An introduction to hidden markov models for time series

FISH 507 – Applied Time Series Analysis

Eric Ward

4 Feb 2021

Overview of today's material

- Gentle introduction to HMMs
- Theory and notation
- Examples of univariate HMMs in R
- Examples of multivariate HMMs in R

Overview of today's material

For additional background, see

- Zucchini et al. (2008, 2016) "Hidden Markov Models for Time Series: An Introduction Using R"
- Jackson (2011) "Multi-State Models for Panel Data: The msm Package for R"
- Visser and Speekenbrink (2010) "depmixS4: An R Package for Hidden Markov Models"
- McClintock et al. (2020) "Uncovering ecological state dynamics with hidden Markov models"

State space models

We've already discussed state space models. These models include

- a latent process model (we don't directly observe)
- a data or observation model
- we've generally assumed both models be normal (or multivariate normal)

State space models

Process model:

$$x_t = x_{t-1} + \varepsilon_{t-1}$$

Observation model:

$$y_t = x_t + \delta_t$$

where $\varepsilon_t \sim \mathit{Normal}\,(0, \sigma_\varepsilon)$ and $\delta_t \sim \mathit{Normal}\,(0, \sigma_\delta)$

State space models

Adding AR coefficients can make these models stationary with respect to the mean,

$$x_t = p \cdot x_{t-1} + \varepsilon_{t-1}$$

however they may not be able to explain some datasets very well.

- Specifically, these models are not well designed to model regimes

Regimes

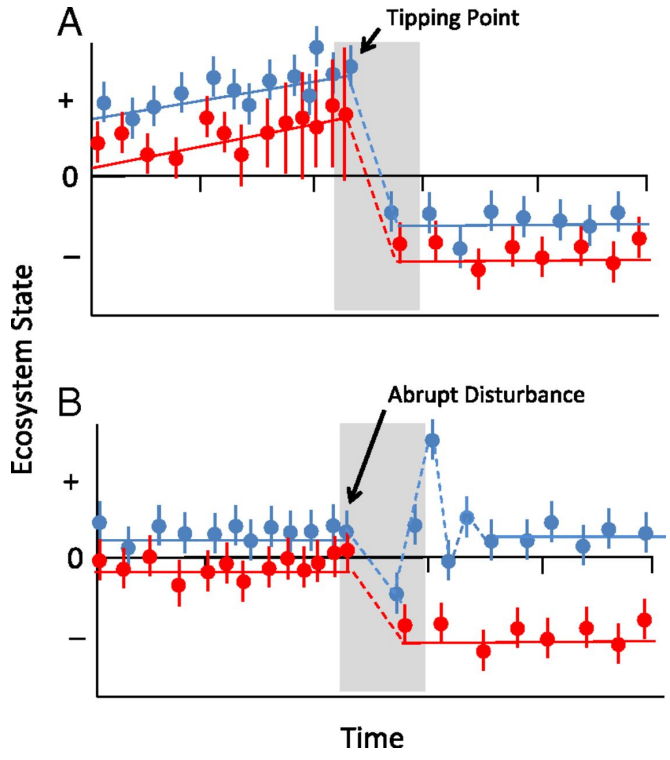


Figure 1: Example from Moore (2018), "Predicting tipping points in complex environmental systems"

Regimes / behavioral states

Many examples of time series in ecology & fisheries may alternate between multiple states (2+)

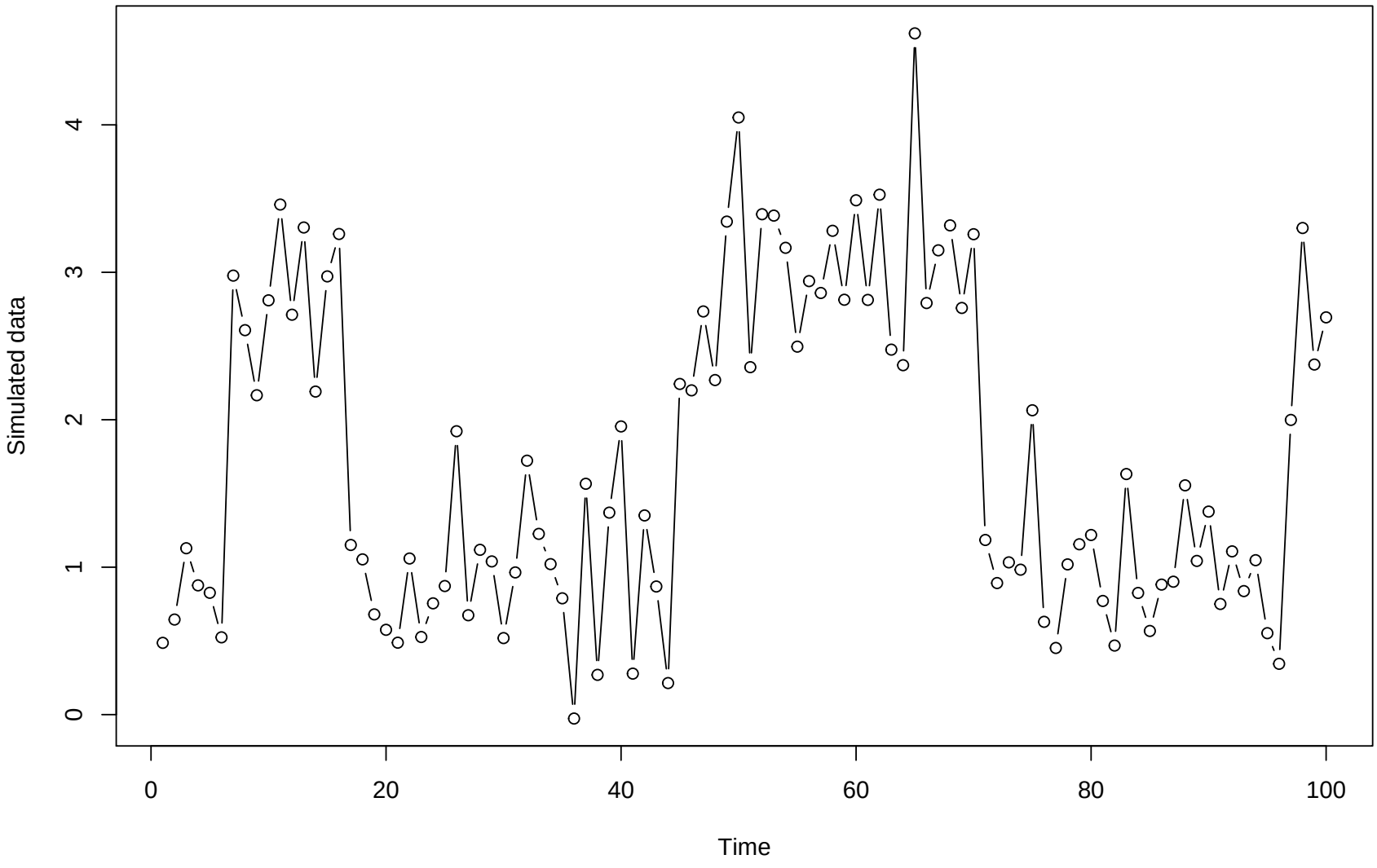
- Vert-pre et al. (2013) https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3562848/
- Francis et al. (2012) https://onlinelibrary.wiley.com/doi/full/10.1111/j.1365-2486.2012.02702.x
- Leos-Barajas et al. (2017) "Analysis of animal accelerometer data using hidden Markov models"

