# Sistemas de equações em diferenças de primeira ordem

*Henri Makika*

*May 21, 2019*

### Sistema de equações em diferença de 2 × 2

O sistema mais simples na sua forma normal é tal que:

$$x_{t+1} = a_{11}x_t + a_{12}y_t + g_1(t)$$
$$y_{t+1} = a_{21}x_t + a_{22}y_t + g_2(t)$$

### Autovalores e autovetores

Para identificar os autovalores e autovetores de uma matriz quadrada, fazemos uso da função eigen do pacote base do R:

```
library(Matrix)
library(matlib)
```

Seja matrix :

```
v = c(2, 1, 1, 2)
A = matrix(v, nrow=2)

r = eigen(A)

lambda = r$values

P = r$vectors
print(P)
```

```
##           [,1]       [,2]
## [1,] 0.7071068 -0.7071068
## [2,] 0.7071068  0.7071068
```

```
det(P)  # Diferente de zero
```

```
## [1] 1
```

```
inv(P) %*% A %*% P
```

```
##      [,1] [,2]
## [1,]    3    0
## [2,]    0    1
```

```r
Ginv(P) %*% A %*% P
```

```
##      [,1] [,2]
## [1,]    3    0
## [2,]    0    1
```

Havendo autovalores reais e diferentes, há garantia para obter $n$ autovetores linearmente independentes, o que garante a existência da matriz P não singular.

Havendo multiplicidade, precisamos verificar se é possível obter a matriz $P$. Obter a matriz $P$ implica ter uma matriz $A$ diagonalizável.

## Diagonalização de matrizes

Vejamos o exemplo a seguir:

```r
A = matrix(c(1, 0, 0, 2, 1, 0, 3, 2, 1), nrow = 3)

r = eigen(A)

lambda = r$values

P = r$vectors
print(P)
```

```
##      [,1]          [,2]          [,3]
## [1,]    1 -1.000000e+00  1.000000e+00
## [2,]    0  1.110223e-16 -1.110223e-16
## [3,]    0  0.000000e+00  1.232595e-32
```

```r
det(P) # Será que det de P é diferente de zero ?
```

```
## [1] 1.368456e-48
```

```r
det(P) != 0 # Sim, det de P é diferente de zero
```

```
## [1] TRUE
```

```r
detP = format(det(P), scientific = FALSE, trim = TRUE)
```

```r
tol = 1e-5
detP = ifelse(detP<tol, 0, round(detP, 4))
detP != 0
```

```
## [1] FALSE
```

Verificamos o grau de liberdade

```r
Ahat = A - lambda[1]*diag(1, nrow(A) )

rankMatrix(Ahat)[1]
```

```
## [1] 2
```

```r
echelon(Ahat)
```

```
##      [,1] [,2] [,3]
## [1,]    0    1    0
## [2,]    0    0    1
## [3,]    0    0    0
```

### Simulando um sistema com raízes reais e diferentes

Neste caso, há certeza da diagonalização da matriz de coeficientes.

```r
library(tidyr)
library(dplyr)
library(ggplot2)
```

```r
A = matrix(c(0, 0.125, 1, .25), nrow=2)

r = eigen(A)

lambda = r$values
lambda
```

```
## [1]  0.50 -0.25
```

```r
P = r$vectors
P
```

```
##            [,1]       [,2]
## [1,] -0.8944272 -0.9701425
## [2,] -0.4472136  0.2425356
```

```r
det(P)
```

```
## [1] -0.6507914
```

```r
Z0 = c(1,.1) # condicao inicial

tmax = 20

Z = matrix(0, nrow=2, ncol=tmax)

Z[,1] = Z0
```

