Bayesian Dynamic Models

Models for Socio-Environmental Data

Chris Che-Castaldo, Mary B. Collins, N. Thompson Hobbs

June 11, 2019

Roadmap

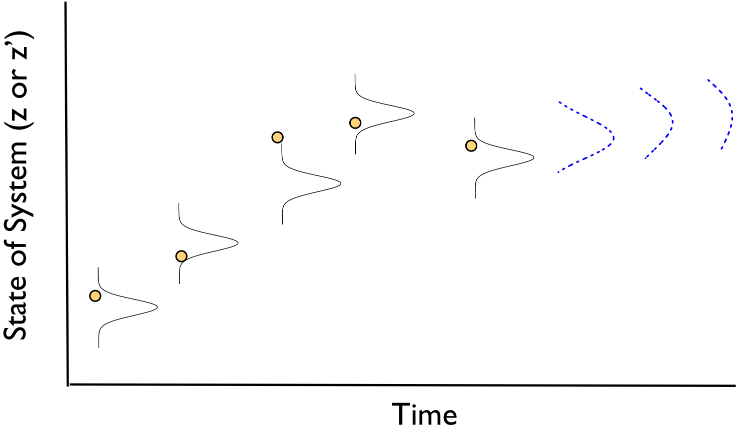
- Overview
- Model types with examples
  - discrete time
    - single state
    - multiple states
  - continuous time (briefly)
- Autocorrelation
- Forecasting
- Coding tips

Dynamic hierarchical models (aka state space models)

Also called “state space” models

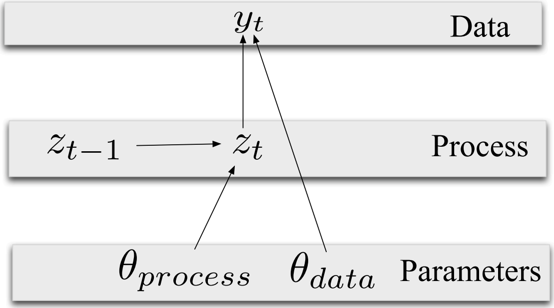
$$[y_t|\boldsymbol{\theta}_d, z_t]$$
$$[z_t|\boldsymbol{\theta}_p, z_{t-1}]$$

The idea is simple. We have a stochastic model of an unobserved, true state (z_t) and a stochastic model that relates our observations (y_t) to the true state.


State of System (z or z')
Time

A broadly applicable approach to modeling dynamic processes in ecology and social science

$$[\mathbf{z}, \theta_{process}, \theta_{data} | \mathbf{y}] \propto \prod_{t=2}^{T} [y_t | \theta_{data}, z_t] \, [z_t | \theta_{process}, z_{t-1}] \, [\theta_{process}, \theta_{data}, z_1]$$



Sources of uncertainty in state space models

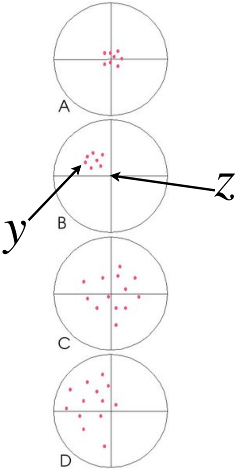
Process uncertainty

- Failure to perfectly represent process
- Propagates in time
- Decreases with model improvement
- Basis for forecasting

Observation uncertainty

- Failure to perfectly observe process
- Does not propagate
- Sampling uncertainty decreases with increased sampling effort.
- Observation (calibration) uncertainly decreases with improved instrumentation, calibration, etc.

Components of observation uncertainty



- Observation (aka calibration) $\left[y|h\left(z,\boldsymbol{\theta}_d\right),\boldsymbol{\sigma}_o^2\right]$
- Sampling $\left[y|z,\boldsymbol{\sigma}_s^2\right]$

When can we separate process variance from observation variance?

