# Missing and censored data in Bayesian models: optional practical exercise

Christopher Jackson

## 1 Outcomes missing not at random

The Mini Mental State Examination (MMSE) is a measure of cognitive function used in the diagnosis of dementia. It can take values from 30 (representing no cognitive impairment) to 0 (the most severe), and we assume it is continuous.

A person in a longitudinal study had four MMSE observations of 28, 26, 27 and 25 respectively at baseline, and years 5, 10 and 15 after baseline. A planned fifth observation at 20 years was missing. The data are specified in R as

```
dat <- list(t = c(0, 5, 10, 15, 20),
            y = c(28, 26, 27, 25, NA) )
plot(dat$t, dat$y, ylim=c(0, 30)) # quick visualisation
```

We wish to fit a standard linear regression model of the form $y_i \sim N(\mu_i, \sigma_i^2)$, $\mu_i = \alpha + \beta x_i$, where $y_i$ is the MMSE value, and $x_i$ is the year after the baseline measurement.

(a) Devise suitable priors for the intercept $\alpha$, slope $\beta$ and error standard deviation $\sigma = 1/\sqrt{\tau}$, given that

- MMSE is bounded by definition
- from previous knowledge, we are 97.5% certain that MMSE cannot increase over time, and 97.5% certain that after 1 year it cannot drop more than 20 points (note that a 95% credible interval is about $\pm 2$ standard deviations in the normal distribution).

*Note there is no single correct answer: the point is to practice translating prior beliefs to distributions. Indeed the normal error model is not strictly realistic given that MMSE is bounded.*

(b) Fit the model with the given priors, and the template JAGS model, initial values and data provided, and compare the posteriors to the priors.

```
ini <- list(alpha=20, beta=-10, sigma=1)
mmse.mod <- "
model {
 for (i in 1:5){
  y[i] ~ dnorm(mu[i], tau)
  mu[i] <- alpha + beta*t[i]
 }

 ### INSERT PRIOR DISTRIBUTIONS HERE
 alpha ~ ??
 beta ~  ??
 sigma ~ ??

 tau <- 1/(sigma*sigma)
}
"
```

(c) Suppose now that the odds of missing a MMSE measurement depend on the measurement itself, such the odds are doubled if MMSE is 5 units lower. Adapt the model, data and initial values to include the non-random missingness mechanism (see lecture slides), run and compare the results.

## Solutions: outcomes missing not at random

(a) Priors.

$\alpha$: MMSE is bounded on this range, so the intercept (equal to the baseline value) is bounded by definition on this range. We have no other prior information to restrict it, so a uniform is reasonable. Alternatively we could use a distribution truncated on this range, or a linearly-transformed beta distribution, if we wanted a peak at somewhere between 0 and 30.

$\beta$: This is the expected change in MMSE after one year. "97.5% certain that MMSE cannot increase over time, and 97.5% certain that after 1 year it cannot drop more than 20 points" implies a prior mean halfway between 0 and -20 for the expected change after one year. Then interpreting the range of (-20, 0) as the 2.5% and 97.5% quantiles, or +- 2 standard deviations, the prior standard deviation is then the range width divided by 4, equal to 5 units of MMSE, equivalently a prior precision of $1/(5*5) = 0.04$.

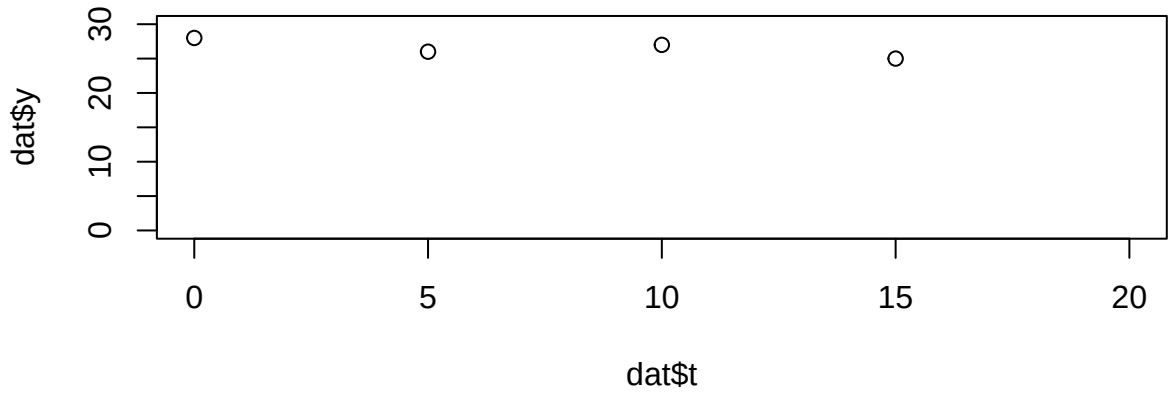
$\sigma$: As noted in the question, the normal error model is not strictly realistic given that MMSE is bounded on (0,30). Even so, we do not want to allow a-priori standard deviations for MMSE which permit MMSE values outside (0,30) with a large probability. Thus it is unlikely to be much over 10. A uniform is reasonable, alternatively we might use a positive distribution (sugh as the exponential or gamma) for the SD, variance or precision, concentrated on values below about 10.