Regimes

Lots of non-HMM approaches for detecting regimes

- STARS algorithm
  - Sequential t-test approach for detecting changes in the mean
  - Rodionov (2015) https://www.mdpi.com/2225-1154/3/3/474
- Brute force model selection approach
  - iterate through change points, evaluating data support for each
  - how do we do change points with regression? $Y_t = BX_t + \varepsilon_t$

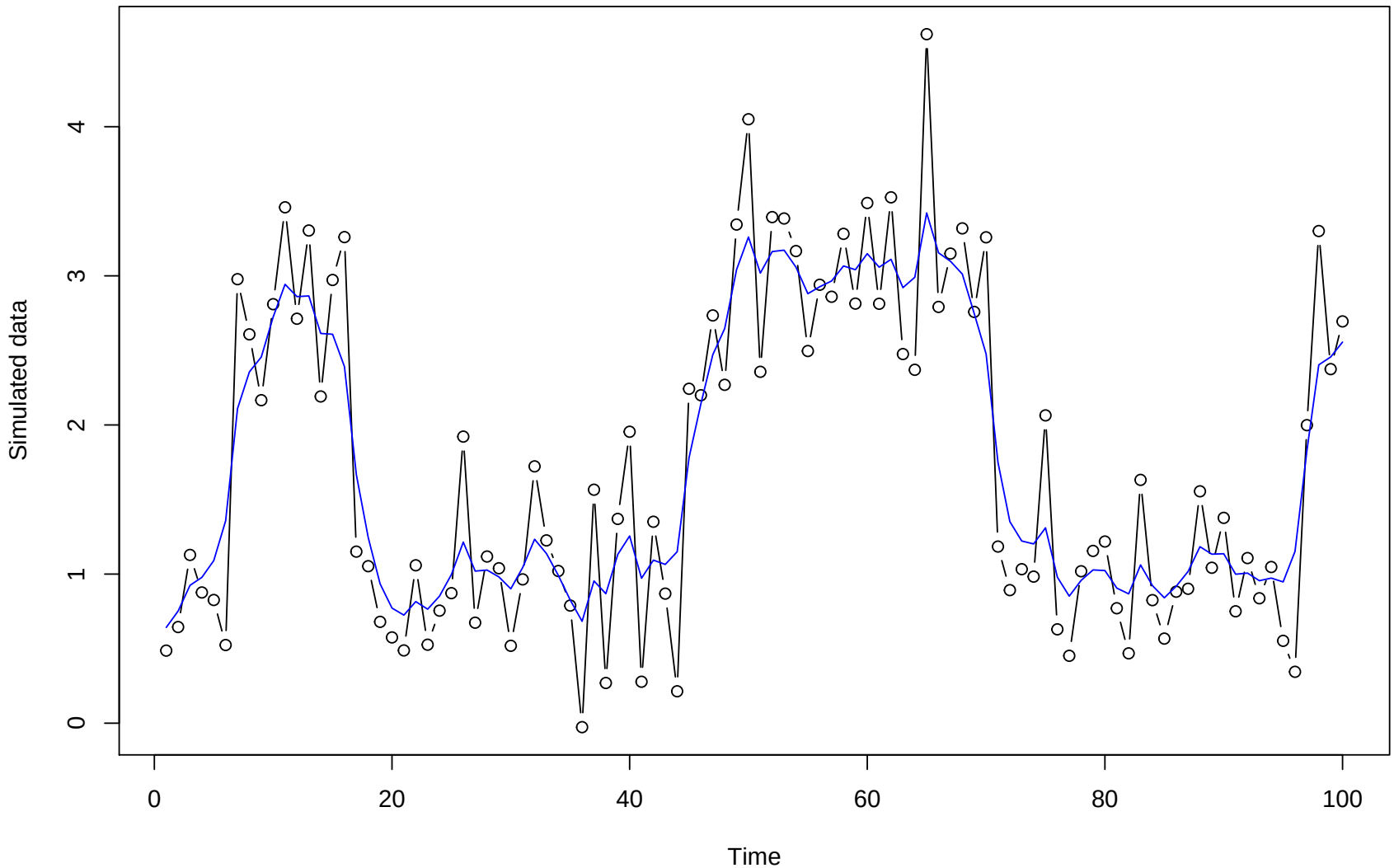
Regimes: simulating data



Fitting these data with state space models

They can actually fit the data from a regime model quite well.

- Via MARSS()



Limitations of state space models with continuous states

What's lacking is inference about:

- What's the probability of transition between regimes?
- How long are regimes expected to last?
- What regimes might we expect in the future?

Lots of applications: speech recognition, bioinformatics, animal movement, environmental data (rain), finance

HMM: theory

Markov Process

- time may be discrete or continuous (we'll focus on discrete)
- Markov if at each time step, the state x_t is only dependent on the previous state x_{t-1}
- x_t does not depend on future values
- these ideas should be familiar, with the exception of states changing from continuous to discrete

Entire history can be written as

$$\{x_1, x_2, x_3, ..., x_T\}$$

HMM: theory

A key piece of the Markov process are the transition probabilities.

- The probability of transitioning from state j to i given the current state is

$$P\left(x_{t+1} = j | x_t = i\right) = \gamma_{ij}$$

- And then these probabilities can be summarized in a transition matrix,

$$\Gamma = \begin{bmatrix} \gamma_{11} & \gamma_{12} & \gamma_{13} \\ \gamma_{21} & \gamma_{22} & \gamma_{23} \\ \gamma_{31} & \gamma_{32} & \gamma_{33} \end{bmatrix}$$