- Replication of the observation for the latent state with sufficient n
- Calibration model with properly estimate prediction variance
- Strongly differing “structure” in process and observation models
- We may not need to separate them–sometimes the observed state and the true state are the same.

General joint and posterior distribution for single state model

$$
\begin{aligned}
\text{Deterministic model} &= g(\boldsymbol{\theta}_{process}, z_{t-1}, \mathbf{x}_{t-1}) \\
\left[\mathbf{z}, \boldsymbol{\theta}_{process}, \boldsymbol{\theta}_{data}, \boldsymbol{\sigma}_p^2, \boldsymbol{\sigma}_d^2 | \mathbf{y}\right] &\propto \prod_{t=2}^{T} \left[y_t | \boldsymbol{\theta}_{data}, z_t, \boldsymbol{\sigma}_d^2\right] \\
&\quad \times \left[z_t | g(\boldsymbol{\theta}_{process}, z_{t-1}, \mathbf{x}_{t-1}), \boldsymbol{\sigma}_d^2\right] \\
&\quad \times \left[\boldsymbol{\theta}_{process}, \boldsymbol{\theta}_{data}, \boldsymbol{\sigma}_p^2, \boldsymbol{\sigma}_d^2, z_1\right]
\end{aligned}
$$

Deterministic matrix model

Process model:

$$\begin{pmatrix} z_1 \\ z_2 \\ z \\ . \\ z_n \end{pmatrix}_t = \hat{} \begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ . \\ z_n \end{pmatrix}_{t-1} \tag{1}$$

where $\boldsymbol{\Theta}$ is an $n \times n$ matrix governing the transitions among states. The product $\boldsymbol{\Theta}\mathbf{z}_t$ defines a system of n linked, difference equations. We can learn a great deal about the dynamics of the system from analyzing the properties of $\boldsymbol{\Theta}$, its eigenvalues, eignvectors, characteristic polynomials, etc. We can make inference on these using derived quantities.

Posterior and joint distribution

$$[\mathbf{z}, \boldsymbol{\Theta}, \boldsymbol{\theta}_{data} | \mathbf{Y}] \propto \prod_{t=2}^{T} [\mathbf{y}_t | \boldsymbol{\theta}_{data}, \mathbf{z}_t] [\mathbf{z}_t | \boldsymbol{\Theta}, \mathbf{z}_{t-1}] [\boldsymbol{\Theta}, \boldsymbol{\theta}_{data}, \mathbf{z}_1]$$

Systems of differential equations

$$
\begin{aligned}
\frac{dz_1}{dt} &= k_1 z_1 - k_2 z_1 z_2 \\
\frac{dz_2}{dt} &= - k_3 z_1 + \alpha k_2 z_1 z_2 \\
\frac{dz_3}{dt} &= \frac{k_4 z_3}{k_5 + z_3}
\end{aligned}
$$

Implementing the process model usually needs a numerical solver to align the states with the data.

Continuous time models

- Must deterministically update states at discrete intervals to match with data
- To estimate states:
  - Use analytical solutions to ODE system if available.
  - For models without analytical solutions:
    - STAN has superb ODE solver. [1]
    - R's Nimble package [2] allows you to embed functions in JAGS. A sturdy ODE solver (Runge-Kutta IV) can be written in 6-8 lines of code.
    - Write your own MCMC sampler with embedded numerical solver (e.g. `lsoda()` in R). [3]

[1] https://mc-stan.org/events/stancon2017-notebooks/stancon2017-margossian-gillespie-ode.html

[2] https://r-nimble.org/

[3] See: Campbell, E. E., W. J. Parton, J. L. Soong, K. Paustian, N. T. Hobbs, and M. F. Cotrufo. 2016. Using litter chemistry controls on microbial processes to partition litter carbon fluxes with the Litter Decomposition and Leaching (LIDEL) model. Soil Biology & Biochemistry 100:160-174.

