Bayesian Inference for Regression Parameters

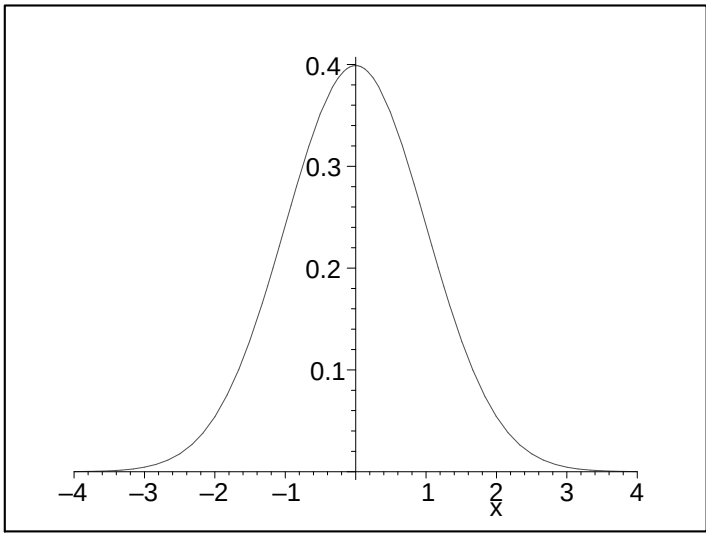
Bayesian inference for simple linear regression parameters follows the usual pattern for all Bayesian analyses:

1. Form a prior distribution over all unknown parameters.
2. Write down the likelihood function of the data.
3. Use Bayes theorem to find the posterior distribution of all parameters.

- We have applied this generic formulation so far to problems with binomial distributions, normal means (with variance known), multinomial parameters, and to the Poisson mean parameter. All of these problems involved only one parameter at a time (strictly speaking, more than one in the multinomial case, but only a single Dirichlet distribution was used, so all parameters were treated together, as if it were just one parameter).
- What makes regression different is that we have three unknown parameters, since the intercept and slope of the line, α and β are unknown, and the residual standard deviation, σ is also unknown.
- Hence our Bayesian problem becomes slightly more complicated, since we are in a multi-parameter situation.
- Before detailing the steps involved in Bayes Theorem for regression problems, we need to look at multi-parameter problems in general.

Joint and Marginal Distributions

When we have only one parameter, we speak of its density. For example, if $x \sim N(0, 1)$, then the graph of the probability density is:



When we have two or more parameters, we speak of a *joint probability density*. For example, let x and y be *jointly multivariately* normally distributed, which is notated by:

$$\begin{pmatrix} x \\ y \end{pmatrix} \sim N\left(\begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \Sigma\right)$$

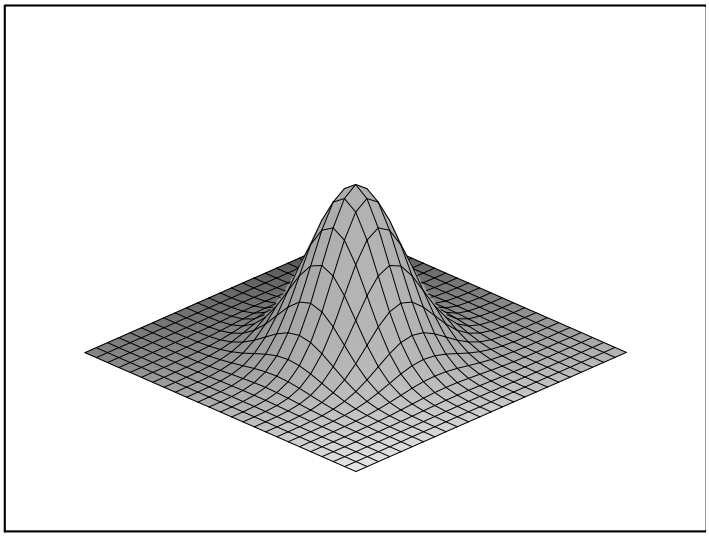
where

$$\Sigma = \begin{pmatrix} \sigma_x^2 & \rho_{xy} \\ \rho_{xy} & \sigma_y^2 \end{pmatrix}$$

