Session 6: Matrices and Model Simulation

Foundations of Quantitative Ecology (EEOB 8896.11)

Paul J. Hurtado
(hurtado.10@mbi.osu.edu)

Mathematical Biosciences Institute (MBI)
The Ohio State University

Last compile: September 25, 2013

Matrix basics

Three common ways in which we use matrices:

1. Notation for multivariate linear relationships
2. Data storage and efficient computation
3. Geometric intuition

Matrix basics

Notation for multivariate linear relationships

$$
\begin{aligned}
y_1 =& A_{11}x_1 + A_{12}x_2 + \ldots + A_{1n}x_n \\
y_2 =& A_{21}x_1 + A_{22}x_2 + \ldots + A_{2n}x_n \\
& \ldots \\
y_n =& A_{n1}x_1 + A_{n2}x_2 + \ldots + A_{nn}x_n
\end{aligned}
$$

Matrix basics

Notation for multivariate linear relationships

$$\begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{bmatrix} = \begin{bmatrix} A_{11}x_1 + A_{12}x_2 + \dots + A_{1n}x_n \\ A_{21}x_1 + A_{22}x_2 + \dots + A_{2n}x_n \\ \dots \\ A_{n1}x_1 + A_{n2}x_2 + \dots + A_{nn}x_n \end{bmatrix}$$

Matrix basics

Notation for multivariate linear relationships

$$\begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{bmatrix} = \begin{bmatrix} A_{11} \; A_{12} \; \dots \; A_{1n} \\ A_{21} \; A_{22} \; \dots \; A_{2n} \\ \dots \\ A_{n1} \; A_{n2} \; \dots \; A_{nn} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix}$$

Matrix basics

Notation for multivariate linear relationships

$$\vec{y} = A\vec{x}$$

Matrix basics

Matrix multiplication:

$$A \cdot B = \begin{bmatrix} A_{11} & A_{12} & \ldots & A_{1n} \\ A_{21} & A_{22} & \ldots & A_{2n} \\ & \ldots & & \\ A_{n1} & A_{n2} & \ldots & A_{nn} \end{bmatrix} \cdot \begin{bmatrix} B_{11} & B_{12} & \ldots & B_{1n} \\ B_{21} & B_{22} & \ldots & B_{2n} \\ & \ldots & & \\ B_{n1} & B_{n2} & \ldots & B_{nn} \end{bmatrix}$$

$$= \begin{bmatrix} A_{11} & A_{12} & \ldots & A_{1n} \\ A_{21} & A_{22} & \ldots & A_{2n} \\ & \ldots & & \\ A_{n1} & A_{n2} & \ldots & A_{nn} \end{bmatrix} \cdot \left[\begin{array}{c|c|c|c} & & \ldots & \\ \vec{B}_1 & \vec{B}_2 & \ldots & \vec{B}_n \\ & & \ldots & \\ & & \ldots & \end{array} \right]$$

Matrix basics

Matrix multiplication:

$$A \cdot B = \begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ & & \dots & \\ A_{n1} & A_{n2} & \dots & A_{nn} \end{bmatrix} \cdot \left[\begin{array}{c|c|c|c} & & \dots & \\ \vec{B}_1 & \vec{B}_2 & \dots & \vec{B}_n \\ & & \dots & \\ & & \dots & \end{array} \right]$$

$$= \left[\begin{array}{c|c|c|c} & & \dots & \\ A \cdot \vec{B}_1 & A \cdot \vec{B}_2 & \dots & A \cdot \vec{B}_n \\ & & \dots & \\ & & \dots & \end{array} \right]$$

Regression

Linear model:

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i \quad \text{where } \epsilon_i \sim N(0, \sigma^2)$$

Regression

Linear model:

$$
\begin{aligned}
Y_1 &= \beta_0 + \beta_1 X_1 + \epsilon_1 \\
Y_2 &= \beta_0 + \beta_1 X_2 + \epsilon_2 \\
&\dots \\
Y_n &= \beta_0 + \beta_1 X_n + \epsilon_n
\end{aligned}
$$

Regression

Linear model:

$$\begin{bmatrix} Y_1 \\ Y_2 \\ \dots \\ Y_n \end{bmatrix} = \begin{bmatrix} \beta_0 + \beta_1 X_1 \\ \beta_0 + \beta_1 X_2 \\ \dots \\ \beta_0 + \beta_1 X_n \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \dots \\ \epsilon_n \end{bmatrix}$$

Regression

Linear model:

$$\begin{bmatrix} Y_1 \\ Y_2 \\ \dots \\ Y_n \end{bmatrix} = \begin{bmatrix} 1 & X_1 \\ 1 & X_2 \\ & \dots \\ 1 & X_n \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \dots \\ \epsilon_n \end{bmatrix}$$

Regression

In matrix form,

$$Y = X\beta + \epsilon$$

Goal:

$$\text{Minimize } \epsilon'\epsilon = (Y - X\beta)'(Y - X\beta)$$

This is the same as solving $X'Y = (X'X)\beta$.