The problem:

Assume for simplicity that the state is observed perfectly. The simplest model of the change in state with time is

$$y_t = \alpha y_{t-1} + \varepsilon_t \tag{2}$$

where $\mathsf{E}(y_t) = 0$ and $\varepsilon_t \sim \text{normal}(0, \sigma^2)$. We might introduce effects of predictor variables using

$$y_t = g(\boldsymbol{\theta}, \mathbf{x}_t) + \alpha y_{t-1} + \varepsilon_t. \tag{3}$$

What if ε_t depends on previous errors, that is, $e_t = h(e_{t-1})$? In this case, there is structural variation in the data, also called temporal dependence. The assumptions of independent errors does not hold. We have two choices:

1. Improve $g(\boldsymbol{\theta}, \mathbf{x}_t)$ so that the deterministic model accounts for the temporal dependence via the covariates.
2. Model the temporal dependence in the errors directly.

Detecting temporal dependence

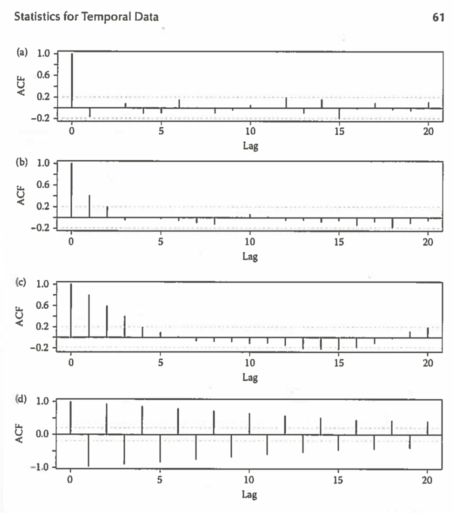
The empirical autocorrelation function (ACF):

$$\rho_g = \frac{\sum_{i=1}^{n-g}(\varepsilon_i - \bar{\varepsilon})(\varepsilon_{i+g} - \bar{\varepsilon})}{\sum_{i=1}^{N}(\varepsilon_i - \bar{\varepsilon})^2}$$

where n is the number of steps in the time series and g is the "lag," the number of steps examined for temporal dependence, $-1 \leq \rho_g \leq 1$

The notation ACF(g) means the correlation between points separated by g time periods.

ACF plots


Statistics for Temporal Data
61
(a)
(b)
(c)
(d)
ACF
Lag
1.0
0.6
0.2
-0.2
0.0
-1.0
0
5
10
15
20

ACF in MCMC

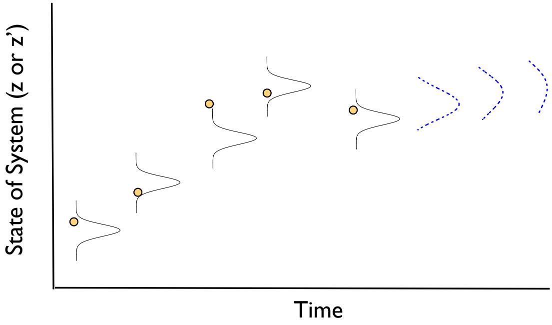
$\boldsymbol{\mu}_t = g(\boldsymbol{\theta}, z_{t-1}, \mathbf{x}_{t-1})$

1. Compute residuals at each MCMC iteration, $e_t^{(k)} = y_t - \boldsymbol{\mu}_t^{(k)}$
2. Compute $\boldsymbol{\rho}_g^{(k)}$ at each MCMC iteration and plot posterior means of $\boldsymbol{\rho}_g^{(k)}$ as a function of g.
3. Or, better and easier, sample from MCMC output for $e_t^{(k)}$, use `acf()` function in R to find posterior distributions of $\boldsymbol{\rho}_g$. Make statements like "Mean autocorrelation was .21 (BCI = .23,.18) at lag 3, revealing minimal temporal dependence in the residuals."

