Intro to Univariate State-Space Models

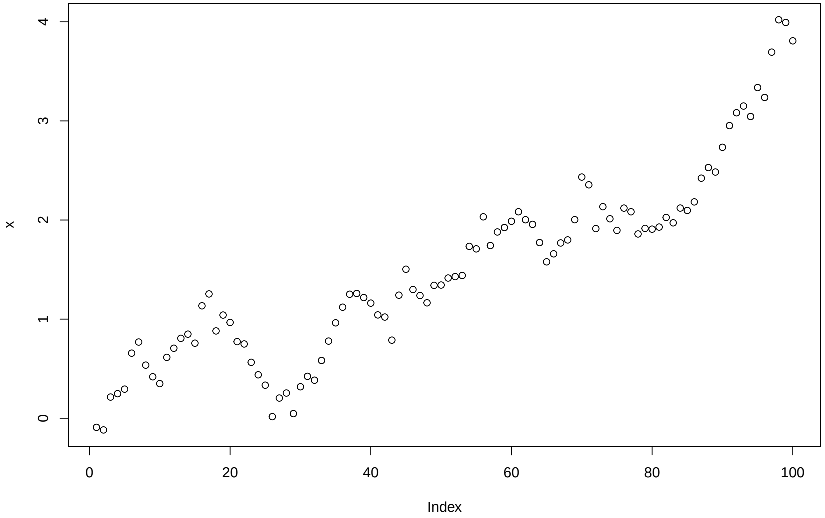
FISH 507 – Applied Time Series Analysis

Eli Holmes

19 Jan 2021

Dickey-Fuller test with ‘ur.df’

```
x <- cumsum(rnorm(100, 0.02, 0.2))
plot(x)
```



Dickey-Fuller stationarity test

$x_t = \phi x_{t-1} + \mu + at + e_t$

$x_t - x_{t-1} = \gamma x_{t-1} + \mu + at + e_t$

Test is for unit root is whether $\gamma = 0$.

- Standard linear regression test statistics won't work since the response variable is correlated with our explanatory variable.
- `ur.df()` reports the critical values we want in the summary info or `attr(test,"cval")`.

```r
library(urca)
test <- ur.df(x, type="trend", lags=0)
summary(test)
```

```
Value of test-statistic is: -3.1375 4.6773 4.9583

Critical values for test statistics:
      1pct  5pct 10pct
tau3 -4.04 -3.45 -3.15
phi2  6.50  4.88  4.16
phi3  8.73  6.49  5.47
```

```r
attr(test, "teststat")
```

```
##                 tau3     phi2     phi3
## statistic -1.936144 2.970519 2.110123
```

```r
attr(test,"cval")
```

```
##       1pct  5pct 10pct
## tau3 -4.04 -3.45 -3.15
## phi2  6.50  4.88  4.16
## phi3  8.73  6.49  5.47
```

The `tau3` is the one we want. This is the test that $\gamma = 0$ which would mean that $\phi = 0$ (random walk).

$x_t = \phi x_{t-1} + \mu + at + e_t$

$x_t - x_{t-1} = \gamma x_{t-1} + \mu + at + e_t$

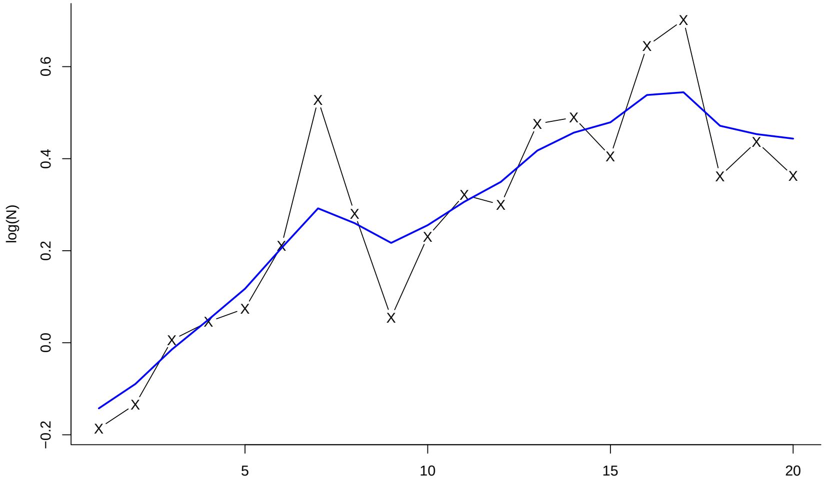
The hypotheses reported in the output are

- `tau` (or `tau2` or `tau3`): $\gamma = 0$
- `phi` reported values: are for the tests that $\gamma = 0$ and/or the other parameters a and μ are also 0.

