ADVANCED BAYESIAN MODELING

JAGS Analysis: Second Attempt

New Loglinear Rate Model

$$y_{ijk} \mid \beta, t_{ijk} \sim \text{indep. Poisson}(\lambda_{ijk}) \qquad \lambda_{ijk} = t_{ijk}\, r_{ijk}$$

$$\log \lambda_{ijk} = \log(t_{ijk}\, r_{ijk}) = \log t_{ijk} + \beta^t_i + \beta^e_j + \beta^p_k$$

$$\beta^t_i \mid \sigma^2_t \sim \text{iid N}(0, \sigma^2_t) \qquad \beta^e_j \mid \sigma^2_e \sim \text{iid N}(0, \sigma^2_e)$$

$$\beta^p_k \sim \text{iid N}(0, 100^2) \qquad \sigma_t, \sigma_e \sim \text{iid U}(0, 100)$$

Note: No explicit intercept.

Role of intercept is implicitly represented by β_1^p and β_2^p.

In file ships2.bug:

```
model {
  for (i in 1:length(incidents)) {
    incidents[i] ~ dpois(lambda[i])
    log(lambda[i]) <- logservice[i] + beta.t[type[i]] + beta.e[era[i]] +
                      beta.p[period[i]]

    incidentsrep[i] ~ dpois(lambda[i])
  }
```

(continued ...)

```
  for (i in 1:max(type)) {
    beta.t[i] ~ dnorm(0, 1/sigma.t^2)
  }
  for (j in 1:max(era)) {
    beta.e[j] ~ dnorm(0, 1/sigma.e^2)
  }
  beta.p[1] ~ dnorm(0, 0.0001)
  beta.p[2] ~ dnorm(0, 0.0001)

  sigma.t ~ dunif(0, 100)
  sigma.e ~ dunif(0, 100)
}
```

Set up data and initializations for four chains:

```
> d2 <- list(incidents = shipssub$incidents,
+            logservice = log(shipssub$service),
+            type = unclass(shipssub$type),
+            era = unclass(factor(shipssub$year)),
+            period = unclass(factor(shipssub$period)))

> inits2 <- list(list(sigma.t=90, sigma.e=90, beta.p=c(100,100)),
+                list(sigma.t=0.01, sigma.e=90, beta.p=c(-100,100)),
+                list(sigma.t=90, sigma.e=0.01, beta.p=c(100,-100)),
+                list(sigma.t=0.01, sigma.e=0.01, beta.p=c(-100,-100)))
```

```
> library(rjags)
...

> m2 <- jags.model("ships2.bug", d2, inits2, n.chains=4, n.adapt=1000)
...

> update(m2, 1000)  # burn-in
  |**************************************************| 100%

> x2 <- coda.samples(m2, c("sigma.t","sigma.e","beta.p"), n.iter=2000)
  |**************************************************| 100%
```

```
> gelman.diag(x2, autoburnin=FALSE)
Potential scale reduction factors:

          Point est. Upper C.I.
beta.p[1]      45.12      96.38
beta.p[2]      45.06      96.37
sigma.e         1.44       2.05
sigma.t         1.30       1.79

Multivariate psrf

37
```

After repeatedly running the chains and checking for convergence, doubling the number of iterations each time, I found approximate convergence after more than a million iterations.

Run more to check effective sample size:

```
> x2 <- coda.samples(m2, c("sigma.t","sigma.e","beta.p","lambda","incidentsrep"),
+                    n.iter=1000)
  |************************************************| 100%

> effectiveSize(x2[,c("sigma.t","sigma.e","beta.p[1]","beta.p[2]")])
  sigma.t   sigma.e beta.p[1] beta.p[2]
681.84694 753.16071  63.25891  62.15367
```

Not enough for β_1^p and β_2^p.

