In [50]:

```
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt

import warnings
warnings.filterwarnings('ignore')
```

In [3]:

```
filepath = './cases-brazil-states.csv'

brasil_data = pd.read_csv(filepath)

brasil_data.head()
```

Out[3]:

	epi_week	date	country	state	city	newDeaths	deaths	newCases	totalCases	deat
0	9	2020-02-25	Brazil	SP	TOTAL	0	0	1	1	
1	9	2020-02-25	Brazil	TOTAL	TOTAL	0	0	1	1	
2	9	2020-02-26	Brazil	SP	TOTAL	0	0	0	1	
3	9	2020-02-26	Brazil	TOTAL	TOTAL	0	0	0	1	
4	9	2020-02-27	Brazil	SP	TOTAL	0	0	0	1	

5 rows × 26 columns

In [6]:

```
brasil_data["state"].unique()
```

Out[6]:

```
array(['SP', 'TOTAL', 'RJ', 'BA', 'ES', 'DF', 'AL', 'MG', 'RS', 'GO',
       'PE', 'PR', 'RN', 'SC', 'AM', 'MS', 'SE', 'CE', 'AC', 'PA', 'PB',
       'TO', 'PI', 'AP', 'MA', 'MT', 'RO', 'RR'], dtype=object)
```

In [22]:

```
df = brasil_data[brasil_data["state"] == "GO"]

df.head()
```

Out[22]:

	epi_week	date	country	state	city	newDeaths	deaths	newCases	totalCases	dea
70	11	2020-03-12	Brazil	GO	TOTAL	0	0	3	3	
85	11	2020-03-13	Brazil	GO	TOTAL	0	0	0	3	
100	11	2020-03-14	Brazil	GO	TOTAL	0	0	1	4	
117	12	2020-03-15	Brazil	GO	TOTAL	0	0	0	4	
135	12	2020-03-16	Brazil	GO	TOTAL	0	0	5	9	

5 rows × 26 columns

In [180…

```
df[["recovered"]].iloc[100:140]
```

Out[180…

	recovered
date_fmt	
2020-06-20	2686.0
2020-06-21	2686.0
2020-06-22	2686.0
2020-06-23	2686.0
2020-06-24	2686.0
2020-06-25	4170.0
2020-06-26	4170.0
2020-06-27	4170.0
2020-06-28	4170.0
2020-06-29	4170.0
2020-06-30	4170.0
2020-07-01	4170.0
2020-07-02	6073.0
2020-07-03	6073.0
2020-07-04	6073.0
2020-07-05	6073.0
2020-07-06	6073.0
2020-07-07	6073.0
2020-07-08	6073.0
2020-07-09	9598.0
2020-07-10	10223.0
2020-07-11	10407.0
2020-07-12	10480.0
2020-07-13	11011.0
2020-07-14	11521.0
2020-07-15	12119.0
2020-07-16	12427.0
2020-07-17	12516.0
2020-07-18	12513.0
2020-07-19	12513.0
2020-07-20	12934.0
2020-07-21	14815.0
2020-07-22	15951.0
2020-07-23	44774.0

	recovered
date_fmt	
2020-07-24	47069.0
2020-07-25	49001.0
2020-07-26	49998.0
2020-07-27	52262.0
2020-07-28	54425.0
2020-07-29	56413.0

In [185…

```
df[["totalCases" , "totalCases_per_100k_inhabitants"]].iloc[100:140]
```

Out[185…

	totalCases	totalCases_per_100k_inhabitants
date_fmt		
2020-06-20	15450	220.13709
2020-06-21	15904	226.60584
2020-06-22	17085	243.43315
2020-06-23	18014	256.66987
2020-06-24	19280	274.70829
2020-06-25	20377	290.33873
2020-06-26	21441	305.49898
2020-06-27	21865	311.54028
2020-06-28	22165	315.81479
2020-06-29	23379	333.11229
2020-06-30	24910	354.92653
2020-07-01	26145	372.52324
2020-07-02	26318	374.98821
2020-07-03	27502	391.85826
2020-07-04	28526	406.44858
2020-07-05	28526	406.44858
2020-07-06	30187	430.11510
2020-07-07	32001	455.96161
2020-07-08	33367	475.42486
2020-07-09	34627	493.37779
2020-07-10	35706	508.75177
2020-07-11	36273	516.83058
2020-07-12	36556	520.86287
2020-07-13	37333	531.93384
2020-07-14	38299	545.69775
2020-07-15	39474	562.43957