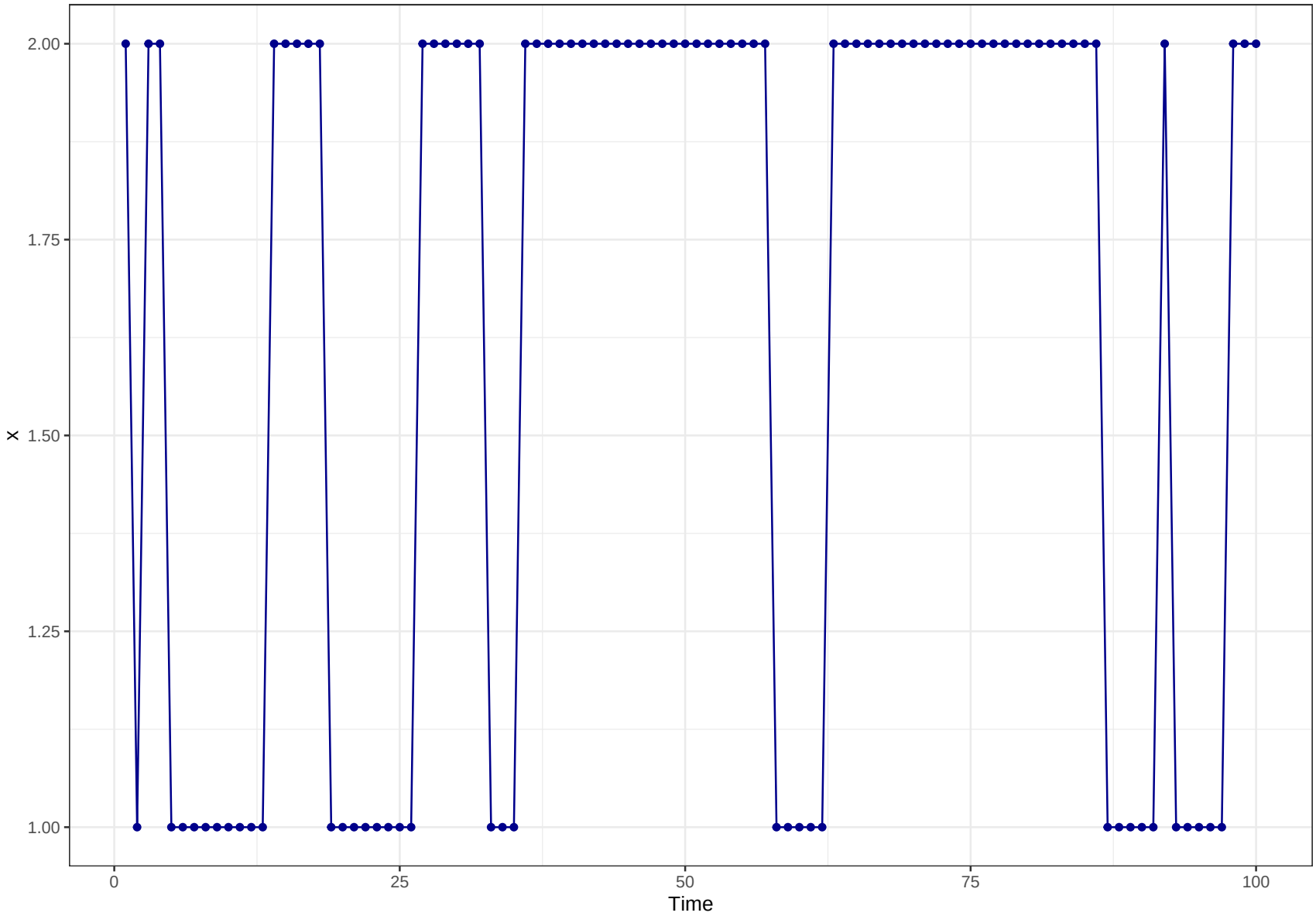
- Elements of Γ can be fixed a priori (0, etc)

HMMs theory

- Let's try to simulate this in R

```
# initial state
x <- sample(1:2, size=1)
# transition matrix
Gamma = matrix(c(0.9,0.1,0.2,0.8),2,2)
for(t in 2:100) {
  x[t] <- sample(1:2, size=1,prob = Gamma[x[t-1],])
}
```

HMMs theory



HMM theory

Matrix Γ is the 1-step transitions. However k-step transition probabilities can be generated,

$\Gamma(k) = \Gamma^k$

- From this, we can also calculate the stationary distribution of the chain
  - See Zucchini et al. (2006) Chapter 2

HMM theory

There are two general flavours of transition matrices:

- Homogenous (or stationary)
  - transition probabilities don't depend on t
- Non-homogeneous
  - transition probabilities are time-varying
- In this class, we'll only consider **homogeneous** cases

HMM theory

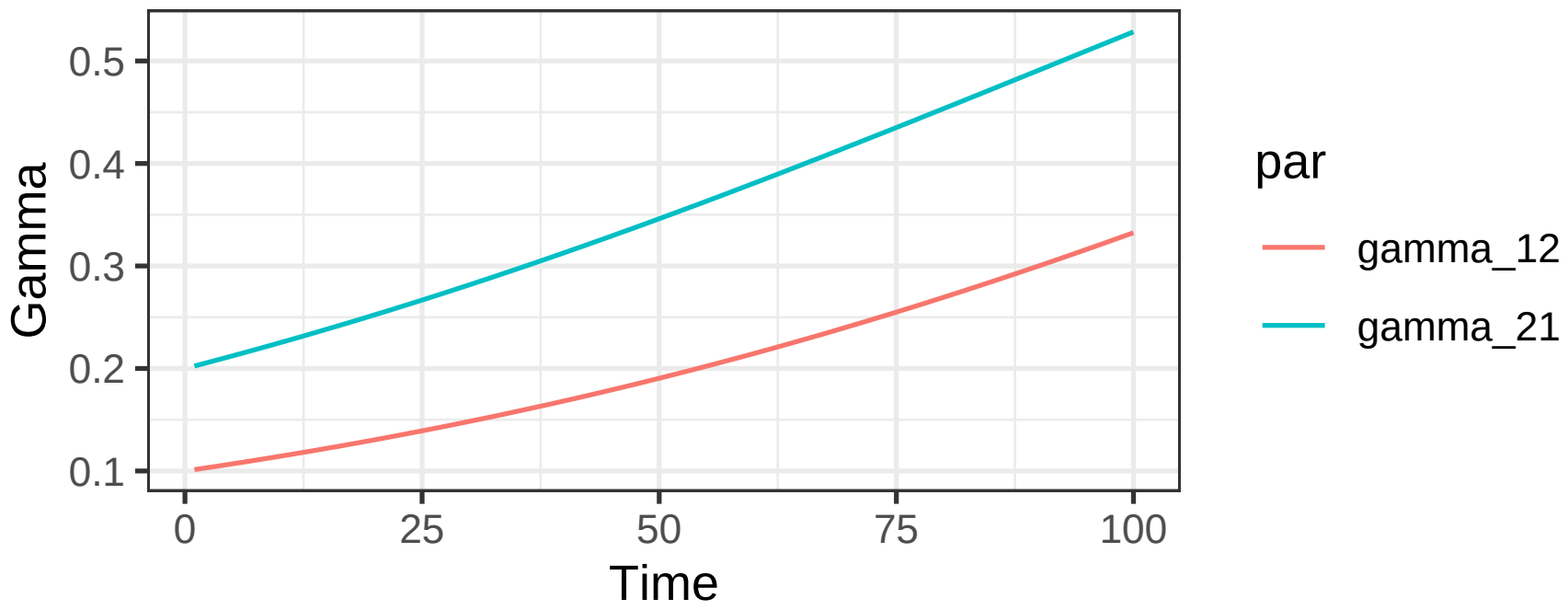
Covariates may enter HMMs in one of two ways

- Effects on the mean
- Effects on the transition matrix Γ
- For effects on Γ, probabilities are constrained (0,1) and constrained to sum to 1
  - multivariate logit regression used to relate covariates to elements of Γ
  - reference / base case is fixed at zero (Agresti 2002)

HMMs theory

- Returning to our simulation example, let's include a temporal trend in the transition probabilities
- We assume trend is linear in logit space
- For simplicity, we assume that slope is same on γ_{12} and γ_{21}

