Autoregressive Models for Capture-Recapture Data
A Bayesian Approach

RK

June 11, 2015

Abstract

This document summarizes the main points of the paper, "Autoregressive Models for Capture-Recapture Data : A Bayesian Approach'.

Contents

1. Context

Capture-Recapture models are a part of staple diet for *ecological researchers*. Here is my naive interpretation of the model :

- You capture a set of animals, tag them with unique ids and release them in to the population at t_0.
- At regular intervals, you have a chance to capture a set of animals, observe whether any of the captured animals are tagged, record them if any.
- At every recapture process, the animals who are not tagged are tagged and released in to the population.

The data are recorded in a upper triangular matrix with rows representing the *capture occasions* when marking is performed, columns representing the *recording occasions* when recaptures or recoveries occur. In the above scenario typically there are two types of probabilities that come in to play. One is the survival probability of an animal from one time period to another and the other is the probability of an animal being observed, given that animal is alive at that specific time interval. The more complicated the model is, the more parameters that a modeler includes. For example the probabilities of survival can be made time dependent, covariate dependent. So also are the conditional probabilities. There is a widespread belief in the ecological research community that survival probabilities are not fixed but they are a realization of random processes with covariates such as age, weather and time factors. This paper deals with a model in which survival probabilities are time dependent.

What are the advantage of using Bayes estimation ?

- Allow for estimation of unobserved random effects
- Survival probabilities can be estimated for each individual period

The paper talks about applying autoregressive frame for two types of models

- Open population mark-recapture models (Cormack-Jolly-Seber Model)
- Band recover models

2. Likelihood function

Notation

- R_i denotes the number of marked or banded animals released at each capture occasion at time t_i
- m_{ij} denotes the number of animals released at time t_i and subsequently recaptured or reported at time t_j
- I denotes the number of capture occasions when marking or banding is performed
- J denotes the number of recording sessions, either recaptures or recoveries

The probability model used is a multinomial model

$$(m_{i1}, m_{i2}, \ldots, m_{iJ}) \sim \text{Multinomial}(R_i, p = f(\text{survival, recovery}))$$

2.1. Mark-Recapture Likelihood

The mark recapture likelihood for the observed data can be written as

$$\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{p}; \boldsymbol{R}, \boldsymbol{m}) = \prod_{i=1}^{I} \binom{R_i}{m_{i,i+1}, m_{i,i+2}, \ldots, m_{i,J+1}, v_i} \xi^{v_i} \prod_{j=i+1}^{J+1} \left\{ \phi_i p_j \prod_{k=i+1}^{j-1} \phi_k (1 - p_k) \right\}^{m_{ij}}$$

where

- ϕ_i is the probability that an animal survives from capture occasion t_i to t_{i+1} given that it is alive at t_i
- p_j is the probability that an animal is captured at t_j given that it is alive at t_j, $j = 2, \ldots, J+1$
- ξ_i is the probability that an animal is never captured after released at t_i
- v_i is the number of animals captures at t_i and never subsequently recaptured during the study

2.2. Band-Recovery Likelihood

The mark recapture likelihood for the observed data can be written as

$$\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\lambda}; \boldsymbol{R}, \boldsymbol{m}) = \prod_{i=1}^{I} \binom{R_i}{m_{i,i+1}, m_{i,i+2}, \ldots, m_{i,J+1}, v_i} \xi^{v_i} \prod_{j=i}^{J} \left\{ \lambda_i \prod_{k=i}^{j-1} \phi_k \right\}^{m_{ij}}$$

where

- ϕ_i is the probability that an animal survives from capture occasion t_i to t_{i+1} given that it is alive at t_i
- λ_j is the probability that a marked animal is hunted between t_j and t_{j+1}
- ξ_i is the probability that an animal is never recaptured after released at t_i
- v_i is the number of animals captures at t_i and never subsequently recaptured during the study