Example: Suppose

$$\begin{pmatrix} x \\ y \end{pmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\right)$$

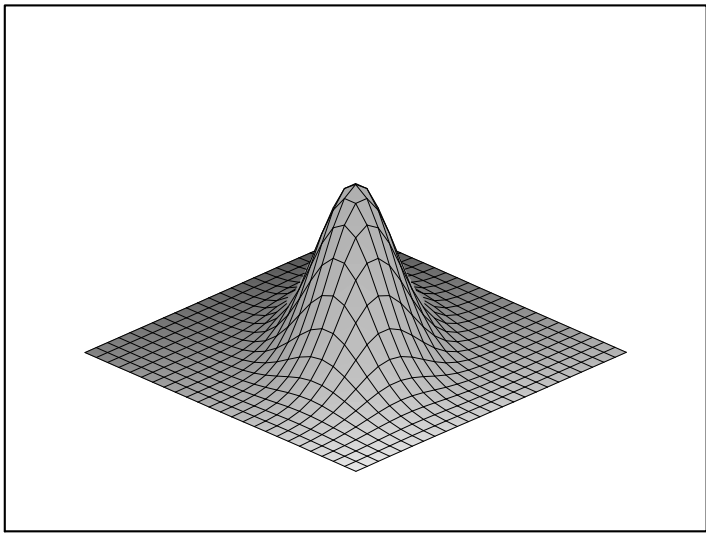
which is equivalent to two independently normally distributed variables, with no correlation between them. Then the picture is:



Note how the "slices" resemble univariate normal densities in all directions. These "slices" are *conditional densities*, which are averaged over to produce *marginal densities*, which will be defined later. In the presence of correlations, for example a correlation of 0.5, we have

$$\begin{pmatrix} x \\ y \end{pmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}\right)$$

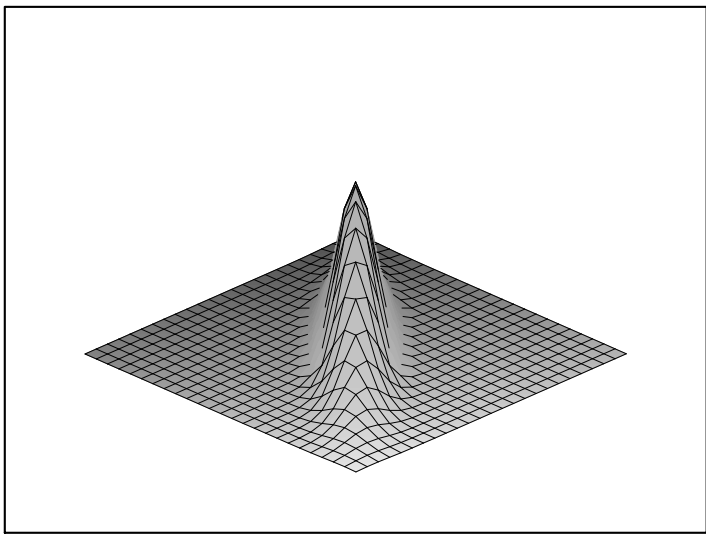
and the picture is:



Similarly, with very high correlation of 0.9, we have

$$\begin{pmatrix} x \\ y \end{pmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{pmatrix} 1 & 0.9 \\ 0.9 & 1 \end{pmatrix}\right)$$

and the picture is:



The *bivariate* normal density formula is:

$$f(x,y) = \frac{\exp\left\{-\frac{1}{2(1-\rho_{xy}^2)}\left[\left(\frac{x-\mu_x}{\sigma_x}\right)^2 - 2\rho_{xy}\left(\frac{x-\mu_x}{\sigma_x}\right)\left(\frac{y-\mu_y}{\sigma_y}\right) + \left(\frac{y-\mu_y}{\sigma_y}\right)^2\right]\right\}}{2\pi\sigma_x\sigma_y\sqrt{1-\rho_{xy}^2}}$$

This is a joint density between two variables, since we look at the distribution of x and y at the same time, i.e., jointly. An example where such a distribution might be useful would be looking at both age and height together. (Another example is looking at the joint posterior distribution of α and β, which is where we are heading with all of this!!)

