

▼ LISTA 8 MAE0399

- O número de instantes em que ocorre uma nova matrícula segue uma Poisson de parâmetro 2.
- O tempo entre duas matrículas consecutivas segue uma distribuição exponencial de parâmetro $1/2$.
- O valor das mensalidades de cada segurado segue um uniforme em 50 e 150 reais, o que define a variável $R(t)$ que segue um distribuição de Poisson composta.
- Os sinistros ocorrem segundo uma Poisson de parâmetro $1/30$.
- O tempo entre dois sinistros consecutivos segue uma distribuição exponencial com parâmetro 30.
- O valor de reparo dos sinistros estamos considerando assumindo uma uniforme entre 2000 e 4000 reais, o que define a variável $D(t)$ com distribuição Poisson Composta.

```
##### IMPORTA AS BIBLIOTECAS #####
import random
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.stats import poisson
```

- item a) Considere 1.000 simulações para as trajetórias de $R(t)$, $D(t)$ e $R(t) - D(t)$, para $t \in [0, 500]$. Para
- ▼ cada $t \in \{50, 100, \dots, 500\}$ encontre os percentis 0, 05 e 99, 5 para os valores observados de S_t . Apresente estes valores em tabelas.

```
##### TABELAS DOS QUANTIS #####
```

```

tabela_quantis_tempo_r = [[0 for k in range(3)] for i in range(10)]
tabela_quantis_tempo_s = [[0 for i in range(3)] for k in range(10)]
tabela_quantis_tempo_rs = [[0 for i in range(3)] for k in range(10)]

##### TEMPOS PEDIDOS NO ENUNCIADO #####

tempos_pedidos = [i for i in range(50,550,50)]

#cada linha representa um tempo pedido, as colunas representam o s de cada iteração
tabela_ts_r = pd.DataFrame([[0 for i in range(1000)] for k in range(10)])
tabela_ts_s = pd.DataFrame([[0 for i in range(1000)] for k in range(10)])
tabela_ts_rs = pd.DataFrame([[0 for i in range(1000)] for k in range(10)])

##### 1000 SIMULAÇÕES #####

for k in range(1000):

    #TEMPO LIMITE
    tempo_limite = 500

    #INSTANTES DE NOVOS CLIENTES
    t_novo_cliente = 0

    #INSTANTES DE NOVOS SINISTROS
    t_novo_sinistro = 0

    #DIFERENÇAS ENTRE TEMPOS
    t_dif = 0

    #LISTA PARA OS TEMPOS
    lista_tempos_clientes = []
    lista_tempos_sinistros = []
    lista_diferenca_tempos = []

    #CONTADOR DE CLIENTES
    numero_clientes = 0

```

```
#VARIAVEIS DE CUSTO
R_t = 0
D_t = 0

#LISTA PRA VARIAVEIS DE CUSTOS
lista_R_t = []
lista_D_t = []

#PERCCORE ATÉ TEMPO LIMITE
while min([t_novo_cliente,t_novo_sinistro]) <= tempo_limite:

    #SE TEMPO FOR MENOR QUE LIMITE ENTRE
    if t_novo_cliente <= tempo_limite:

        #INCREMENTO DO CONTADOR
        numero_clientes += 1

        #APPENDA NO LISTA DE TEMPOS
        lista_tempos_clientes.append(t_novo_cliente)

        #INCREMENTOS TEMPOS NOVOS CLIENTES
        t_novo_cliente += random.expovariate(2)

        #APPENDA NA LISTA
        lista_R_t.append(R_t)

        #INCREMENTO VARIÁVEL DE LUCRO
        R_t += random.uniform(50,150)

    #SE O INSTANTE DO SINISTRO FOR MENOR Q LIMITE ENTRE
    if t_novo_sinistro <= tempo_limite:

        #APPENDA NA LISTA
        lista_tempos_sinistros.append(t_novo_sinistro)
```

```
#INCREMENTO TEMPOS DOS SINISTROS
t_novo_sinistro += random.expovariate (1/30)
```

```
#APPENDA NA LISTA
lista_D_t.append(D_t)
```

```
#INCREMENTO VARIABEL DE PERDA
D_t += random.uniform(2000,4000)
```

```
##### TEMPOS DO ENUNCIADO #####
```

```
for tempo in tempos_pedidos:
```

```
#LISTAS PARA GUARDAR TEMPOS QUE NÃO EXCEDEM O TEMPO ESPECIFICO
lista_boa_clientes = []
lista_boa_sinistros = []
```

```
#PRA CADA TEMPO SE FOR MENOR ENTRE NA LISTA
for t1 in lista_tempos_clientes:
    if t1 <= tempo: lista_boa_clientes.append(t1)
```

```
#PRA CADA TEMPO SE FOR MENOR ENTRE NA LISTA
for t2 in lista_tempos_sinistros:
    if t2 <= tempo: lista_boa_sinistros.append(t2)
```

```
#APPENDA NA TABELA DE 1000 INTERAÇÕES OS CUSTOS/LUCROS RELATIVOS AO ULTIMO TEMPO QUE NAO ULTRAPASSOU
tabela_ts_r.iloc[(tempo//50)-1,k] = lista_R_t[len(lista_boa_clientes)-1]
tabela_ts_s.iloc[(tempo//50)-1,k] = lista_D_t[len(lista_boa_sinistros)-1]
tabela_ts_rs.iloc[(tempo//50)-1,k] = lista_R_t[len(lista_boa_clientes)-1] - lista_D_t[len(lista_boa_sinistros)-1]
```

```
#AGORA PRA CADA TEMPO CALCULA O QUANTIL DA SUA LINHA E APPENDA NA TABELA QUANTIS
```

```
for tempo in tempos_pedidos:
```

```

tabela_quantis_tempo_r[(tempo//50)-1][0] = tempo
tabela_quantis_tempo_r[(tempo//50)-1][1] = tabela_ts_r.iloc[(tempo//50)-1].quantile(0.05)
tabela_quantis_tempo_r[(tempo//50)-1][2] = tabela_ts_r.iloc[(tempo//50)-1].quantile(0.95)

tabela_quantis_tempo_s[(tempo//50)-1][0] = tempo
tabela_quantis_tempo_s[(tempo//50)-1][1] = tabela_ts_s.iloc[(tempo//50)-1].quantile(0.05)
tabela_quantis_tempo_s[(tempo//50)-1][2] = tabela_ts_s.iloc[(tempo//50)-1].quantile(0.95)

tabela_quantis_tempo_rs[(tempo//50)-1][0] = tempo
tabela_quantis_tempo_rs[(tempo//50)-1][1] = tabela_ts_rs.iloc[(tempo//50)-1].quantile(0.05)
tabela_quantis_tempo_rs[(tempo//50)-1][2] = tabela_ts_rs.iloc[(tempo//50)-1].quantile(0.95)