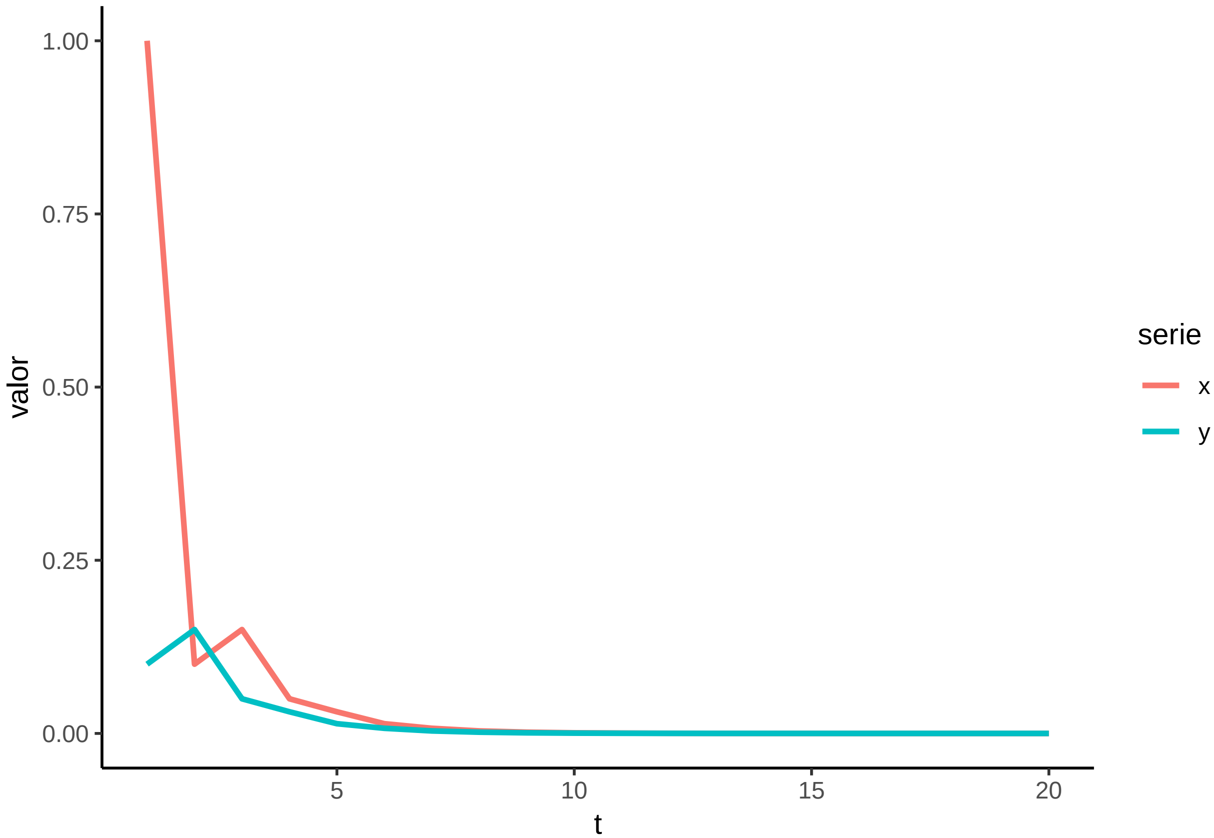
```r
for (t in 2:tmax){
  Z[,t] = A %*% Z[ , t-1]
}

t(Z)
```

```
##                [,1]         [,2]
##  [1,] 1.000000e+00 1.000000e-01
##  [2,] 1.000000e-01 1.500000e-01
##  [3,] 1.500000e-01 5.000000e-02
##  [4,] 5.000000e-02 3.125000e-02
##  [5,] 3.125000e-02 1.406250e-02
##  [6,] 1.406250e-02 7.421875e-03
##  [7,] 7.421875e-03 3.613281e-03
##  [8,] 3.613281e-03 1.831055e-03
##  [9,] 1.831055e-03 9.094238e-04
## [10,] 9.094238e-04 4.562378e-04
## [11,] 4.562378e-04 2.277374e-04
## [12,] 2.277374e-04 1.139641e-04
## [13,] 1.139641e-04 5.695820e-05
## [14,] 5.695820e-05 2.848506e-05
## [15,] 2.848506e-05 1.424104e-05
## [16,] 1.424104e-05 7.120892e-06
## [17,] 7.120892e-06 3.560353e-06
## [18,] 3.560353e-06 1.780200e-06
## [19,] 1.780200e-06 8.900941e-07
## [20,] 8.900941e-07 4.450485e-07
```

```r
t = seq(1, tmax, 1)
series = data.frame(t, x = Z[1, ], y = Z[2,])
series
```

```
##     t            x            y
## 1   1 1.000000e+00 1.000000e-01
## 2   2 1.000000e-01 1.500000e-01
## 3   3 1.500000e-01 5.000000e-02
## 4   4 5.000000e-02 3.125000e-02
## 5   5 3.125000e-02 1.406250e-02
## 6   6 1.406250e-02 7.421875e-03
## 7   7 7.421875e-03 3.613281e-03
## 8   8 3.613281e-03 1.831055e-03
## 9   9 1.831055e-03 9.094238e-04
## 10 10 9.094238e-04 4.562378e-04
## 11 11 4.562378e-04 2.277374e-04
## 12 12 2.277374e-04 1.139641e-04
## 13 13 1.139641e-04 5.695820e-05
## 14 14 5.695820e-05 2.848506e-05
## 15 15 2.848506e-05 1.424104e-05
## 16 16 1.424104e-05 7.120892e-06
## 17 17 7.120892e-06 3.560353e-06
## 18 18 3.560353e-06 1.780200e-06
## 19 19 1.780200e-06 8.900941e-07
## 20 20 8.900941e-07 4.450485e-07
```

```r
series_tidy = gather(series, -t, key = "serie", value = "valor")

ggplot(series_tidy, aes(x=t, y=valor, color=serie)) +
  geom_line(size=1) +
  theme_classic()
```




