Projeto 5 - Modelo de Leslie

João Victor Dell Agli Floriano, 10799783

O Modelo de Leslie, introduzido por Patrick H. Leslie em 1945 em seu artigo, *On the use of matrices in certain population mathematics*, apresenta uma maneira de estuturar e modelar o crescimento populacional por idade. O modelo, de comportamento de tempo discreto, se baseia na ideia de fertilidade por faixas de idade, separando a população com útero fértil em faixas, que vão de **0** a **49** anos de idade.

Considerando uma população com faixas de idade separadas de 5 em 5 anos, as quais tais faixas são referidas como $P_{k,t}$, a população da faixa k no tempo t. O crescimento de cada faixa aqui é baseado em dois parâmetros:

- f_k : Fertilidade na faixa k, sendo $f_k \geq 0$.
- s_k : Probabilidade de um animal passar do tempo t para o tempo $t+1$, definida entre $[0.0, 1.0]$.

A partir desses dois parâmetros, conseguimos organizar um sistema linear que, para cada faixa de idade, existe uma fertilidade e probabilidade de sobrevivência associadas, que se organizam da seguinte maneira:

$$\begin{bmatrix} P_{0,t+1} \\ P_{1,t+1} \\ \vdots \\ P_{K,t+1} \end{bmatrix} = \begin{bmatrix} f_0 & f_1 & \cdots & f_K \\ s_0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} P_{0,t} \\ P_{1,t} \\ \vdots \\ P_{K,t} \end{bmatrix}$$

Nesse sistema linear, a matriz do meio, que organiza as fertilidades e probabilidades, é a matriz de Leslie.

$$\mathbf{P}_{t+1} = \mathbf{M}\mathbf{P}_t$$

Para resolver esse sistema, testemos um chute:

$$\mathbf{P}_t = r^t\mathbf{V}$$

$$r^{t+1}\mathbf{V} = \mathbf{M}r^t\mathbf{V}$$

$$r\mathbf{V} = \mathbf{M}\mathbf{V}$$

$$\mathbf{M}\mathbf{V} = r\mathbf{V}$$

Que indica que esse chute é equivalente à solução de um sistema de autovalores e autovetores, portanto, basta calculá-los para encontrar valores de r, e as distribuições estáveis.

O que Leslie provou foi que, calculando esses autovalores, apenas o maior autovalor indicaria o comportamento assintótico de $\mathbf{P(t)}$.

Implementação

In [ ]:
```
import numpy as np
```

Para a Matriz de Leslie de exemplo:

In [ ]:
```
leslie = np.array([[0.00000, 0.00105, 0.08203, 0.28849, 0.37780, 0.26478, 0.14055, 0.05857, 0.01344, 0.00081],
                   [0.99694, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
                   [0.0, 0.99842, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
                   [0.0, 0.0, 0.99785, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
                   [0.0, 0.0, 0.0, 0.99671, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
                   [0.0, 0.0, 0.0, 0.0, 0.99614, 0.0, 0.0, 0.0, 0.0, 0.0],
                   [0.0, 0.0, 0.0, 0.0, 0.0, 0.99496, 0.0, 0.0, 0.0, 0.0],
                   [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.99247, 0.0, 0.0, 0.0],
                   [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.98875, 0.0, 0.0],
                   [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.98305, 0.0]])
```

Calculemos os autovalores e autovetores:

In [ ]:
```
val, vec = np.linalg.eig(leslie)

print(val)
print(vec)
```

```
[ 1.03765055+0.j          0.30981229+0.73740434j  0.30981229-0.73740434j
  0.01057436+0.51101901j  0.01057436-0.51101901j -0.3879696 +0.37153279j
 -0.3879696 -0.37153279j -0.40819376+0.10637076j -0.40819376-0.10637076j
 -0.08609712+0.j          ]
[[ 3.74609824e-01+0.00000000e+00j -3.55671587e-02-7.59189277e-02j
  -3.55671587e-02+7.59189277e-02j  3.98101929e-04+2.11317904e-03j
   3.98101929e-04-2.11317904e-03j  2.73952324e-03-1.83686145e-03j
   2.73952324e-03+1.83686145e-03j -2.67214306e-04-3.02537556e-04j
  -2.67214306e-04+3.02537556e-04j -2.73592209e-10+0.00000000e+00j]
 [ 3.59912610e-01+0.00000000e+00j -1.04411391e-01+4.21802682e-03j
  -1.04411391e-01-4.21802682e-03j  4.13687176e-03-6.91048604e-04j
   4.13687176e-03+6.91048604e-04j -6.02988393e-03-1.05435819e-03j
  -6.02988393e-03+1.05435819e-03j  4.30820000e-04+8.51160579e-04j
   4.30820000e-04-8.51160579e-04j  3.16799244e-09+0.00000000e+00j]
 [ 3.46305361e-01+0.00000000e+00j -4.56293858e-02+1.22198734e-01j
  -4.56293858e-02-1.22198734e-01j -1.18240272e-03-8.10701477e-03j
  -1.18240272e-03+8.10701477e-03j  6.73906898e-03+9.16689706e-03j
   6.73906898e-03-9.16689706e-03j -4.78735133e-04-2.20664607e-03j
  -4.78735133e-04+2.20664607e-03j -3.67374322e-08+0.00000000e+00j]
 [ 3.33022330e-01+0.00000000e+00j  1.18499617e-01+1.11531647e-01j
   1.18499617e-01-1.11531647e-01j -1.58712812e-02+1.98041937e-03j
  -1.58712812e-02-1.98041937e-03j  2.73619306e-03-2.09568038e-02j
   2.73619306e-03+2.09568038e-02j -2.20423132e-04+5.33681649e-03j
  -2.20423132e-04-5.33681649e-03j  4.25780187e-07+0.00000000e+00j]
 [ 3.19882919e-01+0.00000000e+00j  1.85330850e-01-8.23048841e-02j
   1.85330850e-01+8.23048841e-02j  3.22074132e-03+3.10225680e-02j
   3.22074132e-03-3.10225680e-02j -3.05609977e-02+2.45726553e-02j
  -3.05609977e-02-2.45726553e-02j  3.68385034e-03-1.20712389e-02j
   3.68385034e-03+1.20712389e-02j -4.92907757e-06+0.00000000e+00j]
 [ 3.07086206e-01+0.00000000e+00j -5.09819687e-03-2.52500490e-01j
  -5.09819687e-03+2.52500490e-01j  6.05769138e-02-5.02475881e-03j
   6.05769138e-02+5.02475881e-03j  7.24476497e-02+6.28624611e-03j
   7.24476497e-02-6.28624611e-03j -1.56066056e-02+2.53912685e-02j
  -1.56066056e-02-2.53912685e-02j  5.70292186e-05+0.00000000e+00j]
 [ 2.94452204e-01+0.00000000e+00j -2.92033458e-01-1.15815771e-01j
  -2.92033458e-01+1.15815771e-01j -7.33954367e-03-1.18095836e-01j
  -7.33954367e-03+1.18095836e-01j -8.88630205e-02-1.01219500e-01j
  -8.88630205e-02+1.01219500e-01j  5.07240956e-02-4.86723168e-02j
   5.07240956e-02+4.86723168e-02j -6.59044032e-04+0.00000000e+00j]
 [ 2.81631402e-01+0.00000000e+00j -2.72848088e-01+2.78412731e-01j
  -2.72848088e-01-2.78412731e-01j -2.29555212e-01+9.50429972e-03j
  -2.29555212e-01-9.50429972e-03j -1.07661284e-02+2.48620891e-01j
  -1.07661284e-02-2.48620891e-01j -1.44363967e-01+8.07207582e-02j
  -1.44363967e-01-8.07207582e-02j  7.59701893e-03+0.00000000e+00j]
 [ 2.68359178e-01+0.00000000e+00j  1.86655118e-01+4.44269962e-01j
   1.86655118e-01-4.44269962e-01j  9.19474246e-03+4.44347353e-01j
   9.19474246e-03-4.44347353e-01j  3.30823974e-01-3.16808205e-01j
   3.30823974e-01+3.16808205e-01j  3.75162560e-01-9.77631910e-02j
   3.75162560e-01+9.77631910e-02j -8.72451099e-02+0.00000000e+00j]
 [ 2.54238278e-01+0.00000000e+00j  5.92266093e-01+0.00000000e+00j
   5.92266093e-01-0.00000000e+00j  8.54793381e-01+0.00000000e+00j
   8.54793381e-01-0.00000000e+00j -8.38252551e-01+0.00000000e+00j
  -8.38252551e-01-0.00000000e+00j -9.03501200e-01+0.00000000e+00j
  -9.03501200e-01-0.00000000e+00j  9.96157688e-01+0.00000000e+00j]]
```

A partir dos autovalores, conseguimos encontrar apenas os valores reais:

In [ ]:
```
val_r = val[val.imag == 0.0]

print(val_r)
```

```
[ 1.03765055+0.j -0.08609712+0.j]
```

E então, procuramos pelo maior valor:

In [ ]:
```
r = val_r[val_r == np.max(val_r)][0].real

print(r)
```

```
1.037650550443967
```

Que, no caso dessa matriz, é $r_1 \approx 1.0376$, que indica que a população de mulheres de idade menor que 50 anos, cresce exponencialmente com o tempo já que $r_1 > 1$, na taxa de 3.76% a cada 5 anos.

A partir desse autovalor, seu autovetor associado que indica a distribuição estável das faixas é:

In [ ]:
```
dist = vec.T[val.imag == 0.0]

dist = dist[val_r == np.max(val_r)].real[0]

dist = dist/np.sum(dist)
print(dist)
```

```
[0.11932148 0.11464009 0.11030589 0.10607495 0.10188976 0.09781372
 0.09378951 0.0897058  0.08547831 0.08098049]
```

Que, traduzido em porcentagens é:

$$D = \begin{bmatrix} 11.93\% \\ 11.46\% \\ 11.03\% \\ 10.60\% \\ 10.18\% \\ 9.78\% \\ 9.37\% \\ 8.97\% \\ 8.54\% \\ 8.09\% \end{bmatrix}$$