```

#TRANFORMA EM DATAFRAME E RENOMEIA AS COLUNAS

```

tabela_quantis_tempo_r = pd.DataFrame(tabela_quantis_tempo_r)
tabela_quantis_tempo_r.rename(columns={0: 't',1:'Quantil 5%',2:'Quantil 95%'},inplace = True)


tabela_quantis_tempo_s = pd.DataFrame(tabela_quantis_tempo_s)
tabela_quantis_tempo_s.rename(columns={0: 't',1:'Quantil 5%',2:'Quantil 95%'},inplace = True)

tabela_quantis_tempo_rs = pd.DataFrame(tabela_quantis_tempo_rs)
tabela_quantis_tempo_rs.rename(columns={0: 't',1:'Quantil 5%',2:'Quantil 95%'},inplace = True)

```

▼ Tabela de quantis $S(t)$

```
tabela_quantis_tempo_s
```

	t	Quantil 5%	Quantil 95%	
0	50	0.000000	12191.847107	
1	100	2355.731665	19507.497666	
2	150	5289.711158	27164.948098	
3	200	8229.326534	33544.391784	
4	250	11781.865320	40217.357163	
5	300	16031.726475	47586.475286	
6	350	18693.136805	53048.331466	
7	400	22273.786866	59161.984441	

▼ Tabela de quantis $R(t)$

tabela_quantis_tempo_r

	t	Quantil 5%	Quantil 95%
0	50	8322.250478	11719.091889



▼ Tabela de quantis $R(t) - S(t)$

2	150	27104.412545	32901.977889
---	-----	--------------	--------------

tabela_quantis_tempo_rs

	t	Quantil 5%	Quantil 95%
0	50	-2314.915622	10893.427856
1	100	-31.392695	18564.356328
2	150	2698.067398	25430.864403
3	200	6479.212200	32603.356409
4	250	9438.197166	38399.011707
5	300	12222.552129	44453.246426
6	350	16003.407209	51267.110956
7	400	19814.369079	57771.159428
8	450	24318.257166	64057.080706
9	500	27632.919626	70342.315187



▼ item b) Apresente em um só gráfico:

*A trajetória de 5 simulações, plotando os valores de $R(t) - D(t)$

*Os valores da tabela do item a, apenas para $R(t) - D(t)$

*A função $E[R(t) - D(t)]$

```
##### 5 SIMULAÇÕES #####
```

```
for k in range(5):
```

```
    #TEMPO LIMITE
```

```
    tempo_limite = 500
```

```
    #INSTANTES DE NOVOS CLIENTES
```

```
    t_r = 0
```

```
    #INSTANTES DE NOVOS SINISTROS
```

```
    t_s = 0
```

```
    #CONTADOR DE CLIENTES
```

```
    clientes = 0
```

```
    #VARIAVEIS DE CUSTO
```

```
    R_t = 0
```

```
    D_t = 0
```

```
    #LISTA PRA VARIAVEIS DE CUSTOS
```

```
    lista_R_t = []
```

```
    lista_D_t = []
```

```
    #LISTA PARA O TEMPO
```

```
    tempo_linear = [i for i in range(1,500,1)]
```

```
    #LISTA PARA OS INSTANTES DE OCORRENCIAS SINISTROS E CLIENTES
```

```
    tempos_clientes = []
```

```
    tempos_sinistros = []
```

```
    #PERCCORE ATÉ TEMPO LIMITE
```

```
    while min([t_r,t_s]) <= tempo_limite:
```

```
        #SE FOR MENOS QUE O TEMPO LIMITE ENTRA
```

```
        if t_r <= tempo_limite:
```



```
random.seed()

#INCREMENTA CLIENTES
clientes +=1

#APPENDA NAS RESPECTIVAS LISTAS
lista_R_t.append(R_t)
tempos_clientes.append(t_r)

#INCREMENTA INSTANTE DE NOVOS CLIENTES
t_r += random.expovariate(2)
#INCREMENTA GANHO
R_t += random.uniform(50,150)


#PERCORRE ATÉ TEMPO LIMITE
if t_s <= tempo_limite:

    random.seed()

    #APPENDA NAS RESPECTIVAS LISTAS
    lista_D_t.append(D_t)
    tempos_sinistros.append(t_s)
    #INCREMENTA INSTANTES DE SINISTROS
    t_s += random.expovariate(1/30)
    #INCREMENTA VALOR DOS SINISTROS
    D_t += random.uniform(2000,4000)


#LISTA PRAS DIFERENÇAS ENTRE S(t) e R(t)
diferencas = []

#PERCORRE TEMPO 500 DE UM EM UM
for tempo_especifico in tempo_linear:

    #LISTAS PARA GUARDAR TEMPOS QUE NÃO EXCEDEM O TEMPO ESPECIFICO
    lista_boa_clientes = []
```

```

lista_boa_sinistros = []

#PRA CADA TEMPO SE FOR MENOR ENTRE NA LISTA
for t in tempos_clientes:
    if t <= tempo_especifico: lista_boa_clientes.append(t)

#PRA CADA TEMPO SE FOR MENOR ENTRE NA LISTA
for tt in tempos_sinistros:
    if tt <= tempo_especifico: lista_boa_sinistros.append(tt)

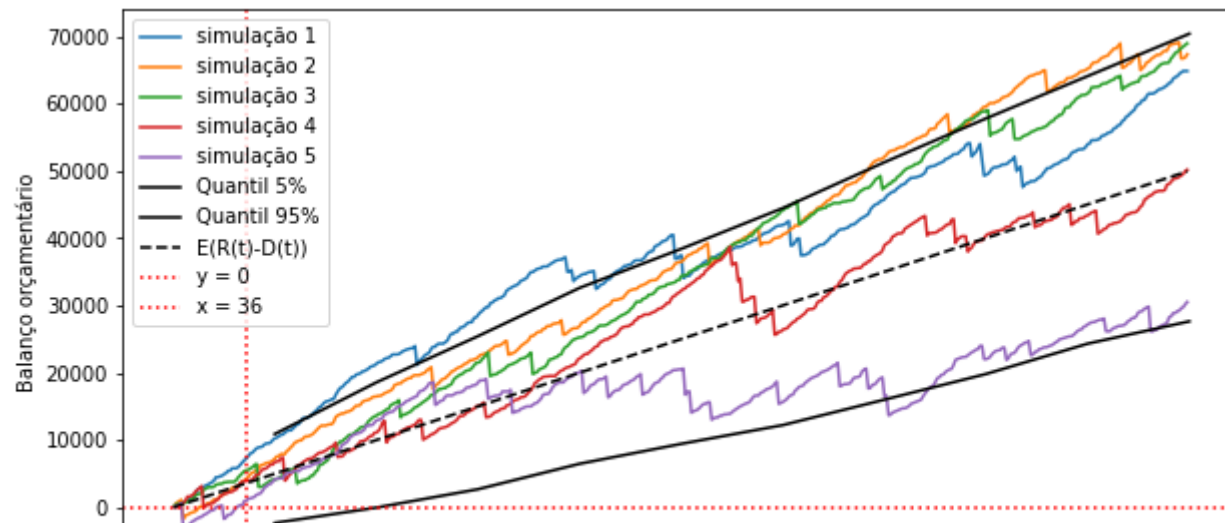
#CALCULA DIF E APPENDA NA LISTA
parte_r = lista_R_t[len(lista_boa_clientes)-1]
parte_s = lista_D_t[len(lista_boa_sinistros)-1]
diferencas.append(parte_r - parte_s)

#PRA CADA SIMULAÇÃO PLOTA A TRAJETÓRIA
plt.plot([tempo for tempo in range(1,500,1)],diferencas, label = f'simulação {k+1}')

#GRAFICO DOS QUANTIS 95 E 5 DA DIFERENÇA
plt.plot([i for i in range(50,550,50)],tabela_quantis_tempo_rs['Quantil 5%'], color = 'black', label = 'Quantil 5%')
plt.plot([i for i in range(50,550,50)],tabela_quantis_tempo_rs['Quantil 95%'], color = 'black', label = 'Quantil 95%')

#GRÁFICO DA ESPERANÇA DA DIFERENÇA
plt.plot([i for i in range(0,500,1)],[100*i for i in range(0,500,1)],color = 'black',linestyle = '--', label = 'E(R(t)-D(t))')
plt.rcParams["figure.figsize"] = (10,5)
plt.axhline(y = 0, xmin = 0, xmax = 500,linestyle = ':', color = 'red', label = 'y = 0')
plt.axvline(x= 36, ymin= 0, ymax=20000, linestyle = ':', color = 'red', label = 'x = 36')
plt.ylabel('Balanço orçamentário')
plt.xlabel('tempo')
plt.legend()
plt.show()

