R2WinBUGS tutorial

By
Hedibert Freitas Lopes
Graduate School of Business
University of Chicago

Nonlinear growth curve

Carlin and Gelfand (1991) present a nonconjugate Bayesian analysis of the following data set from Ratkowsky (1983):

Dugong (sea cows)	1	2	3	...	26	27
Age (X)	1.00	1.50	1.50	...	29.0	31.50
Length (Y)	1.80	1.85	1.87	...	2.27	2.57

Carlin and Gelfand (1991) model this data using a nonlinear growth curve with no inflection point and an asymptote as x_i tends to infinity:

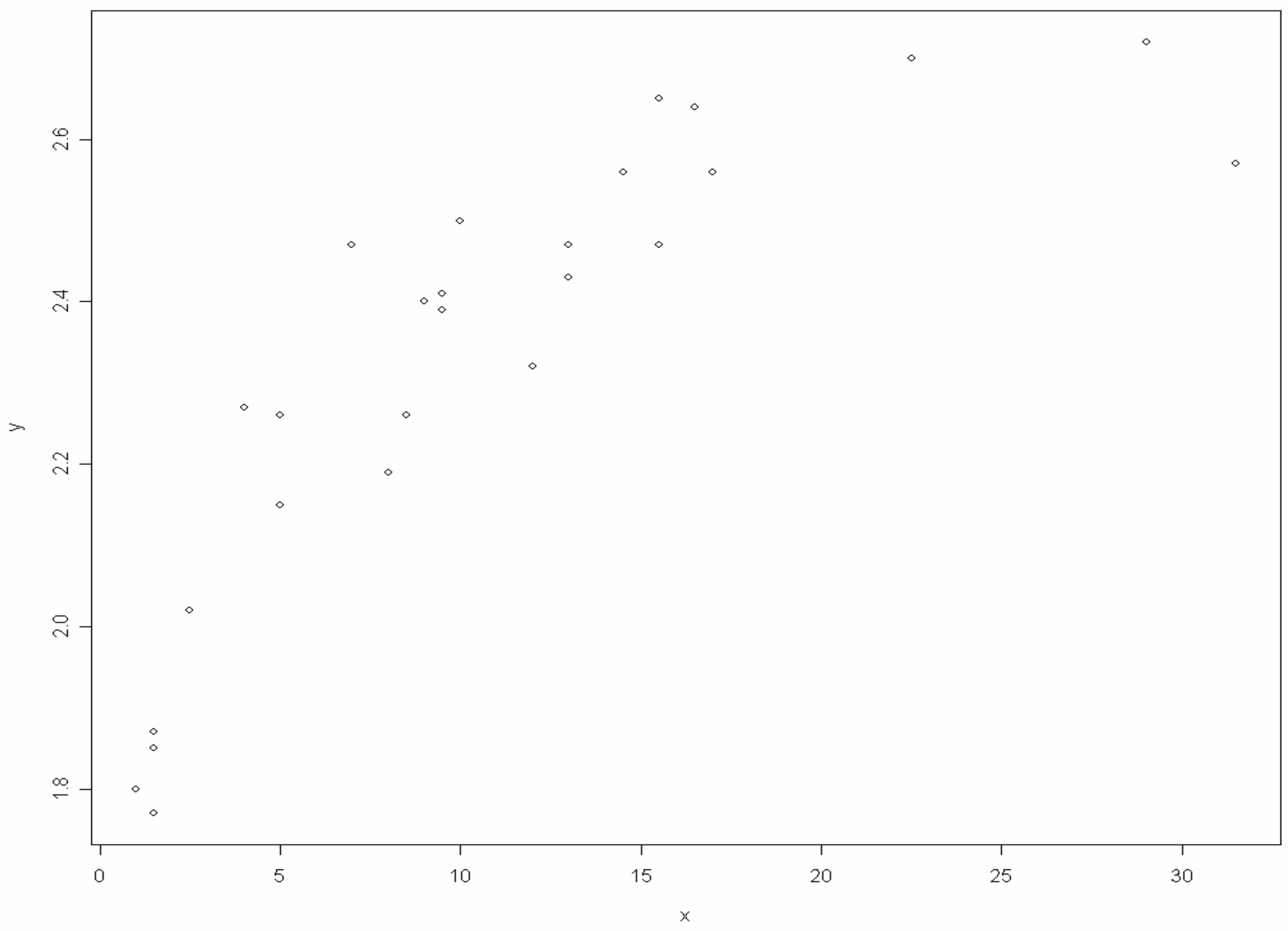
$$\begin{aligned} y_i &\sim N(\mu_i, \tau^{-1}) \\ \mu_i &= \alpha - \beta\gamma^{x_i} \end{aligned}$$

for $i = 1, \ldots, 27$, $\alpha, \beta > 1$ and $0 < \gamma < 1$.