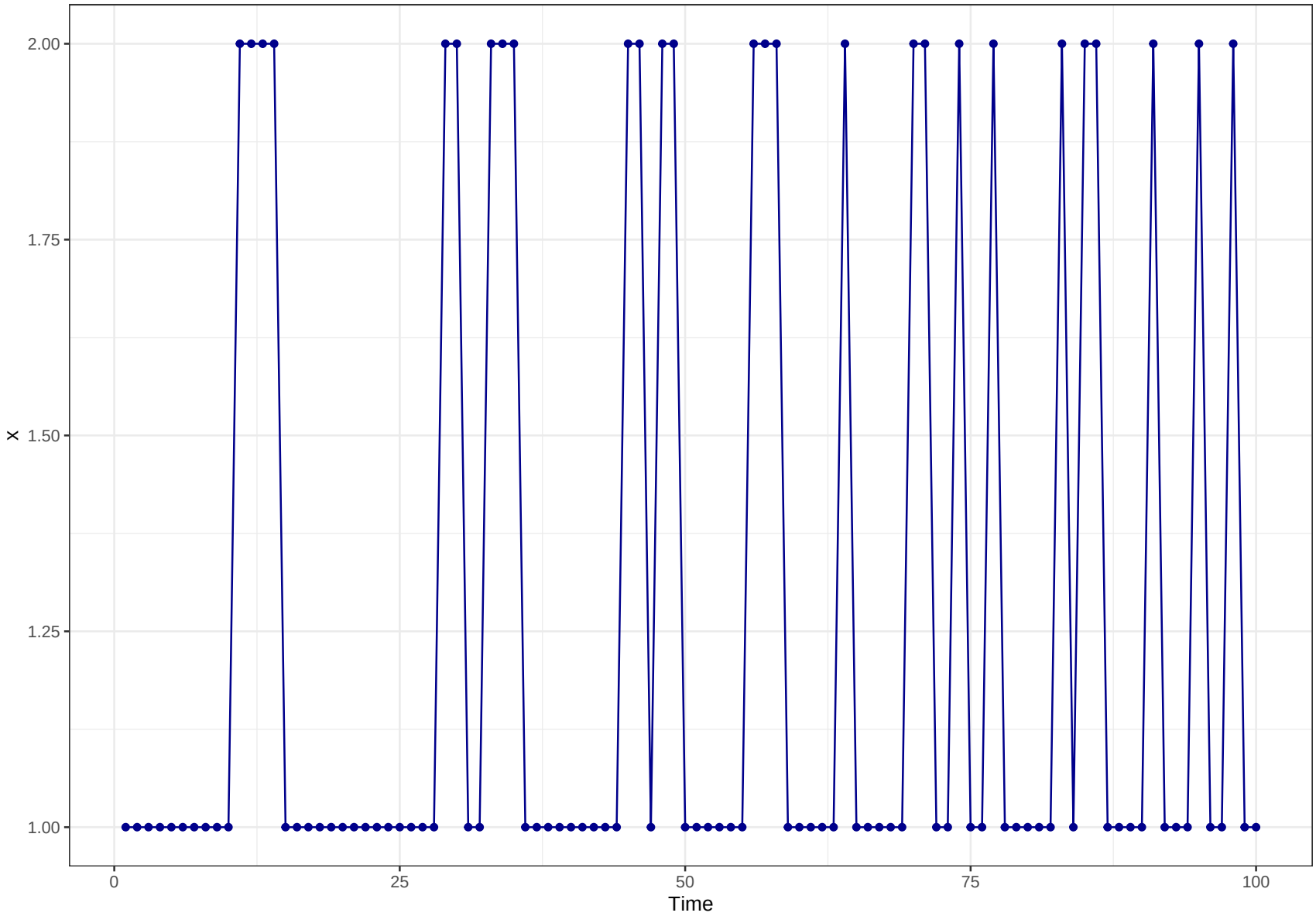
```
plogis(logit(0.1) + 0.015*1:100)
```



HMMs theory

```r
# initial state
x <- sample(1:2, size=1)
# transition matrix
Gamma = matrix(c(0.9,0.1,0.2,0.8),2,2)
for(t in 2:100) {
  Gamma[1,2] = plogis(logit(0.1) + 0.015*t)
  Gamma[1,1] = 1 - Gamma[1,2]
  Gamma[2,1] = plogis(logit(0.2) + 0.015*t)
  Gamma[2,2] = 1 - Gamma[2,1]
  x[t] <- sample(1:2, size=1,prob = Gamma[x[t-1],])
}
```

HMMs theory



HMM: theory

- Observations: observable data $Y_{i=1,\ldots,N}$
- States: latent (discrete) variables that are not directly observed
  - $x_{t=1,\ldots,T}$
  - N is the number of states possible
- Transition probabilities: transition matrix representing the probability of transitioning between states in the Markov chain
  - Γ and γ_{ij}
- Emission probabilities: how likely the states are at any particular timestep
  - $\theta_{i=1,\ldots,N}$