Pela matriz de coeficiente ter um autovalor negativo, espera-se obervar oscilações. Ainda, como os dois autovalores são menores que a unidade em módulo, espera-se uma trajetória amortecida para o conjunto de séries do sistema.

Lembre do efeito da condição inicial e da janela de tempo considerada na simulação. Obseve também que a raíz negativa possui um módulo pequeno (-0.25), que é atenuado rapidamente a medida que t aumenta.

Analisando cada componente podemos ter uma melhor precepção deste caso.

```r
tmax = 20
Z1 = matrix(0, nrow = 2, ncol = tmax)
Z2 = matrix(0, nrow = 2, ncol = tmax)

A1 = .1
A2 = -.1

v1 = P[, 1]
v2 = P[, 2]

for (t in 1:tmax){
  Z1[,t] = A1*v1*lambda[1]^t
  Z2[,t] = A2*v2*lambda[2]^t
```

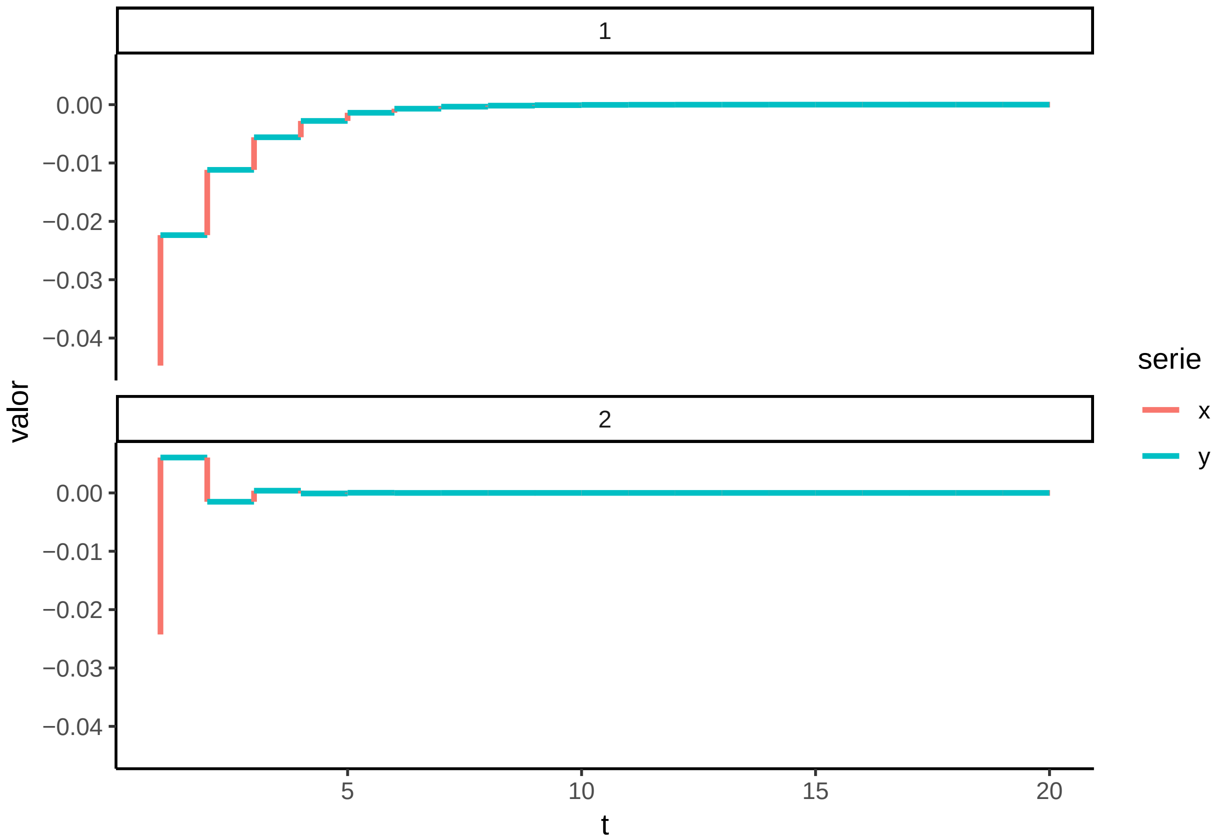
```r
}

t = seq(1, tmax, 1)

series1 = data.frame(t, x = Z1[1, ], y = Z1[2, ], p = rep(1, length(t)),
                     stringsAsFactors = FALSE)

head( series1 )
```

```
##   t            x             y p
## 1 1 -0.044721360 -0.0223606798 1
## 2 2 -0.022360680 -0.0111803399 1
## 3 3 -0.011180340 -0.0055901699 1
## 4 4 -0.005590170 -0.0027950850 1
## 5 5 -0.002795085 -0.0013975425 1
## 6 6 -0.001397542 -0.0006987712 1
```

```r
series2 = data.frame(t, x = Z2[1, ], y = Z2[2, ], p = rep(2, length(t)),
                     stringsAsFactors = FALSE)