3. Bayesian Approach to survival models

The glm considered for probability that an animal survives from time t_j to t_{j+1} is

$$g(\phi_j) = \beta + \epsilon_j, \quad j = 1, 2, \ldots, J$$
$$\epsilon_j = \sum_{i=1}^{k} \rho_k \epsilon_{j-k} + z_j, \quad j = 1, 2, \ldots, J$$

where $\boldsymbol{X}_j'$ is the matrix of covariates, β is a vector of regression coefficients, $\epsilon' \sim N(\mathbf{0}, \boldsymbol{\Sigma})$ and $z_j \sim N(0, \sigma^2)$

The main reason behind using AR(m) model is that it provides a positive or negative correlation between survival probabilities that decrease with an increasing separation of time.

The posterior distribution of parameters and random effects are given by

$$\pi(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\rho}, \boldsymbol{\epsilon}, \boldsymbol{r} | D) = \mathcal{L}(\boldsymbol{\beta}, \boldsymbol{\epsilon}, \boldsymbol{r}; D) \times \pi(\boldsymbol{\beta})\pi(\sigma^2)\pi(\boldsymbol{\rho})\pi(\boldsymbol{r}) \times |\boldsymbol{\Sigma}|^{-1/2} \exp\left(-\boldsymbol{\epsilon}' \boldsymbol{\Sigma}^{-1} \boldsymbol{\epsilon}/2\right)$$

The paper derives the full conditional distributions for the following parameters

- $f(\beta_l | \boldsymbol{\beta}_{-l}, \sigma^2, \boldsymbol{\rho}, \boldsymbol{\epsilon}, \boldsymbol{r}, D)$
- $f(r_l | \boldsymbol{r}_{-l}, \boldsymbol{\beta}, \sigma^2, \boldsymbol{\rho}, \boldsymbol{\epsilon}, D)$
- $f(\sigma^2 | \boldsymbol{r}, \boldsymbol{\beta}, \boldsymbol{\rho}, \boldsymbol{\epsilon}, D)$
- $f(\boldsymbol{\rho} | \sigma^2, \boldsymbol{\beta}, \boldsymbol{\rho}, \boldsymbol{\epsilon}, D)$
- $f(\boldsymbol{\epsilon} | \sigma^2, \boldsymbol{\rho}, D)$

Given a proper choice of conjugate prior, only one of the above posteriors come out to have a standard form. The rest are non standard densities. Hence there is a need to use a combination of Metropolis Hastings steps and Gibbs sampling steps. Thankfully WinBUGS takes care of these steps, once the DAG is set up.

4. Parameter estimation using WinBUGS

4.1. Intercept only Model

```
{
sink("model-int.txt")
cat("model
{
for(i in 1:I){ D[i, 2:(I+2)] ~ dmulti(C[i,], D[i, 1]); }
for(i in 1:(I-1)){
    lphi[i]       <- log(phi[i])
    logit(phi[i]) <- beta + e[i]
   for(j in (i+1):I){
     C[i, j]   <- lambda[j]*exp(sum(lphi[i:(j-1)]))
   }
    for (j in 1:i){
        C[i+1, j] <- 0
    }
}
for(i in 1:I){
    C[i, i]       <- lambda[i]
    C[i, I+1]     <- 1 - sum(C[i, 1:I])
}
e[1]       ~ dnorm(mu[1], tau1)
e[2]       ~ dnorm(mu[2], tau2)
mu[1]      <- 0
mu[2]      <- (rho[1]/(1-rho[2]))*e[1]
tau1       <- ((1+rho[2])/(1-rho[2]))*((1-rho[2])*(1-rho[2]) - rho[1]*rho[1])*tau
tau2       <- tau*(1 - rho[2]*rho[2])
for(i in 3:(I-1)){
    e[i]   ~ dnorm(mu[i],tau)
    mu[i]  <- rho[1]*e[i-1] + rho[2]*e[i-2]
}
sigma      <- 1/sqrt(tau)
beta       ~ dnorm(0, 0.01)
for(i in 1:I){lambda[i] ~ dunif(0, 1)}
tau        ~ dgamma(0.001, 0.001)
rho[1]     ~ dunif(l, u)
u          <- abs(1 - rho[2])
l         <- -u
rho[2]     ~ dunif(-1, 1)
}",fill = TRUE)
sink()}
```