When one starts with a joint density, it is often of interest to calculate *marginal* densities from the joint densities. Marginal densities look at each variable one at a time, and can be directly calculated from joint densities through integration:

$$f(x) = \int f(x,y)dy, \text{ and}$$

$$f(y) = \int f(x, y)dx.$$

In higher dimensions,

$$f(x) = \int f(x, y, z)dydz,$$

and so on.

Normal Marginals are Normal

If $f(x, y)$ is a bivariate normal density, for example, it can be proven that both the marginal densities for x and y are also normally distributed. For example, if

$$\begin{pmatrix} x \\ y \end{pmatrix} \sim N\left(\begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \begin{pmatrix} \sigma_x^2 & \rho_{xy} \\ \rho_{xy} & \sigma_y^2 \end{pmatrix}\right)$$

then

$$x \sim N(\mu_x, \sigma_x^2)$$

and

$$y \sim N(\mu_y, \sigma_y^2)$$

Summary

- Joint densities describe multi-dimensional probability distributions for two or more variables.

- If one has a joint density, then if it is of interest to look at each variable separately, one can find marginal probability distributions by integrating the joint densities. If one wants the marginal distribution of x, for example, then one would "integrate out" all of the parameters except x, and so on.
- For multivariate normal distributions, all marginal densities are again normal distributions, with the same means and variances as the variables have in the joint density.

Before considering Bayesian regression, we will first look at Bayesian inference for a simpler (but related) two parameter model, estimation of a normal mean with variance also unknown. We will then see how to extend this model to a regression model.

Simple Bayesian two parameter problem - normal mean with variance unknown

Recall that we last looked a Bayesian analysis of a normal mean when the variance was known, through the following problem:

EXAMPLE: Consider the situation where we are trying to estimate the mean diastolic blood pressure of Americans living in the United States from a sample of 27 patients. The data are:

76, 71, 82, 63, 76, 64, 64, 74, 70, 64, 75, 81, 75, 78, 66, 62, 79, 82, 78, 62, 72, 83, 79, 41, 80, 77, 67.

We found $\overline{x} = 71.89$, and $s^2 = 85.18$, so that $s = \sqrt{85.18} = 9.22$, but we in fact assumed that σ was known exactly, and just did inference in the unknown mean μ. For that, we found the posterior distribution $\mu \sim N(71.69, 2.68)$, so that the standard deviation was $\sqrt{2.68} = 1.64$.

However, in real practice, σ is unknown, so we have a two parameter problem. Following the usual Bayesian steps, we need to:

Write down the likelihood function for the data: We have already seen the likelihood function for this problem, when we looked at the single

parameter normal mean problem, and it is the same here (but now σ is viewed as a variable, rather than as a known constant):

$$f(x_1, x_2, \ldots, x_n|\mu) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{(x_i-\mu)^2}{2\sigma^2}) = \left(\frac{1}{\sqrt{2\pi}\sigma^2}\right)^n \exp(-\frac{\sum_{i=1}^{n}(x_i-\mu)^2}{2\sigma^2}).$$

Write down the prior distribution for the unknown parameters, μ and σ: As before, we will use a normal prior for μ, and we will use the same one as we used previously,

$$\mu \sim N(\theta, \tau^2) = N(70, 5^2 = 25)$$

We also need a prior distribution for σ. Many choices are possible, the most popular, however, is a uniform over an appropriate range. Here we will use the range

$$\sigma \sim \text{Uniform}(0, 50)$$

We doubt the standard deviation can be more than 50, since that implies a very large range.

Use Bayes theorem to derive the posterior distribution: With the choice of priors given above, there is no simple conjugate solution similar to those we have seen for one parameter problems. There is in fact a conjugate solution, if one uses an inverse gamma prior distribution on σ^2, but for various technical reasons this is not as good a choice as a uniform prior.

With no "closed form" analytical formula to write down, we need to use computer methods to find the posterior distributions for our unknown parameters μ and σ. We will use a computer algorithm called the Gibbs sampler as implemented automatically in software called WinBUGS. We will now see details about how WinBUGS works.