	totalCases	totalCases_per_100k_inhabitants
date_fmt		
2020-07-16	40544	577.68531
2020-07-17	40718	580.16452
2020-07-18	40795	581.26165
2020-07-19	40782	581.07642
2020-07-20	41433	590.35210
2020-07-21	45254	644.79506
2020-07-22	48682	693.63842
2020-07-23	51423	732.69316
2020-07-24	54154	771.60542
2020-07-25	55567	791.73835
2020-07-26	56250	801.46997
2020-07-27	59043	841.26563
2020-07-28	61613	877.88390
2020-07-29	64250	915.45682

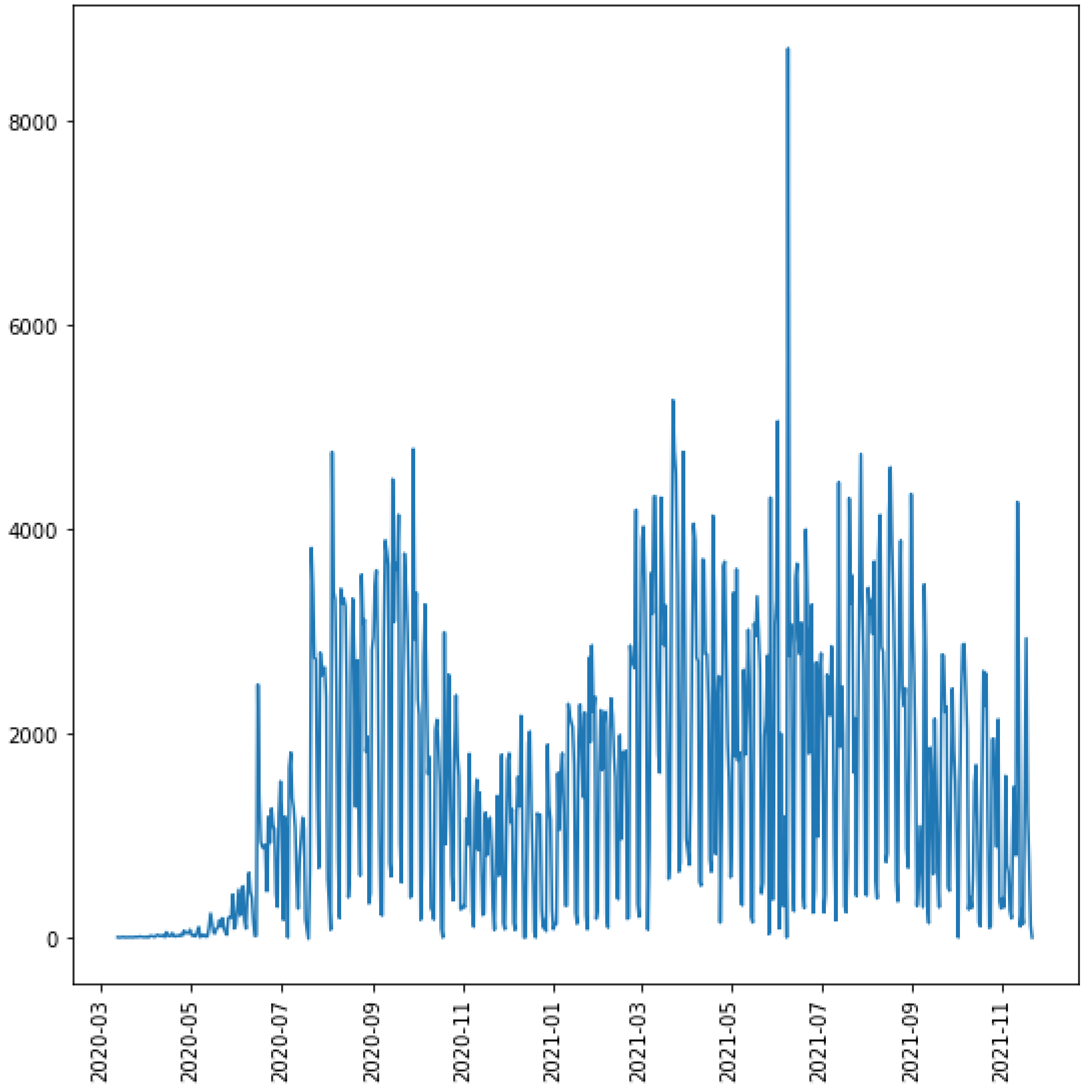
```
In [51]: df["date_fmt"] = df["date"].apply(pd.to_datetime);
```

```
In [55]: df = df.set_index("date_fmt")
```

```
In [ ]: df.
```

```
In [69]: plt.figure(figsize=(9, 9))

         plt.xticks(rotation=90)
         plt.plot(df["newCases"])
```

```
Out[69]: [<matplotlib.lines.Line2D at 0x7f9815d62370>]
```



