Numerical Solutions to Differential Equations in R

Nessy Tania, Erica Graham

In this tutorial, you will build R codes for obtaining numerical solutions to ordinary differential equations (ODEs). Here are the steps:

1. Load the `deSolve` package in R (this package must be installed in order to load).
2. Create a function with your ODE.
3. Use R ODE solvers such as `lsoda`, `lsode`, `ode23`, `ode45`, etc.

Each step is outlined on this tutorial below.

1. Start a new R script.
2. Load the `deSolve` package in R. (This must be isntalled before you load it.)
   `load(deSolve)`
3. In order to solve an ODE, we need to first define a function that takes in independent and dependent variables, as well as a set of parameters (where applicable).

```
example1 = function (t, y, parameters) {
    list(
        t*y + t
    )
}
```

Numerical ODE Solvers

R has a number of different ODE solvers that can be used for initial-value problems:

- `lsoda`: Automatically selects method for solving ODEs. This should be the first solve you try in R. Use for either stiff or nonstiff ODEs.
- `lsode`: If using `lsoda` turns out to be a slow process, then the DEs you are solving may be stiff (the solver needs to take very small steps each time). Use the 'stiff' option to solve this type of DE using the backward differentiation formula. The 'non-stiff' option will use the Adams method to solve.
- `ode45`: Simultaneously uses fourth and fifth order RK formulas to make error estimates and adjust the time step accordingly. For nonstiff ODEs.
- `ode23`: Uses simultaneously second and third order Runge Kutta formulas to make estimates of the error, and calculate the time step size. Since the second and third order RK require less steps, `ode23` is "less expensive" in terms of computation demands than `ode45`, but is also lower order. Use for nonstiff ODEs.

Example 1: Let's solve the following ode:

$$\frac{dy}{dt} = ty + t$$

with the initial condition $y(0) = 0.5$. To solve, type

```
library(deSolve)
example1 = function (t, Y, parameters) {
    y = Y[1]    # define the independent variable (optional)
    list(
```

```
        t*y + t
    )
}
yinit = 0.5
times = seq(0,5,0.1)
out = as.data.frame(ode(y = yinit, func = example1, times, parms = c()))
names(out) = c("t","y")
```

The commands
`out = as.data.frame(ode(y = yinit, func = example1, times, parms = c()))`
can be broken down as follows:

- the output is `out`, which defines the solution.
- `yinit` is the initial value. If you have a system of equations, you input the initial conditions as a vector.
- `func` tells R which function defines the differential equation
- `times` tells R to give the solution to the differential equation for the specified time points.
- `parms` provides an optional list of parameters to the model. In practice, `parameters` would contain a list of any model parameters.
- Optionally, we may specify the method to be used within the `ode` command, e.g. `method="lsoda"`.

Finally, we save the solution as a data frame, so that we can label the time points (always the first column) and the independent variables.

Example 2: Let's solve the following predator system

$$\frac{dx}{dt} = -\beta xy + x$$
$$\frac{dy}{dt} = \beta xy - \alpha y,$$

with initial conditions $x(0) = 0.1$ and $y(0) = 0.6$, from $t = 0$ to $t = 50$. We can define the function

```
example2 = function (t, Y, parameters) {
    alpha = parameters[1]
    beta = parameters[2]
    x = Y[1]
    y = Y[2]
    dxdt = -beta*x*y + x
    dydt = beta*x*y - alpha*y
    list(
        c(dxdt, dydt) # this is the output of the function
    )
}
```

Now, to solve the system (assuming the `deSolve` package is loaded), we type

```
yinit = c(0.1,0.6)
times = seq(0,50,0.01)
out = as.data.frame(ode(y = yinit, func = example2, times, parms = c(0.3, 0.4)))
names(out) = c("t","x","y")
```

You can plot the solutions too:

```
matplot(out$t,out[,2:3],type='l',col=c('red','blue'), xlab='t', ylab=F,lty=1)
legend('topleft',c('x','y'),col=c('red','blue'),lty=1)
```

Alternatively, you may plot the solution in the phase plane:

```
plot(out$x, out$y, type='l',xlab='x',ylab='y')
```

Exercise 3: Use `ode23` to solve the following system. Then plot $y_1,\ y_2,\ y_3$ over time.

$$
\begin{aligned}
y_1' &= 2y_1 + y_2 + 5y_3 + e^{-2t} \\
y_2' &= -3y_1 - 2y_2 - 8y_3 + 2e^{-2t} - \cos(3t) \\
y_3' &= 3y_1 + 3y_2 + 2y_3 + \cos(3t) \\
& y_1(0) = 1, \quad y_2(0) = -1, \quad y_3(0) = 0 \\
& t \in [0, \pi/2]
\end{aligned}
$$