Introduction to WinBUGS

WinBUGS is a free program available from the Biostatistics Unit of the Medical Research Council in the UK (see link on course web page). Installing

WinBUGS is straightforward, one downloads the single file required, typically called winbugs14.exe, and runs that file to install in typical windows fashion. However, this allows only a limited use of WinBUGS, and one also need to install a key (again, this is free) for unlimited use. Instructions are on the WinBUGS home page, and please let me know if you experience any difficulty installing the program or the key. All computers in the basement of Purvis Hall have WinBUGS installed on them for your use.

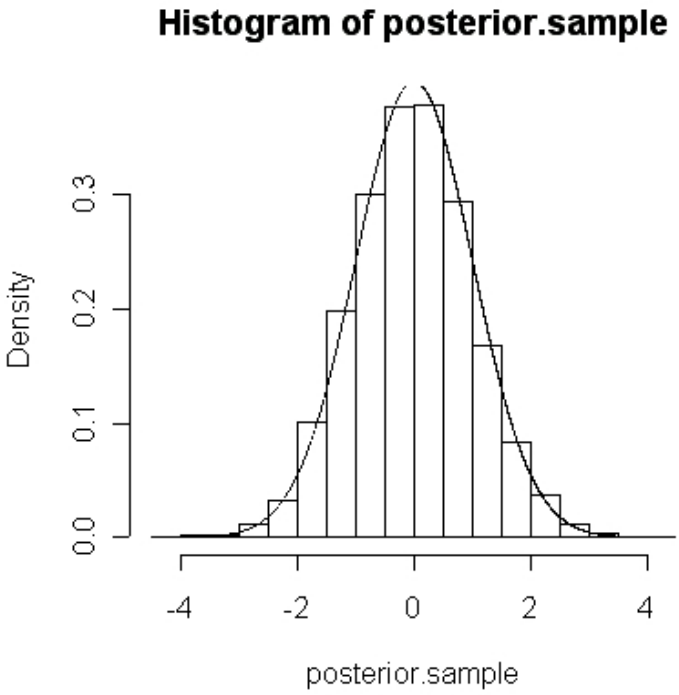
Once installed, a WinBUGS program consists of three parts, all of which can be placed into a single file, or as three separate files (or two files):

Main program: A set of lines that let WinBUGS know what the prior and likelihood of the model are.

Data set: The data that will be used, either formatted as an R like list, or in rectangular format. Note that you can have as many data sets as you wish, and can mix formatting types when entering data.

Initial Values: These are optional (but generally a good idea to include), and are used by WinBUGS to start its algorithm.

WinBUGS solves problems in Bayesian analysis by multiplying the prior by the likelihood, and then taking samples from the posterior distributions via an iterative algorithm called the Gibbs sampler. Thus, rather than get an exact formula for the posterior distribution, WinBUGS returns samples from it, as shown below:



WinBUGS can be used to solve a *very* wide class of problems, much wider than standard software packages. It must be used carefully, as the Gibbs sampler algorithm does not always converge. Good practice dictates to run WinBUGS several times (say, three times) to ensure that the answer is always the same (or extremely close). For problems in this course, convergence will seldom, if ever, be a problem.

Running WinBUGS

Follow these steps to produce analyses in WinBUGS:

1. Open WinBUGS by clicking on the WinBUGS icon on desktop (after installation).
2. Either open an existing WinBUGS file (typical extension is *.odc) if continuing from an already created file, or open a new window from the file menu.

3. Type in the program (or cut and paste if a program already exists, typically will be the case in this course).

4. Below the program (or if you prefer, in another window inside of WinBUGS), type in or cut and paste in the data set, remembering that the format must be either rectangular or list-like (see examples on course web page).

