- alpha[4])
diff25 <- step(alpha[2]- alpha[5])
diff34 <- step(alpha[3]- alpha[4])
diff35 <- step(alpha[3]- alpha[5])
diff45 <- step(alpha[4]- alpha[5])
}

# Inits

list(alpha=c(0,0,0,0,0), beta=1, mu=0, sigma = 1)

#  Data

list(age = c(78, 67, 62, 74, 64, 59, 74, 55, 68, 77, 65, 73, 77, 67, 62,
74, 57, 58, 75, 59, 74, 70, 66, 60, 62, 68, 64, 67, 68, 73, 56, 75, 71,
57, 61, 55, 63, 75, 64, 69, 77, 65, 65, 74, 71, 79, 67, 64, 70, 60, 77,
75, 60, 60, 60, 60, 57, 68, 61, 64, 70, 70, 59, 73, 65, 56, 74, 69, 65,
.............etc.................
62, 65, 68, 57, 70, 79, 58, 55, 66, 55, 74, 76, 73, 69, 71, 57, 69, 76,
```

```
57, 70, 60, 63, 70, 76, 56, 64, 73, 59, 58, 68, 57, 67, 59, 68, 66),
country = c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
.....etc.....
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5),
heart = c(1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0,
0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0,
1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1,
.....etc.....
1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1))

# Results

node      mean    sd      MC error     2.5%    median  97.5%   start   sample
alpha[1] -0.2867  0.8739  0.05466     -2.03   -0.285   1.368   1001    20000
alpha[2] -0.407   0.8965  0.05617     -2.206  -0.404   1.295   1001    20000
alpha[3] -1.028   0.8985  0.05613     -2.831  -1.024   0.6557  1001    20000
alpha[4] -1.813   0.875   0.0536      -3.566  -1.808  -0.1614  1001    20000
alpha[5] -2.284   0.9188  0.05554     -4.131  -2.283  -0.5388  1001    20000
beta      0.00377 0.01275 8.132E-4    -0.0203  0.00377 0.02938 1001    20000
diff12    0.6652  0.4719  0.003906     0.0     1.0     1.0     1001    20000
diff13    0.9953  0.06803 4.838E-4     1.0     1.0     1.0     1001    20000
diff14    1.0     0.0     7.071E-13    1.0     1.0     1.0     1001    20000
diff15    1.0     0.0     7.071E-13    1.0     1.0     1.0     1001    20000
diff23    0.9839  0.1261  8.079E-4     1.0     1.0     1.0     1001    20000
diff24    1.0     0.0     7.071E-13    1.0     1.0     1.0     1001    20000
diff25    1.0     0.0     7.071E-13    1.0     1.0     1.0     1001    20000
diff34    0.9916  0.09127 6.557E-4     1.0     1.0     1.0     1001    20000
diff35    0.9998  0.01414 9.869E-5     1.0     1.0     1.0     1001    20000
diff45    0.8854  0.3185  0.002532     0.0     1.0     1.0     1001    20000
mu       -1.158   1.148   0.05572    -3.38    -1.154   1.04    1001    20000
p[1]      0.5019  0.06056 0.002184     0.3828  0.5021  0.6201  1001    20000
p[2]      0.4916  0.04936 3.066E-4     0.3955  0.4914  0.5878  1001    20000
p[3]      0.4869  0.0515  0.00109      0.3882  0.4869  0.5885  1001    20000
p[4]      0.4981  0.05433 0.001398     0.3915  0.4985  0.6044  1001    20000
p[5]      0.4888  0.05008 7.129E-4     0.3926  0.4886  0.5876  1001    20000
.....etc.....
p[497]    0.1198  0.03225 2.502E-4     0.06377 0.1176  0.1895  1001    20000
p[498]    0.1171  0.03367 7.964E-4     0.06009 0.1142  0.1917  1001    20000
p[499]    0.1202  0.03232 2.292E-4     0.0639  0.118   0.1898  1001    20000
p[500]    0.1194  0.03223 2.947E-4     0.06327 0.1172  0.1891  1001    20000
sigma     1.421   0.9857  0.01617      0.5323  1.16    3.926   1001    20000
tau       1.011   0.9366  0.009583     0.065   0.7431  3.532   1001    20000
```

There are enormous differences in heart rates between countries, and note that sigma is quite large.

3. There exists drugs or other treatments that are known to work for some subjects but not for others. Sometimes the reason for this can be ascertained, but not always. This can happen, for example, if an as yet undiscovered gene affects response to the substance. In clinical trials of these substances, any effects are typically reported as an average over non-responders and non-responders.

One example of a substance that seems to work for some subjects but not others is calcium supplementation for reduction in blood pressure; some subjects seem to respond, some do not. While the reason is not at this time known, we will suppose that there is an undiscovered gene responsible for this effect.

Suppose that a multi-centre clinical trial will be carried out to estimate the effect of calcium supplementation on lowering blood pressure. Two towns will participate, A and B. Suppose that the gene tends to be present in town A, but not town B.

Download the data file "calcium.txt" from the course web page. [While there, you might want to download calcium.missing.txt, which will be used later.]

(a) Using WinBUGS, do a simple linear regression of blood pressure reduction on calcium intake. Report the average effect of calcium supplements, with 95% credible interval.

Model and results are below. Average effect over cities A and B is a reduction of about 7.9 mm Hg, p5% CrI is (-9.0, -6.8).