Bayesian forecasting future states z'

$$\underbrace{\left[z'_{T+1}|\mathbf{y}\right]}_{\text{predictive process distribution}} =$$

$$\int_{\boldsymbol{\theta}_1\ldots\boldsymbol{\theta}_P}\int_{z_1\ldots}\int_{z_T}\left[z'_{T+1}|\mathbf{z},\boldsymbol{\theta}_{process},\mathbf{y}\right]\underbrace{\left[\mathbf{z},\boldsymbol{\theta}_{process},\boldsymbol{\theta}_{data}|\mathbf{y}\right]}_{\text{posterior distribution}}dz\ldots dz_t\,d\boldsymbol{\theta}_1\ldots d\boldsymbol{\theta}_P$$



Predictive process distribution

The MCMC output:

i	$\boldsymbol{\theta}_1$	$\boldsymbol{\theta}_1$	$\boldsymbol{\theta}_3$								
1	.42	3.3	20.3	$z_{1,1}$	$z_{1,2}$	$\cdots$	$z_{1,T}$	$z'_{1,T+1}$	$z'_{1,T+2}$	$\cdots$	$z'_{1,T+F}$
2	.41	2.3	18.5	$z_{2,1}$	$z_{2,2}$	$\cdots$	$z_{2,T}$	$z'_{2,T+1}$	$z'_{,2T+2}$	$\cdots$	$z'_{2,T+F}$
3	.46	3.1	16.6	$z_{3,1}$	$z_{3,2}$	$\cdots$	$z_{3,T}$	$z'_{3,T+1}$	$z'_{3,T+2}$	$\cdots$	$z'_{3T+,F}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
n	.39	3.4	22.1	$z_{n,1}$	$z_{n,2}$	$\cdots$	$z_{n,T}$	$z'_{n,T+1}$	$z'_{n,T+2}$	$\cdots$	$z'_{n,T+F}$

n = number of iterations

T = final time with data

F = number of forecasts beyond data

Posterior and joint distribution with forecasts

$$\mu_t = g(\boldsymbol{\theta}_{process}, z_{t-1}, \mathbf{x}_{t-1})$$

$$[\mathbf{z}, \boldsymbol{\theta}_{process}, \boldsymbol{\theta}_{data} | \mathbf{y}] \propto$$

$$\prod_{t=2}^{T} [y_t | \boldsymbol{\theta}_{data}, z_t] \prod_{t=2}^{T+F} [z_t | \boldsymbol{\mu}_t] [\boldsymbol{\theta}_{process}, \boldsymbol{\theta}_{data}, z_1]$$

Posterior and joint distribution with missing data

$$\mu_t = g(\boldsymbol{\theta}_{process}, z_{t-1}, \mathbf{x}_{t-1})$$

$$[\mathbf{z}, \boldsymbol{\theta}_{process}, \boldsymbol{\theta}_{data} | \mathbf{y}] \propto$$

$$\prod_{\forall t \in \mathbf{y}.i}^{T} [y_t | \boldsymbol{\theta}_{data}, z_t] \prod_{t=2}^{T} [z_t | \mu_t] [\boldsymbol{\theta}_{process}, \boldsymbol{\theta}_{data}, z_1]$$

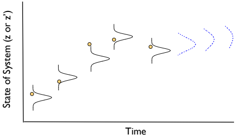
Can put NA's in data for all missing values or use the indexing trick shown below.