```



item c) Repita o exercício inserindo a ideia de franquia, ou seja, que o sinistro so seja coberto pela seguradora caso o seu valor seja superior a \$ 3000,00.

```
##### TABELAS DOS QUANTIS #####
```

```
#uma coluna a mais para armazenar a media
```

```
tabela_quantis_tempo_r = [[0 for k in range(4)] for i in range(10)]
```

```
tabela_quantis_tempo_s = [[0 for i in range(4)] for k in range(10)]
```

```
tabela_quantis_tempo_rs = [[0 for i in range(4)] for k in range(10)]
```

```
##### TEMPOS PEDIDOS NO ENUNCIADO #####
```

```
tempos_pedidos = [i for i in range(50,550,50)]
```

```
#cada linha representa um tempo pedido, as colunas representam o s de cada iteração
```

```
tabela_ts_r = pd.DataFrame([[0 for i in range(1000)] for k in range(10)])
```

```
tabela_ts_s = pd.DataFrame([[0 for i in range(1000)] for k in range(10)])
```

```
tabela_ts_rs =pd.DataFrame([[0 for i in range(1000)] for k in range(10)])
```

```
##### 1000 SIMULAÇÕES #####
```

```
for k in range(1000):

    #TEMPO LIMITE
    tempo_limite = 500

    #INSTANTES DE NOVOS CLIENTES
    t_novo_cliente = 0

    #INSTANTES DE NOVOS SINISTROS
    t_novo_sinistro = 0

    #DIFERENÇAS ENTRE TEMPOS
    t_dif = 0

    #LISTA PARA OS TEMPOS
    lista_tempos_clientes = []
    lista_tempos_sinistros = []
    lista_diferenca_tempos = []

    #CONTADOR DE CLIENTES
    numero_clientes = 0

    #VARIÁVEIS DE CUSTO
    R_t = 0
    D_t = 0

    #LISTA PARA VARIÁVEIS DE CUSTOS
    lista_R_t = []
    lista_D_t = []

    #PERCCORE ATÉ TEMPO LIMITE
    while min([t_novo_cliente,t_novo_sinistro]) <= tempo_limite:

        #SE TEMPO FOR MENOR QUE LIMITE ENTRE
        if t_novo_cliente <= tempo_limite:
```

```
#INCREMENTO DO CONTADOR
```

```
numero_clientes += 1
```

```
#APPENDA NO LISTA DE TEMPOS
```

```
lista_tempos_clientes.append(t_novo_cliente)
```

```
#INCREMENTOS TEMPOS NOVOS CLIENTES
```

```
t_novo_cliente += random.expovariate(2)
```

```
#APPENDA NA LISTA
```

```
lista_R_t.append(R_t)
```

```
#INCREMENTO VARIÁVEL DE LUCRO
```

```
R_t += random.uniform(50,150)
```

```
if t_novo_sinistro <= tempo_limite:
```

```
    valor_sinistro = random.uniform(2000,4000)
```

```
    if valor_sinistro >= 3000:
```

```
        #APPENDA NA LISTA
```

```
        lista_tempos_sinistros.append(t_novo_sinistro)
```

```
        #INCREMENTO TEMPOS DOS SINISTROS
```

```
        t_novo_sinistro += random.expovariate (1/30)
```

```
        #APPENDA NA LISTA
```

```
        lista_D_t.append(D_t)
```

```
        #INCREMENTO VARIÁVEL DE PERDA
```

```
        D_t += valor_sinistro
```

```
##### TEMPOS DO ENUNCIADO #####
```

```
for tempo in tempos_pedidos:
```

```
    #LISTAS PARA GUARDAR TEMPOS QUE NÃO EXCEDEM O TEMPO ESPECIFICO
```

```
    lista_boa_clientes = []
```

```
    lista_boa_sinistros = []
```

```
    #PRA CADA TEMPO SE FOR MENOR ENTRE NA LISTA
```

```
    for t1 in lista_tempos_clientes:
```

```
        if t1 <= tempo: lista_boa_clientes.append(t1)
```

```
    #PRA CADA TEMPO SE FOR MENOR ENTRE NA LISTA
```

```
    for t2 in lista_tempos_sinistros:
```

```
        if t2 <= tempo: lista_boa_sinistros.append(t2)
```

```
    #APPENDA NA TABELA DE 1000 INTERAÇÕES OS CUSTOS/LUCROS RELATIVOS AO ULTIMO TEMPO QUE NAO ULTRAPASSOU
```

```
    tabela_ts_r.iloc[(tempo//50)-1,k] = lista_R_t[len(lista_boa_clientes)-1]
```

```
    tabela_ts_s.iloc[(tempo//50)-1,k] = lista_D_t[len(lista_boa_sinistros)-1]
```

```
    tabela_ts_rs.iloc[(tempo//50)-1,k] = lista_R_t[len(lista_boa_clientes)-1] - lista_D_t[len(lista_boa_sinistros)-1]
```

```
#AGORA PRA CADA TEMPO CALCULA O QUANTIL DA SUA LINHA E APPENDA NA TABELA QUANTIS
```

```
for tempo in tempos_pedidos:
```

```
    tabela_quantis_tempo_r[(tempo//50)-1][0] = tempo
```

```
    tabela_quantis_tempo_r[(tempo//50)-1][1] = tabela_ts_r.iloc[(tempo//50)-1].quantile(0.05)
```

```
    tabela_quantis_tempo_r[(tempo//50)-1][2] = tabela_ts_r.iloc[(tempo//50)-1].quantile(0.95)
```

```
    tabela_quantis_tempo_r[(tempo//50)-1][3] = tabela_ts_r.iloc[(tempo//50)-1].mean()
```

```
    tabela_quantis_tempo_s[(tempo//50)-1][0] = tempo
```

```
    tabela_quantis_tempo_s[(tempo//50)-1][1] = tabela_ts_s.iloc[(tempo//50)-1].quantile(0.05)
```

```
    tabela_quantis_tempo_s[(tempo//50)-1][2] = tabela_ts_s.iloc[(tempo//50)-1].quantile(0.95)
```

```
    tabela_quantis_tempo_s[(tempo//50)-1][3] = tabela_ts_s.iloc[(tempo//50)-1].mean()
```

```
    tabela_quantis_tempo_rs[(tempo//50)-1][0] = tempo
```

```
tabela_quantis_tempo_rs[(tempo//50)-1][1] = tabela_ts_rs.iloc[(tempo//50)-1].quantile(0.05)
tabela_quantis_tempo_rs[(tempo//50)-1][2] = tabela_ts_rs.iloc[(tempo//50)-1].quantile(0.95)
tabela_quantis_tempo_rs[(tempo//50)-1][3] = tabela_ts_rs.iloc[(tempo//50)-1].mean()
```

#TRANFORMA EM DATAFRAME E RENOMEIA AS COLUNAS

```
tabela_quantis_tempo_r = pd.DataFrame(tabela_quantis_tempo_r)
tabela_quantis_tempo_r.rename(columns={0: 't',1:'Quantil 5%',2:'Quantil 95%',3:'Média'},inplace = True)
```

```
tabela_quantis_tempo_s = pd.DataFrame(tabela_quantis_tempo_s)
tabela_quantis_tempo_s.rename(columns={0: 't',1:'Quantil 5%',2:'Quantil 95%',3:'Média'},inplace = True)
```

```
tabela_quantis_tempo_rs = pd.DataFrame(tabela_quantis_tempo_rs)
tabela_quantis_tempo_rs.rename(columns={0: 't',1:'Quantil 5%',2:'Quantil 95%',3:'Média'},inplace = True)
```