In [77]:

```
df.columns
```

Out[77]:

```
Index(['epi_week', 'date', 'country', 'state', 'city', 'newDeaths', 'deaths',
       'newCases', 'totalCases', 'deathsMS', 'totalCasesMS',
       'deaths_per_100k_inhabitants', 'totalCases_per_100k_inhabitants',
       'deaths_by_totalCases', 'recovered', 'suspects', 'tests',
       'tests_per_100k_inhabitants', 'vaccinated',
       'vaccinated_per_100_inhabitants', 'vaccinated_second',
       'vaccinated_second_per_100_inhabitants', 'vaccinated_single',
       'vaccinated_single_per_100_inhabitants', 'vaccinated_third',
       'vaccinated_third_per_100_inhabitants'],
      dtype='object')
```

In [100…

```
a = df.groupby(["epi_week"]).agg({
    "newCases": "mean",
    "newDeaths": "mean",
    "recovered": "max",
    "totalCases": "max"
})

a
```

Out[100…

	newCases	newDeaths	recovered	totalCases
epi_week				
11	1.333333	0.000000	NaN	4

	newCases	newDeaths	recovered	totalCases
epi_week				
12	2.285714	0.000000	NaN	20
13	5.142857	0.142857	NaN	56
14	6.714286	0.142857	NaN	103
15	15.142857	1.142857	NaN	209
...	...	...	...	...
143	1276.000000	18.285714	873149.0	904909
144	595.571429	16.571429	874557.0	909078
145	1484.857143	11.428571	874557.0	919472
146	993.857143	3.857143	874557.0	926429
147	0.000000	0.000000	874557.0	926429