(b) Fit model

```
mmse.mod <- "
model {
 for (i in 1:5){
  y[i] ~ dnorm(mu[i], tau)
  mu[i] <- alpha + beta*t[i]
 }

 ### PRIOR DISTRIBUTIONS
 alpha ~ dunif(0, 30)
 beta ~ dnorm(-10, 0.04)
 sigma ~ dunif(0, 10)

 tau <- 1/(sigma*sigma)
}
"
dat <- list(t = c(0, 5, 10, 15, 20),
        y = c(28, 26, 27, 25, NA) )
plot(dat$t, dat$y, ylim=c(0, 30)) # quick visualisation
```



```
ini <- list(list(alpha=20, beta=-10, sigma=1),
        list(alpha=29, beta=10, sigma=5))

library(rjags)
mmse.jag <- jags.model(textConnection(mmse.mod), dat, ini, n.chains=2)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 4
##    Unobserved stochastic nodes: 4
##    Total graph size: 31
##
## Initializing model
##
##   |                                                              |
```

```
update(mmse.jag, 1000)
```

```
##   |                                                              |
```

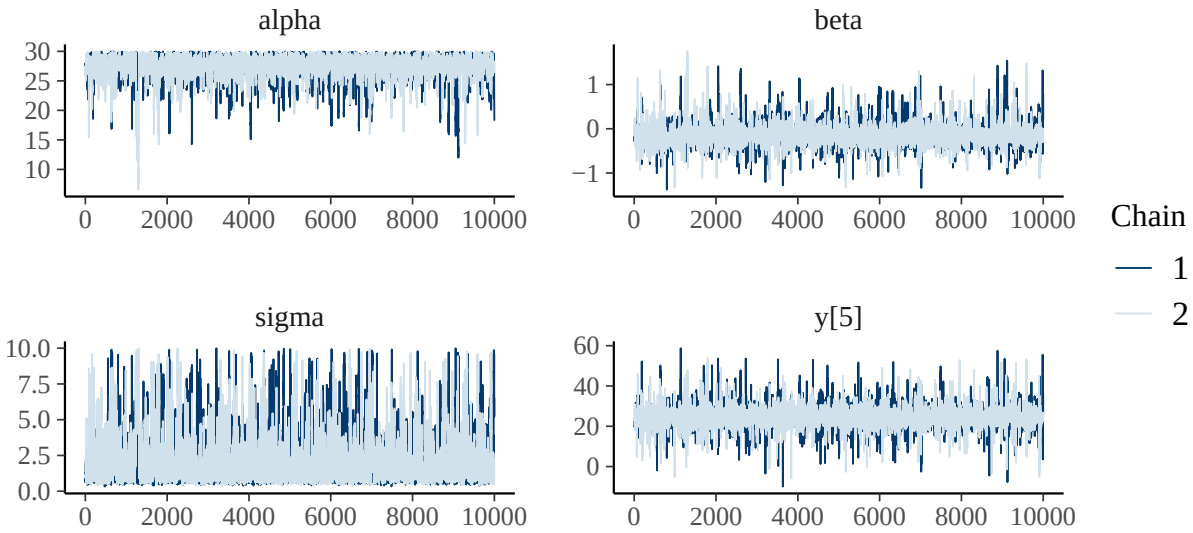
```
sam <- coda.samples(mmse.jag, var=c("sigma","alpha","beta","y[5]"), n.iter=10000)
```

```
##   |                                                              |
```