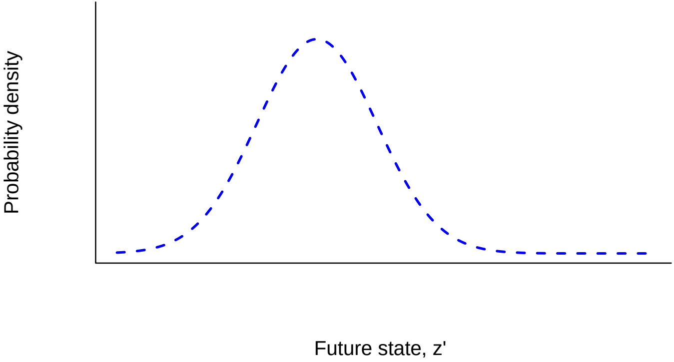
Forecasting

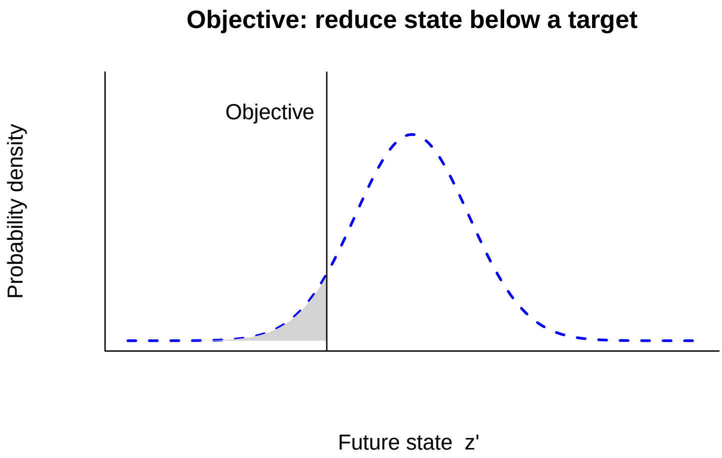
The fundamental problem of management:

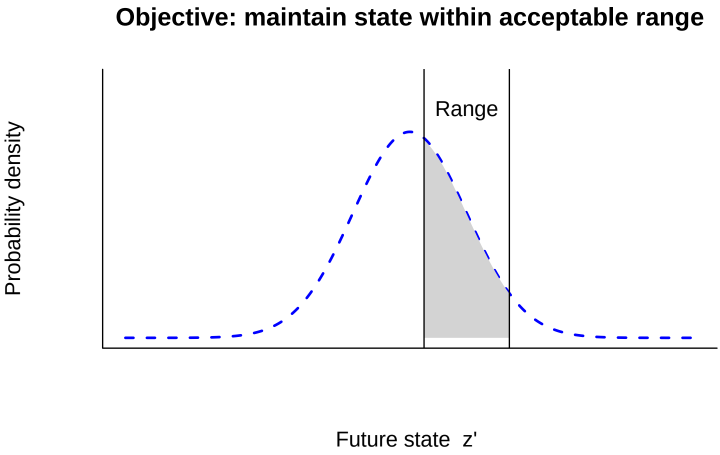
What actions can we take today that will allow us to meet goals for the future?

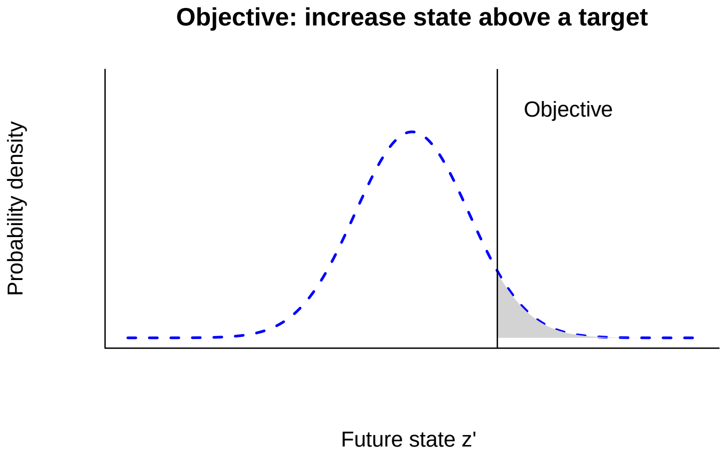
Predictive process distribution of z'

