Supplementary Information for MMC Statistical Review of Becker et al. 2011.
B. Becker

Purpose: To facilitate a statistical review of Becker et al. 2011. A PDF of the original paper is annotated with notes to orient the reader to the associated data (Part A) and R code (Part B) below.

PART A. Data File descriptions (Pages 1 – 3). These are separate files.
- A1. DATA FILE: oyster harvest by year and spring 1982-2009.xls
- A2. DATA FILE: oyster acre vs. harvest 1993 – 2009-latest.csv
- A3. DATA FILE: Disturbances_subsite_year_source_1997-2009.csv
- A4. DATA FILE: 1982-2009.subsite+86-93.csv
- A5. DATA FILE: binom.09.csv

PART B. Model Code (R or WinBUGS) for basic (usually full) models (Pages 3 – 15)
- B1. glmmML
- B2. GEEs
- B3. GLMs for Regional Models
- B4. Bootstrapping
- B5. R2WinBUGS Code for Bayesian Analysis

PART A. DATA FILES

I am providing 5 data files which pertain to different levels of analysis in the paper. Harbor Seal Data are generally derived from the NPS Pinniped database:

A1. DATA FILE: oyster harvest by year and spring 1982-2009.xls

DESCRIPTION: DFG reported annual and spring (March–May) oyster harvest in lbs. 2004 has no reported seasonal harvest value.

A2. DATA FILE: oyster acre vs. harvest 1993 – 2009-latest.csv

DESCRIPTION: Acres of oyster equipment derived from aerial photos
USE: Figure 4
Field Key:
Year
Oyster Harvest – annual in lbs. obtained from California Dept fish and Game
Acres of Equipment on or near sandbars – which are seal haul outs (UEN, OB, UEF. Near = < 300m).

A3. DATA FILE: Disturbances_subsite_year_source_1997-2009.csv

DESCRIPTION: Human disturbances by subsite within Drakes Estero
USE: Figure 5
Field Key:
Subsite – One of the 8 haul out sites in Drakes Estero.
max.count – of total harbor seals in that year during the breeding season.
max.pups - of total harbor pups seal in that year during the breeding season.

area – subsites UEN, OB, and UEF = UPPER, else = LOWER.
mam.dist – human caused seal disturbances during march-may each year. (excludes airplanes).
mam.surv – number of harbor seal surveys during march-may each year.
oyst – annual oyster harvest in lbs from DFG. 2009 estimated, but actual was 458,000 lbs.

A4. DATA FILE: 1982-2009.subsite+86-93.csv

DESCRIPTION: Seal count data within Drakes Estero. Used for colony scale analyses
USE: Tables 1, 2
Field Key:

Date - Date of seal survey
Year
Pupnot.de – total max count of seal pups that year not in Drakes Estero
Totnot.de – annual max count of all seals that year not in Drakes Estero
ENSO.YRS – years since the last major ENSO event
Mam.u.dist – march-may human disturbance rate in the upper estero
Mam.l.dist – mar-may human disturbance rate in the lower estero
A.Adult – max daily count of seals at subsite A
A.Pup – max daily count of seal pups at subsite A
Next 14 fields - are similar for different subsites
OB.Total, UEN.Total, UEF.Total = adults + pups max count of day
Upper.pup = daily sum of pup max counts of upper estero subsites
Lower.pup = daily sum of pup max counts of lower estero subsites.
Lower.tot = daily sum of max seal counts of lower estero subsites.
Upper.tot = daily sum of max seal counts of upper estero subsites.
Oyster.new = Annual Oyster harvest values (lbs)from Dept. Fish and Game
Oyst.hl – high/low classification of annual oyster harvest levels based on Appendix A.
LowerTot.year – Annual max seals in the lower estero.
Post92 – post year 1992
Survs – annual number of surveys used in the analysis

A5. DATA FILE: binom.09.csv

DESCRIPTION: Seal Count data for Drakes Estero and regional colonies. Used for regional scale analyses
USE: Figures 2, 3AB, 6
Tables: 3, 4, 5
Field Key:

Year
Adult.de – breeding season max count of adult seals in drakes estero
Adultnot.de - breeding season max count of adult seals not in drakes estero
Pup.de - breeding season max count of pup seals in drakes estero
Pupnot.de - breeding season max count of pup seals not in drakes estero
Tot.de – total breeding season max count seals in drakes estero
Totnot.de - total breeding season max count seals not in drakes estero

a.max – max seal count at subsite A in Drakes Estero
prop.a.de – proportion of Drakes Estero seals using subsite A
a.avail – subsite A is used by seals (1 = yes)
bags.6 – low oyster harvest from 1999-2004 (0 = low)
bags.5 - low oyster harvest from 2000-2004 (0 = low)
tot.dp – max breeding season count seals at double point
pup.dp - max breeding season count seal pups at double point
prop.dp – proportion of regional seals at double point
dp.change – % change in number of seals at Double point from previous year (0 if no info)
dist.rate – human disturbance rate for entire estero
mam.dist.rate – march-may human disturbance rate in Drakes estero
weight = weighting value
enso – years since last major ENSO

PART B: MODEL CODE:

Here, I reproduce the model code which is generally for the full models. In some instances, a dozen or more *a priori* competing models of different covariates (=hypotheses) were run which can be replicated as needed by referencing the tables in the paper. Modeling techniques, code, and philosophy closely followed those of Zuur et al. 2009 for glmmML, GEEs, and bootstrapping; Burnham and Anderson (2002) for ranking GLMs by QAICc; and Kéry 2010 for the Bayesian analysis. Models also represent statistical guidance and input from J. Laake, T. Gerrodette, A. Solow (WHOI), B. Halstead (USGS), and several anonymous reviewers.

B1. glmmML: To produce Table 1 (this is single example of multiple models run):

```
> Library(glmmML)
> Model -> glmmML(formula = prop.pup.ul ~ oyst.hl + log1p(ENSO.YRS), family =
binomial, data = data, cluster = factor(Year))
      ## this was one of the best models, other models included upper and
      lower estero
      ## disturbance rates, etc.
      ## Note that prop.pup.ul is a cbind() file with two vectors: seals in
      ## the upper stero and lower estero
      ## similar model with prop.seals.ul for total column in Table 1.
      ## glmmML reports AIC
```

B2. GEEs: To Produce Table 2

```
      > library(geepack)
      > library(MASS)
      > library(lme4)
      > prop.tot<-cbind(upper.tot,lower.tot)
      > prop.pup<-cbind(upper.pup,lower.pup)
      > geeglm(formula = prop.pup ~ log1p(ENSO.YRS) + Oyster.new, family =
      binomial, data = subsite.all, id = fYear, corstr = "ar1")
            ##fYear <- factor(year)
```

B3. GLMs for Regional Models: To produce Figure 6, Tables, 3 and 4

Quasibinomial models to assess overdispersion values done separately for total seal and just pups.

```
> model.tot<-glm(formula = prop.pup ~ prop.a.de+oyst+tot.dp, family =
quasibinomial, weight = weight,binom.09.s)
```

For QAICc rankings, same formula but using binomial distribution. Then derive `logLik()` and compute QAICc using formula from Burnham and Anderson (2002). Note that QAICc is corrected both for sample size and overdispersion.

