Computer Coding Review

July 13, 2019

1 Practice problems

This assignment's focus is to get you to practice using a computer to help solve mathematical problems. Please submit a printout that includes all relevant computer code. This question asks you to simulate the growth of a single population with an Allee effect. The population has n individuals at time t, and its' growth is governed by the following equation:

$$\frac{dN}{dt} = f(n) = rn(1 - \frac{n}{k})(\frac{n}{a} - 1) \tag{1}$$

a Assign values 1, 10, and 100 to variables r, a, and k respectively (i.e. $r = 1$, $a = 10$, $k = 100$).

b Define the function, $f(n)$, for the growth rate of the population, with n (the population size) as an input. Evaluate this function at $n = 20$.

c Plot $f(n)$ with n values ranging from 0 to 110.

d Define a new function, $f(n, r, a, k)$ with 4 inputs, and plot $f(n, 1, 25, 100)$ with n values ranging from 0 to 110.

e Simulate the growth of a population over 100 years, with 15 individuals to begin with (i.e. at $n_0 = 15$), and $r = 1$, $a = 10$, and $k = 100$. Plot the resulting population vs. time dynamics.

f Repeat this procedure with 5 individuals to start with (i.e. n[0]=5).

g Create a function with inputs r, a, k, and n_0 that simulates this the growth of the population over 100 years and prints a graph of the population trajectory.

Answers For each of these questions, example code is included below for completing the tasks in *Mathematica*, *R*, and *Maxima*.

a *Mathematica code:*

```
r = 1; a = 10; k = 100;
```

R code:

```
r = 1; a = 10; k = 100;
```

Maxima code:

```
r: 1; a: 10; k: 100;
```

b *Mathematica code:*

```
F = Function[{n}, 1*n*(1 - n/100)*(n/10 - 1)];
F[20]
```

R code:

```
F = function(n) {1*n*(1 - n/100)*(n/10 - 1)}
F(20)
```

Maxima code:

```
F(n):= 1*n*(1 - n/100)*(n/10 - 1);
F(20);
```

c *Mathematica code:*

```
Plot[F[n], {n, 0, 110}]
```

R code:

```
x<-seq(0, 110, length=100)
plot(x, F(x), type="l")
```

Maxima code:

```
plot2d (F(n), [n, 0, 110]);
```

Note, if you are using *wxMaxima*, then you need to use the command "wxplot2d" d *Mathematica code:*

```
F = Function[{n, r, a, k}, r*n*(1 - n/k)*(n/a - 1)];
Plot[F[n, 1, 25, 100], {n, 0, 110}]
```

R code:

```
F = function(n, r, a, k) {r*n*(1 - n/k)*(n/a - 1)}
x<-seq(0, 110, length=100)
plot(x, F(x, 1, 25, 100), type="l")
```

Maxima code:

```
F(n,r,a,k):= r*n*(1 - n/k)*(n/a - 1);
plot2d (F(n, 1, 25, 100), [n, 0, 110]);
```

e *Mathematica code:*

```
s = NDSolve[{y'[x] == F[y[x], 1, 10, 100], y[0] == 15},
  y, {x, 0, 100}]
Evaluate[y[1] /. s]
Plot[Evaluate[y[x] /. s], {x, 0, 100}, PlotRange -> All]
```

R code:

```
require(deSolve)
odefun<-function(time, state, pars) {
  return(list(F(state, r=1, a=10, k=100)))
}
out<-ode(y = 15, times = seq(0, 100, length=100),
  func = odefun)
plot(out[,1], out[,2], type="l", xlab="time", ylab="n")
```

Note - if you do not already have "deSolve" installed, you will need to run the command: *install.packages("deSolve")* before you can do anything else.

Maxima code:

```
plotdf(F(n, 1, 10, 100), [t, n], [trajectory_at, 0, 15],
  [direction, forward], [t, 0, 100], [n, 0, 100]);
```

f *Mathematica code:*

```
s = NDSolve[{y'[x] == F[y[x], 1, 10, 100], y[0] == 5},
  y, {x, 0, 100}]
Evaluate[y[1] /. s]
Plot[Evaluate[y[x] /. s], {x, 0, 100}, PlotRange -> All]
```

R code:

```
odefun<-function(time, state, pars) {
  return(list(F(state, r=1, a=10, k=100)))
}
out<-ode(y = 5, times = seq(0, 100, length=100),
  func = odefun)
plot(out[,1], out[,2], type="l", xlab="time", ylab="n")
```

Maxima code:

```
plotdf(F(n, 1, 10, 100), [t, n], [trajectory_at, 0, 5],
  [direction, forward], [t, 0, 100], [n, 0, 100]);
```

g *Mathematica code:*

```
plotF = Function[{n0, r, a, k}, s = NDSolve[{y'[x] == F[y[x], r, a, k],
    y[0] == n0},y, {x, 0, 100}];
  Plot[Evaluate[y[x] /. s], {x, 0, 100}, PlotRange -> All]]
```

R code:

```
pltfun<-function(n0, r, a, k) {
  odefun<-function(time, state, pars) {
    return(list(F(state, r=pars[1], a=pars[2], k=pars[3])))
  }
  out<-ode(y = n0, times = seq(0, 100, length=100),
    func = odefun, parms=c(r, a, k))
  plot(out[,1], out[,2], type="l", xlab="time", ylab="n")
}
```

Maxima code:

```
plotF(n0, r, a, k):=plotdf(F(n, r, a, k), [t, n],
  [trajectory_at, 0, n0], [direction, forward],
  [t, 0, 100], [n, 0, k]);
```