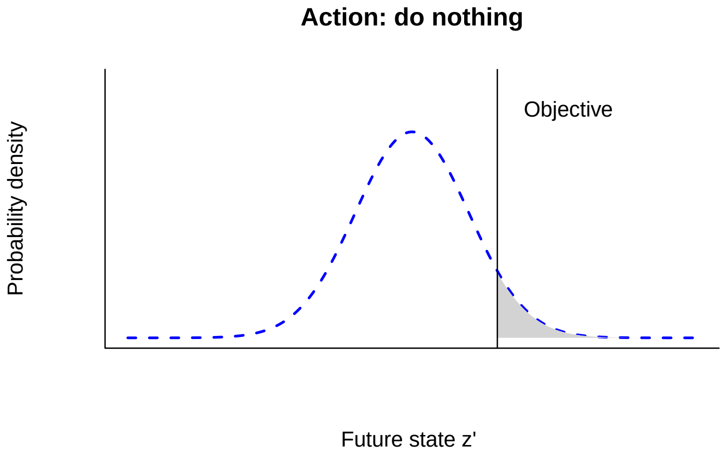


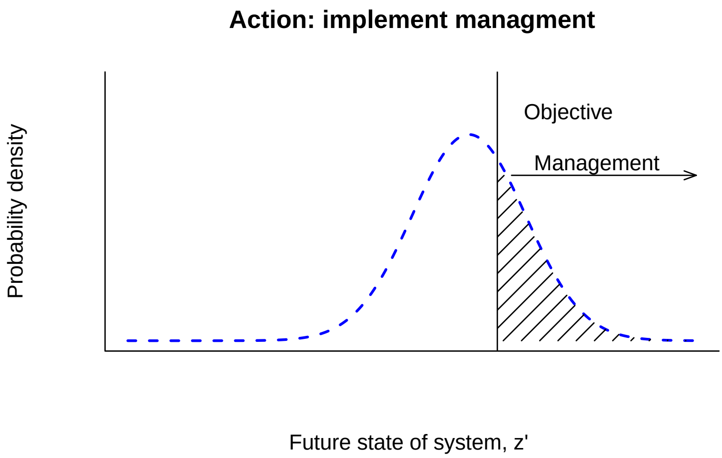


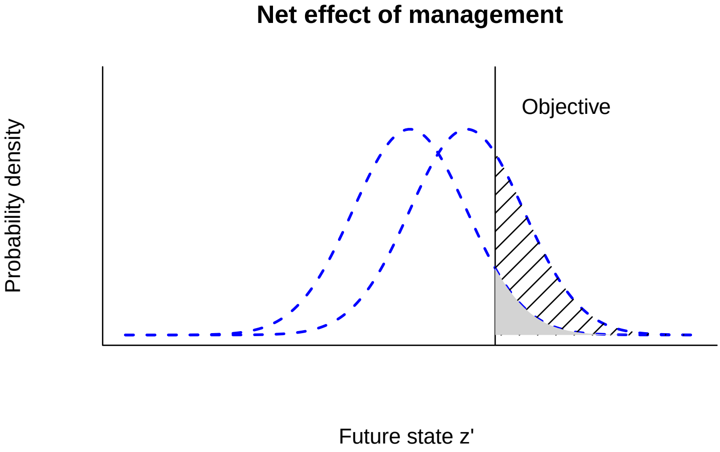

Objective: reduce state below a target
Objective
Probability density
Future state z'

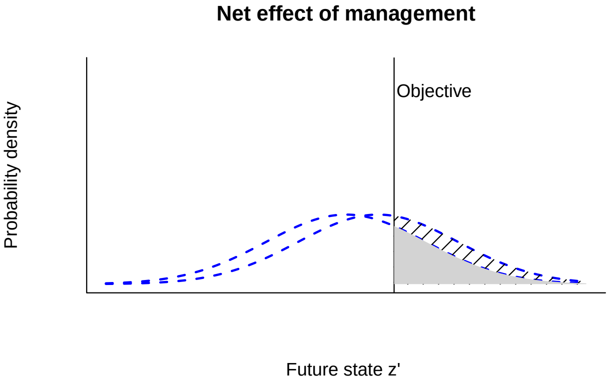

Objective: maintain state within acceptable range
Range
Probability density
Future state z'