B4. Bootstrapping methods for best models in Table 3.

```
  > prop.pup<-cbind(pup.de,pupnot.de)

  ##NOW FOR THE LOWEST QAICC MODEL:
  > fbags <- factor(bags.5)
  > m2a7<-glm(prop.pup~fbags+tot.dp,family=binomial,data=binom.09.s)
  > summary(m2a7)

  Call:
  glm(formula = prop.pup ~ fbags + tot.dp, family = binomial, data =
  binom.09.s)

  Deviance Residuals:
      Min       1Q   Median       3Q      Max
  -4.3354  -1.3206   0.5579   1.2126   3.3042

  Coefficients:
                Estimate Std. Error z value Pr(>|z|)
  (Intercept) -0.2016268  0.0967679  -2.084   0.0372 *
  fbags[T.1]  -0.2619859  0.0369756  -7.085 1.39e-12 ***
  tot.dp      -0.0005901  0.0001035  -5.703 1.17e-08 ***
  ---
  Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

  (Dispersion parameter for binomial family taken to be 1)

      Null deviance: 129.269  on 14  degrees of freedom
  Residual deviance:  63.205  on 12  degrees of freedom
  AIC: 176.26
Number of Fisher Scoring iterations: 3
  ##AND THE MODEL BOOTSTRAPPED
  > model.boot<-function(binom.09.s,indices){
  + sub.data<-binom.09.s[indices,]
  + model<-glm(prop.pup~fbags+tot.dp,family=binomial,data=sub.data)
  + coef(model)}
  > glim.boot<-boot(binom.09.s,model.boot,R=2000)
  > glim.boot

  ORDINARY NONPARAMETRIC BOOTSTRAP
```

```
Call:
boot(data = binom.09.s, statistic = model.boot, R = 2000)

Bootstrap Statistics:
         original        bias     std. error
t1* -0.2016267859 -0.5391331210 0.3271317906
t2* -0.2619859317  0.0534001370 0.0332789989
t3* -0.0005901444  0.0005960157 0.0003868234
##LOOKS GOOD, LOW BIAS FOR THE BEST PUP MODEL
##NOW DO TOTAL SEALS MODEL
> prop.tot<-cbind(tot.de,totnot.de)

##NOW WE’LL DO THE TOTAL MODEL
> model.boot<-function(binom.09.s,indices){
+ sub.data<-binom.09.s[indices,]
+ model<-
glm(prop.tot~fbags+tot.dp,family=binomial,data=sub.data,weights=weight
)
+ coef(model)}
> glim.boot<-boot(binom.09.s,model.boot,R=2000)
> glim.boot

ORDINARY NONPARAMETRIC BOOTSTRAP

Call:
boot(data = binom.09.s, statistic = model.boot, R = 2000)

Bootstrap Statistics :
         original        bias     std. error
t1* -0.3428364902 -0.4323597175 0.2724916757
t2* -0.2255795680  0.0666238556 0.0326915006
t3* -0.0004734991  0.0004785484 0.0003194529

##LOW BIAS AGAIN
```

B5. R2WinBUGS Code for Bayesian Analysis. To construct Table 5. Note that all these models use oyster harvest as a continuous covariate (not high/low).

