Day 2 exercise

James Thorson

2025-05-20

Arrow-and-lag interface for simultaneous and lagged effects

Agenda section E: Exercise Goal:

1. Understand arrow-and-lag notation
2. Fit simultaneous and lagged effects in a time-series model

Structure:

1. Students are to self-organize into groups of 2-3 people, and work in a group on step-1 for 20 minutes.
2. For Step-0, we will walk through code as a group;
3. For later steps, the instructor will then share code for that step on github, and groups will continue with step-2 for 20 minutes, then repeating for step-3.

Step 1: Try fitting a linear model (20 minutes)

Try simulating data from a linear model:

$$x_i \sim \text{Normal}(2, \sigma_X^2)$$
$$y_i \sim \text{Normal}(1 + 0.5x_i, \sigma_Y^2)$$

Where:

- $i \in 1, 2, \ldots, 100$, x and y are vectors with length 100
- $\sigma_X^2 = 1$ is the variance in the predictor variable x and $\sigma_Y^2 = 1$ is the residual variance in the response variable y
- $E(x_i) = 2$ and $E(y_i) = 1 + 0.5x_i$
- $x_i \sim \text{Normal}(2, \sigma_X^2)$ indicates that x_i is simulated from a normal distribution, which can be done in R using `rnorm( n = 100, mean = 2, sd = 1)`

```
# Simulate x and y
x = rnorm( n = 100, mean = 2, sd = 1 )
y = rnorm( n = 100, mean = 1 + 0.5*x, sd = 1 )
```

Then try fitting in `dsem`:

```r
#install.packages("dsem")
library(dsem)
```

```
## Warning: package 'dsem' was built under R version 4.4.3
```

```r
# Define arrow-and-lag notation
time_term = "
# predictor -> response, lag, parameter_name, start_value
x -> y, 0, slope, 1
x <-> x, 0, sd_x, 1
y <-> y, 0, sd_y, 1
"

# format data into ts object
tsdata = ts(data.frame(x=x, y=y))

# fit in dsem
dsem_fit = dsem(
  tsdata = tsdata,
  sem = time_term,
  control = dsem_control(quiet = TRUE)
)

# Summarize dsem output
summary(dsem_fit)
```

```
##      path lag  name start parameter first second direction  Estimate  Std_Error
## 1   x -> y   0 slope     1         1     x      y          1 0.3960125 0.09323020
## 2 x <-> x   0  sd_x     1         2     x      x          2 0.9587272 0.06813361
## 3 y <-> y   0  sd_y     1         3     y      y          2 0.8893430 0.06320267
##     z_value      p_value
## 1  4.247685 2.159906e-05
## 2 14.071282 5.702638e-45
## 3 14.071287 5.702188e-45
```

You can then compare this with a linear model

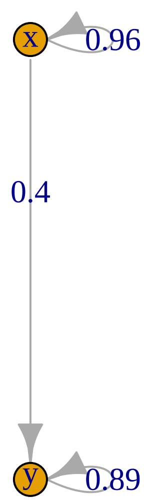
```r
# linear model
lm_fit = lm( y ~ 1 + x )
summary(lm_fit)
```

```
##
## Call:
## lm(formula = y ~ 1 + x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.51668 -0.45656 -0.07594  0.56075  2.03060
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.3023     0.2017   6.457 4.11e-09 ***
```

```
## x              0.3960     0.0937   4.226 5.34e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8939 on 98 degrees of freedom
## Multiple R-squared:  0.1542, Adjusted R-squared:  0.1455
## F-statistic: 17.86 on 1 and 98 DF,  p-value: 5.336e-05
```

You can also plot output to see the graphical model

```r
# plot dsem
plot( dsem_fit,
      edge_label = "value",
      style = "igraph" )
```



Step 2: Simulate autoregressive data (20 minutes)

Next, please simulate data from an first-order autoregressive process:

$$x_t \sim \begin{cases} \text{Normal}(0, \sigma^2) & \text{if } t = 1 \\ \text{Normal}(\rho x_{t-1}, \sigma^2) & \text{if } t > 2 \end{cases}$$

using $\rho = 0.8$ and $\sigma = 1$. Try simulating x_t for $t \in \{1, 2, ..., 40\}$ times. Then fit this in `dsem` using

```
time_term = "
x <-> x, 0, sigma
x -> x, 1, rho
"
```

How well can you estimate the parameters ρ and σ?

Step 3: Simulate age-structured data (20 minutes)

Please write an R script that simulates abundance at age $n_{a,t}$ for ages $a \in 1, 2, \ldots, 10$ and years $t \in 1, 2, \ldots, 40$ with the following simplified dynamics:

$$n_{a,t} = \begin{cases} \mu_R \times e^{\sigma_1 \epsilon_{a,t} - ma} & \text{if } a = 1 \text{ } or \text{ } t = 1 \\ n_{a-1,t-1} \times e^{\sigma_2 \epsilon_{a,t} - m} & \text{otherwise} \end{cases}$$

Where process errors $\epsilon_{a,t} \sim \text{Normal}(0, \sigma^2)$, median age-0 recruitment $\mu_R = 10^9$ [individuals], recruitment variation $\sigma_1 = 0.6$ [dimensionless], demographic variation $\sigma_2 = 0.1$ [dimensionless], and natural mortality $m = 0.4$ [per year].

Now imagine that we can only sample abundance-at-age from a sampling gear with logistic selectivity:

$$s_a = \frac{1}{1 + e^{-\theta_2(a - \theta_1)}}$$

Where age at 50% selection $\theta_1 = 3$ [years] and the logit-slope $\theta_2 = 1$ [year^(-1)]. We then observe:

$$n^*_{a,t} = n_{a,t} s_a$$

Step 2: Fit model using dsem (20 minutes)

Write out the arrow-and-lag notation whereby $n_{a,t}$ is caused by abundance $n_{a-1,t-1}$ the preceding age and year. This will presumably involve several one-headed arrows:

```
X -> Y, 1, parameter
```

Where `X` is the predictor, `Y` the response, `1` represents a time-lag, and `parameter` is a parameter name (you can use the same name multiple times to force a single parameter value to be estimated and shared across lines)

Next, modify this arrow-and-lag notation to represent the assumption that exogenous variation at age1 (representing recruitment deviations) will be different than subsequent ages (representing demographic variation). This will presumably involve several two-headed arrows:

```
X <-> X, 0, parameter
```

where these always have a lag of zero.

Step 4: Retrospective skill testing (20 minutes)

Finally, conduct a "leave-future-out" crossvalidation by dropping the final five years of simulated data. To do so, modify `tsdata` by replacing values (measurements) with NAs (representing missing data). This will ensure that the model still predicts those missing values.