ADVANCED BAYESIAN MODELING

Shakespeare Plays: JAGS Analysis

We use JAGS to fit the Shakespeare plays model – for illustration only.

The classical (non-Bayes) linear regression analysis would be adequate in this case – essentially equivalent to the noninformative prior model.

Start with classical analysis, since its estimates can be used to help initialize and make sure priors are not over-constrained ...

Initial Classical Analysis

```
> mod <- lm(Speeches ~ 0 + Genre + scale(Words), data=ss)
```

Note:

- "`0 +`" prevents an explicit intercept from being added
- `Genre` is automatically coded using indicators, since it is a `factor`
- "`scale`" standardizes its argument

```
> summary(mod)

...

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
GenreComedy   1004.75      41.18  24.397  < 2e-16 ***
GenreHistory   799.31      43.42  18.409  < 2e-16 ***
GenreTragedy  1003.01      42.00  23.880  < 2e-16 ***
scale(Words)   156.89      26.98   5.815 1.66e-06 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 140.4 on 33 degrees of freedom
Multiple R-squared:  0.9811,Adjusted R-squared:  0.9789
F-statistic: 429.2 on 4 and 33 DF,  p-value: < 2.2e-16
```

All β estimates are about 100 to 1000, with much smaller standard errors.

We choose to set

- Initial β values at ± 10000
- Prior β variance at

$$\sigma_\beta^2 \quad = \quad (10000)^2$$

Regression error variance (σ^2) estimate is about $(140)^2 \approx 20000$.

We'll choose initial σ^2 values of 100 and 1000000.

In JAGS we use the equivalent prior (BDA3, Table A.1)

$$(\sigma^2)^{-1} \mid X \quad \sim \quad \text{Gamma}(\nu_0/2,\, \nu_0\sigma_0^2/2)$$

which is less informative as $\nu_0 \to 0$.

Arbitrarily, we set

$$\nu_0 \;\; = \;\; 0.0002 \qquad\qquad \sigma_0^2 \;\; = \;\; 1$$

JAGS Analysis 1

```
model {

  for (i in 1:length(y)) {
    y[i] ~ dnorm(beta1*x1[i] + beta2*x2[i] + beta3*x3[i] + beta4*x4[i],
                 sigmasqinv)
  }

  beta1 ~ dnorm(0, 0.00000001)
  beta2 ~ dnorm(0, 0.00000001)
  beta3 ~ dnorm(0, 0.00000001)
  beta4 ~ dnorm(0, 0.00000001)
  sigmasqinv ~ dgamma(0.0001, 0.0001)

  sigmasq <- 1/sigmasqinv

}
```

The data (in data frame ss) must be converted to appropriate numerical variables:

```
> d1 <- list(y  = ss$Speeches,
+             x1 = as.numeric(ss$Genre == "Comedy"),
+             x2 = as.numeric(ss$Genre == "History"),
+             x3 = as.numeric(ss$Genre == "Tragedy"),
+             x4 = as.vector(scale(ss$Words)))
```

Set up initial values for 4 chains:

```
> inits1 <- list(list(beta1=10000, beta2=10000, beta3=10000, beta4=10000,
+                     sigmasqinv=0.01),
+                list(beta1=10000, beta2=10000, beta3=-10000, beta4=-10000,
+                     sigmasqinv=0.01),
+                list(beta1=10000, beta2=-10000, beta3=10000, beta4=-10000,
+                     sigmasqinv=0.000001),
+                list(beta1=10000, beta2=-10000, beta3=-10000, beta4=10000,
+                     sigmasqinv=0.000001))
```

Set up and run chains as usual:

```
> library(rjags)
...

> m1 <- jags.model("shakespeare1.bug", d1, inits1, n.chains=4, n.adapt=1000)

...

> update(m1, 1000)  # burn-in
  |**************************************************| 100%

> x1 <- coda.samples(m1, c("beta1","beta2","beta3","beta4","sigmasq"),
+                    n.iter=2000)
  |**************************************************| 100%
```

```
> gelman.diag(x1, autoburnin=FALSE)
Potential scale reduction factors:

        Point est. Upper C.I.
beta1            1          1
beta2            1          1
beta3            1          1
beta4            1          1
sigmasq          1          1

Multivariate psrf

1
```