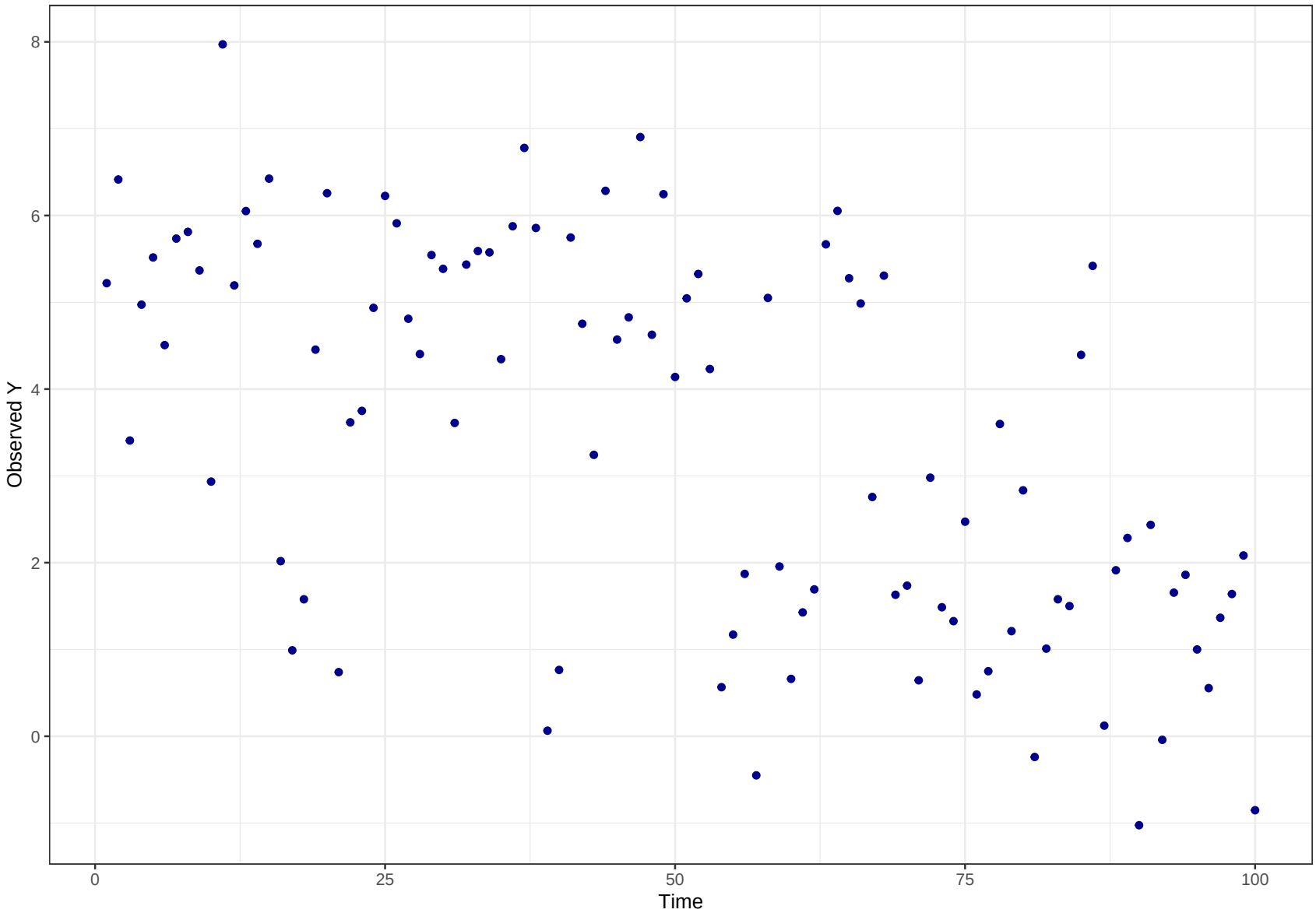
HMM: theory

- Our simulations haven't yet included an observation component. Let's add observed data that's normally distributed

```
# initial state
x <- sample(1:2, size=1)
# transition matrix
Gamma = matrix(c(0.9,0.1,0.2,0.8),2,2)
for(t in 2:100) {
  x[t] <- sample(1:2, size=1,prob = Gamma[x[t-1],])
}

u = c(1.2, 5.4)
y = rnorm(length(x), u[x], 1)
```

HMM: theory



HMM: theory

One quantity of interest from HMMs is the marginal distribution, $P(Y_t)$

Output from the model will give us the conditional distribution, $P(Y_t|x_t = i)$. But we need to marginalize over all possible states

Solution:

$$P(Y_t) = \sum_{i=1}^{N} P(x_t = i)P(Y_t|x_t = i)$$

which can also be expressed as

$$P(Y_t) = \sum_{i=1}^{N} \delta_i P(Y_t|x_t = i)$$

where δ represents the stationary distribution (Zucchini et al. 2006)

HMM: theory

Estimation of HMM parameters can be quite complicated

Dealing with joint likelihood of observed data and unobserved states

$$P(x, Y) = \prod_{i=1}^{T} P(x_t|x_{t-1})P(Y_t|x_t)$$

HMM: theory

Estimation most commonly done with maximum likelihood

1. Forward-backward algorithm: calculate probability of observed data sequence
2. Viterbi algorithm: used to generate most likely states
3. EM-algorithm: estimate / update parameters

- For forward-backward algorithm we can factor conditional state probabilties

$$P\left(x_t|Y_{1:T}\right) = P(x_t|Y_{1:t}, Y_{t+1:T}) = P(x_t|Y_{1:t})P(Y_{t+1:T}|x_t)$$

* Probability of state given the data is the probability of the state given the data up to that time step multiplied by the probability of future data given state

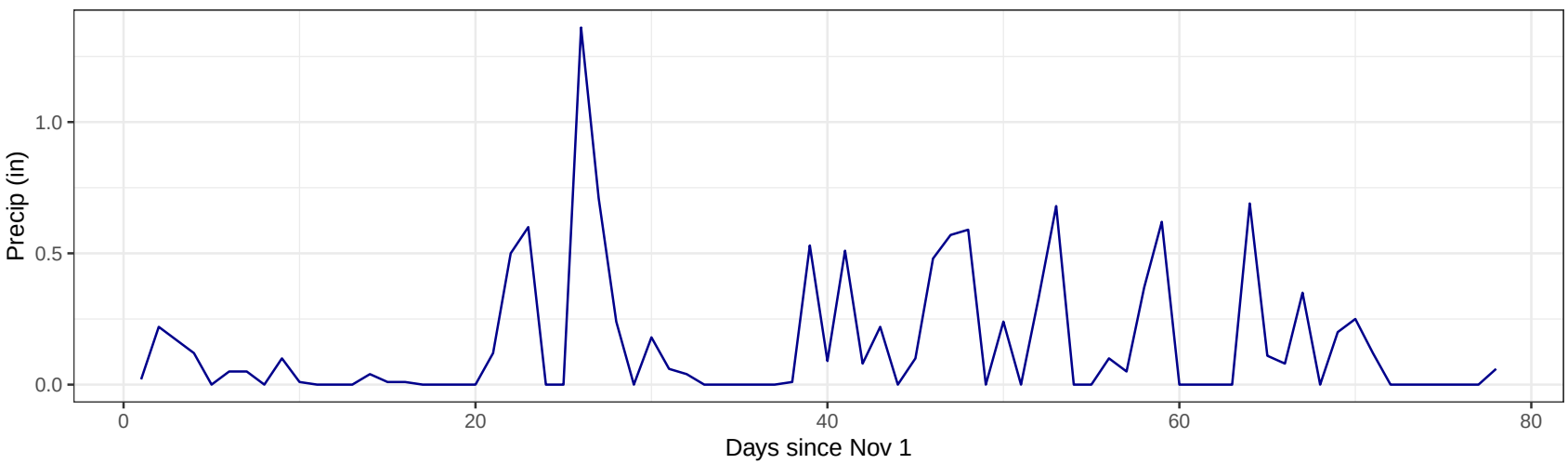
HMM: theory

Forward-backward algorithm has 3 steps (sometimes 2/3 combined):

1. Forward probabilities
   - from the last slide, this step calculates $P(x_t|Y_{1:t})$
2. Backward probabilities
   - from the last slide, this step calculates $P(Y_{t+1:T}|x_t)$
3. Smoothing
   - compute marginal likelihood of sequence of state variables $P(x_t|Y)$

Examples of univariate HMMs in R

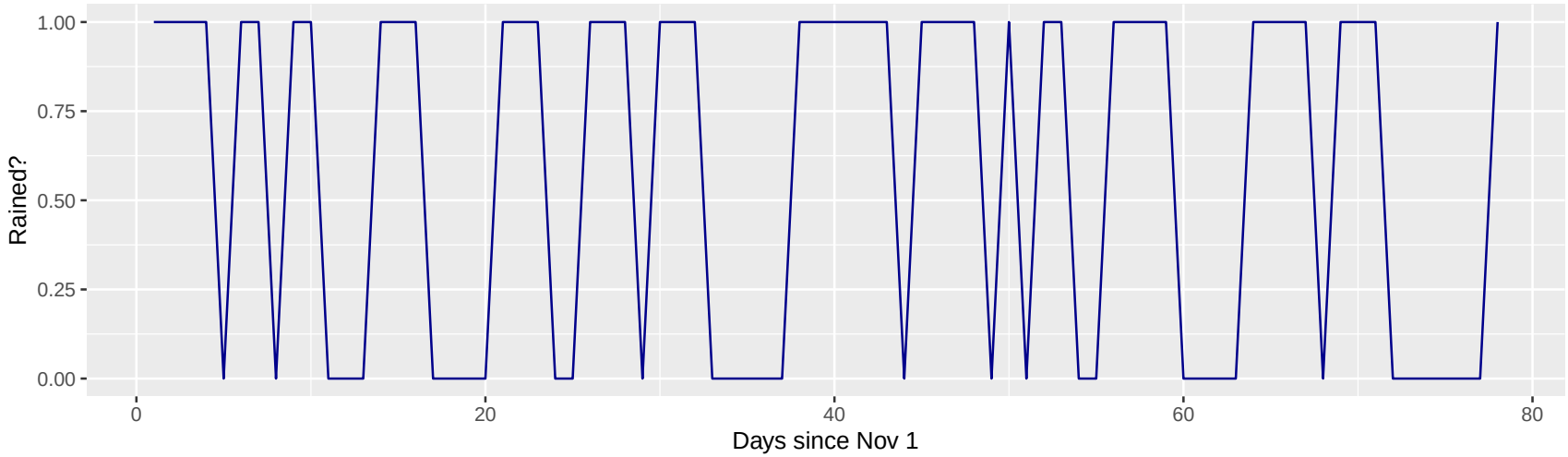
As a first example, let's use an example of rainfall data in Seattle from the end of 2018 and beginning of 2019 (accessed from wunderground.com)