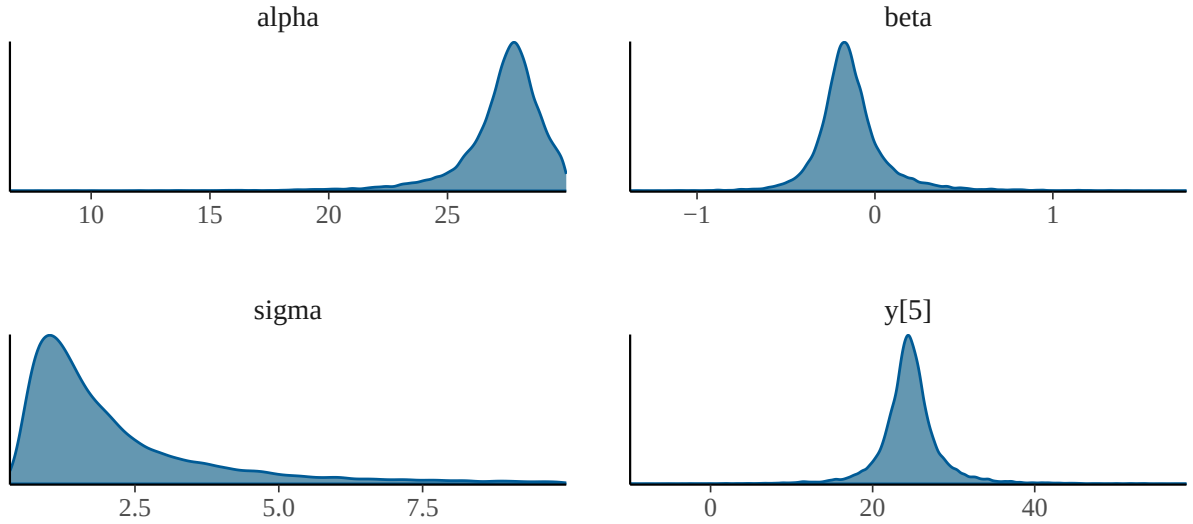
```
summary(sam)
```

```
##
## Iterations = 2001:12000
## Thinning interval = 1
## Number of chains = 2
## Sample size per chain = 10000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##         Mean     SD Naive SE Time-series SE
## alpha 27.278 1.8948  0.01340       0.051428
## beta  -0.134 0.2121  0.00150       0.004393
## sigma  2.164 1.7160  0.01213       0.045338
## y[5]  24.597 4.2215  0.02985       0.043599
##
## 2. Quantiles for each variable:
##
##          2.5%     25%     50%      75%   97.5%
## alpha 22.3393 26.7119 27.6291 28.36061 29.6965
## beta  -0.4859 -0.2387 -0.1575 -0.06224  0.3737
## sigma  0.5755  1.0414  1.5699  2.61234  7.4471
## y[5]  16.1273 22.9043 24.4997 26.13128 33.7240
```

```
# convergence check
library(bayesplot)
mcmc_trace(sam)
```




```
mcmc_dens(sam)
```




```
library(posterior)
draws <- as_draws(sam)
summary(draws,
    ~quantile(.x, probs=c(0.025, 0.5, 0.975)),
    ~mcse_quantile(.x, probs=c(0.025, 0.5, 0.975)))
```

```
## # A tibble: 4 x 7
##   variable `2.5%`  `50%` `97.5%` mcse_q2.5 mcse_q50 mcse_q97.5
##   <chr>     <num>  <num>   <num>     <num>    <num>      <num>
## 1 alpha    22.3   27.6    29.7     0.444    0.0203      0.0167
## 2 beta     -0.486 -0.157   0.374   0.00752  0.00204     0.0342
## 3 sigma     0.576  1.57    7.45    0.00478  0.0274      0.232
## 4 y[5]     16.1   24.5    33.7     0.361    0.0265      0.396
```

```
y5_mar <- extract_variable(sam,"y[5]") # save for comparison in (c)
```

The posterior distributions are concentrated compared to their priors, but they are all likely to be sensitive to the exact choice of prior, given only 4 data points.

Note the large Monte Carlo error for the upper quantile of $\sigma$. This posterior is skewed, and with this

number of samples there is only about 1 significant figure of accuracy.