(NOTE: The remainder of this document consists of these models).

```
 >library(r2WinBUGS)
## Code Modified from Kery 18.2 for Phoca DE vs non DE
## Becker 07/09/2010

#### varibles for phoca analysis: (pup.de "C") vs. (pup.de + pupnot.de:
"N"), oyst, tot.dp, a.max

## remember to setwd()
## must load dataset binom.09.csv into R
## next build multiple models and rank by DIC
## need to get model diagnostics (check correlation of betas)
#### variables for phoca analysis: (pup.de "C") vs. (pup.de + pupnot.de:
"N"), oyst, tot.dp, a.max

####################################################
```

```
### NO OVERDISPERSION, SIMPLE MODEL ############18.4. Analysis using
WinBUGS    oyst (beta1) + DP(beta2)
####################################################
# Define model
sink("binom.txt")
cat("
model {

# Priors

    alpha ~ dnorm(0, 0.001)          # Intercept
    beta1 ~ dnorm(0, 0.001)          # Slope
    beta2 ~ dnorm(0, 0.001)          # Slope


# Likelihood
 for (i in 1:15) {
    C[i] ~ dbin(p[i], N[i])
    logit(p[i]) <- alpha + beta1 * oyst[i] + beta2 * dp[i]

 }

}
",fill=TRUE)
sink()


# Bundle data
win.data <- list(C = pup.de, N = (pup.de + pupnot.de), oyst =
(oyst/100000), dp = (tot.dp/100))

inits <- function(){list(alpha = rnorm(1), beta1 = rnorm(1), beta2 =
rnorm(1))}


# Parameters to estimate
params <- c("alpha", "beta1", "beta2")

# MCMC settings
ni <- 100000
nb <- 50000
nt <- 25
nc <- 3

# Start Gibbs sampler
out.12 <- bugs(data = win.data, inits = inits, parameters = params,
model.file = "binom.txt",
n.thin = nt, n.chains = nc, n.burnin = nb, n.iter = ni, debug = TRUE,
DIC = TRUE)



####################################################
### NO OVERDISPERSION, SIMPLE MODEL ############18.4. Analysis using
WinBUGS    Oyst(beta1) only
####################################################
# Define model
```

```
sink("binom.txt")
cat("
model {

# Priors

    alpha ~ dnorm(0, 0.001)          # Intercept
    beta1 ~ dnorm(0, 0.001)          # Slope


# Likelihood
 for (i in 1:15) {
    C[i] ~ dbin(p[i], N[i])
    logit(p[i]) <- alpha + beta1 * oyst[i]

 }

}
",fill=TRUE)
sink()


# Bundle data
win.data <- list(C = pup.de, N = (pup.de + pupnot.de), oyst =
(oyst/100000))

inits <- function(){list(alpha = rnorm(1), beta1 = rnorm(1))}


# Parameters to estimate
params <- c("alpha", "beta1")

# MCMC settings
ni <- 100000
nb <- 50000
nt <- 25
nc <- 3

# Start Gibbs sampler
out.1 <- bugs(data = win.data, inits = inits, parameters = params,
model.file = "binom.txt",
n.thin = nt, n.chains = nc, n.burnin = nb, n.iter = ni, debug = TRUE,
DIC = TRUE)



####################################################
### NO OVERDISPERSION, SIMPLE MODEL ############18.4. Analysis using
WinBUGS    DP(beta2)
####################################################
# Define model
sink("binom.txt")
cat("
model {

# Priors
```

```
    alpha ~ dnorm(0, 0.001)          # Intercept
    beta2 ~ dnorm(0, 0.001)          # Slope


# Likelihood
 for (i in 1:15) {
    C[i] ~ dbin(p[i], N[i])
    logit(p[i]) <- alpha + beta2 * dp[i]

 }

}
",fill=TRUE)
sink()


# Bundle data
win.data <- list(C = pup.de, N = (pup.de + pupnot.de), dp =
(tot.dp/100))

inits <- function(){list(alpha = rnorm(1), beta2 = rnorm(1))}


# Parameters to estimate
params <- c("alpha", "beta2")

# MCMC settings
ni <- 100000
nb <- 50000
nt <- 25
nc <- 3

# Start Gibbs sampler
out.2 <- bugs(data = win.data, inits = inits, parameters = params,
model.file = "binom.txt",
n.thin = nt, n.chains = nc, n.burnin = nb, n.iter = ni, debug = TRUE,
DIC = TRUE)






########################################
### 18.4. Analysis using WinBUGS + Overdispersion: DP(beta2) +
oyst(beta1)
######################################
# Define model
sink("binom.txt")
cat("
model {

# Priors

    alpha ~ dnorm(0, 0.001)          # Intercept
    beta1 ~ dnorm(0, 0.001)          # Slope
    beta2 ~ dnorm(0, 0.001)          # Slope
    sigma ~ dunif(0,10)
    tau <- 1 / (sigma * sigma)


# Likelihood
 for (i in 1:15) {
    C[i] ~ dbin(p[i], N[i])
    logit(p[i]) <- alpha + beta1 * oyst[i] + beta2 * dp[i] + eps[i]
    eps[i] ~ dnorm(0, tau)

 }

}
",fill=TRUE)
sink()


# Bundle data
win.data <- list(C = pup.de, N = (pup.de + pupnot.de), oyst =
(oyst/100000), dp = (tot.dp/100))

inits <- function(){ list(alpha = rnorm(1), beta1 = rnorm(1), beta2 =
rnorm(1))}


# Parameters to estimate
params <- c("alpha", "beta1", "beta2", "sigma", "tau", "eps")

# MCMC settings
ni <- 200000
nb <- 100000
nt <- 25
nc <- 3

# Start Gibbs sampler
out.od.12 <- bugs(data = win.data, inits = inits, parameters = params,
model.file = "binom.txt",
n.thin = nt, n.chains = nc, n.burnin = nb, n.iter = ni, debug = TRUE,
DIC = TRUE)
```