head( series2 )
```

```
##   t             x             y p
## 1 1 -2.425356e-02  6.063391e-03 2
## 2 2  6.063391e-03 -1.515848e-03 2
## 3 3 -1.515848e-03  3.789619e-04 2
## 4 4  3.789619e-04 -9.474048e-05 2
## 5 5 -9.474048e-05  2.368512e-05 2
## 6 6  2.368512e-05 -5.921280e-06 2
```

```r
series = bind_rows(series1, series2)

series_tidy = gather(series, -t, -p, key = "serie", value = "valor")

ggplot(series_tidy, aes(x = t, y = valor, color = serie, group = p)) +
  geom_line(size = 1) +
  facet_wrap(~p, nrow = 2) +
  theme_classic()
```




### Simulando um sistema com raízes reais e iguais

```r
library(limSolve)
```

```r
A <- matrix(c(4, -1, 1, 2), nrow = 2)

r <- eigen(A) # autovalores

lambda <- r$values
print(lambda)
```

```
## [1] 3 3
```

```r
P <- r$vectors
print(P)
```

```
##            [,1]       [,2]
## [1,]  0.7071068 -0.7071068
## [2,] -0.7071068  0.7071068
```

```r
det(P)
```

```
## [1] 3.140185e-16
```

```r
det(P) != 0 # Determinante de P é diferente de zero
```

```
## [1] TRUE
```

```r
v1 <- P[,1]


#D <- zapsmall(ginv(P)%*%A%*%P ) # note que P deve ser inversivel
#D

#(A-lambda*I)^2*v2 = v1 # precisamos apenas uma solucao possivel

M <- A - diag(lambda[1], 2)
M
```

```
##      [,1] [,2]
## [1,]    1    1
## [2,]   -1   -1
```