Standard noninformative priors are adopted for α, β and τ, and a uniform prior on (0,1) is assumed for γ.

Example from the winbugs manual at http://www.mrc-bsu.cam.ac.uk/bugs/

Data



Why use R2WinBUGS?

No need to open WinBUGS.

Automatic procedure.

Direct interface with R and its potentials.

Only one piece of WinBUGS code necessary

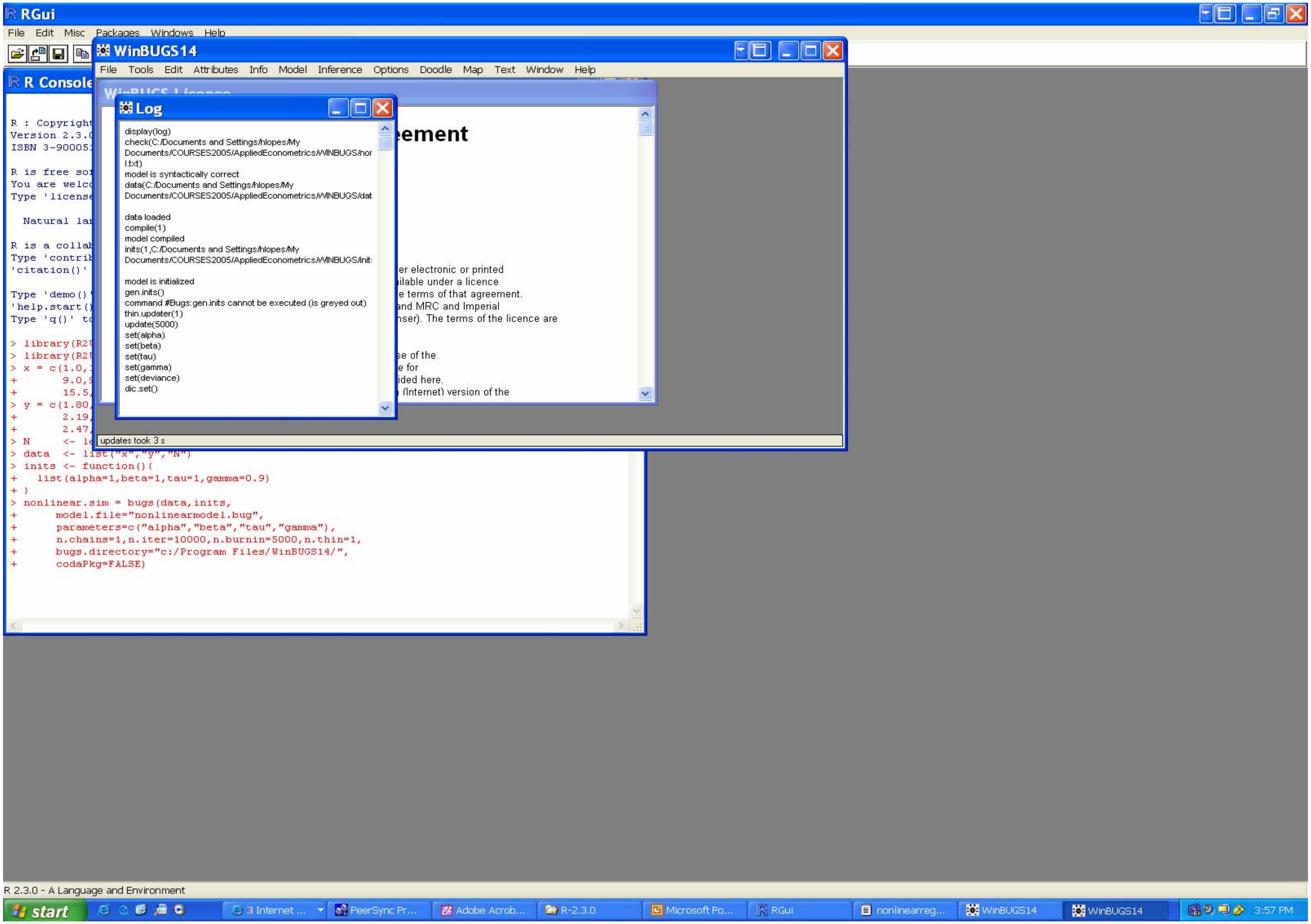
```
# This is file “nonlinearmodel.bug”
model{
  for( i in 1 : N ) {
    y[i] ~ dnorm(mu[i], tau)
    mu[i] <- alpha - beta * pow(gamma,x[i])
  }
  alpha ~ dnorm(0.0, 1.0E-6)
  beta ~ dnorm(0.0, 1.0E-6)
  gamma ~ dunif(0.0, 1.0)
  tau ~ dgamma(0.01, 0.01)
}
```