▼ Tabela de quantis de $S(t)$

```
tabela_quantis_tempo_s
```

	t	Quantil 5%	Quantil 95%	Média
0	50	0.000000	14142.707232	5730.391944
1	100	3069.247262	23880.043287	11620.130324
2	150	6511.207363	31554.245756	17358.421734

▼ Tabela de quantis $R(t)$

tabela_quantis_tempo_r

	t	Quantil 5%	Quantil 95%	Média
0	50	8411.892658	11799.461867	10016.702759
1	100	17620.399234	22612.174078	20080.867402
2	150	27169.515160	33080.209390	30069.684254
3	200	36688.982188	43493.531944	40072.198480
4	250	46454.987999	53933.163025	50113.895936
5	300	56072.221833	64424.338469	60105.055672
6	350	66020.479887	74567.996853	70115.134068
7	400	75566.441748	84912.443544	80122.352902
8	450	85000.846605	95446.602895	90099.119548
9	500	94686.231412	105362.147782	100058.133032

▼ Tabela de quantis $R(t) - D(t)$

tabela_quantis_tempo_rs

tempo_quantil5_tempo_10

	t	Quantil 5%	Quantil 95%	Média
0	50	-4709.720577	10670.258379	4286.310815
1	100	-3682.624069	18417.506071	8460.737078
2	150	-1737.698496	24365.529888	12711.262520
3	200	-166.204539	31133.866367	16746.016579
4	250	3153.763111	36856.279840	20850.467757
5	300	5057.829155	43347.528746	24994.130264
6	350	8207.753482	48745.225118	29323.515446
7	400	9800.229538	53919.054771	33503.149613
8	450	13317.177037	59296.578850	37812.881739
9	500	15687.655402	65074.310264	41992.390028



5 SIMULAÇÕES

```
for k in range(5):
```

```
    #TEMPO LIMITE
```

```
    tempo_limite = 500
```

```
    #INSTANTES DE NOVOS CLIENTES
```

```
    t_r = 0
```

```
    #INSTANTES DE NOVOS SINISTROS
```

```
    t_s = 0
```

```
    #CONTADOR DE CLIENTES
```

```
    clientes = 0
```

```
    #VARIÁVEIS DE CUSTO
```

```

#VARIÁVEIS DE CUSTO
R_t = 0
D_t = 0

#LISTA PRA VARIÁVEIS DE CUSTOS
lista_R_t = []
lista_D_t = []

#LISTA PARA O TEMPO
tempo_linear = [i for i in range(1,500,1)]

#LISTA PARA OS INSTANTES DE OCORRENCIAS SINISTROS E CLIENTES
tempos_clientes = []
tempos_sinistros = []

#PERCCORE ATÉ TEMPO LIMITE
while min([t_r,t_s]) <= tempo_limite:

    #SE FOR MENOS QUE O TEMPO LIMITE ENTRA
    if t_r <= tempo_limite:

        random.seed()

        #INCREMENTA CLIENTES
        clientes +=1
        #APPENDA NAS RESPECTIVAS LISTAS
        lista_R_t.append(R_t)
        tempos_clientes.append(t_r)
        #INCREMENTA INSTANTE DE NOVOS CLIENTES
        t_r += random.expovariate(2)
        #INCREMENTA GANHO
        R_t += random.uniform(50,150)

    #PERCORRE ATÉ TEMPO LIMITE
    if t_s <= tempo_limite:

        random.seed()

```

```

valor_sinistro = random.uniform(2000,4000)

if valor_sinistro >= 3000:
    #APPENDA NAS RESPECTIVAS LISTAS
    lista_D_t.append(D_t)
    tempos_sinistros.append(t_s)
    #INCREMENTA INSTANTES DE SINISTROS
    t_s += random.expovariate(1/30)
    #INCREMENTA VALOR DOS SINISTROS
    D_t += valor_sinistro

#LISTA PRAS DIFERENÇAS ENTRE S(t) e R(t)
diferencas = []

#PERCORRE TEMPO 500 DE UM EM UM
for tempo_especifico in tempo_linear:

    #LISTAS PARA GUARDAR TEMPOS QUE NÃO EXCEDEM O TEMPO ESPECIFICO
    lista_boa_clientes = []
    lista_boa_sinistros = []

    #PRA CADA TEMPO SE FOR MENOR ENTRE NA LISTA
    for t in tempos_clientes:
        if t <= tempo_especifico: lista_boa_clientes.append(t)

    #PRA CADA TEMPO SE FOR MENOR ENTRE NA LISTA
    for tt in tempos_sinistros:
        if tt <= tempo_especifico: lista_boa_sinistros.append(tt)

    #CALCULA DIF E APPENDA NA LISTA
    parte_r = lista_R_t[len(lista_boa_clientes)-1]
    parte_s = lista_D_t[len(lista_boa_sinistros)-1]
    diferencas.append(parte_r - parte_s)