Examples of univariate HMMs in R

We could model rainfall directly, but for starters let's just model whether or not it rained on a given day (0/1).

```
rain$rained = ifelse(rain$Rainfall > 0, 1, 0)
```



Examples of univariate HMMs in R

Questions that we might be interested in:

- Conditional probabilities
  - What's the probability of it raining tomorrow given that it's raining today?
  - What's the probability of it raining tomorrow given that it's not today?
- Persistence
  - On average, how long might we expect to go without rain?

Examples of univariate HMMs in R

We don't really need HMMs to address these questions

- daily rainfall may be measured with a tiny amount of uncertainty
- probably safe to say that there's almost no uncertainty in whether or not it rained on a given day

Examples of univariate HMMs in R

Transition probabilities can be just calculated directly,

- $P(rain_{t+1}|rain_t)$

$$\frac{\#consecutive \quad rainy \quad days}{\#rainy \quad days}$$

For example, we can create a differenced variable to indicate no change (0), and compute

```
rain$diff = c(diff(rain$rained), NA)
p_rain_rain = length(which(rain$diff == 0 &
    rain$rained==1)) / length(which(rain$rained==1))
```

Examples of univariate HMMs in R

Ok, now let's assume that there's some minor observation or measurement error in the rain gauge data. For this case, it's best to use a HMM.

Let's start with using the R package `depmixS4`. There's a good vignette on the package that you can refer back to,

https://cran.r-project.org/web/packages/depmixS4/vignettes/depmixS4.pdf

Examples of univariate HMMs in R

There's two functions we'll use to set up the HMM with `depmixS4`.

First we'll structure the model using the same formula notation you're hopefully familiar with,

```
mod = depmix(rained ~ 1,
             nstates = 2,
             transition = ~ 1,
             family = binomial(),
             data=rain)
```

Examples of univariate HMMs in R

Stepping through this `rained` is our response (yes / no)

```
mod = depmix(rained ~ 1,
             nstates = 2,
             transition = ~ 1,
             family = binomial(),
             data=rain)
```

Examples of univariate HMMs in R

`nstates` is the number of alternative states (> 2), specified a priori

```
mod = depmix(rained ~ 1,
             nstates = 2,
             transition = ~ 1,
             family = binomial(),
             data=rain)
```

Examples of univariate HMMs in R

`transition` is a formula to specify whether any covariates are to be included in the transition probabilities. The default is no covariates, and that these transitions aren't time varying.

```
mod = depmix(rained ~ 1,
             nstates = 2,
             transition = ~ 1,
             family = binomial(),
             data=rain)
```

Examples of univariate HMMs in R

`family` is family or list of families (a list for multiple response variables) of the families associated with each response. The majority of common families are supported

```
mod = depmix(rained ~ 1,
             nstates = 2,
             transition = ~ 1,
             family = binomial(),
             data=rain)
```

For a complete list, see

```
?depmixS4::GLMresponse
```

Examples of univariate HMMs in R

Ok, now that we've set up the model, we can do the estimation and look at the output

```
set.seed(123)
fitmod = fit(mod)
```

	To_1	To_2
From_1	0.77	0.23
From_2	0.41	0.59

Examples of univariate HMMs in R

As a warning, note that the results from the estimation are a bit sensitive to the initial seed. Look at how much the transition probabilties change,

```
set.seed(121)
fitmod = fit(mod)
```

	To_1	To_2
From_1	0.59	0.41
From_2	0.23	0.77

Examples of univariate HMMs in R

There's a couple practical ways to try to overcome this seed issue.

- First, we can change the control parameters

Unfortunately for this example, this doesn't solve the issue

Examples of univariate HMMs in R

- A second option is to run the estimation across a large number (> 100) of random starting values

Pseudocode:

```
best = 1.0e10
best_model = NA
for(i in 1:iter) {
  # fit the model
  fitmod = fit(mod)

  # check to see if this is the best solution?
  if(AIC(fitmod) < best) {
    best_model = fitmod
    best = AIC(fitmod)
  }
}
```

Examples of univariate HMMs in R

Let's move on to a more complex example.

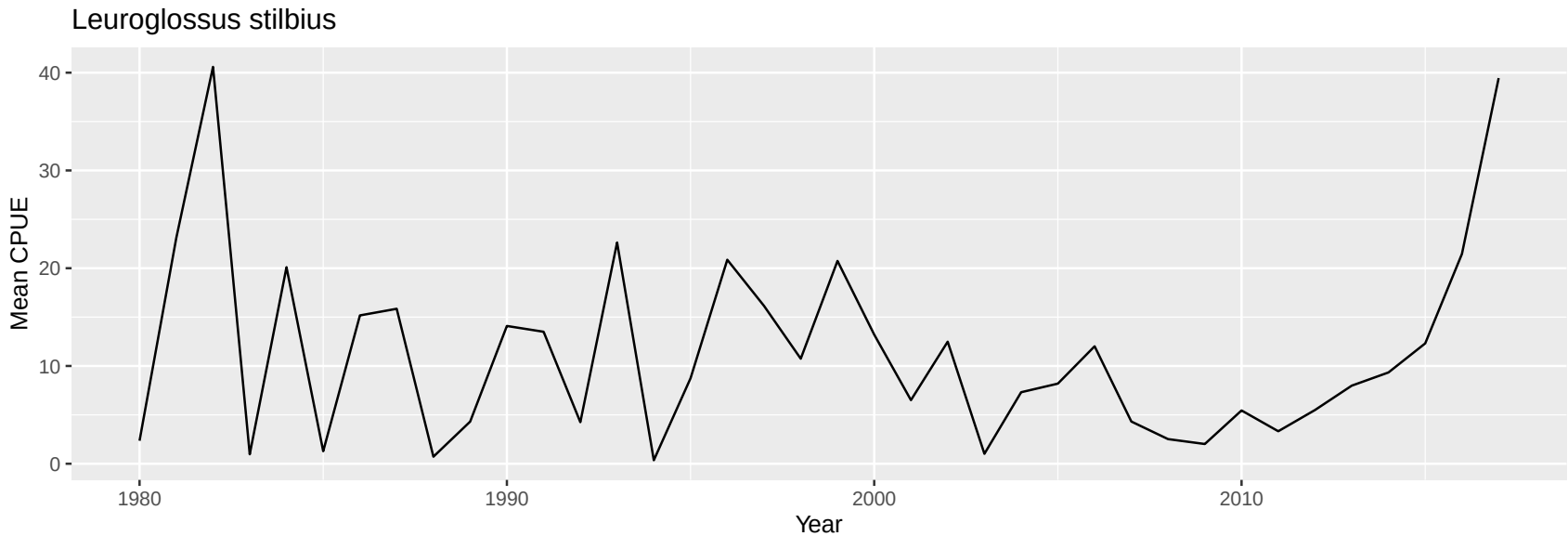
We'll pull some data from the CalCOFI ichthyoplankton cruises in Southern California. Some species have been used to indicate *cool* versus *warm* regimes.