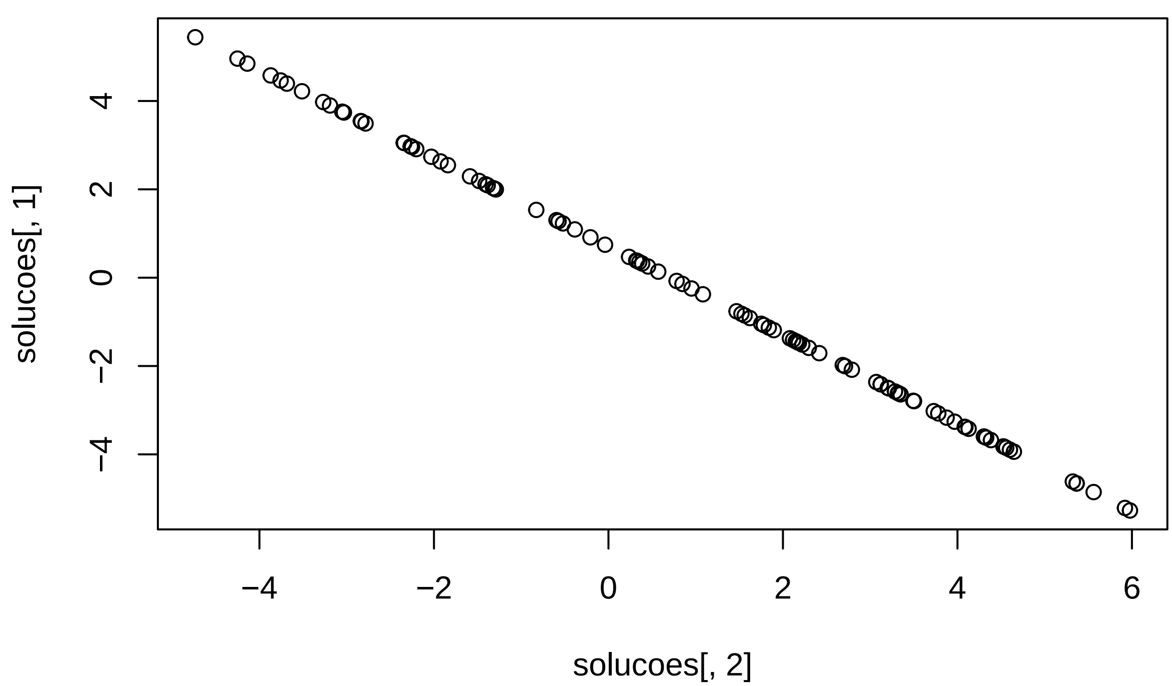
```r
B = matrix(v1, nrow = 2)
solucoes = xsample(E = M, F = B, iter = 100, jmp = 1)$X
print(solucoes)
```

```
##              [,1]        [,2]
##   [1,]  0.3535534  0.35355339
##   [2,]  0.7460241 -0.03891727
##   [3,]  0.2533449  0.45376191
##   [4,]  0.3802973  0.32680952
##   [5,]  0.3206507  0.38645611
##   [6,] -0.3757073  1.08281412
##   [7,] -0.7608091  1.46791587
##   [8,]  1.0927402 -0.38563337
##   [9,]  0.9135780 -0.20647125
##  [10,]  1.9979812 -1.29087446
##  [11,]  2.0079978 -1.30089105
##  [12,]  3.0476173 -2.34051056
##  [13,]  3.4910491 -2.78394237
##  [14,]  2.9730211 -2.26591435
##  [15,]  3.7389571 -3.03185031
##  [16,]  3.0539601 -2.34685333
##  [17,]  3.5456284 -2.83852162
##  [18,]  4.9591726 -4.25206586
##  [19,]  5.4429342 -4.73582737
##  [20,]  4.8462051 -4.13909827
##  [21,]  4.4646017 -3.75749491
##  [22,]  4.5786874 -3.87158059
##  [23,]  3.8981684 -3.19106162
##  [24,]  3.9763014 -3.26919465
##  [25,]  4.3927502 -3.68564347
##  [26,]  4.2197071 -3.51260029
##  [27,]  3.7578092 -3.05070247
##  [28,]  2.6314817 -1.92437493
```

```
## [29,]  2.9729726 -2.26586583
## [30,]  2.2939947 -1.58688792
## [31,]  2.1152642 -1.40815738
## [32,]  2.1891013 -1.48199451
## [33,]  2.9582785 -2.25117172
## [34,]  3.5380816 -2.83097478
## [35,]  2.9098376 -2.20273077
## [36,]  2.0909998 -1.38389302
## [37,]  0.3888745  0.31823233
## [38,]  1.5345506 -0.82744380
## [39,]  2.7372853 -2.03017848
## [40,]  2.5472185 -1.84011170
## [41,]  1.2786068 -0.57150007
## [42,]  2.0288203 -1.32171357
## [43,]  1.3024394 -0.59533264
## [44,]  1.2287217 -0.52161496
## [45,]  0.4706502  0.23645658
## [46,] -0.0739312  0.78103798
## [47,]  0.1365272  0.57057961
## [48,] -0.9140030  1.62110979
## [49,] -2.6158150  3.32292179
## [50,] -2.7867074  3.49381420
## [51,] -2.4938263  3.20093307
## [52,] -2.5062794  3.21338623
## [53,] -2.7956291  3.50273587
## [54,] -3.0212057  3.72831245
## [55,] -3.8193869  