5. In list format, type in initial values (see examples on web page).

   For our normal problem with unknown variance, the WinBUGS window will typically look like this:

```
model
{                          #  Program must start with model
                           #  statement and open {
  for (i in 1:27)          #  Loop over number of data points
   {
    x[i] ~ dnorm(mu,tau) #  Enter the 27 likelihood terms
   }
  mu  ~ dnorm(70,0.04)     #  Prior for the normal mean, mu, 0.04=1/25
  sigma ~ dunif(0,50)      #  Prior for the normal standard deviation, sigma
  tau<-1/(sigma*sigma)     #  WinBUGS normal form needs 1/sigma^2
      }                    #  End of program part

#  Data                    #  Data section

list(x=c(76, 71, 82, 63, 76, 64, 64, 74, 70, 64, 75, 81, 75, 78, 66,
62, 79, 82, 78, 62, 72, 83, 79, 41, 80, 77, 67))

#  Initial Vlaues          #  Initial Values section

list(mu=60, sigma = 20)
```

6. Once all input is ready, WinBUGS must be run to produce the output, following these steps:

   (a) Open the Model $\rightarrow$ Specification menu item, highlight the word "model" in the program, and then click on "check model". This checks the model for syntax errors. Watch for (often quite cryptic) error messages on bottom status line, if any errors occur.

(b) Highlight the word "list" in the data section of the program, and click on "load data". This loads in your data. Repeat this step as many times as needed to load in all your data. If in rectangular array style, highlight the first line (variable names) rather than the word "list".

(c) Click on compile (assuming all is fine so far...if not, find error(s) and repeat first two steps).

(d) Highlight the word "list" in the initial values section of the program, and click on "load inits". This loads in your initial values. If you did not provide all initial values, click on "gen inits" to have WinBUGS do this automatically for you (not always a good idea).

(e) WinBUGS can now be run to produce posterior distributions. First, run some "burn-in" or "throw away" values, to ensure convergence. Typically, 2000 should be (more than) sufficient for models in this course. To create the burn-in, open the Model → Update menu item, change the default 1000 value to 2000, and click on the "update" button.

(f) To run further iterations "to keep", open the Inference → Samples menu item. Enter all unknown variables you wish inferences on into the window (in this case, mu and sigma), clicking "set" after each one. Then go back to the update box, which should still be open, and run the number of iterations you want. We will use 10,000, so enter 10000 and click again on the update button.

(g) Finally, we can look at summaries of the posterior distributions. Going back to the Samples dialogue box, type a " * " in the window (meaning look at all variables tracked), and then click on the various items to get the desired summaries. Most useful for us will be "Stats" and "density", to get basic posterior inferences such as means, medians and 95% credible intervals, and density plots (somewhat crude). The results are:

```
node  mean  sd    MC error 2.5%  median 97.5% start sample
mu    71.63 1.727 0.01698  68.24 71.62  74.97 2001  10000
sigma 9.69  1.43  0.01567  7.358 9.532  12.92 2001  10000
```

For the mean, note that our posterior has mean of about 71.6 (compare to 71.69 found previously), and the standard deviation for this parameter is 1.73, compared to 1.63 found previously. As we have an extra parameter to estimate, this decrease in the accuracy in

estimating μ is expected. Note that we have a new parameter to estimate, σ, which was assumed known before.

Having been introduced to WinBUGS, we can now look at Bayesian regression, first in general terms, then in terms of WinBUGS programming and inferences.

Brief Sketch of Bayesian regression

Recall the three steps: prior $\rightarrow$ likelihood $\rightarrow$ posterior.

1. We need a joint prior distribution over α, β, and σ. We will specify these as three independent priors [which when multiplied together will produce a joint prior]. One choice is:

- $\alpha \sim \text{uniform}[-\infty, +\infty]$
- $\beta \sim \text{uniform}[-\infty, +\infty]$
- $log(\sigma) \sim \text{uniform}[-\infty, +\infty]$

With this choice, we can approximate the results from a frequentist regression analysis. Another possible choice is (more practical for WinBUGS):

- $\alpha \sim \text{Normal}(0, 0.0001)$
- $\beta \sim \text{Normal}(0, 0.0001)$
- $\sigma \sim \text{uniform}[0, 100]$

These are very diffuse (typically, depends on scale of data), so approximate the first set of priors. Of course, if real prior information is available, it can be incorporated.