____

```
###############################
######18.4. Analysis using WinBUGS + Overdispersion with oyst (beta1)
only
################################
# Define model
sink("binom.txt")
cat("
model {

# Priors

    alpha ~ dnorm(0, 0.001)          # Intercept
    beta1 ~ dnorm(0, 0.001)          # Slope
    sigma ~ dunif(0,10)
    tau <- 1 / (sigma * sigma)


# Likelihood
 for (i in 1:15) {
    C[i] ~ dbin(p[i], N[i])
    logit(p[i]) <- alpha + beta1 * oyst[i] + eps[i]
    eps[i] ~ dnorm(0, tau)

 }

}
",fill=TRUE)
sink()


# Bundle data
win.data <- list(C = pup.de, N = (pup.de + pupnot.de), oyst =
(oyst/100000))

inits <- function(){ list(alpha = rnorm(1), beta1 = rnorm(1))}


# Parameters to estimate
params <- c("alpha", "beta1", "sigma", "tau", "eps")

# MCMC settings
ni <- 100000
nb <- 50000
nt <- 25
nc <- 3

# Start Gibbs sampler
out.od.1 <- bugs(data = win.data, inits = inits, parameters = params,
model.file = "binom.txt",
n.thin = nt, n.chains = nc, n.burnin = nb, n.iter = ni, debug = TRUE,
DIC = TRUE)



###############################
######18.4. Analysis using WinBUGS + Overdispersion with dp (Beta2) only
################################

# Define model
sink("binom.txt")
cat("
model {

# Priors

    alpha ~ dnorm(0, 0.001)          # Intercept
    beta2 ~ dnorm(0, 0.001)          # Slope
    sigma ~ dunif(0,10)        # for overdispersion
    tau <- 1 / (sigma * sigma)       # for overdispersion
```

```
# Likelihood
 for (i in 1:15) {
    C[i] ~ dbin(p[i], N[i])
    logit(p[i]) <- alpha + beta2 * dp[i] + eps[i]
    eps[i] ~ dnorm(0, tau)  #Overdispersion parameter

 }

}
",fill=TRUE)
sink()


# Bundle data
win.data <- list(C = pup.de, N = (pup.de + pupnot.de), dp =
(tot.dp/100))

inits <- function(){ list(alpha = rnorm(1), beta2 = rnorm(1))}


# Parameters to estimate
params <- c("alpha", "beta2", "sigma", "tau", "eps")

# MCMC settings
ni <- 200000
nb <- 100000
nt <- 25
nc <- 3

# Start Gibbs sampler
out.od.2 <- bugs(data = win.data, inits = inits, parameters = params,
model.file = "binom.txt",
n.thin = nt, n.chains = nc, n.burnin = nb, n.iter = ni, debug = TRUE,
DIC = TRUE)



#######################################
### Analysis using WinBUGS + Overdispersion: covariates = a.max(beta3) +
DP(beta2) + oyst(beta1)
#####################################
# Define model
sink("binom.txt")
cat("
model {

# Priors

    alpha ~ dnorm(0, 0.001)          # Intercept
    beta1 ~ dnorm(0, 0.001)          # Slope
    beta2 ~ dnorm(0, 0.001)          # Slope
    beta3 ~ dnorm(0, 0.001)          # Slope
    sigma ~ dunif(0, 10)
    tau <- 1 / (sigma * sigma)
```

```
# Likelihood
 for (i in 1:15) {
    C[i] ~ dbin(p[i], N[i])
    logit(p[i]) <- alpha + beta1 * oyst[i] + beta2 * dp[i] + beta3 *
a[i] + eps[i]
    eps[i] ~ dnorm(0, tau)

 }

}
",fill=TRUE)
sink()


# Bundle data
win.data <- list(C = pup.de, N = (pup.de + pupnot.de), oyst =
(oyst/100000), dp = (tot.dp/100), a = (a.max/100))

inits <- function(){ list(alpha = rnorm(1), beta1 = rnorm(1), beta2 =
rnorm(1), beta3 = rnorm(1))}


# Parameters to estimate
params <- c("alpha", "beta1", "beta2", "beta3", "sigma", "tau", "eps")

# MCMC settings
ni <- 200000
nb <- 100000
nt <- 25
nc <- 3

# Start Gibbs sampler
out.od.123 <- bugs(data = win.data, inits = inits, parameters = params,
model.file = "binom.txt",
n.thin = nt, n.chains = nc, n.burnin = nb, n.iter = ni, debug = TRUE,
DIC = TRUE)


#######################################