- http://calcofi.org/publications/calcofireports/v58/Vol58-State_of_the_Current_pages_1-55.pdf

For this example, we'll focus on the California smoothtongue (*Leuroglossus stilbius*)

Examples of univariate HMMs in R

Caveat: the survey is spatially gridded, and you'd want to perform index standardization or use spatial models to generate indices. For simplicty, we're just taking the mean abundance across stations in April-May.



Examples of univariate HMMs in R

We'll start with fitting a 2-state model with `depmix`. Assumptions:

- Model fit to ln transformed data, assumed Gaussian

```
set.seed(123)
calcofi$ln = log(calcofi$m)
mod = depmix(ln ~ 1,
             nstates = 2,
             data=calcofi)
fitmod = fit(mod)
```

Examples of univariate HMMs in R

First let's look at how to get predictions out of a `depmix.fitted` object

We'll start with the state probabilities. Remember we could work with either

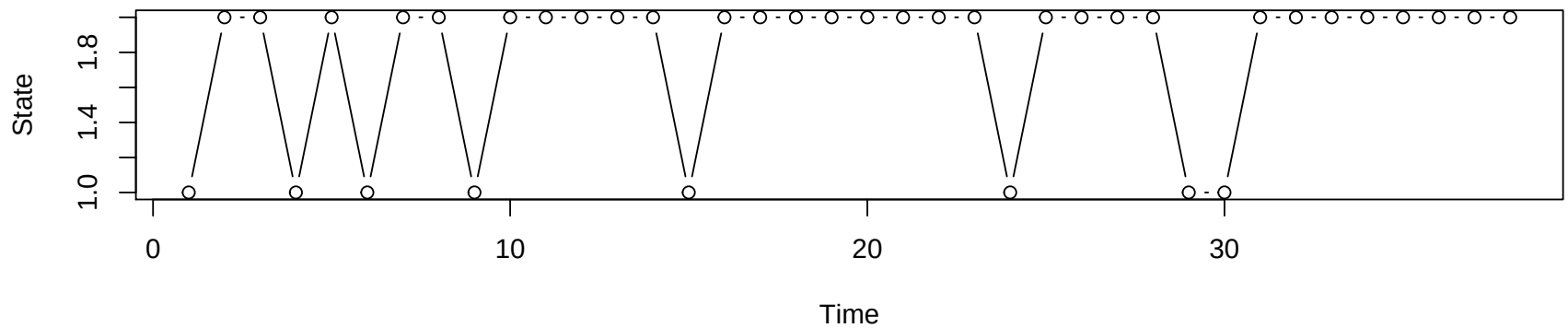
- Most probable state trajectory - via the `viterbi` function
- Marginals of $P(x_t)$ - via the `posterior` function

Examples of univariate HMMs in R

The most probable states are

```
prstates = apply(posterior(fitmod)[,c("S1","S2")],
  1, which.max)
```

```
plot(prstates, type="b", xlab="Time", ylab="State")
```

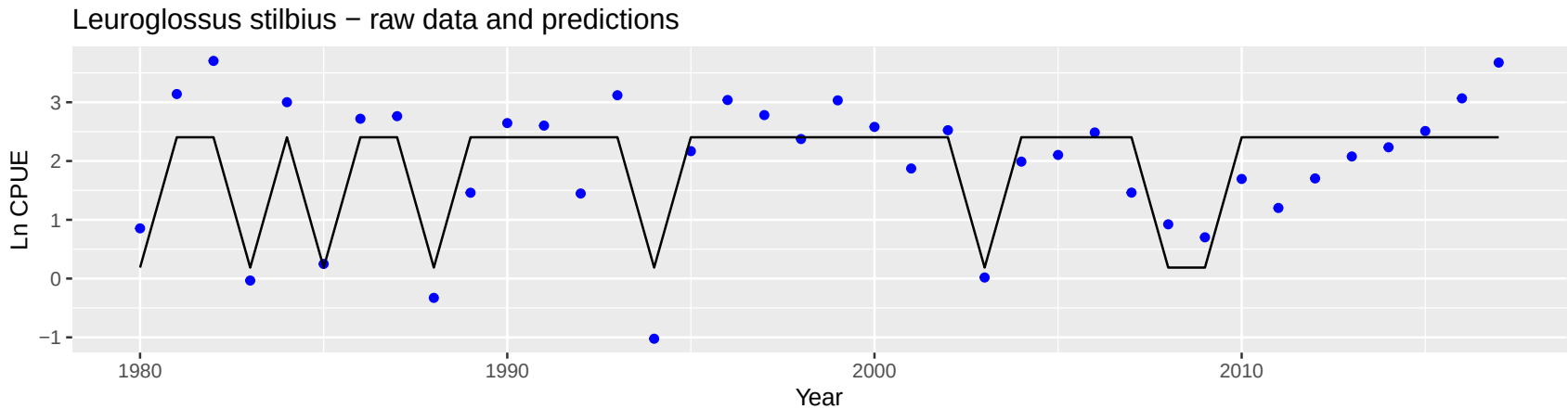


Examples of univariate HMMs in R

depmixS4 doesn't have a `predict()` or `forecast()` function, but creating estimated data is pretty straightforward. We can get the means out with

```
mu = summary(fitmod)[,1]
```

```
pred = data.frame("year"=seq(min(calcofi$year),
  max(calcofi$year)),
  "fit" = mu[prstates])
```



Examples of univariate HMMs in R

Some diagnostics (like autocorrelation) may not look great. What if we compared the previous model to one with 3 states? AIC from the 2-state model was

```
## [1] 118.994
```

```
set.seed(123)
calcofi$ln = log(calcofi$m)
mod = depmix(ln ~ 1,
             nstates = 3,
             data=calcofi)
fitmod = fit(mod)
```

This seems like increasing the states doesn't result in lower AIC

```
## [1] 122.5181
```

Model selection & diagnostics

- In comparing 2 vs 3 state HMM, we found lower AIC for 2-state model
- However, opposite is often true – more states fit better, and result in lower AIC
- How to decide? Occam's razor & metrics of predictive ability
- Pohle et al. 2017. "Selecting the Number of States in Hidden Markov Models: Pragmatic Solutions Illustrated Using Animal Movement"

Examples of multivariate HMMs in R

If the univariate examples make sense, it's not that different to extend these models for multiple time series.