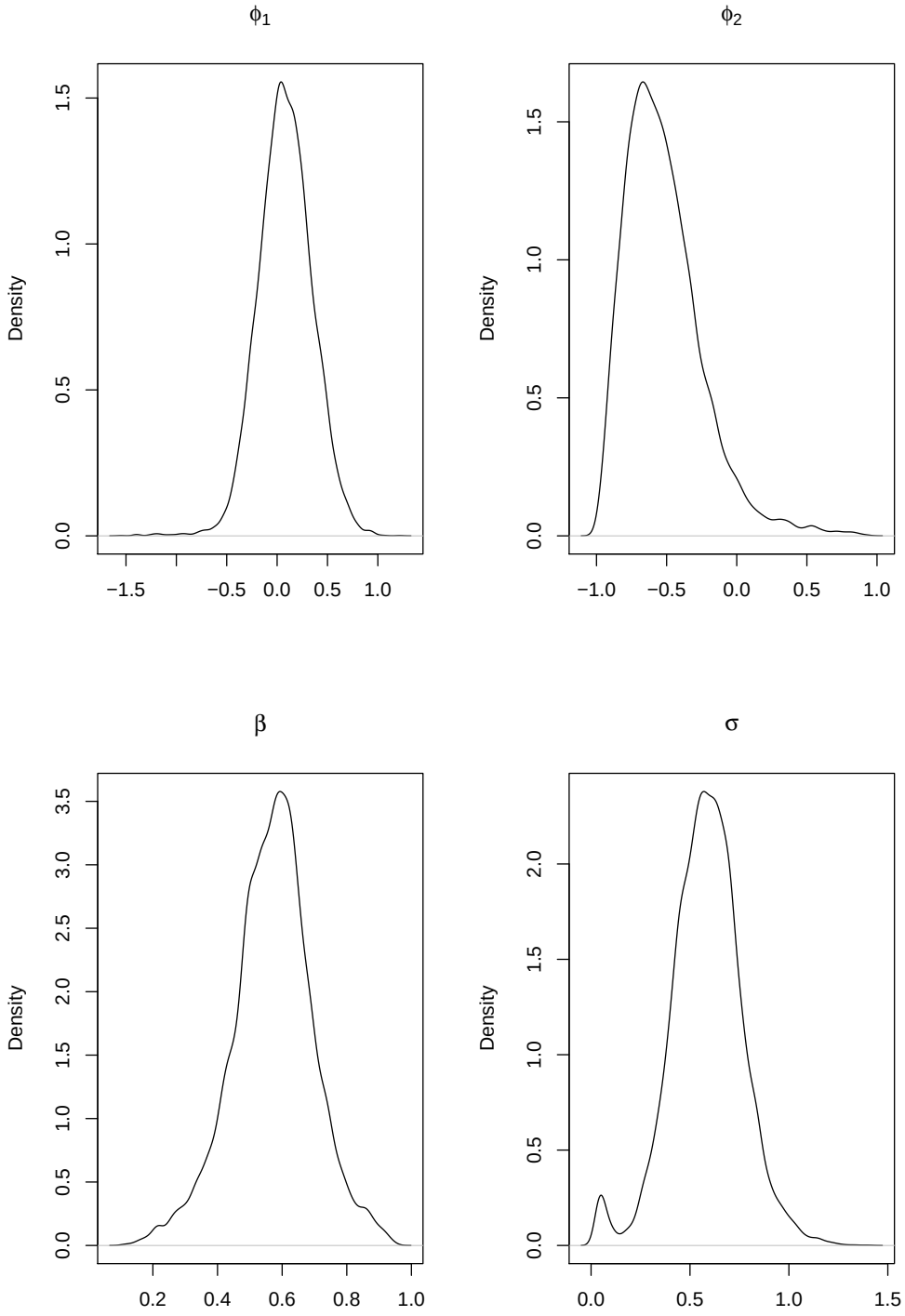
```
#dataset
data           <- list(I=28,D = matrix(data = c(
  270, 7, 6, 3, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 252, 693, 0, 21, 10, 4, 2,
  3, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 651, 1612, 0, 0, 32, 20, 8, 5, 1, 2, 0, 2, 1, 1, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1540, 858, 0, 0, 0, 26,
        12, 5, 6, 4, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 804, 1471, 0, 0, 0, 0, 21, 18, 6, 5, 0, 0, 1, 0, 1, 1,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1418, 1051, 0, 0, 0,
        0, 0, 18, 4, 6, 4, 1, 2, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 1013, 796, 0, 0, 0, 0, 0, 0, 24, 6, 4, 0, 3, 3, 0,
        0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 755, 277, 0, 0, 0,
        0, 0, 0, 0, 10, 9, 6, 6, 4, 1, 2, 4, 2, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 233, 903, 0, 0, 0, 0, 0, 0, 0, 0, 15, 8, 1, 8, 4, 0,
        2, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 862, 621, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 6, 4, 1, 6, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 602, 584, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 10, 4, 3, 7, 3,
        1, 2, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 553, 822, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 25, 6, 10, 4, 4, 2, 0, 0, 2, 0, 0, 1, 0, 0, 0,
        0, 0, 768, 1344, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 28, 27, 8,
        11, 3, 1, 4, 1, 2, 0, 0, 1, 0, 0, 0, 0, 1258, 566, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 10, 13, 6, 2, 2, 1, 1, 1, 1, 0, 0, 0,
        0, 0, 0, 529, 481, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 9,
        7, 3, 2, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 459, 695, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 11, 11, 5, 2, 2, 1, 1, 0, 1, 1, 0,
        0, 0, 660, 632, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        22, 10, 2, 4, 0, 1, 2, 0, 0, 0, 0, 0, 591, 1114, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 21, 11, 8, 3, 5, 3, 2, 1, 0,