Collect more iterations, but with thinning, so that subsequent analyses will be easier to manage (fewer iterates):

```
> x2 <- coda.samples(m2, c("sigma.t","sigma.e","beta.p","lambda","incidentsrep"),
+                    n.iter=100000, thin=50)
  |************************************************| 100%

> effectiveSize(x2[,c("sigma.t","sigma.e","beta.p[1]","beta.p[2]")])
  sigma.t   sigma.e beta.p[1] beta.p[2]
3983.6399 1668.9500  937.6624  937.3023
```

```
> summary(x2[,c("sigma.t","sigma.e","beta.p[1]","beta.p[2]")])

Iterations = 1025050:1125000
Thinning interval = 50
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
sigma.t    0.6052 0.4737 0.005296        0.01108
sigma.e    0.7468 0.7657 0.008561        0.02151
beta.p[1] -6.1178 0.5761 0.006441        0.02025
beta.p[2] -5.7240 0.5693 0.006365        0.01992
```

```
2. Quantiles for each variable:

              2.5%      25%      50%      75%  97.5%
sigma.t     0.1910   0.3504   0.4852   0.7045  1.728
sigma.e     0.2021   0.3665   0.5316   0.8350  2.634
beta.p[1]  -7.2758 -6.4061 -6.1158 -5.8325 -4.964
beta.p[2]  -6.9084 -5.9981 -5.7196 -5.4462 -4.576
```

σ_t and σ_e do not appear to be overly constrained by their prior upper bound of 100, nor do β_1^p and β_2^p seem overly affected by their prior standard deviation of 100.

Perform a chi-square posterior predictive check for overdispersion:

```
> lambdas <- as.matrix(x2)[, paste("lambda[",1:nrow(shipssub),"]", sep="")]

> incidentsrep <- as.matrix(x2)[, paste("incidentsrep[",1:nrow(shipssub),"]",
+                                       sep="")]

> Tchi <- numeric(nrow(incidentsrep))
> Tchirep <- numeric(nrow(incidentsrep))
> for(s in 1:nrow(incidentsrep)){
+   Tchi[s] <- sum((shipssub$incidents - lambdas[s,])^2 / lambdas[s,])
+   Tchirep[s] <- sum((incidentsrep[s,] - lambdas[s,])^2 / lambdas[s,])
+ }

> mean(Tchirep >= Tchi)
[1] 0.071125
```

The posterior predictive p-value is suspiciously small, although not exceedingly so.

This could alternatively be an indicator of outliers, but a separate check (results not shown) reveals no such problem.

We conclude that there could be a bit of overdispersion (consistent with classical analyses published elsewhere), but we will ignore the problem for now.

What is the posterior probability that the rate of damaging incidents was higher in the earlier period (1960–74) than in the later (75–79)?

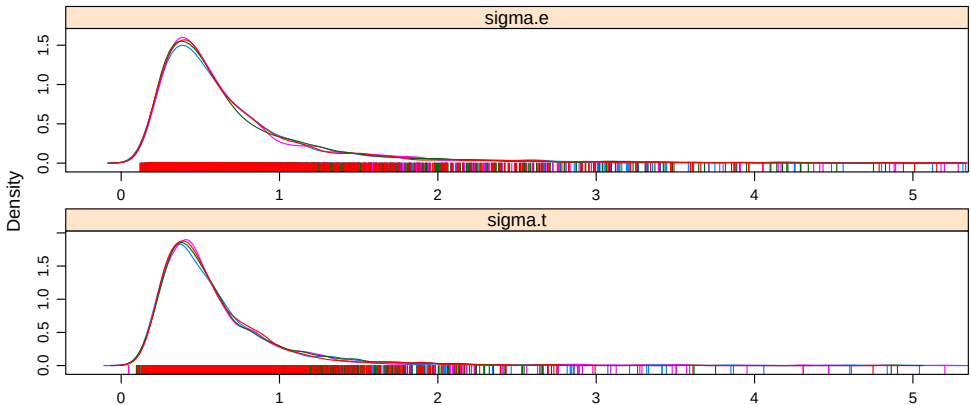
```
> beta.p <- as.matrix(x2)[, c("beta.p[1]","beta.p[2]")]

> mean(beta.p[,1] > beta.p[,2])
[1] 5e-04
```

So it seems that the rate was actually *lower* in the earlier period.

What do densities for the square roots of the variance components look like?

```
> densityplot(x2[, c("sigma.t","sigma.e")], xlim=c(0,5))
```


sigma.e
sigma.t
Density
0.0
0.5
1.0
1.5
0
1
2
3
4
5

The densities seem very low near zero, supporting the idea that type of ship and era of construction really are related to rate of damaging incidents.

Computing DIC (the JAGS version) gives:

```
> dic.samples(m2, 100000)
  |**************************************************| 100%
Mean deviance:  145.7
penalty 8.514
Penalized deviance: 154.2
```

There are effectively about 8.5 parameters.

Compare with the corresponding classical fixed-effect ANOVA (main effects only), which has effectively nine parameters.