4.52649366
## [56,] -3.3842975  4.09140432
## [57,] -3.1690039  3.87611070
## [58,] -3.6017540  4.30886081
## [59,] -2.6106411  3.31774788
## [60,] -1.3724558  2.07956261
## [61,] -2.4124249  3.11953168
## [62,] -1.5891296  2.29623640
## [63,] -1.9783931  2.68549990
## [64,] -3.3760139  4.08312064
## [65,] -3.9400263  4.64713311
## [66,] -3.8287273  4.53583409
## [67,] -2.4136993  3.12080603
## [68,] -2.0828742  2.78998096
## [69,] -2.6413085  3.34841526
## [70,] -3.0732346  3.78034137
## [71,] -3.5958926  4.30299936
## [72,] -3.8945028  4.60160955
## [73,] -3.6769331  4.38403992
## [74,] -3.6224315  4.32953826
## [75,] -4.6154964  5.32260320
## [76,] -4.6579196  5.36502641
## [77,] -5.2703353  5.97744207
## [78,] -4.8539207  5.56102745
## [79,] -4.6593735  5.36648030
## [80,] -5.2129477  5.92005452
## [81,] -3.8521448  4.55925160
## [82,] -3.4209668  4.12807362
```

```
##  [83,] -3.2611379  3.96824467
##  [84,] -2.3627098  3.06981657
##  [85,] -1.4756833  2.18279007
##  [86,] -1.4088036  2.11591041
##  [87,] -1.4405244  2.14763121
##  [88,] -1.4551642  2.16227096
##  [89,] -2.0042946  2.71140139
##  [90,] -2.5770940  3.28420080
##  [91,] -1.0440989  1.75120573
##  [92,] -0.8541931  1.56129990
##  [93,] -1.1875558  1.89466260
##  [94,] -1.7093293  2.41643608
##  [95,] -1.5142581  2.22136489
##  [96,] -1.0743598  1.78146663
##  [97,] -1.1307498  1.83785656
##  [98,] -0.2450405  0.95214725
##  [99,] -0.1412167  0.84832344
## [100,] -0.8171996  1.52430642
```

```
plot(solucoes[,2], solucoes[,1])
```




```
c2 = matrix( solucoes[1,], nrow = 2 )

tmax = 10

Z = matrix(0,2, tmax)

Z[, 1] = c(2,-1)

for (t in 2:tmax){
  Z[,t] = A %*% Z[, t-1]
```

```
}

t(Z)
```

```
##         [,1]   [,2]
##  [1,]      2     -1
##  [2,]      7     -4
##  [3,]     24    -15
##  [4,]     81    -54
##  [5,]    270   -189
##  [6,]    891   -648
##  [7,]   2916  -2187
##  [8,]   9477  -7290
##  [9,]  30618 -24057
## [10,]  98415 -78732
```

