Multivariate non-linear continuous-time deterministic models

Basic model: $\frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x})$, where $\boldsymbol{f}$ is a vector-valued function of a vector-valued state variable, $\boldsymbol{x}$. OK, at this point, you should probably be able to guess what's about to happen. Here's a toy example that we use below,

$$\begin{aligned} \frac{dN}{dt} =& aN(1-N)(M-0.5) \\ \frac{dM}{dt} =& bM(1-M)(N-0.5) \end{aligned}$$

Fixed points

$\boldsymbol{x}\star$ is a fixed point if $\boldsymbol{f}(\boldsymbol{x}\star) = \boldsymbol{0}$. These can be hard to find. But they are easy for the toy model, which we can state in two conditions: (1) $N\star = M\star$ and (2) $N\star \in \{0, 0.5, 1\}$.

Stability

A fixed point $\boldsymbol{x}\star$ is stable if the real part of the eigenvalues of the Jaccobian at $\boldsymbol{x}\star$ are all less than zero. Unlike the linear case, it is difficult to say much in general about eigenvalues with real part exactly equal to zero. The Jaccobian for the 2×2 case is,

$$\begin{pmatrix} \frac{\partial f_1(x_1,x_2)}{\partial x_1} & \frac{\partial f_1(x_1,x_2)}{\partial x_2} \\ \frac{\partial f_2(x_1,x_2)}{\partial x_1} & \frac{\partial f_2(x_1,x_2)}{\partial x_2} \end{pmatrix}$$

which for the example is,

$$\begin{pmatrix} a(1-2N)(M-0.5) & aN(1-N) \\ bM(1-M) & b(1-2M)(N-0.5) \end{pmatrix}$$

Let's look at each fixed point. For both $N = M = 0$ and $N = M = 1$,

$$\begin{pmatrix} -a/2 & 0 \\ & -b/2 \end{pmatrix}$$

which gives eigenvalues $-a/2$ and $-b/2$. Therefore, the origin is stable if both a and b are positive. Now for $N = M = 0.5$,

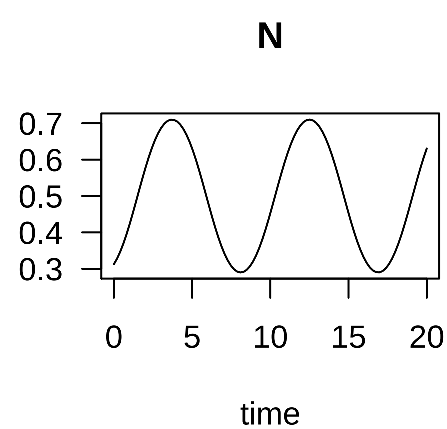
$$\begin{pmatrix} 0 & a/4 \\ b/4 & 0 \end{pmatrix}$$

which gives eigenvalues $\pm 0.25\sqrt{ab}$. If a and b have the same signs, then at least one of these eigenvalues has positive real part, and so the fixed point is not stable. However, if a and b have different signs, both of these eigenvalues have real parts of zero, and so we need to explore this fixed point using simulations.

Time dependent solution

Use `deSolve` in `R`. For the above model,

```
library(deSolve)
# start at difficult FP + perturbation
set.seed(1)
state <- c(N = 0.5, M = 0.5) +
    runif(2,-0.4,0.4)
pars <- c(a = -3, b = 3)
times <- seq(0,20,length=100)
func <- function(Time, State, Pars){
    with(as.list(c(State, Pars)), {
        dN <- a*N*(1-N)*(M-0.5)
        dM <- b*M*(1-M)*(N-0.5)
        return(list(c(dN, dM)))
    })
}
out <- ode(state, times, func, pars)
plot(out, which = "N", las = 1)
```



Cycles!