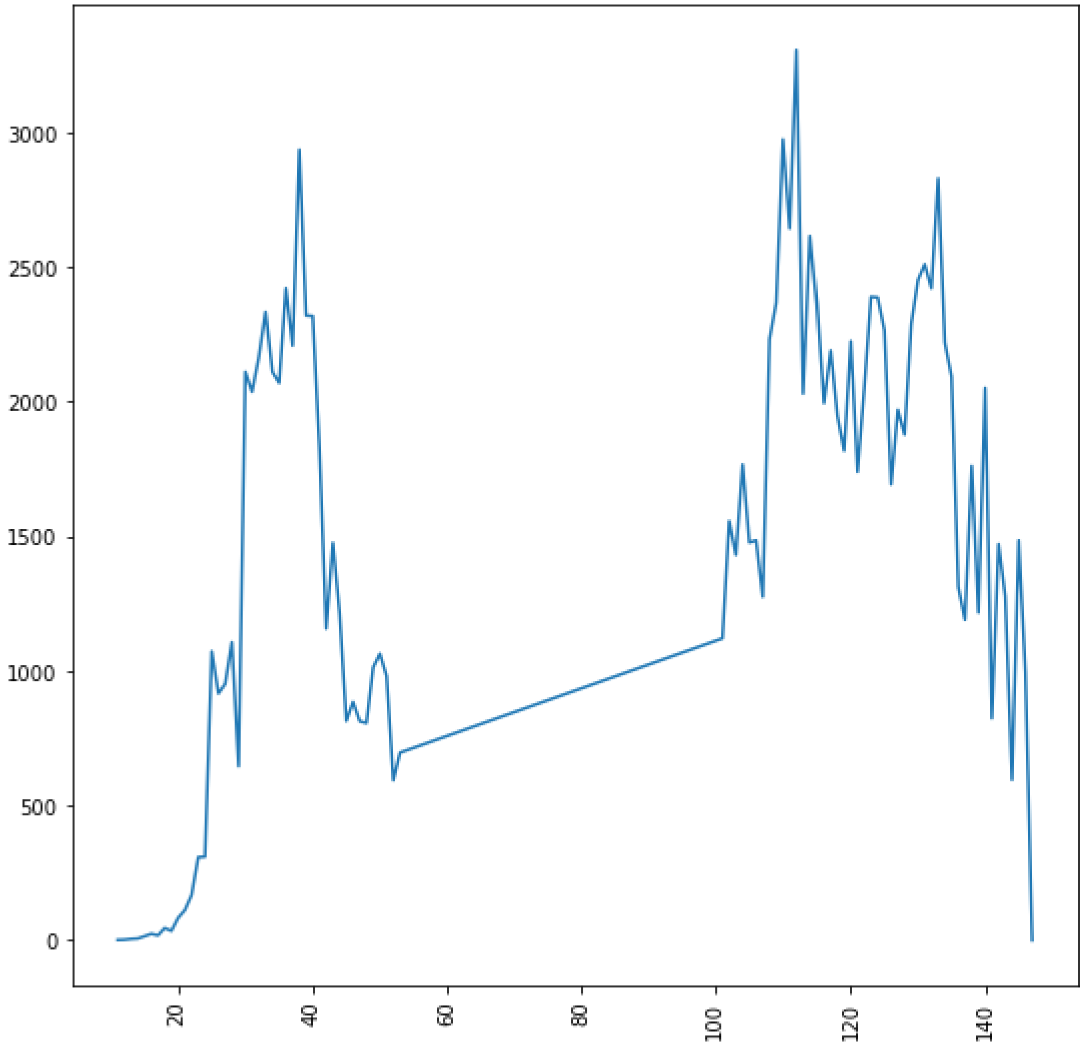
90 rows × 4 columns

In [74]:

```
plt.figure(figsize=(9, 9))

plt.xticks(rotation=90)
plt.plot(a["newCases"])

# 0 - 40
```

Out[74]:

```
[<matplotlib.lines.Line2D at 0x7f9816380460>]
```



In [ ]:

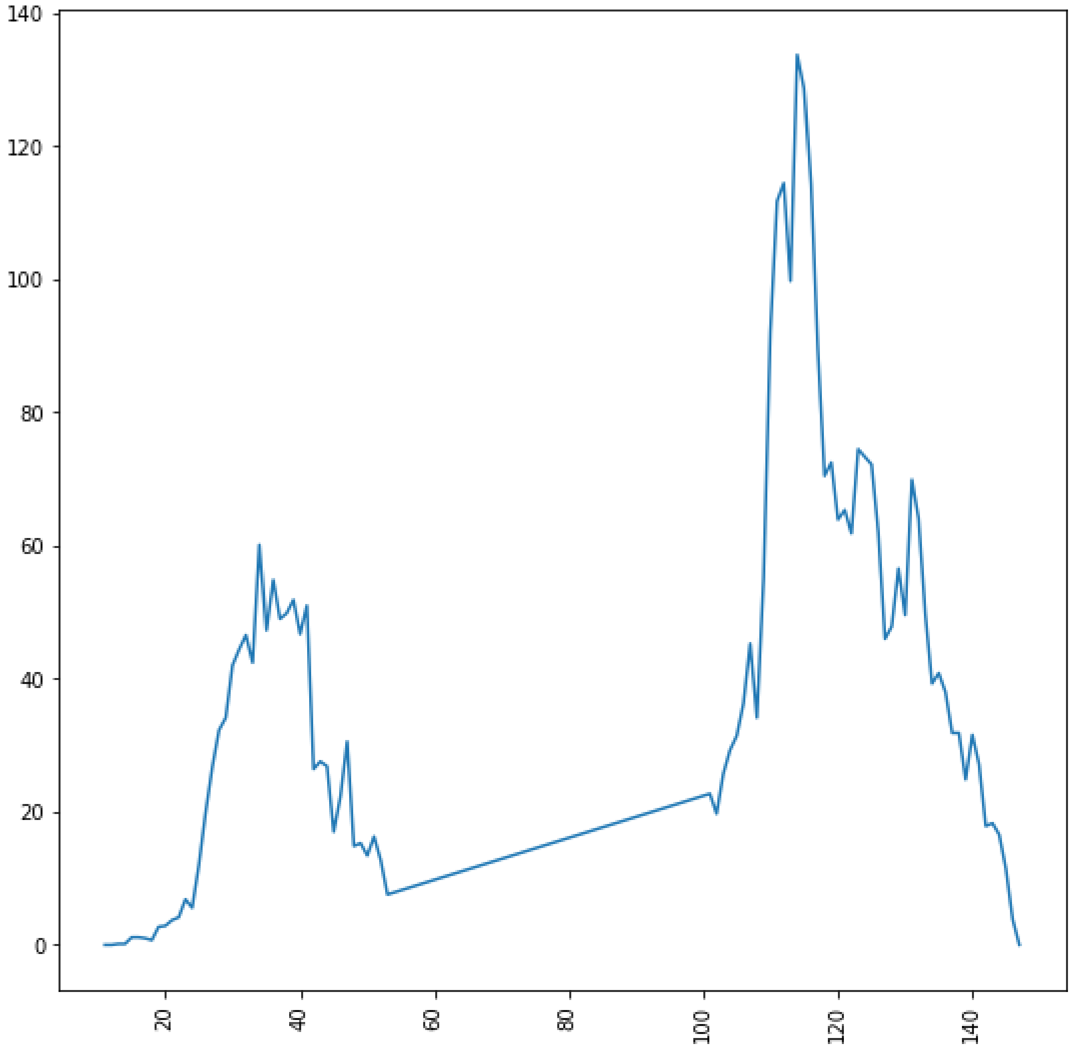
```
```

In [76]:

```
plt.figure(figsize=(9, 9))

plt.xticks(rotation=90)
plt.plot(a["newDeaths"])
```

Out[76]:

```
[<matplotlib.lines.Line2D at 0x7f98167bd2e0>]
```



In [ ]:

In [91]:

```
POP = 6523000

S = 6523000 # 6,523 milhões (População de Goias)
I = 1.33    # semana 11
R = 0       # nossos dados começam na EPI 21


S, I, R
```

Out[91]:

```
(6523000, 1.33, 0)
```

In [89]:

```
# k ∈ [1/15,1/5] e b ∈ [0,5].

b = 0.998 # taxa de contatio
k = 1/14 # tempo de recuperacao

b, k
```

Out[89]:

```
(0.998, 0.07142857142857142)
```

In [93]:

```
s = S / POP
i = I / POP
r = R / POP
```

```
s, i, r
```

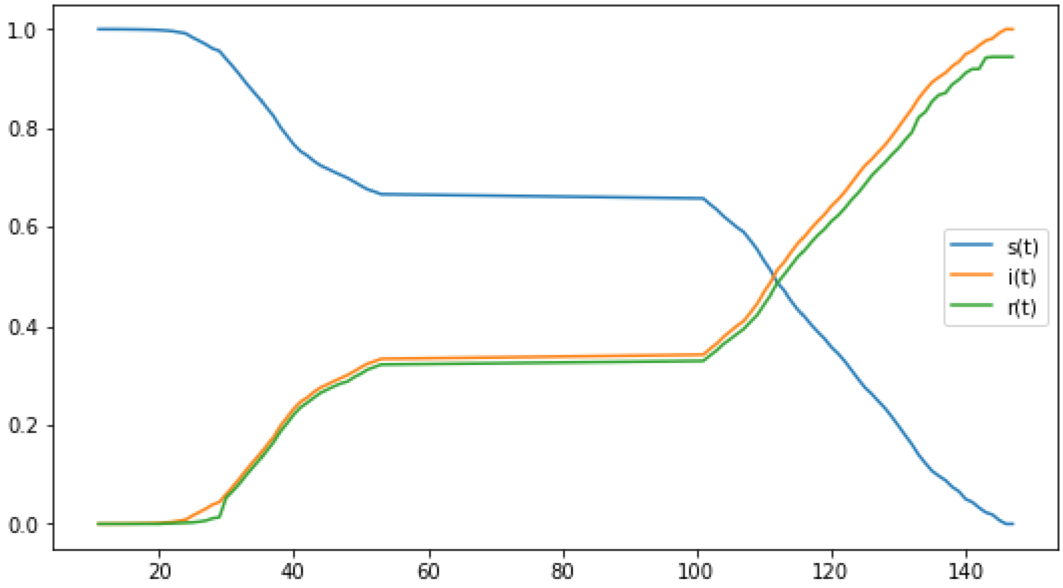
Out[93]:
```
(1.0, 2.0389391384332365e-07, 0.0)
```