```
model
{
    for (i in 1:600) {
          mu[i] <- alpha + beta*calcium[i]
          bp.change[i]   ~ dnorm(mu[i],tau)
              }
    alpha    ~ dnorm(0.0,1.0E-4)
    beta     ~ dnorm(0.0,1.0E-4)
    tau <-      1/(sigma*sigma)
    sigma   ~ dunif(0,100)
}

#  Initial values
list(alpha=0, beta=0, sigma=20)
```

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
alpha	0.0704	0.4091	0.002964	-0.7327	0.07206	0.8725	1001	20000
beta	-7.905	0.5605	0.004349	-8.997	-7.906	-6.788	1001	20000
sigma	6.876	0.1988	0.00149	6.497	6.871	7.276	1001	20000
tau	0.0212	0.00122	9.149E-6	0.01889	0.02118	0.02369	1001	20000

(b) Do separate linear regressions within each town. Compare the effects of calcium in town A versus town B. Note that you can do all of this within a single WinBUGS program, by simply

looping twice, once over the first 300 subjects, all from town A, and then over the next 300 subjects, all from town B. By creating a new parameter such as

```
beta.calcium.a - beta.calcium.b
```

you can directly monitor the difference in effects of calcium supplementation between the two towns.

Model and results are below. There is about twice as large an effect ini city A compared to city B.

```
model
{
    for (i in 1:300) {
          mu[i] <- alpha.a + beta.calcium.a*calcium[i]
          bp.change[i]   ~ dnorm(mu[i],tau)
                    }
   for (i in 301:600) {
          mu[i] <- alpha.b + beta.calcium.b*calcium[i]
          bp.change[i]   ~ dnorm(mu[i],tau)
                     }
    alpha.a    ~ dnorm(0.0,1.0E-4)
    beta.calcium.a     ~ dnorm(0.0,1.0E-4)
    alpha.b    ~ dnorm(0.0,1.0E-4)
    beta.calcium.b     ~ dnorm(0.0,1.0E-4)
    tau <-      1/(sigma*sigma)
    sigma   ~ dunif(0,100)
    calcium.diff <- beta.calcium.a - beta.calcium.b
      }