Notes:

- The need for the log in first set of priors comes from the fact the the variance must be positive. The prior on σ is equivalent to a density that is proportional to $\frac{1}{\sigma^2}$.
- We specify a non-informative prior distribution of these three parameters. Of course, we can also include prior information when available, but this is beyond the scope of this course.
- First set of priors is in fact "improper" because their densities do not integrate to one, since the area under these curves in infinite! In general this is to be avoided since sometimes it can cause problems with posterior distributions. This is *not* one of those problem cases, however, and it is sometimes convenient to use a "flat" prior everywhere, so it is mentioned here (even if we use proper priors for the rest of this lecture).

2. Likelihood function in regression:

- As is often the case, the likelihood function used in a Bayesian analysis is the same as the one used for the frequentist analysis.
- Recall that we have normally distributed residuals, $\epsilon \sim N(0, \sigma^2)$
- Recall that the mean of the regression line, given that we know α and β is $y = \alpha + \beta \times x$.
- Putting this together, we have $y \sim N(\alpha + \beta \times x, \sigma^2)$.
- So for a single patient with observed value x_i, we have $y \sim N(\alpha + \beta \times x_i, \sigma^2)$
- So for a single patient, the likelihood function is:

$$f(y_i) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{\frac{(y_i - (\alpha + \beta \times x_i))^2}{\sigma^2}\right\}$$

- So for a group of n patients each contributing data (x_i, y_i), the likelihood function is given by

$$\prod_{i=1}^{n} f(y_i) = f(y_1) \times f(y_2) \times f(y_3) \times \ldots \times f(y_n)$$

- So the likelihood function is simply a bunch of normal densities multiplied together ... a multivariate normal likelihood of dimension n.

3. Posterior densities in regression

- Bayes theorem now says to multiply the likelihood function (multivariate normal) with the prior. For the first prior set, the joint prior simply is $1 \times 1 \times \frac{1}{\sigma^2}$. For the second, it is two normal distributions multiplied by a uniform. Since the uniform is equal to 1 everywhere, this reduced to a product of two normal distributions.
- For the first prior set, the posterior distribution simply is:

$$\prod_{i=1}^{n} f(y_i) \times \frac{1}{\sigma^2}.$$

- This is a three dimensional posterior involving α, β, and σ^2.
- By integrating this posterior density, we can obtain the marginal densities for each of α, β, and σ^2.
- After integration (tedious details omitted):
  - $\alpha \sim t_{n-2}$
  - $\beta \sim t_{n-2}$
  - $\sigma^2 \sim$ Inverse Chi-Square (so $1/\sigma^2 \sim$ Chi-Square)
- Note the similar results given by Bayesian and frequentist approaches for α and β, and, in fact, the means and variances are the same as well, assuming the "infinite priors" listed above are used.
- When the second prior set is used, the resulting formulae are more complex, and WinBUGS will be used. However, as both sets of priors are very "diffuse" or "non-informative", numerically, results will be similar to each other, and both will be similar to the inferences given by the frequentist approach (but interpretations will be very different).
- Although the t distributions listed above can be used, Bayesian computations usually carried out via computer programs. We will use WinBUGS to compute Bayesian posterior distributions for regression and logistic problems.

- Bayes approach also suggests different ways to assess goodness of fit and model selection (we will see this later).

Revisit problem with DMF teeth

Recall that when we looked at simple linear regression from a frequentist view, we looked at a problem from public health in dentistry, with the following data:

Community Number	DMF per 100 children	Fluoride Concentration in ppm
1	236	1.9
2	246	2.6
3	252	1.8
4	258	1.2
5	281	1.2
6	303	1.2
7	323	1.3
8	343	0.9
9	412	0.6
10	444	0.5
11	556	0.4
12	652	0.3
13	673	0.0
14	703	0.2
15	706	0.1
16	722	0.0
17	733	0.2
18	772	0.1
19	810	0.0
20	823	0.1
21	1027	0.1