```

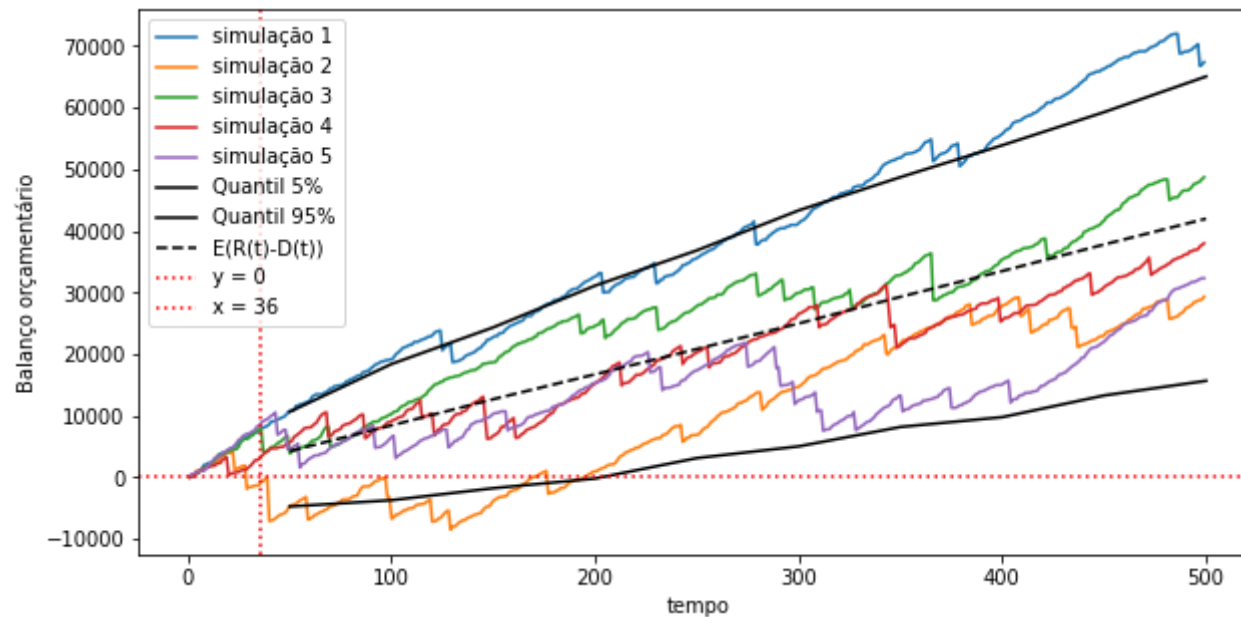
```

#PRA CADA SIMULAÇÃO PLOTA A TRAJETÓRIA
plt.plot([tempo for tempo in range(1,500,1)],diferencas, label = f'simulação {k+1}')

#GRAFICO DOS QUANTIS 95 E 5 DA DIFERENÇA
plt.plot([i for i in range(50,550,50)],tabela_quantis_tempo_rs['Quantil 5%'], color = 'black', label = 'Quantil 5%')
plt.plot([i for i in range(50,550,50)],tabela_quantis_tempo_rs['Quantil 95%'], color = 'black', label = 'Quantil 95%')

#GRÁFICO DA ESPERANÇA DA DIFERENÇA
plt.plot([i for i in range(50,550,50)],tabela_quantis_tempo_rs['Média'], color = 'black', label = 'E(R(t)-D(t))',linestyle = '--')
plt.rcParams["figure.figsize"] = (10,5)
plt.axhline(y = 0, xmin = 0, xmax = 500,linestyle = ':', color = 'red', label = 'y = 0')
plt.axvline(x= 36, ymin= 0, ymax=20000, linestyle = ':', color = 'red', label = 'x = 36')
plt.ylabel('Balanço orçamentário')
plt.xlabel('tempo')
plt.legend()
plt.show()

```



- item d) Repita o exercício fazendo alguma modificação (a seu critério) na distribuição do valor das mensalidades e/ou no valor da franquia, objetivando diminuir a chance de $R(t) < D(t)$ para algum t mas mantendo valores (baixos) em condições de competir com outras empresas.

```
##### COMENTÁRIO #####
```

```
''' AQUI UTILIZEI A DISTRIBUIÇÃO DE PAGAMENTOS COMO UMA UNIFORME ENTRE 100 E 150, MANTENDO O PREÇO RELATIVAMENTE BAIXO. E TAMBÉM AUME PARA A SEGURADORA PAGAR SINISTROS APENAS ACIMA DE 3500, COMO PODE SER VISTO NO GRAFICO AO FINAL, O BALANÇO FICA POSITIVO PARA TEMPO M
```

```
##### TABELAS DOS QUANTIS #####
```

```
tabela_quantis_tempo_r = [[0 for k in range(4)] for i in range(10)]
tabela_quantis_tempo_s = [[0 for i in range(4)] for k in range(10)]
tabela_quantis_tempo_rs = [[0 for i in range(4)] for k in range(10)]
```

```
##### TEMPOS PEDIDOS NO ENUNCIADO #####
```

```
tempos_pedidos = [i for i in range(50,550,50)]
```

```
#cada linha representa um tempo pedido, as colunas representam o s de cada iteração
```

```
tabela_ts_r = pd.DataFrame([[0 for i in range(1000)] for k in range(10)])
tabela_ts_s = pd.DataFrame([[0 for i in range(1000)] for k in range(10)])
tabela_ts_rs = pd.DataFrame([[0 for i in range(1000)] for k in range(10)])
```

```
##### 1000 SIMULAÇÕES #####
```

```
for k in range(1000):
```

```
    #TEMPO LIMITE
    tempo_limite = 500
```

```
    #INSTANTES DE NOVOS CLIENTES
    t_novo_cliente = 0
```

```
#INSTANTES DE NOVOS SINISTROS
t_novo_sinistro = 0

#DIFERENÇAS ENTRE TEMPOS
t_dif = 0

#LISTA PARA OS TEMPOS
lista_tempos_clientes = []
lista_tempos_sinistros = []
lista_diferenca_tempos = []

#CONTADOR DE CLIENTES
numero_clientes = 0

#VARIAVEIS DE CUSTO
R_t = 0
D_t = 0

#LISTA PRA VARIAVEIS DE CUSTOS
lista_R_t = []
lista_D_t = []

#PERCCORE ATÉ TEMPO LIMITE
while min([t_novo_cliente,t_novo_sinistro]) <= tempo_limite:

    #SE TEMPO FOR MENOR QUE LIMITE ENTRE
    if t_novo_cliente <= tempo_limite:

        #INCREMENTO DO CONTADOR
        numero_clientes += 1

        #APPENDA NO LISTA DE TEMPOS
        lista_tempos_clientes.append(t_novo_cliente)

        #INCREMENTOS TEMPOS NOVOS CLIENTES
        t_novo_cliente += random.expovariate(2)
```

```
#APPENDA NA LISTA
lista_R_t.append(R_t)

#INCREMENTO VARIÁVEL DE LUCRO
R_t += random.uniform(100,150)
```

```
if t_novo_sinistro <= tempo_limite:
```

```
    valor_sinistro = random.uniform(2000,4000)
```

```
    if valor_sinistro >= 3500:
```

```
        #APPENDA NA LISTA
        lista_tempos_sinistros.append(t_novo_sinistro)
```

```
        #INCREMENTO TEMPOS DOS SINISTROS
        t_novo_sinistro += random.expovariate (1/30)
```

```
        #APPENDA NA LISTA
        lista_D_t.append(D_t)
```

```
        #INCREMENTO VARIÁVEL DE PERDA
        D_t += valor_sinistro
```

```
##### TEMPOS DO ENUNCIADO #####
```

```
for tempo in tempos_pedidos:
```

```
    #LISTAS PARA GUARDAR TEMPOS QUE NÃO EXCEDEM O TEMPO ESPECIFICO
    lista_boa_clientes = []
    lista_boa_sinistros = []
```

```
    #PRA CADA TEMPO SE FOR MENOR ENTRE NA LISTA
```

```

for t1 in lista_tempos_clientes:
    if t1 <= tempo: lista_boa_clientes.append(t1)

#PRA CADA TEMPO SE FOR MENOR ENTRE NA LISTA
for t2 in lista_tempos_sinistros:
    if t2 <= tempo: lista_boa_sinistros.append(t2)

#APPENDA NA TABELA DE 1000 INTERAÇÕES OS CUSTOS/LUCROS RELATIVOS AO ULTIMO TEMPO QUE NAO ULTRAPASSOU
tabela_ts_r.iloc[(tempo//50)-1,k] = lista_R_t[len(lista_boa_clientes)-1]
tabela_ts_s.iloc[(tempo//50)-1,k] = lista_D_t[len(lista_boa_sinistros)-1]
tabela_ts_rs.iloc[(tempo//50)-1,k] = lista_R_t[len(lista_boa_clientes)-1] - lista_D_t[len(lista_boa_sinistros)-1]