### 18.4. Analysis using WinBUGS + Overdispersion: covariates =
a.max(beta3) + DP(beta2)
#####################################
# Define model
sink("binom.txt")
cat("
model {

# Priors

    alpha ~ dnorm(0, 0.001)          # Intercept
    beta2 ~ dnorm(0, 0.001)          # Slope
    beta3 ~ dnorm(0, 0.001)          # Slope
    sigma ~ dunif(0, 10)
    tau <- 1 / (sigma * sigma)


# Likelihood
```

```
 for (i in 1:15) {
    C[i] ~ dbin(p[i], N[i])
    logit(p[i]) <- alpha + beta2 * dp[i] + beta3 * a[i] + eps[i]
    eps[i] ~ dnorm(0, tau)

 }

}
",fill=TRUE)
sink()


# Bundle data
win.data <- list(C = pup.de, N = (pup.de + pupnot.de), dp =
(tot.dp/100), a = (a.max/100))

inits <- function(){ list(alpha = rnorm(1), beta2 = rnorm(1), beta3 =
rnorm(1))}


# Parameters to estimate
params <- c("alpha", "beta2", "beta3", "sigma", "tau", "eps")

# MCMC settings
ni <- 200000
nb <- 100000
nt <- 25
nc <- 3

# Start Gibbs sampler
out.od.23 <- bugs(data = win.data, inits = inits, parameters = params,
model.file = "binom.txt",
n.thin = nt, n.chains = nc, n.burnin = nb, n.iter = ni, debug = TRUE,
DIC = TRUE)


#######################################
### Analysis using WinBUGS + Overdispersion: DP(beta2) + oyst(beta1) +
pupnot.de(Beta4)
#####################################
# Define model
sink("binom.txt")
cat("
model {

# Priors

    alpha ~ dnorm(0, 0.001)          # Intercept
    beta1 ~ dnorm(0, 0.001)          # Slope
    beta2 ~ dnorm(0, 0.001)          # Slope
    beta4 ~ dnorm(0, 0.001)          # Slope
    sigma ~ dunif(0,10)
    tau <- 1 / (sigma * sigma)


# Likelihood
 for (i in 1:15) {
    C[i] ~ dbin(p[i], N[i])
    logit(p[i]) <- alpha + beta1 * oyst[i] + beta2 * dp[i] + beta4 * 
pup[i] + eps[i]
    eps[i] ~ dnorm(0, tau)

 }

}
",fill=TRUE)
sink()


# Bundle data
win.data <- list(C = pup.de, N = (pup.de + pupnot.de), oyst = 
(oyst/100000), dp = (tot.dp/100), pup = (pupnot.de/100))

inits <- function(){ list(alpha = rnorm(1), beta1 = rnorm(1), beta2 = 
rnorm(1), beta4 = rnorm(1))}


# Parameters to estimate
params <- c("alpha", "beta1", "beta2", "beta4", "sigma", "tau", "eps")

# MCMC settings
ni <- 200000
nb <- 100000
nt <- 25
nc <- 3

# Start Gibbs sampler
out.od.124 <- bugs(data = win.data, inits = inits, parameters = params, 
model.file = "binom.txt", 
n.thin = nt, n.chains = nc, n.burnin = nb, n.iter = ni, debug = TRUE, 
DIC = TRUE)




#######################################
### 18.4. Analysis using WinBUGS + Overdispersion: pupnot.de(Beta4)
#####################################
# Define model
sink("binom.txt")
cat("
model {

# Priors

    alpha ~ dnorm(0, 0.001)          # Intercept
    beta4 ~ dnorm(0, 0.001)          # Slope
    sigma ~ dunif(0,10)
    tau <- 1 / (sigma * sigma)


# Likelihood
 for (i in 1:15) {
    C[i] ~ dbin(p[i], N[i])
    logit(p[i]) <- alpha + beta4 * pup[i] + eps[i]
    eps[i] ~ dnorm(0, tau)

 }

}
",fill=TRUE)
sink()


# Bundle data
win.data <- list(C = pup.de, N = (pup.de + pupnot.de), pup =
(pupnot.de/100))

inits <- function(){ list(alpha = rnorm(1), beta4 = rnorm(1))}


# Parameters to estimate
params <- c("alpha", "beta4", "sigma", "tau", "eps")

# MCMC settings
ni <- 100000
nb <- 50000
nt <- 20
nc <- 3

# Start Gibbs sampler
out.od.4 <- bugs(data = win.data, inits = inits, parameters = params,
model.file = "binom.txt",
n.thin = nt, n.chains = nc, n.burnin = nb, n.iter = ni, debug = TRUE,
DIC = TRUE)
```