Matrix basics: Geometric intuition

Consider a linear system of differential equations:

$$\begin{bmatrix} dx_1/dt \\ dx_2/dt \\ \dots \\ dx_n/dt \end{bmatrix} = \begin{bmatrix} A_{11}x_1 + A_{12}x_2 + \dots + A_{1n}x_n \\ A_{21}x_1 + A_{22}x_2 + \dots + A_{2n}x_n \\ \dots \\ A_{n1}x_1 + A_{n2}x_2 + \dots + A_{nn}x_n \end{bmatrix}$$

$$D_t\vec{x} = A\vec{x} \quad or \quad \dot{x} = Ax$$

Q: How do the entries in A affect the long-term values of x?

Matrix basics

Consider the 1-dimensional case: $\dot{x} = ax$ which has the solution $x(t) = x(0)exp(at)$. If $a > 0$ then x grows exponentially. If $a < 0$, it decays exponentially towards $x = 0$.

In the n-dimensional case, something similar happens, but to see it we need to understand the *eigenstructure* of matrix A.

Matrix basics: Eigenstuff

Pick a random* matrix A. It can (usually) be written:

$$A = QDQ^{-1}$$

where $D = diag(\lambda_1, ..., \lambda_n)$ are **eigenvalues**, and the columns of Q are their corresponding **eigenvectors**.

Matrix basics: Eigenstuff

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dots \\ \dot{x}_n \end{bmatrix} = \overbrace{\begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ & \dots & & \\ A_{n1} & A_{n2} & \dots & A_{nn} \end{bmatrix}}^{A=QDQ^{-1}} \cdot \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix}$$

Matrix basics: Eigenstuff

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dots \\ \dot{x}_n \end{bmatrix} = Q \overbrace{\begin{bmatrix} \lambda_1 \; 0 \; \dots \; 0 \\ 0 \; \lambda_1 \; \dots \; 0 \\ \dots \\ 0 \; 0 \; \dots \; \lambda_1 \end{bmatrix}}^{D} Q^{-1} \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix}$$

Matrix basics: Eigenstuff

$$Q^{-1}\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dots\\ \dot{x}_n\end{bmatrix} = Q^{-1}Q\overbrace{\begin{bmatrix}\lambda_1 \; 0 \;\dots\; 0\\ 0 \;\lambda_1 \;\dots\; 0\\ \dots\\ 0 \; 0 \;\dots\; \lambda_1\end{bmatrix}}^{D} Q^{-1}\begin{bmatrix}x_1\\ x_2\\ \dots\\ x_n\end{bmatrix}$$

Matrix basics: Eigenstuff

$$\overbrace{Q^{-1}\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dots\\ \dot{x}_n\end{bmatrix}}^{\dot{y}} = Q^{-1}Q\overbrace{\begin{bmatrix}\lambda_1\ 0\ \dots\ 0\\ 0\ \lambda_1\ \dots\ 0\\ \dots\\ 0\ 0\ \dots\ \lambda_1\end{bmatrix}}^{D}\overbrace{Q^{-1}\begin{bmatrix}x_1\\ x_2\\ \dots\\ x_n\end{bmatrix}}^{y}$$

Matrix basics: Eigenstuff

$$\begin{bmatrix} \dot{y}_1 \\ \dot{y}_2 \\ \ldots \\ \dot{y}_n \end{bmatrix} = \begin{bmatrix} \lambda_1 \; 0 \; \ldots \; 0 \\ 0 \; \lambda_1 \; \ldots \; 0 \\ \ldots \\ 0 \; 0 \; \ldots \; \lambda_1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \ldots \\ y_n \end{bmatrix}$$

Matrix basics: Eigenstuff

$$\begin{aligned} \dot{y}_1 &= \lambda_1 y_1 \\ \dot{y}_2 &= \lambda_1 y_2 \\ &\dots \\ \dot{y}_n &= \lambda_1 y_n \end{aligned} \implies \begin{aligned} y_1(t) &= y_1(0) exp(\lambda_1 t) \\ y_2(t) &= y_2(0) exp(\lambda_2 t) \\ &\dots \\ y_n(t) &= y_n(0) exp(\lambda_n t) \end{aligned}$$

Recalling that $y = Q^{-1}x$, then $x = Ay$, which yields...