### Modelo de Cournot

```
set.seed(1)

n = 2
a = 10
b = 0.5
d = 10
c = runif(n)

A = matrix(-1/2, nrow = n, ncol=n) + diag(1/2, n)
print(A)
```

```
##      [,1] [,2]
## [1,]  0.0 -0.5
## [2,] -0.5  0.0
```

```
B = (a-c)/(2*b)

tmax = 20

X = matrix(0, nrow = n, ncol = tmax)
X[, 1] = 100*runif(n) # condicoes iniciais para cada uma das n firmas

# simulando

for (t in 2:tmax){
  X[, t] = A %*% X[ , t-1] + B
}

t(X)
```

```
##              [,1]          [,2]
##  [1,]   57.285336   90.820778999
```

```
##  [2,] -35.675898 -19.014792067
##  [3,]  19.241887  27.465825182
##  [4,]  -3.998421   0.006932415
##  [5,]   9.731025  11.627086727
##  [6,]   3.920948   4.762363536
##  [7,]   7.353310   7.667402114
##  [8,]   5.900790   5.951221316
##  [9,]   6.758881   6.677480960
## [10,]   6.395751   6.248435761
## [11,]   6.610273   6.430000672
## [12,]   6.519491   6.322739372
## [13,]   6.573122   6.368130600
## [14,]   6.550426   6.341315275
## [15,]   6.563834   6.352663082
## [16,]   6.558160   6.345959251
## [17,]   6.561512   6.348796202
## [18,]   6.560093   6.347120245
## [19,]   6.560931   6.347829483
## [20,]   6.560577   6.347410493
```

```r
series = as.data.frame(t(X))
print(series)
```

```
##            V1           V2
## 1   57.285336  90.820778999
## 2  -35.675898 -19.014792067
## 3   19.241887  27.465825182
## 4   -3.998421   0.006932415
## 5    9.731025  11.627086727
## 6    3.920948   4.762363536
## 7    7.353310   7.667402114
## 8    5.900790   5.951221316
## 9    6.758881   6.677480960
## 10   6.395751   6.248435761
## 11   6.610273   6.430000672
## 12   6.519491   6.322739372
## 13   6.573122   6.368130600
## 14   6.550426   6.341315275
## 15   6.563834   6.352663082
## 16   6.558160   6.345959251
## 17   6.561512   6.348796202
## 18   6.560093   6.347120245
## 19   6.560931   6.347829483
## 20   6.560577   6.347410493
```