        0, 0, 1060, 639, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 9, 10, 10, 2, 3, 0, 2, 0, 0, 0, 603, 926, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 16, 9, 9, 2, 5, 1, 2,
        1, 0, 881, 858, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 14, 12, 3, 5, 1, 1, 1, 0, 821, 369, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 13, 2, 4, 4, 1,
        1, 0, 344, 450, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 8, 3, 4, 1, 2, 1, 431, 212, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 6, 0, 0, 1,
        0, 205, 1680, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 18, 28, 8, 4, 1622, 421, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 14, 1,
        2, 404, 118, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 116, 60, 0, 0, 0, 0, 0, 0, 0,
```

```
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
        59), nrow = 28, ncol = 30, byrow=TRUE))

# Initialize parameters
inits          <- function() { list(
              beta = 0.6, tau = 1, rho = c(0.4, -0.4),
              e = c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
              lambda= c(0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,
              0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,
              0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01))}

# Parameters to track
params         <- c("phi", "rho", "beta" ,"sigma","phi")

# MCMC settings
nc             <- 1
ni             <- 10000
nb             <- 1000
nt             <- 1

# bugs directory
bugs.directory <- "C:/Cauldron/garage/WinBUGS14"

# Use WinBUGS
samples_int_only <- bugs(data = data, inits = inits, parameters = params,
                 model="model-int.txt", n.thin=nt, n.chains=nc, n.burnin=nb,
                 n.iter = ni, debug = FALSE,codaPkg = FALSE,
                 bugs.directory = bugs.directory)

coef_int_only   <- round(samples_int_only$summary[28:31,c(1,2,3,5,7)],3)
```

```
print(coef_int_only)
```

```
##          mean    sd   2.5%    50% 97.5%
## rho[1]  0.083 0.268 -0.420  0.080 0.605
## rho[2] -0.519 0.277 -0.907 -0.563 0.168
## beta    0.570 0.125  0.298  0.576 0.820
## sigma   0.583 0.181  0.180  0.585 0.943
```