We will now reanalyze these data from a Bayesian viewpoint, using WinBUGS. The program we need is:

```
model
{
 for (i in 1:21)                    # loop over cities
    {
     mu.dmf[i] <- alpha + beta*fl[i] # regression equation
     dmf[i] ~ dnorm(mu.dmf[i],tau)   # distribution individual values
    }
    alpha ~ dnorm(0.0,0.000001)      # prior for intercept
    beta  ~ dnorm(0.0,0.000001)      # prior for slope
    sigma ~ dunif(0,400)             # prior for residual SD
    tau     <- 1/(sigma*sigma)       # precision required by WinBUGS
 for (i in 1:21)
    {                                # calculate residuals
     residual[i] <- dmf[i] - mu.dmf[i]
    }
pred.mean.1.7 <- alpha + beta*1.7         # mean prediction for fl=1.7
pred.ind.1.7 ~ dnorm(pred.mean.1.7, tau)  # individual pred for fl=1.7
}
```

Data are entered in one of the following two formats (both equivalent, note blank line required after "END" statement):

```
 dmf[]   fl[]
 236     1.9
 246     2.6
 252     1.8
 258     1.2
 281     1.2
 303     1.2
 323     1.3
 343     0.9
 412     0.6
 444     0.5
 556     0.4
 652     0.3
 673     0.0
 703     0.2
 706     0.1
 722     0.0
```

```
 733     0.2
 772     0.1
 810     0.0
 823     0.1
 1027    0.1
END
```

or

```
list(dmf=c( 236, 246, 252, 258, 281, 303, 323, 343, 412, 444, 556, 652,
673, 703, 706, 722, 733, 772, 810, 823, 1027), fl=c( 1.9, 2.6, 1.8, 1.2,
1.2, 1.2, 1.3, 0.9, 0.6, 0.5, 0.4, 0.3, 0.0, 0.2, 0.1, 0.0, 0.2, 0.1,
0.0, 0.1, 0.1))
```

We will use these initial values (rather arbitrary, just to get into right general area to start):

```
list(alpha=100, beta = 0, sigma=100)
```

Running 2000 burn-in values and then 10,000 further iterations, produces the following results:

node	mean	sd	2.5%	median	97.5%
alpha	730.1	41.45	646.8	730.6	809.7
beta	-277.4	40.86	-358.8	-277.4	-196.8
mu.dmf[1]	203.0	57.16	89.49	203.0	314.1
mu.dmf[2]	8.869	82.93	-152.8	8.885	171.1
mu.dmf[3]	230.8	53.72	124.3	230.6	335.6
mu.dmf[4]	397.2	35.98	325.8	397.5	467.1
mu.dmf[5]	397.2	35.98	325.8	397.5	467.1
mu.dmf[6]	397.2	35.98	325.8	397.5	467.1
mu.dmf[7]	369.5	38.42	293.3	369.6	444.2
mu.dmf[8]	480.4	30.81	418.9	480.4	540.8
mu.dmf[9]	563.7	30.09	503.0	563.4	621.8
mu.dmf[10]	591.4	30.94	529.2	591.4	651.0

mu.dmf[11]	619.1	32.29	554.6	619.3	680.9
mu.dmf[12]	646.9	34.08	578.9	647.1	712.3
mu.dmf[13]	730.1	41.45	646.8	730.6	809.7
mu.dmf[14]	674.6	36.24	602.0	675.1	744.8
mu.dmf[15]	702.4	38.72	624.6	702.7	777.0
mu.dmf[16]	730.1	41.45	646.8	730.6	809.7
mu.dmf[17]	674.6	36.24	602.0	675.1	744.8
mu.dmf[18]	702.4	38.72	624.6	702.7	777.0
mu.dmf[19]	730.1	41.45	646.8	730.6	809.7
mu.dmf[20]	702.4	38.72	624.6	702.7	777.0
mu.dmf[21]	702.4	38.72	624.6	702.7	777.0
pred.ind.1.7	257.3	145.8	-33.89	255.3	546.8
pred.mean.1.7	258.5	50.37	158.7	258.5	356.8
residual[1]	32.96	57.16	-78.05	32.97	146.5
residual[2]	237.1	82.93	75.25	237.1	398.8
residual[3]	21.22	53.72	-83.55	21.39	127.9
residual[4]	-139.2	35.98	-209.0	-139.5	-67.76
residual[5]	-116.2	35.98	-186.0	-116.5	-44.76
residual[6]	-94.22	35.98	-164.0	-94.46	-22.76
residual[7]	-46.48	38.42	-121.0	-46.57	29.72
residual[8]	-137.4	30.81	-197.8	-137.4	-75.79
residual[9]	-151.7	30.09	-209.8	-151.4	-90.83
residual[10]	-147.4	30.94	-206.9	-147.4	-85.25
residual[11]	-63.13	32.29	-124.9	-63.27	1.432
residual[12]	5.126	34.08	-60.23	4.903	73.22
residual[13]	-57.09	41.45	-136.7	-57.55	26.23
residual[14]	28.39	36.24	-41.83	27.93	101.0
residual[15]	3.647	38.72	-71.02	3.295	81.4
residual[16]	-8.092	41.45	-87.68	-8.55	75.23
residual[17]	58.39	36.24	-11.83	57.93	131.0
residual[18]	69.65	38.72	-5.02	69.3	147.4
residual[19]	79.91	41.45	0.324	79.45	163.2
residual[20]	120.6	38.72	45.98	120.3	198.4
residual[21]	324.6	38.72	250.0	324.3	402.4
sigma	135.0	21.83	98.52	132.5	183.2