Graphical diagnostics (not shown) also show no problems with chains.

```
> effectiveSize(x1)
   beta1    beta2    beta3    beta4  sigmasq
5263.724 7018.103 6950.444 4451.034 6547.985
```

Sample sizes more than adequate.

```
> summary(x1)

Iterations = 1001:3000
Thinning interval = 1
Number of chains = 4
Sample size per chain = 2000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

           Mean      SD Naive SE Time-series SE
beta1    1005.5   42.17   0.4715         0.5822
beta2     800.2   45.00   0.5031         0.5389
beta3    1002.2   43.55   0.4869         0.5260
beta4     157.6   27.91   0.3120         0.4265
sigmasq 21056.6 5520.99  61.7265        68.2147
```

(Compare with classical results – very similar.)

```
2. Quantiles for each variable:

           2.5%     25%     50%     75%   97.5%
beta1     921.6   977.2  1005.6  1033.4  1087.6
beta2     711.2   770.4   799.9   829.8   889.3
beta3     916.6   973.3  1002.5  1030.6  1088.2
beta4     102.8   139.3   157.5   176.1   212.7
sigmasq 12877.2 17150.2 20174.2 24067.9 34366.2
```

How probable is $\beta_1 < \beta_3$? How about $\beta_2 < \beta_1$?

```
> post.samp <- as.matrix(x1)

> mean(post.samp[,"beta1"] < post.samp[,"beta3"])
[1] 0.486
>   # do comedies have fewer speeches than tragedies, adjusting for words?

> mean(post.samp[,"beta2"] < post.samp[,"beta1"])
[1] 0.999125
>   # do histories have fewer speeches than comedies, adjusting for words?
```

So, after adjusting for words,

- No apparent evidence that comedies have fewer speeches than tragedies
- Strong evidence that histories have fewer speeches than comedies

The once-disputed comedy *The Two Noble Kinsmen* (not in the data set) has 25424 words.

A 95% central posterior interval for the *mean* number of speeches we expect it to have:

```
> quantile(post.samp[,"beta1"] +
+          post.samp[,"beta4"] * (25424 - mean(ss$Words))/sd(ss$Words),
+          c(0.025,0.975))
    2.5%    97.5%
1017.114 1230.824
```

Note: For the *actual* number of speeches, need a posterior *predictive* interval.

JAGS Analysis 2

We can use an alternative version of the JAGS model that generalizes more easily ...

```
model {

  for (i in 1:length(y)) {
    y[i] ~ dnorm(inprod(X[i,], beta), sigmasqinv)
  }

  beta ~ dmnorm(beta0, Sigmabetainv)
  sigmasqinv ~ dgamma(0.0001, 0.0001)

  sigmasq <- 1/sigmasqinv

}
```

- `inprod` is the (vector) inner product
- `dmnorm` is the multivariate normal distribution, in a parameterization using the **precision matrix** (inverse covariance matrix)
- `beta0` and `Sigmabetainv` are specified with the data

Now the data (and constants in the priors) can be set up as

```
> d2 <- list(y = ss$Speeches,
+            X = model.matrix(~ 0 + Genre + scale(Words), data=ss),
+            beta0 = c(0,0,0,0),
+            Sigmabetainv = rbind(c(0.00000001, 0, 0, 0),
+                                 c(0, 0.00000001, 0, 0),
+                                 c(0, 0, 0.00000001, 0),
+                                 c(0, 0, 0, 0.00000001)))
```

Initialization of `beta` is also modified:

```
> inits2 <- list(list(beta=c(10000, 10000, 10000, 10000),
+                      sigmasqinv=0.01),
+                 list(beta=c(10000, 10000, -10000, -10000),
+                      sigmasqinv=0.01),
+                 list(beta=c(10000, -10000, 10000, -10000),
+                      sigmasqinv=0.000001),
+                 list(beta=c(10000, -10000, -10000, 10000),
+                      sigmasqinv=0.000001))
```

Results are almost exactly the same as before, except the Gibbs sampler runs a bit more efficiently (because of block updating).