```

#AGORA PRA CADA TEMPO CALCULA O QUANTIL DA SUA LINHA E APPENDA NA TABELA QUANTIS

```

for tempo in tempos_pedidos:

```

```

    tabela_quantis_tempo_r[(tempo//50)-1][0] = tempo
    tabela_quantis_tempo_r[(tempo//50)-1][1] = tabela_ts_r.iloc[(tempo//50)-1].quantile(0.05)
    tabela_quantis_tempo_r[(tempo//50)-1][2] = tabela_ts_r.iloc[(tempo//50)-1].quantile(0.95)
    tabela_quantis_tempo_r[(tempo//50)-1][3] = tabela_ts_r.iloc[(tempo//50)-1].mean()

```

```

    tabela_quantis_tempo_s[(tempo//50)-1][0] = tempo
    tabela_quantis_tempo_s[(tempo//50)-1][1] = tabela_ts_s.iloc[(tempo//50)-1].quantile(0.05)
    tabela_quantis_tempo_s[(tempo//50)-1][2] = tabela_ts_s.iloc[(tempo//50)-1].quantile(0.95)
    tabela_quantis_tempo_r[(tempo//50)-1][3] = tabela_ts_s.iloc[(tempo//50)-1].mean()

```

```

    tabela_quantis_tempo_rs[(tempo//50)-1][0] = tempo
    tabela_quantis_tempo_rs[(tempo//50)-1][1] = tabela_ts_rs.iloc[(tempo//50)-1].quantile(0.05)
    tabela_quantis_tempo_rs[(tempo//50)-1][2] = tabela_ts_rs.iloc[(tempo//50)-1].quantile(0.95)
    tabela_quantis_tempo_rs[(tempo//50)-1][3] = tabela_ts_rs.iloc[(tempo//50)-1].mean()

```

#TRANFORMA EM DATAFRAME E RENOMEIA AS COLUNAS

```

tabela_quantis_tempo_r = pd.DataFrame(tabela_quantis_tempo_r)