Since we are focused on the random walk (non-stationary) test, we focus on the `tau` (or `tau2` or `tau3`) statistics and critical values

Univariate state-space models

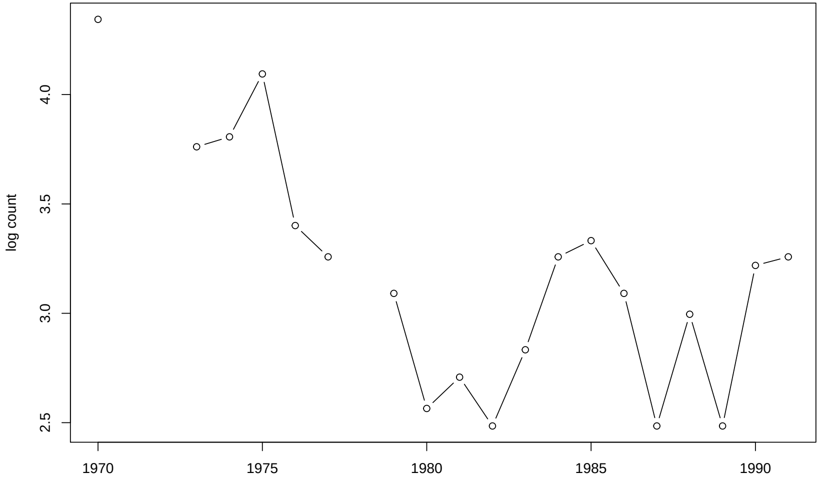
Autoregressive state-space models fit a random walk AR(1) through the data. The variability in the data contains both process and non-process (observation) variability.



PVA example

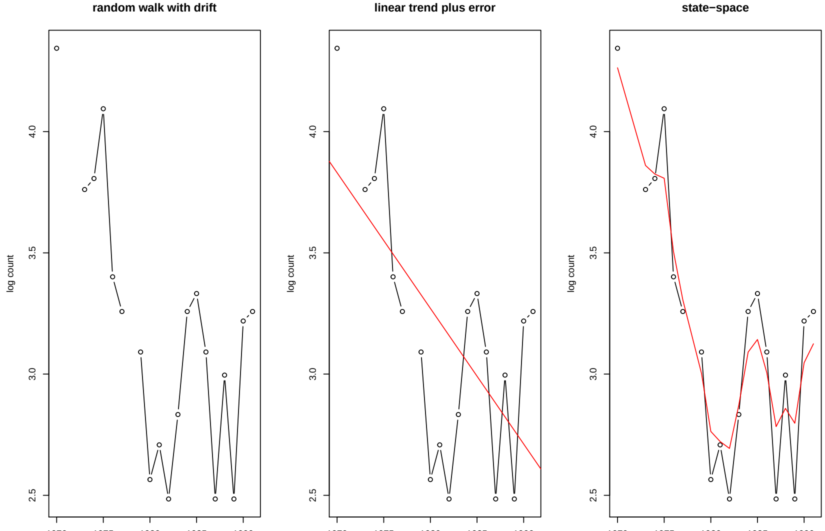
One use of univariate state-space models is "count-based" population viability analysis (chap 7 HWS2014)

Imagine you were tasked with estimating the probability of the population going extinct (N=1) within certain time frames (10, 20, years).