Density Estimates of parameters



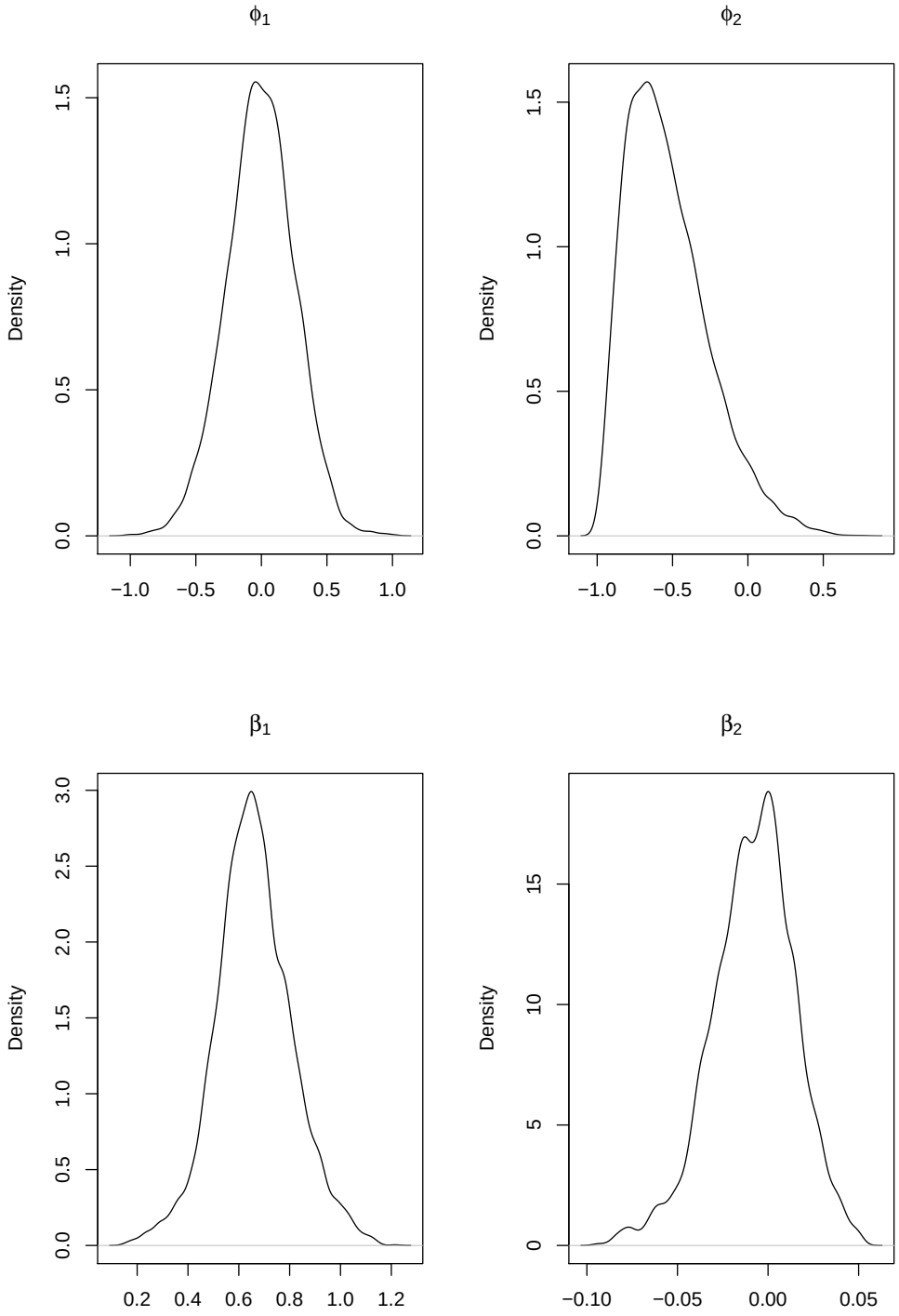
4.2. Intercept and Slope Model

```
{
sink("model-int-slope.txt")
cat("model
{
for(i in 1:I){ D[i, 2:(I+2)] ~ dmulti(C[i,], D[i, 1]); }
for(i in 1:(I-1)){
    lphi[i]       <- log(phi[i])
    logit(phi[i]) <- beta[1] + beta[2]*(i-14) + e[i]
    for(j in (i+1):I){
       C[i, j]   <- lambda[j]*exp(sum(lphi[i:(j-1)]))
   }
    for (j in 1:i){
        C[i+1, j] <- 0
    }
}
for(i in 1:I){
    C[i, i]       <- lambda[i]
    C[i, I+1]     <- 1 - sum(C[i, 1:I])
}
e[1]       ~ dnorm(mu[1], tau1)
e[2]       ~ dnorm(mu[2], tau2)
mu[1]      <- 0
mu[2]      <- (rho[1]/(1-rho[2]))*e[1]
tau1       <- ((1+rho[2])/(1-rho[2]))*((1-rho[2])*(1-rho[2]) - rho[1]*rho[1])*tau
tau2       <- tau*(1 - rho[2]*rho[2])
for(i in 3:(I-1)){
    e[i]   ~ dnorm(mu[i],tau)
    mu[i]  <- rho[1]*e[i-1] + rho[2]*e[i-2]
}
sigma      <- 1/sqrt(tau)
for(i in 1:2){beta[i] ~ dnorm(0, 0.01)}
for(i in 1:I){lambda[i] ~ dunif(0, 1)}
tau        ~ dgamma(0.001, 0.001)
rho[1]     ~ dunif(l, u)
u          <- abs(1 - rho[2])
l         <- -u
rho[2]     ~ dunif(-1, 1)
}",fill = TRUE)
sink()}
```

```
# Initialize parameters
inits          <- function() { list(
              beta = c(0.6, 0.01), tau = 1, rho = c(0.4, -0.4),
              e = c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
              lambda= c(0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,
              0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,