(c) Accounting for non-random missingness mechanism

```
mmse.miss.mod <- "
model {
 for (i in 1:5){
  y[i] ~ dnorm(mu[i], tau)
  mu[i] <- alpha + beta*t[i]

  ## Model for missingness
  miss[i] ~ dbern(p[i])
  logit(p[i]) <- a - b*y[i]/5 # OR doubled (b=log(2)) if MMSE 5 units lower
 }

 alpha ~ dunif(0, 30)
 beta ~ dnorm(-10, 0.04)
 sigma ~ dunif(0, 10)
 tau <- 1/(sigma*sigma)

 ## Priors for missingness model parameters
 a ~ dlogis(0, 1)
 b <- log(2) #
}
"

### Add a new variable "miss" to the data
dat <- list(t = c(0, 5, 10, 15, 20),
        y = c(28, 26, 27, 25, NA),
        miss = c(0, 0 , 0, 0, 1) )
### Extend the initial values
ini <- list(list(alpha=20, beta=-10, sigma=1, a=0),
        list(alpha=29, beta=10, sigma=5, a=1))

mmse.jag <- jags.model(textConnection(mmse.miss.mod), dat, ini, n.chains=2)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 9
##    Unobserved stochastic nodes: 5
##    Total graph size: 60
##
## Initializing model
##
##   |                                                      |
```

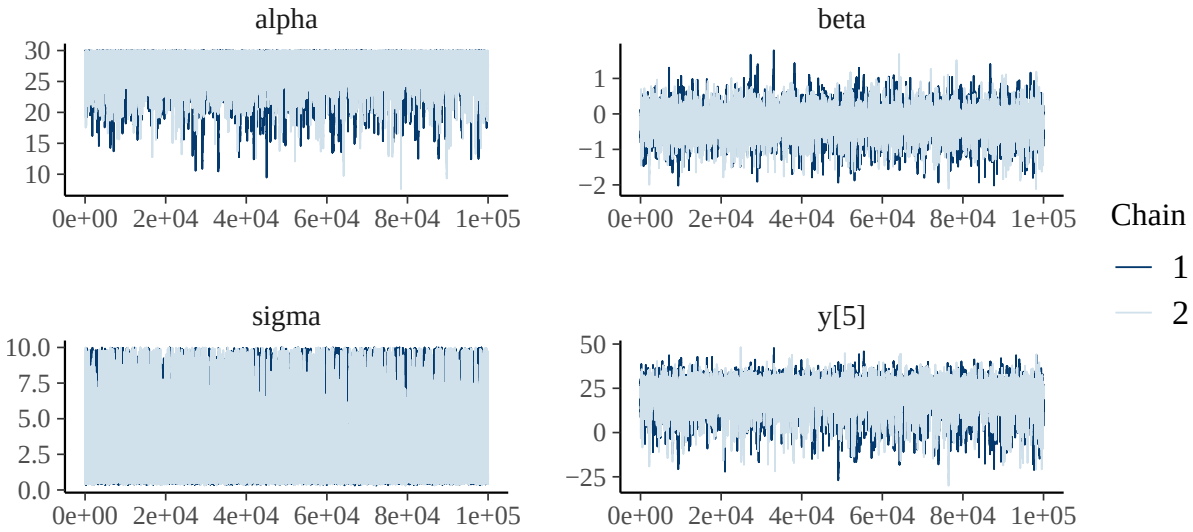
```
update(mmse.jag, 1000)
```

```
##   |                                                      |
```

```
sam.miss <- coda.samples(mmse.jag, var=c("sigma","alpha","beta","y[5]"), n.iter=100000)
```

```
##   |                                                      |
```

```
## convergence check
mcmc_trace(sam.miss)
```




```
draws <- as_draws(sam.miss)
summary(draws,
    ~quantile(.x, probs=c(0.025, 0.5, 0.975)),
    ~mcse_quantile(.x, probs=c(0.025, 0.5, 0.975)))
```

```
## # A tibble: 4 x 7
##   variable `2.5%`  `50%` `97.5%` mcse_q2.5 mcse_q50 mcse_q97.5
##   <chr>     <num>  <num>   <num>     <num>    <num>      <num>
## 1 alpha    22.7   27.7    29.8      0.110    0.00769    0.00423
## 2 beta     -0.705 -0.187   0.217    0.00944  0.00101    0.00695
## 3 sigma     0.584  1.68    8.34     0.00190  0.0125     0.0649
## 4 y[5]      7.23  23.6    29.5      0.351    0.0211     0.0768
```

```
y5 <- extract_variable(sam.miss,"y[5]")
# alternative way to extract sample for a single variable
# that doesn't need the "posterior" package
# y5 <- unlist(sam.miss[,"y[5]"])
mean(y5 <  20)
```

```
## [1] 0.18645
```

Note the estimate of `beta` is lower, that is, the MMSE decline is estimated to be quicker.

The estimated probability of the missing measurement being below 20 has increased from 0.08 to 0.19.