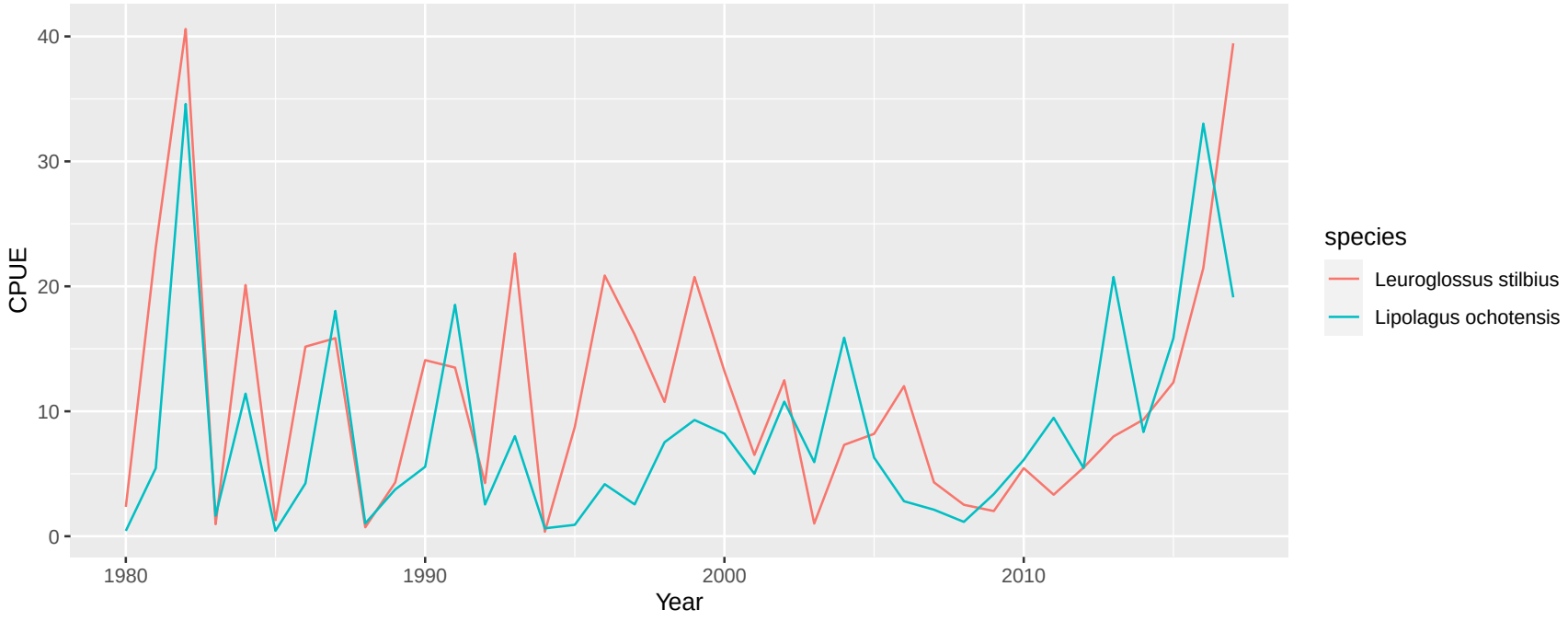
In this setting, the assumption is that the latent state process is the same for all time series, though the time series may differ

1. their lengths
2. their error distributions

Examples of multivariate HMMs in R

In `depmix`, these arguments generally become lists

We'll extend our CalCOFI example to include 2 species now



Examples of multivariate HMMs in R

Fitting a 2-state model with 2 species as responses. First we have to reshape the data,

```
calcofi$ln = log(calcofi$m) # ln transform
calcofi <- dcast(melt(calcofi[,c("year","ln","species")],
                 id.vars = c("year", "species")),
                 year ~ species)
names(calcofi)[2:3] = c("L_stilbius","L_ochotensis")
head(calcofi)
```

```
##   year  L_stilbius L_ochotensis
## 1 1980  0.85449030   -0.8414039
## 2 1981  3.13964169    1.6941728
## 3 1982  3.70359056    3.5435378
## 4 1983 -0.03344793    0.4947281
## 5 1984  3.00082695    2.4346651
## 6 1985  0.24911445   -0.8471716
```

Examples of multivariate HMMs in R

```
set.seed(123)
mod = depmix(list(L_stilbius ~ 1, L_ochotensis~1),
             nstates = 2,
             family = list(gaussian(),gaussian()),
             data=calcofi)
fitmod = fit(mod)
```

Examples of multivariate HMMs in R

We could also have situations where the time series are of different length.

- For example, if L_ochotensis time series was missing first half of values
- The argument `ntimes` is a list that becomes particularly valuable here - representing the length of each time series

Summary

HMMs are a useful approach at identifying latent state vectors that undergo discrete transitions

Estimation of HMMs for time series in R generally done with ML methods

- Fast, but these algorithms may get stuck
- Robust solutions = multiple starting values

Bayesian estimation generally beyond the scope of this class

- Very straightforward to build these models in BUGS/JAGS
- More complicated with newer sofware (Stan), though examples here: https://mc-stan.org/docs/2_26/stan-users-guide/hmms-section.html

JAGS Examples

- Why JAGS? Categorical samplers
- Let's re-fit our CalCOFI example in JAGS. We'll step through this in small chunks to understand what's going on in JAGS syntax

JAGS model

- First we need to specify the initial state. We don't know it, but assume each state (1/2) is equally likely

```
model{
for(i in 1:2){initp[i] <- 0.5}
x[1] ~ dcat(initp)
```

JAGS model

- Next we can think about parameterizing the transition matrix. Γ is 2 x 2, but only has 2 parameters
- We'll specify uniform(0,1) priors on these

```
model{
for(i in 1:2){initp[i] <- 0.5}
x[1] ~ dcat(initp)
p ~ dunif(0,1)
q ~ dunif(0,1)
```

JAGS model

- Next we specify Γ

```
model{
for(i in 1:2){initp[i] <- 0.5}
x[1] ~ dcat(initp)
gamma12 ~ dunif(0,1)
gamma21 ~ dunif(0,1)
Gamma[1,1] <- 1-gamma12
Gamma[1,2] <- gamma12
Gamma[2,1] <- gamma21
Gamma[2,2] <- 1-gamm21
```

JAGS model

- Next, add in the transition model for the latent state x

```
model{
for(i in 1:2){initp[i] <- 0.5}
x[1] ~ dcat(initp)
p ~ dunif(0,1)
q ~ dunif(0,1)
Gamma[1,1] <- 1-p
Gamma[1,2] <- p
Gamma[2,1] <- q
Gamma[2,2] <- 1-q
for(i in 2:n){x[i] ~ dcat(Gamma[x[i-1],])}
}
```

JAGS model

- And finally the observation model for our observed data
- We need to estimate 2 means (u) and a residual error variance term

```
model{
for(i in 1:2){initp[i] <- 0.5}
x[1] ~ dcat(initp)
p ~ dunif(0,1)
q ~ dunif(0,1)
Gamma[1,1] <- 1-p
Gamma[1,2] <- p
Gamma[2,1] <- q
Gamma[2,2] <- 1-q
for(i in 2:n){x[i] ~ dcat(Gamma[x[i-1],])}
resid_tau~dgamma(0.001,0.001)
for(i in 1:2) {u[i] ~ dnorm(0,1)}
for(i in 1:n){y[i] ~ dnorm(u[x[i]],resid_tau)}
}
```

JAGS model

- To run the model, save it in a file, e.g. 'model.txt'
- Create data list

```
n = nrow(calcofi)
y = log(calcofi$m)
jags_data = list("n","y")
```

- Create parameter list

```
jags_params = c("x","u","Gamma","resid_tau")
```

JAGS model

- Running the model

```
jagsfit <- jags(data=jags_data,
parameters.to.save=jags_params,
model.file="model.txt")
```