#  Initial values
list(alpha.a=0, beta.calcium.a=0, alpha.b=0, beta.calcium.b=0, sigma=20)

Results:

node           mean    sd  MC error     2.5%    median  97.5%   start   sample
alpha.a        -0.1539 0.5757  0.004072 -1.278  -0.1556 0.9929  1001    20000
alpha.b        0.2604  0.5455  0.003806 -0.8053 0.2613  1.327   1001    20000
beta.calcium.a -10.16  0.7726  0.00505  -11.67  -10.15  -8.642  1001    20000
beta.calcium.b -5.368  0.7638  0.005236 -6.859  -5.368  -3.879  1001    20000
calcium.diff   -4.789  1.084   0.007266 -6.914  -4.792  -2.621  1001    20000
sigma          6.616   0.1914  0.001324 6.257   6.61    7.008   1001    20000
tau            0.0229  0.00132 9.091E-6 0.02036 0.02289 0.02554 1001    20000
```

We will now repeat the analysis of this clinical trial, but using data sets calcium.missing.txt, which contains some missing data.

(c) Using only the first 400 subjects in the database (i.e., those without any missing data, use a linear regression model to estimate the effect of calcium supplementation. Compare your answer to that obtained in part (a).

Using exactly the same model as in (a), but now using data from only the first 400 subjects, results are given below. Note that the estimate of the effect of calcium supplementation is too large.

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
alpha	-0.0479	0.524	0.003274	-1.077	-0.0453	0.9701	1001	20000
beta	-9.29	0.7173	0.004679	-10.69	-9.293	-7.889	1001	20000
sigma	7.124	0.2526	0.001853	6.649	7.118	7.636	1001	20000
tau	0.01978	0.00139	1.025E-5	0.01715	0.01974	0.02262	1001	20000

(d) Now use multiple imputation to adjust your answer in part (c). Using all subjects in the data set calcium.missing.txt, impute the missing data on the effects. Use a separate prediction equation for each of town A and town B. Now compare your answer to both parts (a) and (c). Has multiple imputation removed the bias in the estimated coefficient (which represents the average effect of calcium supplementation in these two towns)?

Using exactly the same model as in (b), but now with the calcium.missing.txt data set, we see that the bias has been corrected. Note that the average of beta.calcium.a and beta.calcium.b is close to -8, matching the result in (a).

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
alpha.a	-0.1504	0.607	0.004238	-1.33	-0.1561	1.052	1001	20000
alpha.b	0.2393	0.9759	0.006741	-1.697	0.2463	2.153	1001	20000
beta.calcium.a	-10.16	0.8201	0.005671	-11.79	-10.16	-8.564	1001	20000
beta.calcium.b	-6.193	1.408	0.009644	-8.969	-6.205	-3.43	1001	20000
calcium.diff	-3.972	1.631	0.01052	-7.165	-3.953	-0.7944	1001	20000
sigma	7.014	0.2513	0.001751	6.544	7.009	7.525	1001	20000
tau	0.02041	0.00145	1.013E-5	0.01766	0.02036	0.02335	1001	20000

4. Generate a simulated linear regression data set in R that follows the following model (sample size = 100):

$$y = 2 + 5 * x, \quad \sigma = 1, \quad x \sim normal(0, 1)$$

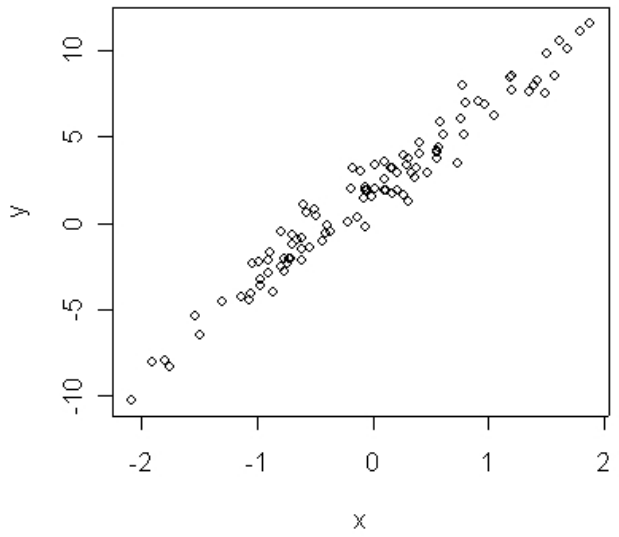
To do this, use lines such as:

```
x <- round(rnorm(100, mean=0, sd=1),2)
y <- round(rnorm(100, mean = 2 + 5*x, sd=1),2)
```

Note that since we are using random numbers, everyone in the class will be using a slightly different data set. I rounded everything to 2 decimal places, which makes for cleaner data sets without losing too much precision.

(a) Plot x versus y.

Plot is:



(b) Use R to run a standard linear regression of x versus y. Provide the estimates and 95% confidence intervals for the intercept and slope (see R class notes if you forget how to do this, and recall that approximate 95% intervals can be derived from the point estimates ± 1.96 times the standard error for each parameter). Are they close to their theoretical values (of 2 and 5, respectively)?

R commands and results are below.

```
> x <- round(rnorm(100, mean=0, sd=1),2)
> y <- round(rnorm(100, mean = 2 + 5*x, sd=1),2)
> plot(x,y)
> summary(lm(y ~ x))

Call:
lm(formula = y ~ x)
Residuals:
    Min      1Q  Median      3Q     Max
-2.0702 -0.6699 -0.1428  0.5959  2.3368
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.78808    0.09392   19.04   <2e-16 ***
x            5.00670    0.10576   47.34   <2e-16 ***
---
```

```
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.9388 on 98 degrees of freedom
Multiple R-Squared: 0.9581,     Adjusted R-squared: 0.9577
F-statistic:  2241 on 1 and 98 DF,  p-value: < 2.2e-16
> 1.78808 - 1.96*0.09392
[1] 1.603997
> 1.78808 + 1.96*0.09392
[1] 1.972163
> 5.00670 - 1.96*0.10576
[1] 4.79941
> 5.00670 + 1.96*0.10576
[1] 5.21399
```

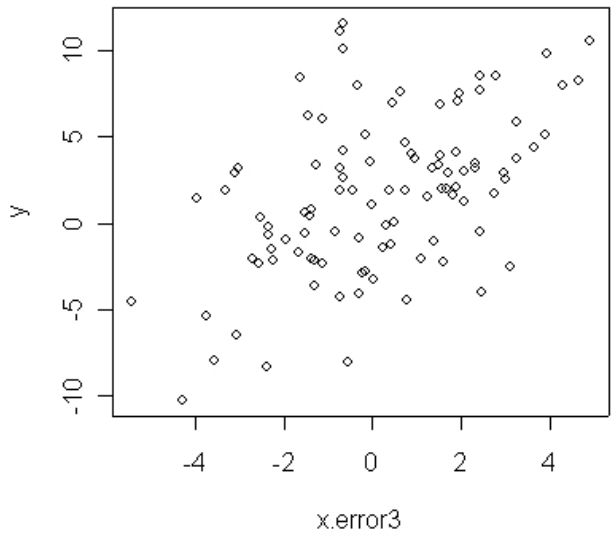
Note that the true value of the slope is well within its 95% CI. The slope is well estimated, but the intercept CI actually misses including the true value of 2 by a tiny bit (one of those 5% of unlucky samples, maybe you will be luckier in your data set).

(c) Now we will add some measurement error to the x values. In particular, we will create a measurement error version of x using the R command

```
x.error <- round(rnorm(100, mean=x, sd=2),2 )
```

Note that the measurement error version of x is centered at the true value of x, but has random noise about the observation. This is typical of measurement error seen when data are generated by an unbiased but imprecise measuring tool. Plot $x.error$ versus y, and note any differences from your plot in part (a).

Plot is below. Note that the regression relationship is much less clear here, because of the extra measurement error.



(d) Rerun the linear regression again, but this time using *x.error* rather than x. Compare the results (point estimates and confidence intervals) you obtain here with those obtained in part (b), and note any differences.

R commands and results are below. Note that the slope estimate is very far from correct, not even close to its 95% interval. The measurement error, even though unbiased, has resulted in extremely poor estimation.