R2WinBUGS code

```
# Copy and paste the following lines into R.  In R, remember to go to “file>change dir…” to enter
# the subdirectory where the file “nonlinearmodel.bug” is located.  I might also want to certify
# yourself where WinBUGS is located in your computer.

library(R2WinBUGS)
x = c(1.0,1.5,1.5,1.5,2.5,4.0,5.0,5.0,7.0,8.0,8.5,
      9.0,9.5,9.5,10.0,12.0,12.0,13.0,13.0,14.5,
      15.5,15.5,16.5,17.0,22.5,29.0,31.5)
y = c(1.80,1.85,1.87,1.77,2.02,2.27,2.15,2.26,2.47,
      2.19,2.26,2.40,2.39,2.41,2.50,2.32,2.32,2.43,
      2.47,2.56,2.65,2.47,2.64,2.56,2.70,2.72,2.57)
N     <- length(x)
data  <- list("x","y","N")
inits <- function(){
  list(alpha=1,beta=1,tau=1,gamma=0.9)
}
nonlinear.sim = bugs(data,inits,
     model.file="nonlinearmodel.bug",
     parameters=c("alpha","beta","tau","gamma"),
     n.chains=1,n.iter=20000,n.burnin=5000,n.thin=1,
     bugs.directory="c:/Program Files/WinBUGS14/",
     codaPkg=FALSE)
```

R evoking WinBUGS



The nonlinear.sim object

This object contains posterior summaries and output.

```
> nonlinear.sim
Inference for Bugs model at "nonlinearmodel.bug"
 1 chains, each with 10000 iterations (first 5000 discarded)
 n.sims = 5000 iterations saved
          mean   sd  2.5%   25%   50%   75% 97.5%
alpha      2.7  0.1   2.5   2.6   2.6   2.7   2.8
beta       1.0  0.1   0.8   0.9   1.0   1.0   1.1
tau      107.8 31.3  55.2  85.1 104.6 127.2 177.4
gamma      0.9  0.0   0.8   0.8   0.9   0.9   0.9
deviance -48.8  3.3 -53.0 -51.3 -49.6 -47.1 -40.7
pD = 5.5 and DIC = -43.4 (using the rule, pD = var(deviance)/2)
DIC is an estimate of expected predictive error (lower deviance is better).

> names(nonlinear.sim)
 [1] "n.chains"        "n.iter"          "n.burnin"        "n.thin"          "n.keep"
 [6] "n.sims"          "sims.array"      "sims.list"       "sims.matrix"     "summary"
[11] "mean"            "sd"              "median"          "root.short"      "long.short"
[16] "dimension.short" "indexes.short"   "last.values"     "pD"              "DIC"
[21] "model.file"      "is.DIC"
```

“sims.array” contains the MCMC chain.

```
> dim(nonlinear.sim$sims.array)
[1] 10000     1     5

> nonlinear.sim$sims.array[1:15,1,]
        alpha   beta      tau   gamma  deviance
 [1,]   2.680 0.9503 128.60  0.8942    -49.87
 [2,]   2.778 1.1040  94.90  0.8986    -48.92
 [3,]   2.740 1.0290  75.43  0.8947    -49.70
 [4,]   2.735 1.0670 101.10  0.8884    -50.87
 [5,]   2.635 0.9631 147.40  0.8788    -47.02
 [6,]   2.678 0.9845  96.95  0.8796    -52.55
 [7,]   2.680 0.9670 162.50  0.8778    -52.15
 [8,]   2.715 1.0380 112.50  0.8727    -49.63
 [9,]   2.672 0.9701 154.70  0.8836    -51.40
[10,]   2.726 1.0130 138.60  0.8810    -49.45
[11,]   2.707 0.9602 105.50  0.8898    -52.52
[12,]   2.745 1.0220 166.00  0.9024    -48.46
[13,]   2.793 1.0700 109.40  0.9048    -50.04
[14,]   2.835 1.1550  66.93  0.9142    -43.54
[15,]   2.776 0.9380 133.00  0.9115    -47.13
```

```
> par(mfrow=c(2,2))
> ts.plot(nonlinear.sim$sims.array[,1,1],xlab="iterations",ylab="",main="alpha")
> ts.plot(nonlinear.sim$sims.array[,1,2],xlab="iterations",ylab="",main="beta")
> ts.plot(nonlinear.sim$sims.array[,1,3],xlab="iterations",ylab="",main="tau")
> ts.plot(nonlinear.sim$sims.array[,1,4],xlab="iterations",ylab="",main="gamma")
```