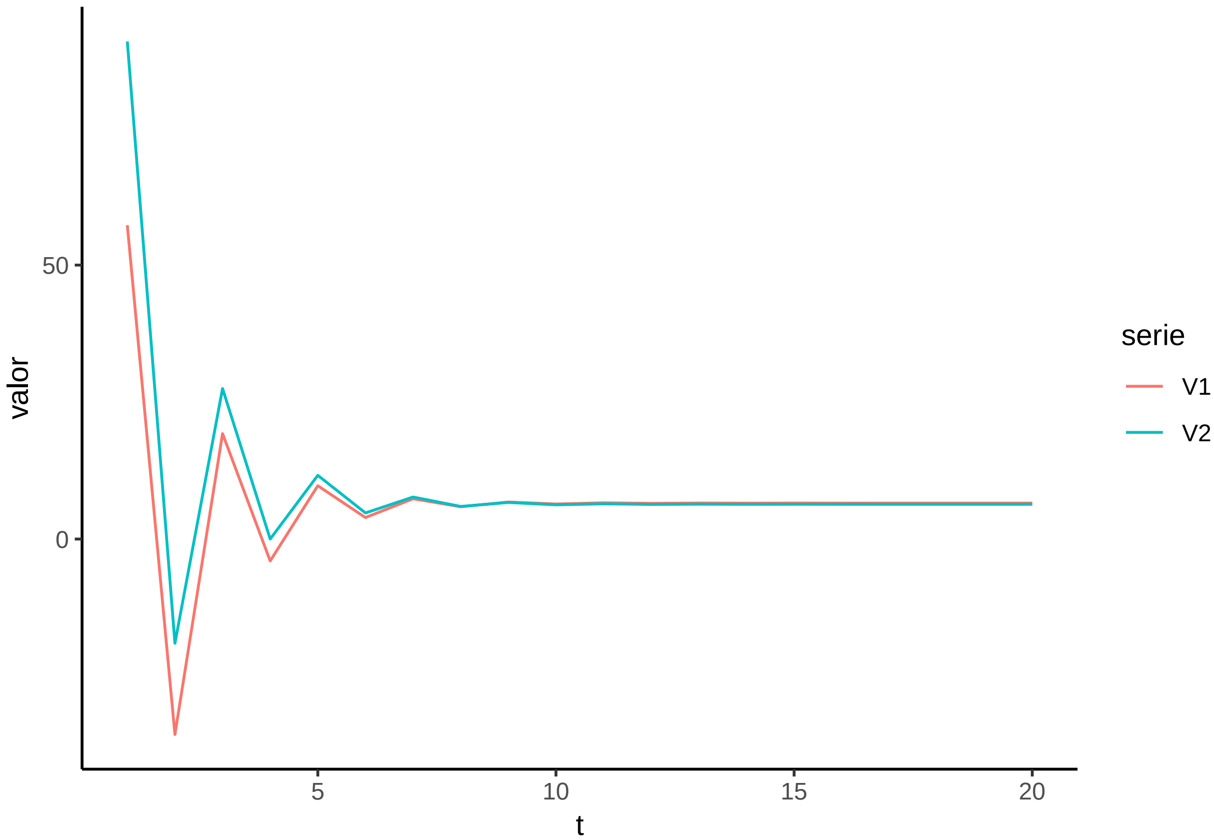
```r
series$t = seq(1, tmax, 1)
print(series$t)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
```

```r
series_tidy = gather(series, -t, key = "serie", value = "valor")
print(series_tidy)
```

```
##     t serie        valor
## 1   1    V1  57.285336335
## 2   2    V1 -35.675898163
## 3   3    V1  19.241887370
## 4   4    V1  -3.998421254
## 5   5    V1   9.731025129
## 6   6    V1   3.920947973
## 7   7    V1   7.353309569
## 8   8    V1   5.900790280
## 9   9    V1   6.758880679
## 10 10    V1   6.395750857
## 11 11    V1   6.610273456
## 12 12    V1   6.519491001
## 13 13    V1   6.573121651
## 14 14    V1   6.550426037
## 15 15    V1   6.563833699
## 16 16    V1   6.558159796
## 17 17    V1   6.561511712
## 18 18    V1   6.560093236
## 19 19    V1   6.560931215
## 20 20    V1   6.560576596
## 21  1    V2  90.820778999
## 22  2    V2 -19.014792067
## 23  3    V2  27.465825182
## 24  4    V2   0.006932415
## 25  5    V2  11.627086727
## 26  6    V2   4.762363536
## 27  7    V2   7.667402114
## 28  8    V2   5.951221316
## 29  9    V2   6.677480960
## 30 10    V2   6.248435761
## 31 11    V2   6.430000672
## 32 12    V2   6.322739372
## 33 13    V2   6.368130600
## 34 14    V2   6.341315275
## 35 15    V2   6.352663082
## 36 16    V2   6.345959251
## 37 17    V2   6.348796202
## 38 18    V2   6.347120245
## 39 19    V2   6.347829483
## 40 20    V2   6.347410493
```

```r
ggplot(series_tidy, aes(x = t, y = valor, color = serie)) +
  geom_line() +
  theme_classic()
```




```r
Xeq = inv(diag(1,n) - A) %*% B
print(Xeq)
```

```
##          [,1]
## [1,] 6.560738
## [2,] 6.347507
```

```r
# Verificando os autovalores
lambda = eigen(A)$values
print(lambda)
```

```
## [1]  0.5 -0.5
```