```
> summary(lm(y ~ x.error))

Call:
lm(formula = y ~ x.error)

Residuals:
    Min      1Q  Median      3Q     Max
-8.9160 -2.1390 -0.4099  2.3939 10.7782

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.4856     0.3994   3.719 0.000333 ***
x.error      1.0356     0.1832   5.652 1.56e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.983 on 98 degrees of freedom
Multiple R-Squared: 0.2458,     Adjusted R-squared: 0.2381
F-statistic: 31.94 on 1 and 98 DF,  p-value: 1.561e-07
> 1.4856  - 1.96*0.3994
```

```
[1] 0.702776
> 1.4856  + 1.96*0.3994
[1] 2.268424
> 1.0356  - 1.96*0.1832
[1] 0.676528
> 1.0356  + 1.96*0.1832
[1] 1.394672
```

(e) Before leaving R, save your data sets for use in WinBUGS in problem 5. To do this, use commands such as:

```
x.list <- list(x=x, y=y)
xerror.list <- list(x.error = x.error,y=y)
dput(x.list, file = "c://temp//x.txt")
dput(xerror.list, file= "c://temp//xerror.txt")
```

You will use the first data set in the (a) of question 5, and the second data set in parts (b) and (c) of question 5.

5. In this question we will analyse the same two data sets as were used in question 4, but now using WinBUGS, with and without correcting for possible measurement error.

(a) Run a straightforward WinBUGS program for the linear regression of x versus y (see class notes of simple WinBUGS programs if you do not recall how to do this). Provide the point estimates and 95% credible intervals for the intercept and slope. Compare these to your estimates in part (b) of question 4 (they should be quite similar).

Model and results are given below. Note that results (both point estimates and CI's) are virtually identical to those given in R, as expected.

```
model
{
    for (i in 1:100)
   {
          y.mean[i] <- alpha + beta*x[i]
          y[i]   ~ dnorm(y.mean[i],tau)
    }
    alpha    ~ dnorm(0 ,0.001)
    beta     ~  dnorm(0 ,0.001)
    tau      <- 1/(sigma*sigma)
    sigma   ~ dunif(0,100)
 }
```

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
alpha	1.786	0.09455	9.678E-4	1.603	1.784	1.974	1001	10000
beta	5.009	0.1066	0.001012	4.8	5.009	5.218	1001	10000
sigma	0.951	0.06892	6.328E-4	0.826	0.9467	1.097	1001	10000
tau	1.123	0.1614	0.001471	0.8313	1.116	1.466	1001	10000

(b) Repeat part (a), but now using *x.error* versus *y*. Provide the point estimates and 95% credible intervals for the intercept and slope. Compare these to your estimates in part (d) of question 4 (again, they should be quite similar).

Program is identical to part (a), except that we change x to x.error, to use the different data set. Results are below. Note that results are very similar to the results from R in 4(d). Since we are not (yet) adjusting for measurement error, this is expected. Thus, the WinBUGS program, not yet knowing that results have measurement error, has not yet adjusted.

```
node   mean    sd     MC error  2.5%    median  97.5%   start   sample
alpha 1.481   0.4016  0.003614  0.6978  1.479   2.282   1001    10000
beta  1.037   0.1855  0.001871  0.6755  1.035   1.408   1001    10000
sigma 4.037   0.2916  0.003042  3.52    4.019   4.65    1001    10000
tau   0.06231 0.00890 9.086E-5  0.04626 0.0619  0.0807  1001    10000
```

(c) Now, we will modify the simple linear regression model to account for any measurement error. To the basic linear regression model (from part (a), NOT part (b), because we want to estimate the true relationship with x, not the one with measurement error variable *x.error*!), add a line such as:

```
x.error[i] ~ dnorm(x[i], tau.error)
```

You will also need to add a line for the prior for tau.error, and for x[i]. As usual, we will define tau.error in terms of sigma.error, and put a uniform prior on sigma. Use the following lines:

```
tau.error <- 1/(sigma.error*sigma.error)
sigma.error ~ dunif(1, 5)
```

to indicate that it is known that the measurement error variance is between 1 and 5 (real value, recall, was SD=2).

Run this model, and report the point estimates and 95% credible intervals for the intercept and slope. Compare these to your estimates in part (b) of question 5 . . . has the model correctly adjusted for the measurement error?

Full Program is below, followed by the results. Program was in fact run three times, first (as illustrated below) with var = 0.5 (precision = 2), then with var = 1 (the unknown but correct value), and finally with var = 2.

```
model
{
    for (i in 1:100)
   {
```

```
        y.mean[i] <- alpha + beta*x[i]
        y[i]   ~ dnorm(y.mean[i],tau)
        x.error[i] ~ dnorm(x[i], tau.error)
        x[i] ~ dnorm(0,2)
    }
    alpha    ~ dnorm(0 ,0.001)
    beta     ~  dnorm(0 ,0.001)
    tau      <- 1/(sigma*sigma)
    sigma   ~ dunif(0,100)
    tau.error <- 1/(sigma.error*sigma.error)
    sigma.error ~ dunif(1,5)
 }
```

Results for var = 0.5

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
alpha	1.545	0.4131	0.0245	0.7482	1.547	2.359	1001	10000
beta	6.003	0.5702	0.03471	4.677	6.026	7.075	1001	10000
sigma	1.097	0.7653	0.06502	0.1518	0.9086	2.892	1001	10000
sigma.error	1.949	0.1424	0.00178	1.695	1.941	2.252	1001	10000
tau	8.457	42.92	3.615	0.1196	1.212	43.84	1001	10000
tau.error	0.2674	0.03869	4.626E-4	0.1972	0.2655	0.3479	1001	10000

Results for var = 1

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
alpha	1.42	0.3973	0.02396	0.6612	1.43	2.203	1001	10000
beta	4.23	0.5116	0.03655	3.015	4.288	5.094	1001	10000
sigma	1.331	0.9239	0.08097	0.06552	1.195	3.271	1001	10000
sigma.error	1.89	0.1447	0.00305	1.631	1.881	2.197	1001	10000
tau	23.09	119.6	9.447	0.09351	0.701	233.3	1001	10000
tau.error	0.2847	0.04328	9.686E-4	0.2073	0.2825	0.3762	1001	10000

Results for var = 2

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
alpha	1.495	0.4008	0.01144	0.7066	1.493	2.279	1001	10000
beta	2.665	0.5332	0.03222	1.583	2.71	3.563	1001	10000
sigma	2.526	0.9348	0.0718	0.4244	2.675	3.986	1001	10000
sigma.error	1.733	0.1972	0.009831	1.341	1.737	2.107	1001	10000
tau	0.5666	1.852	0.1633	0.06295	0.1397	5.554	1001	10000
tau.error	0.3469	0.08555	0.004066	0.2252	0.3316	0.5564	1001	10000

With variance = 0.5 (precision = 2), note that while estimation is not as good as using the correct data and the right model (as in part (a)), our measurement error model has done a very good job of correcting for the measurement error. Both alpha and beta are well within their 95% CrI's, unlike in part (b) where we did not adjust. Note that we also left a wide range for the measurement error, with a prior from 1 to 5, and that the model correctly zeroed in on the correct value of 2,

even though this was not in the centre of the prior distribution. Slightly different adjustments result when variance = 1 or variance = 2. With var = 1 (correct value) all parameters are within their 95% intervals, but not when var = 2. Overall, the degree to which the model correctly adjusts for measurement error depends on knowing the variance of the variable measured with error.