Note how similar these results are to the frequentist results we calculated earlier in the course. Recall also that a quadratic model may fit better. Here is a program for this more complex model, which also illustrates Bayesian

regression for more than one variable.

```
model {
 for (i in 1:21)                  # loop over cities
    {
     mu.dmf[i] <- alpha + beta1*fl[i] + beta2*fl[i]*fl[i]
                                  # regression equation
     dmf[i] ~ dnorm(mu.dmf[i],tau)
                                  # distribution individual values
    }
    alpha ~ dnorm(0.0, 0.000001)  # prior for intercept
    beta1  ~ dnorm(0.0, 0.000001) # prior for slope for fl
    beta2  ~ dnorm(0.0, 0.000001) # prior for slope for fl squared
    sigma ~ dunif(0,200)          # prior for residual SD
   tau     <- 1/(sigma*sigma)     # precision required by WinBUGS
 for (i in 1:21)
   {                              # calculate residuals
    residual[i] <- dmf[i] - mu.dmf[i]
   }
pred.mean.1.7 <- alpha + beta1*1.7 + beta2*1.7*1.7
                                  # mean prediction for fl=1.7
pred.ind.1.7 ~ dnorm(pred.mean.1.7, tau)
                                  # individual pred for fl=1.7
}
```

Running this model, the results are:

node	mean	sd	2.5%	median	97.5%
alpha	808.6	34.31	739.8	808.4	875.5
beta1	-624.2	87.07	-792.6	-624.9	-448.1
beta2	161.5	38.16	85.78	162.1	236.3
pred.ind.1.7	214.7	105.0	3.166	213.7	424.9
pred.mean.1.7	214.2	37.34	140.5	213.8	287.7
sigma	94.4	17.54	67.35	91.79	135.8

Once again, results are very similar to the frequentist approach we calculated earlier.

Comparing Bayesian to Frequentist Regression

- If no prior information is used (i.e., if we use a Bayesian approach with "noninformative" or "flat" or "diffuse"or "reference" priors), then the inferences from Bayesian and frequentist are numerically similar. For example, 95% confidence intervals will be very similar to 95% credible intervals. We have illustrated this in the above examples.

- However, the *interpretations* of these intervals are completely different: Bayesian intervals are directly interpreted as the probability the parameter is in the credible interval, given the data and any prior information. Frequentist confidence intervals cannot be interpreted this way, one can only say that if the confidence interval procedure were to be used repeatedly, then 95% of all intervals will contain the true value.

- If there is prior information, then only the Bayesian approach can formally include this information in the model.

- Both Bayesian and frequentist approaches can be extended to much more complex models. However Bayesian methods tend to be more flexible, in general.

- Using WinBUGS, it is trivially easy to obtain inferences for any function of any estimated parameter. For example, having estimated, say, β_1 and β_2, it is trivial to obtain the posterior distribution for, say, $\beta_1 \times \sqrt{\beta_2}$.

- As we will soon see, Bayesian methods for model selection tend to work better than standard frequentist methods (although there is no single perfect method for model selection developed so far).