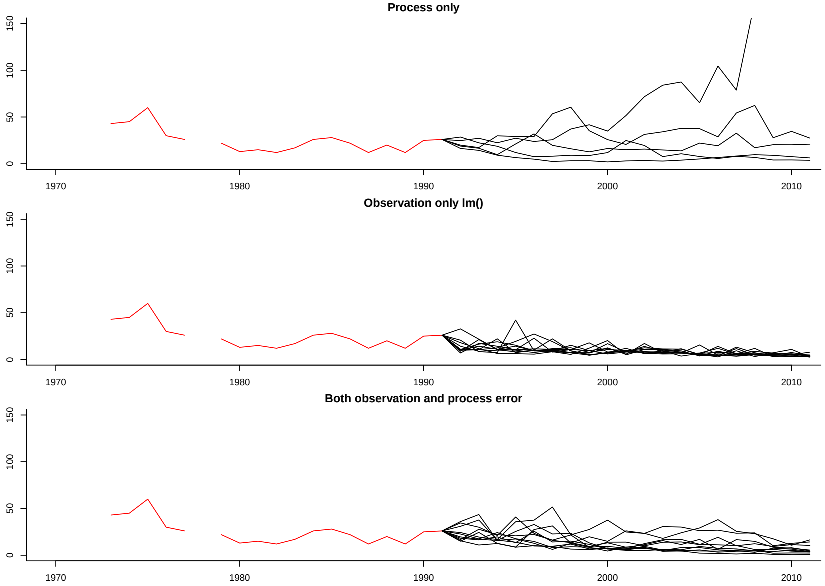


How might we approach our forecast?

- Fit a model
- Simulate with that model many times
- Count how often the simulation hit N=1 (logN=0)



How you model your data has a large impact on your forecasts



Stochastic level models

Flat level

$$x = u$$

$$y_t = x + v_t$$

Linear level

$$x_t = u + c \times t$$

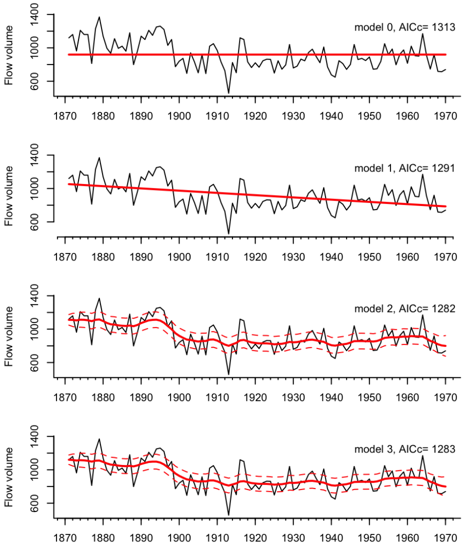
$$y_t = x_t + v_t$$

Stochastic level

$$x_t = x_{t-1} + u + w_t$$

$$y_t = x_t + v_t$$

Nile River example



Kalman filter and smoother

The **Kalman filter** is an algorithm for computing the expected value of the x_t conditioned on the data up to $t-1$ and t and the model parameters.

$$x_t = bx_{t-1} + u + w_t, \;\; w_t \sim N(0, q)$$

$$y_t = zx_t + a + v_t, \;\; v_t \sim N(0, r)$$

The **Kalman smoother** computes the expected value of the x_t conditioned on all the data.

Diagnostics

Innovations residuals $=$

data at time t minus model predictions given data up to $t-1$

$$\hat{y}_t = E[Y_t | y_{t-1}]$$

`residuals(fit)`

Standard diagnostics

- ACF
- Normality

MARSS package

We will be using the MARSS package to fit univariate and multivariate state-space models.

$$\mathbf{x}_t = \mathbf{B}x_{t-1} + \mathbf{U} + \mathbf{w}_t, \;\; \mathbf{w}_t \sim MVN(0, \mathbf{Q})$$
$$\mathbf{y}_t = \mathbf{Z}\mathbf{x}_t + \mathbf{A} + \mathbf{v}_t, \;\; \mathbf{v}_t \sim MVN(0, \mathbf{R})$$

The main function is `MARSS()`:

```
fit <- MARSS(data, model=list())
```

data are a univariate vector, univariate ts or a matrix with time going along the columns.

model list is a list with the structure of all the parameters.

Univariate example

$$x_t = x_{t-1} + u + w_t, \;\; w_t \sim N(0, q)$$
$$y_t = x_t + v_t, \;\; v_t \sim N(0, r)$$

Write where everything bold is a matrix.

$$x_t = \mathbf{B} x_{t-1} + \mathbf{U} + w_t, \;\; w_t \sim MVN(0, \mathbf{Q})$$
$$y_t = \mathbf{Z} x_t + \mathbf{A} + v_t, \;\; v_t \sim MVN(0, \mathbf{R})$$

```
mod.list <- list(
  B = matrix(1), U = matrix("u"), Q = matrix("q"),
  Z = matrix(1), A = matrix(0), R = matrix("r"),
  x0 = matrix("x0"),
  tinitx = 0
)
```

Let's see some examples