Matrix basics: Eigenstuff

$$\begin{bmatrix} x_1(t) \\ x_2(t) \\ \dots \\ x_n(t) \end{bmatrix} = Q \begin{bmatrix} y_1(0)exp(\lambda_1 t) \\ y_2(0)exp(\lambda_2 t) \\ \dots \\ y_n(0)exp(\lambda_n t) \end{bmatrix}$$

Matrix basics: Eigenstuff

$$\begin{bmatrix} x_1(t) \\ x_2(t) \\ \dots \\ x_n(t) \end{bmatrix} = \begin{bmatrix} & \Big| & \Big| & \dots & \Big| & \\ \vec{Q}_1 & \Big| \vec{Q}_2 & \Big| & \dots & \Big| \vec{Q}_n & \\ & \Big| & \Big| & \dots & \Big| & \\ & \Big| & \Big| & \dots & \Big| & \end{bmatrix} \begin{bmatrix} y_1(0) exp(\lambda_1 t) \\ y_2(0) exp(\lambda_2 t) \\ \dots \\ y_n(0) exp(\lambda_n t) \end{bmatrix}$$

Matrix basics: Eigenstuff

Final solution:

$$\begin{bmatrix} x_1(t) \\ x_2(t) \\ \dots \\ x_n(t) \end{bmatrix} = y_1(0)e^{\lambda_1 t}\vec{Q}_1 + y_2(0)e^{\lambda_2 t}\vec{Q}_2 + \dots + y_n(0)e^{\lambda_2 t}\vec{Q}_n$$

Matrix basics: Eigenstuff

Final solution:

$$\begin{bmatrix} x_1(t) \\ x_2(t) \\ \dots \\ x_n(t) \end{bmatrix} = y_1(0)e^{\lambda_1 t}\vec{Q}_1 + y_2(0)e^{\lambda_2 t}\vec{Q}_2 + \dots + y_n(0)e^{\lambda_2 t}\vec{Q}_n$$

Q: How do the entries in A affect the long-term values of x?

Matrix basics: Eigenstuff

Final solution:

$$\begin{bmatrix} x_1(t) \\ x_2(t) \\ \dots \\ x_n(t) \end{bmatrix} = y_1(0)e^{\lambda_1 t}\vec{Q}_1 + y_2(0)e^{\lambda_2 t}\vec{Q}_2 + \dots + y_n(0)e^{\lambda_2 t}\vec{Q}_n$$

Q: How do the entries in A affect the long-term values of x?

A: Exponential growth & decay along different "*eigendirections*" according to the sign of the corresponding eigenvalues. If all are negative, decay towards 0.

Matrix basics: Eigenstuff

Had we considered a difference equation $\vec{x}_{t+1} = A\vec{x}_t$, then

$$\begin{bmatrix} x_1(t) \\ x_2(t) \\ \ldots \\ x_n(t) \end{bmatrix} = y_1(0)\lambda_1^t\vec{Q}_1 + y_2(0)\lambda_2^t\vec{Q}_2 + \ldots + y_n(0)\lambda_2^t\vec{Q}_n$$

Matrix basics: Eigenstuff

Had we considered a difference equation $\vec{x}_{t+1} = A\vec{x}_t$, then

$$\begin{bmatrix} x_1(t) \\ x_2(t) \\ \dots \\ x_n(t) \end{bmatrix} = y_1(0)\lambda_1^t\vec{Q}_1 + y_2(0)\lambda_2^t\vec{Q}_2 + \dots + y_n(0)\lambda_2^t\vec{Q}_n$$

Q: How do the entries in A affect the long-term values of x?

Matrix basics: Eigenstuff

Had we considered a difference equation $\vec{x}_{t+1} = A\vec{x}_t$, then

$$\begin{bmatrix} x_1(t) \\ x_2(t) \\ \dots \\ x_n(t) \end{bmatrix} = y_1(0)\lambda_1^t \vec{Q}_1 + y_2(0)\lambda_2^t \vec{Q}_2 + \dots + y_n(0)\lambda_2^t \vec{Q}_n$$

Q: How do the entries in A affect the long-term values of x?

A: Growth (decay) along different "*eigendirections*" according to whether the corresponding eigenvalues are bigger (less) than 1. If all $\lambda_i < 1$, decay towards 0.