              0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01))}

# Parameters to track
params         <- c("phi", "rho", "beta" ,"sigma","phi")

# MCMC settings
nc             <- 1
ni             <- 10000
nb             <- 1000
nt             <- 1

# bugs directory
bugs.directory <- "C:/Cauldron/garage/WinBUGS14"

# Use WinBUGS
samples_int_slope <- bugs(data = data, inits = inits, parameters = params,
                model="model-int-slope.txt", n.thin=nt, n.chains=nc, n.burnin=nb,
                n.iter = ni, debug = FALSE,codaPkg = FALSE,
                bugs.directory = bugs.directory)

coef_int_slope <- round(samples_int_slope$summary[28:32,c(1,2,3,5,7)],3)
```

```
print(coef_int_slope)

##           mean    sd   2.5%    50% 97.5%
## rho[1]  -0.006 0.261 -0.530 -0.006 0.499
## rho[2]  -0.530 0.272 -0.919 -0.575 0.119
## beta[1]  0.660 0.150  0.359  0.654 0.980
## beta[2] -0.008 0.023 -0.059 -0.007 0.034
## sigma    0.720 0.223  0.368  0.692 1.244
```

Density Estimates of parameters



5. Model comparison

Intercept Model

	mean	sd	2.5pct	50pct	97.5pct
ρ_1	0.083	0.268	-0.420	0.080	0.605
ρ_2	-0.519	0.277	-0.907	-0.563	0.168
β	0.570	0.125	0.298	0.576	0.820
σ	0.583	0.181	0.180	0.585	0.943

Intercept and Slope Model

	mean	sd	2.5pct	50pct	97.5pct
ρ_1	-0.006	0.261	-0.530	-0.006	0.499
ρ_2	-0.530	0.272	-0.919	-0.575	0.119
β_1	0.660	0.150	0.359	0.654	0.980
β_2	-0.008	0.023	-0.059	-0.007	0.034
σ	0.720	0.223	0.368	0.692	1.244

The above stats show that intercept model is good enough

Survival Probability plot