```



```
tabela_quantis_tempo_r.rename(columns={0: 't',1:'Quantil 5%',2:'Quantil 95%',3:'Média'},inplace = True)

tabela_quantis_tempo_s = pd.DataFrame(tabela_quantis_tempo_s)
tabela_quantis_tempo_s.rename(columns={0: 't',1:'Quantil 5%',2:'Quantil 95%',3:'Média'},inplace = True)

tabela_quantis_tempo_rs = pd.DataFrame(tabela_quantis_tempo_rs)
tabela_quantis_tempo_rs.rename(columns={0: 't',1:'Quantil 5%',2:'Quantil 95%',3:'Média'},inplace = True)
```

```
##### 5 SIMULAÇÕES #####
```

```
for k in range(5):
```

```
    #TEMPO LIMITE
    tempo_limite = 500
```

```
    #INSTANTES DE NOVOS CLIENTES
    t_r = 0
```

```
    #INSTANTES DE NOVOS SINISTROS
    t_s = 0
```

```
    #CONTADOR DE CLIENTES
    clientes = 0
```

```
    #VARIÁVEIS DE CUSTO
    R_t = 0
    D_t = 0
```

```
    #LISTA PRA VARIÁVEIS DE CUSTOS
    lista_R_t = []
    lista_D_t = []
```

```
    #LISTA PARA O TEMPO
    tempo_linear = [i for i in range(1,500,1)]
```

```
#LISTA PARA OS INSTANTES DE OCORRENCIAS SINISTROS E CLIENTES
tempos_clientes = []
tempos_sinistros = []

#PERCCORE ATÉ TEMPO LIMITE
while min([t_r,t_s]) <= tempo_limite:

    #SE FOR MENOS QUE O TEMPO LIMITE ENTRA
    if t_r <= tempo_limite:

        random.seed()

        #INCREMENTA CLIENTES
        clientes +=1
        #APPENDA NAS RESPECTIVAS LISTAS
        lista_R_t.append(R_t)
        tempos_clientes.append(t_r)
        #INCREMENTA INSTANTE DE NOVOS CLIENTES
        t_r += random.expovariate(2)
        #INCREMENTA GANHO
        R_t += random.uniform(100,150)

    #PERCORRE ATÉ TEMPO LIMITE
    if t_s <= tempo_limite:

        random.seed()

        valor_sinistro = random.uniform(2000,4000)

        if valor_sinistro >= 3500:
            #APPENDA NAS RESPECTIVAS LISTAS
            lista_D_t.append(D_t)
            tempos_sinistros.append(t_s)
            #INCREMENTA INSTANTES DE SINISTROS
            t_s += random.expovariate(1/30)
```

```

#INCREMENTA VALOR DOS SINISTROS
D_t += valor_sinistro

#LISTA PRAS DIFERENÇAS ENTRE S(t) e R(t)
diferencas = []

#PERCORRE TEMPO 500 DE UM EM UM
for tempo_especifico in tempo_linear:

    #LISTAS PARA GUARDAR TEMPOS QUE NÃO EXCEDEM O TEMPO ESPECIFICO
    lista_boa_clientes = []
    lista_boa_sinistros = []

    #PRA CADA TEMPO SE FOR MENOR ENTRE NA LISTA
    for t in tempos_clientes:
        if t <= tempo_especifico: lista_boa_clientes.append(t)

    #PRA CADA TEMPO SE FOR MENOR ENTRE NA LISTA
    for tt in tempos_sinistros:
        if tt <= tempo_especifico: lista_boa_sinistros.append(tt)

    #CALCULA DIF E APPENDA NA LISTA
    parte_r = lista_R_t[len(lista_boa_clientes)-1]
    parte_s = lista_D_t[len(lista_boa_sinistros)-1]
    diferencas.append(parte_r - parte_s)

#PRA CADA SIMULAÇÃO PLOTA A TRAJETÓRIA
plt.plot([tempo for tempo in range(1,500,1)],diferencas, label = f'simulação {k+1}')

#GRAFICO DOS QUANTIS 95 E 5 DA DIFERENÇA
plt.plot([i for i in range(50,550,50)],tabela_quantis_tempo_rs['Quantil 5%'], color = 'black', label = 'Quantil 5%')
plt.plot([i for i in range(50,550,50)],tabela_quantis_tempo_rs['Quantil 95%'], color = 'black', label = 'Quantil 95%')

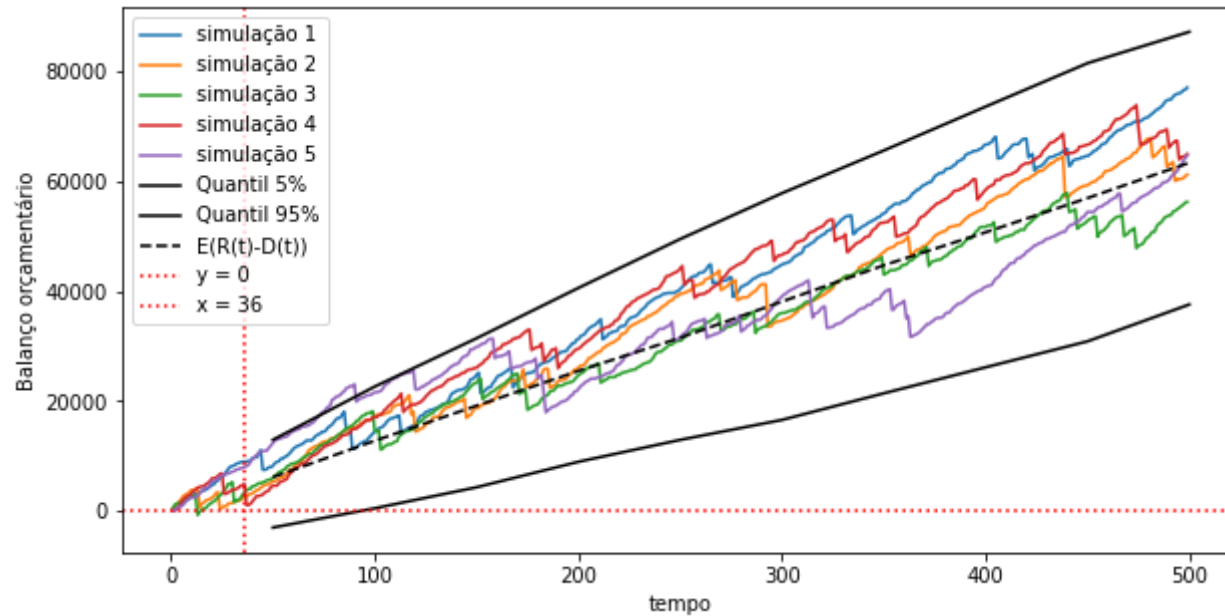
#GRÁFICO DA ESPERANÇA DA DIFERENÇA

```

```

plt.plot([i for i in range(50,550,50)],tabela_quantis_tempo_rs['Média'], color = 'black', label = 'E(R(t)-D(t))',linestyle = '--')
plt.rcParams["figure.figsize"] = (10,5)
plt.axhline(y = 0, xmin = 0, xmax = 500,linestyle = ':', color = 'red', label = 'y = 0')
plt.axvline(x= 36, ymin= 0, ymax=20000, linestyle = ':', color = 'red', label = 'x = 36')
plt.legend()
plt.ylabel('Balanco orçamentário')
plt.xlabel('tempo')
plt.show()

```



✓ 0s completed at 3:27 PM