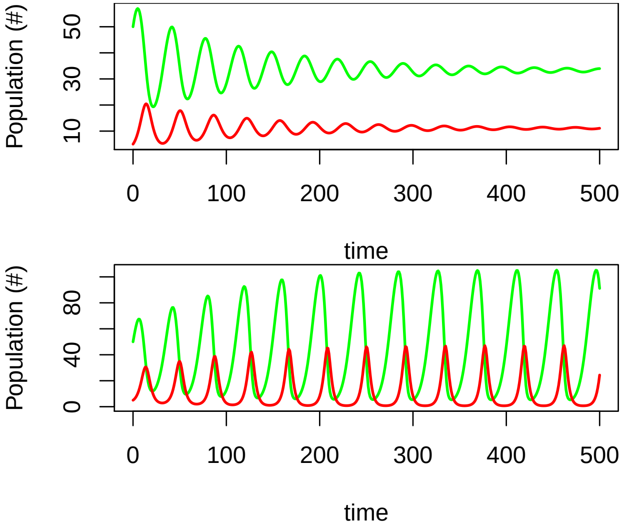
Part II: Simulation

Simulating ODEs

Simulate ODEs using the deSolve package:

```
library(deSolve)  #install.packages('deSolve') if needed
dydt <- function(ts, y, params) {
    r = params[1]
    K = params[2]
    a = params[3]
    Th = params[4]
    conv = params[5]
    mu = params[6]
    dy1dt = r * y[1] * (1 - y[1]/K) - a * y[1] * y[2]/(1 + a * Th * y[1])
    dy2dt = conv * a * y[1] * y[2]/(1 + a * Th * y[1]) - mu * y[2]
    return(list(c(dy1dt, dy2dt)))
}
y0 = c(50, 5)
out1 = ode(y0, 0:500, dydt, c(0.2, 100, 0.02, 1, 1, 0.4), method = "lsoda")
```

Simulating ODEs



Simulating ODEs

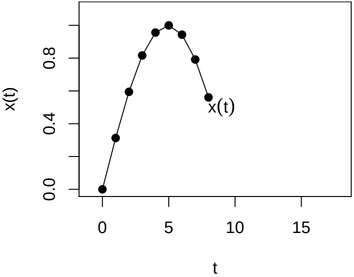
How does `ode()` work? A differential equation (e.g. $\frac{dx}{dt} = f(x)$) models *rates of change in* x. Since the derivative is approximately

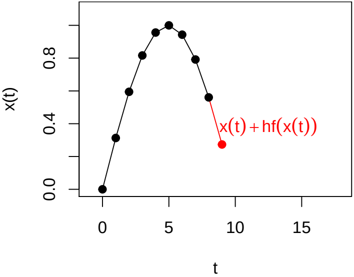
$$\frac{x(t+h) - x(t)}{h} \approx f(x(t))$$

we can rearrange things to approximate a step forward in time by

$$x(t+h) \approx x(t) + f(x(t)) \cdot h$$

That is, we add a change given by rate $f(x)$ times time step, h. The function `ode()` does something similar.





Simulating ODEs

Exercise 1:
Implement the Stochastic Simulation Algorithm (SSA; aka Gillespie Algorighm) for the above model to incorporate demographic stochasticity.

1. Compute a rate for some event (birth, death, predation)
2. Draw a random exponential time step to the next event
3. Use a random uniform to decide which event happened
4. Update the state variables, then repeat.

Exercise 2:
Modify the predator-prey model code (see previous slides) to include a third (top) predator.

Simulating ODEs

Hints for Exercise 1:

```
### Pseudo-code to simulate
##    Use the parameter values from the ODE script
#
## For iterating...
times = c(0) # initialize
y     = y0   # initial conditions from above
i     = 0    # initialize indexing/count variable
#
## Until current time surpasses 500...
  # recompute rates, total event rate, event probs
  # Pick a random time step and update times
  # Pick which event happened and update y values
  # Repeat
#
### Plot and compare with ODE output from above
```

Simulating ODEs

More hints for Exercise 1:

```
event_rates = c(
  r*y[1],            # Increment y[1] only (birth)
  r*y[1]^2/K,        # Decrement y[1] only (natural death)
  a*y[1]*y[2]/(1+a*Th*y[1]), # Decrement y[1] (predation)
  conv*a*y[1]*y[2]/(1+a*Th*y[1]), # Increment y[2] (birth)
  mu*y[2] ## Decrement y[2] (predator death)
)
# Total event rate
S = sum(event_rates)  # rate of "something" happening
# Proportions for picking event type
event_probs = event_rates/S
# Check cases with if-else statments, e.g.
if(event1) { y[,i+1] <- update_accordingly(y[,i]) }
else if(event2) { ... }
...
else if(lastevent) {...}
# OR use something like switch(). Ex:
switch(sample(1:5),"one","two","three","four","five")
switch(which(runif(1) < cumsum(event_probs))[1],
       c(1,0), c(-1,0), c(-1,0), c(0,1),c(0,-1))
```