Objective: increase state above a target
Objective
Probability density
Future state z'


Action: do nothing
Objective
Probability density
Future state z'


Action: implement managment
Objective
Management
Probability density
Future state of system, z'


Net effect of management
Objective
Probability density
Future state z'


Net effect of management
Objective
Probability density
Future state z'

Papers using forecasting relative to goals

- Ketz, A. C., T. L. Johnson, R. J. Monello, and N. T. Hobbs. 2016. Informing management with monitoring data: the value of Bayesian forecasting. Ecosphere 7:e01587-n/a.
- Raiho, A. M., M. B. Hooten, S. Bates, and N. T. Hobbs. 2015. Forecasting the Effects of fertility control on overabundant ungulates: white-tailed deer in the National Capital Region. PLoS ONE 10.
- Hobbs, N. T., C. Geremia, J. Treanor, R. Wallen, P. J. White, M. B. Hooten, and J. C. Rhyan. 2015. State-space modeling to support management of brucellosis in the Yellowstone bison population. Ecological Monographs 85:3-28.

More on forecasting

- M. C. Dietz. Ecological Forecasting. Princeton University Press, Princeton New Jersey, USA, 2017.
- Workshop July 28 - August 2
  https://ecoforecast.wordpress.com/summer-course/

JAGS code for posterior and joint distributions

$$\left[\mathbf{z},\boldsymbol{\beta},\sigma_p^2|\mathbf{y}\right] \propto \underbrace{\prod_{\forall t \in y.i}\left[y_t \,\middle|\, z_t, y.sd_t\right]}_{\text{data model}}$$

$$\times \underbrace{\prod_{t=2}^{48}\left[z_t|g\left(\boldsymbol{\beta}, z_{t-1}, x_{t-1}\right), \sigma_p^2\right]}_{\text{process model}} \times \underbrace{[\beta_0][\beta_1][\beta_2][\beta_3]\left[\sigma_p^2\right][z_1]}_{\text{parameter models}}$$

```
model{
#Priors
b[1] ~ dnorm(.234,1/.136^2)
for(j in 2:n.coef){
b[j] ~ dnorm(0,.0001)
}
tau.p ~ dgamma(.01,.01)
sigma.p <- 1/sqrt(tau.p)
z[1]  ~ dnorm(N.obs[1],tau.obs[1])  #this must be treated as prior so that you have z[t-
1]
##Process model
for(t in 2:(T+F)){
mu[t] <- log(z[t-1]*exp(b[1] + b[2]*z[t-1] + b[3]*x[t] +b[4]*x[t]*z[t-1]))
z[t] ~ dlnorm(mu[t], tau.p)
}

#Data model
for(j in 2:n.obs){
N.obs[j] ~ dnorm(z[index[j]],tau.obs[j]) #index to match z[t] with data
}
}#end of model
```

Posterior predictive checks for time series data

Test statistic:

$$\frac{1}{T-1}\sum_{t=2}^{T} \mid y_t - y_{t-1} \mid \tag{4}$$

Conventional statistics are also used (mean, CV, discrepancy statistic for the y_t.

Reilly, C., A. Gelman, and J. Katz, 2001. Poststratification without Population Level Information 731 on the Poststratifying Variable, with Application to Political Polling. Journal of the American 732 Statistical Association 96:1–11.

Posterior predictive checks and test for autocorrelation

```
#Derived quantities for model evaluation

for(i in 1:n.obs){

    #for autocorrelation test

epsilon.obs[i] <- N.obs[i] - z[index[i]]

 # simulate new data

        N.new[i] ~ dnorm(z[index[i]],tau.obs[i])

sq[i] <- (N.obs[i] - z[index[i]] )^2
sq.new[i] <-(N.new[i] - z[index[i]]) ^2
}
fit  <- sum(sq[])
fit.new <- sum(sq.new[])
pvalue <-step(fit.new-fit)
```