Projeto I

Luiz Rodrigo de Souza

We need to solve the equation $\frac{dy}{dt} = Ay$ with initial condition $y(0) = y_0$, whose solution is $y = e^{At}y_0$. So we need to compute the exponential of a matrix. First, I will define a function that dedices whether a given matrix is diagonalizable. Notice: I use 2x2 matrix for examples, but the methods also work for arbitrary square matrices.

```
library(tidyverse)
```

```
# tries to diagonalize. returns false if there's an error
is_diagonalizable = function(M, tol = 1e-10) {
  p = eigen(M)$vectors
  d = diag(eigen(M)$values)
  possibly(~ norm(p %*% d %*% solve(p) - M) < tol, FALSE)()
}

ex1 = matrix(c(1, 1, 1, 0), byrow = TRUE, nrow = 2) # diag
ex1
```

```
##      [,1] [,2]
## [1,]    1    1
## [2,]    1    0
```

```
ex2 = matrix(c(1, 0, 1, 1), byrow = TRUE, nrow = 2) # non diag
ex2
```

```
##      [,1] [,2]
## [1,]    1    0
## [2,]    1    1
```

We write a function that exponentiates a matrix if it is diagonalizable:

```
exp_diagonalizable = function(M) {
  p = eigen(M)$vectors
  d = diag(exp(eigen(M)$values))
  p %*% d %*% solve(p)
}

exp_diagonalizable(ex1)
```

```
##          [,1]     [,2]
## [1,] 3.798246 2.014323
## [2,] 2.014323 1.783923
```

is_diagonalizable = function(M, tol = 1e-10) { p = eigen(M)$vectorsd = diag(eigen(M)$values) possibly(~ norm(p %% *d* %% solve(p) - M) < tol, FALSE)() }

```
# really naïve implementation of matrix powers
powm = function(M, n) {
  if (n == 0)
    diag(rep(1, nrow(M)))
  else
    map(1:n, ~ M) %>% reduce(`%*%`)
}
```

```r
# produces the fibonnaci sequence
map_dbl(0:10, ~ powm(ex1, .)[1, 1])
```

```
##  [1]  1  1  2  3  5  8 13 21 34 55 89
```

```r
exp_taylor = function(M, steps = 15) {
  map(0:steps, ~ powm(M, .)/factorial(.)) %>% reduce(`+`)
}

exp_taylor(ex2)
```

```
##          [,1]     [,2]
## [1,] 2.718282 0.000000
## [2,] 2.718282 2.718282
```

Finally, we write an `expm` function, and then an DE solver.

```r
expm = function(M, steps = 15) {
  if (is_diagonalizable(M))
    exp_diagonalizable(M)
  else
    exp_taylor(M)
}

solve_init_problem = function(A, y0) {
  function(t) as.vector(expm(A*t) %*% y0)
}
```

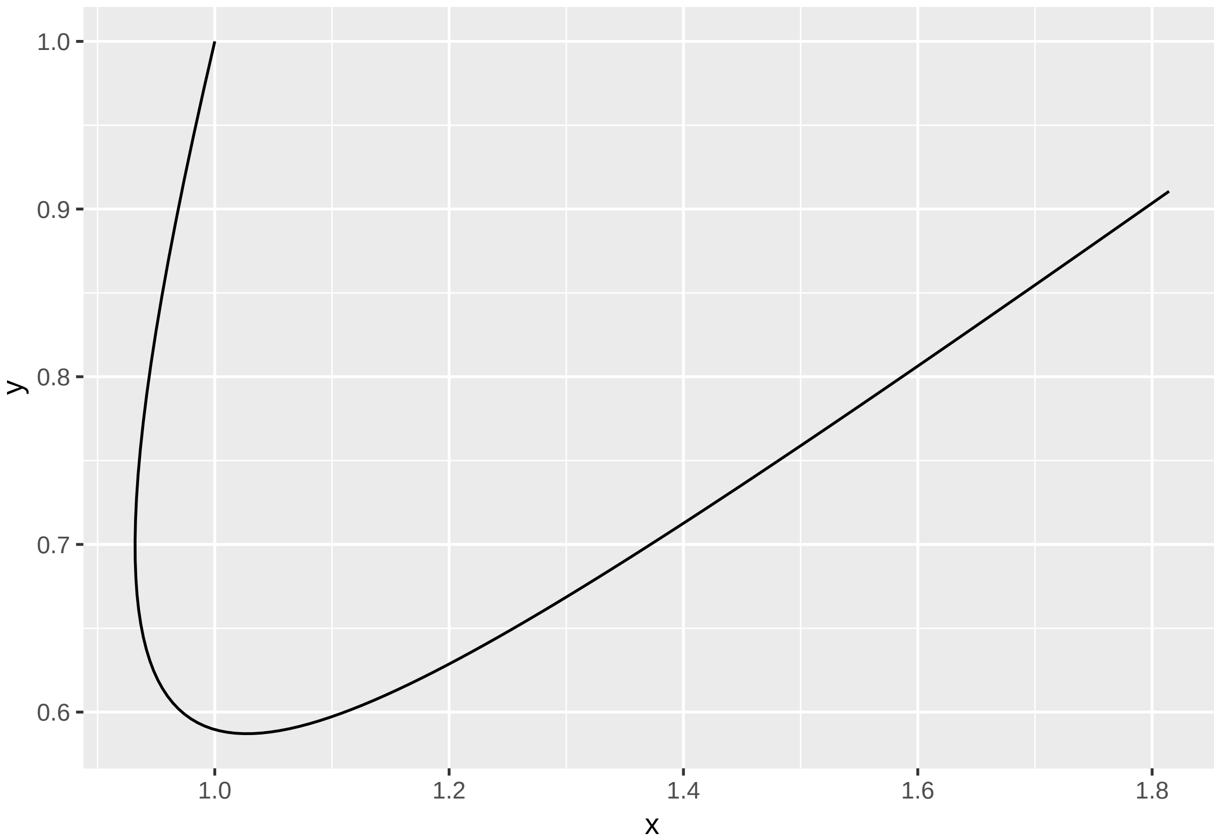
To see it at work, we can, for example, solve the system

$$\frac{dx}{dt} = 3x - 4y, \frac{dx}{dt} = 4x - 7y, x(0) = 1, y(0) = 1$$

```r
A = matrix(c(3, -4, 4, -7), byrow = TRUE, nrow = 2)
f = solve_init_problem(A, c(1, 1))

tb = seq(0, 1, by = 0.01) %>%
  map_dfr(function(t) {u = f(t); data.frame(x = u[1], y = u[2])})

ggplot(tb, aes(x = x, y = y)) + geom_path()
```



1.0
0.9
0.8
0.7
0.6
y
1.0
1.2
1.4
1.6
1.8
x