In [217…
```
# S + I + R = N
# S = N - I - R
# I =

POP = 926429
#POP = 6523000 // 2


a["R"] = a["recovered"].fillna(0)
a["I"] = a["totalCases"]
a["S"] = POP - a["totalCases"]

a["s"] = a["S"] / POP
a["i"] = a["I"] / POP
a["r"] = a["R"] / POP

a
```

Out[217…
```
             newCases  newDeaths  recovered  totalCases       I         R       S
epi_week
      11     1.333333   0.000000        NaN           4       4       0.0  926425  0.9999
      12     2.285714   0.000000        NaN          20      20       0.0  926409  0.9999
      13     5.142857   0.142857        NaN          56      56       0.0  926373  0.9999
      14     6.714286   0.142857        NaN         103     103       0.0  926326  0.9998
      15    15.142857   1.142857        NaN         209     209       0.0  926220  0.9997
     ...          ...        ...        ...         ...     ...       ...     ...
     143  1276.000000  18.285714   873149.0      904909  904909  873149.0   21520  0.0232
     144   595.571429  16.571429   874557.0      909078  909078  874557.0   17351  0.0187
     145  1484.857143  11.428571   874557.0      919472  919472  874557.0    6957  0.0075
     146   993.857143   3.857143   874557.0      926429  926429  874557.0       0  0.0000
     147     0.000000   0.000000   874557.0      926429  926429  874557.0       0  0.0000

90 rows × 10 columns
```

In [218…
```
plt.figure(figsize=(9, 5))


plt.plot(a["s"], label='s(t)')
plt.plot(a["i"], label='i(t)')
plt.plot(a["r"], label='r(t)')


plt.legend()
```

Out[218…
```
<matplotlib.legend.Legend at 0x7f98199c5940>
```



In [206…

```
b = 0.998 # taxa de contatio
k = 1/14 # tempo de recuperacao


i = 0.001
s = 1 - i

dt = 1

print(f"s={s}, i={i}")

dados = []

for t in range(100):
    st, it = -b*s*i, (b*s - k)*i

    s, i = s + st*dt, i + it*dt

    #print(f"s={s}, i={i}")

    dados.append((s,i))


dados = np.array(dados)
```

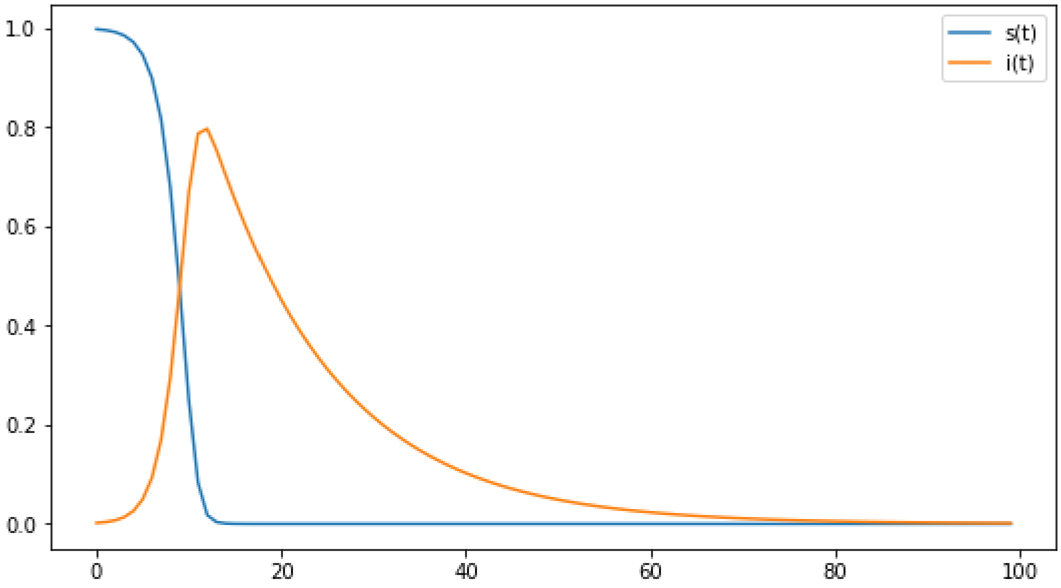
```
s=0.999, i=0.001
```

In [207…

```
plt.figure(figsize=(9, 5))


plt.plot(dados[:,0], label='s(t)')
plt.plot(dados[:,1], label='i(t)')

plt.legend()
```

Out[207…

```
<matplotlib.legend.Legend at 0x7f9819357580>
```



In [ ]:

In [233…

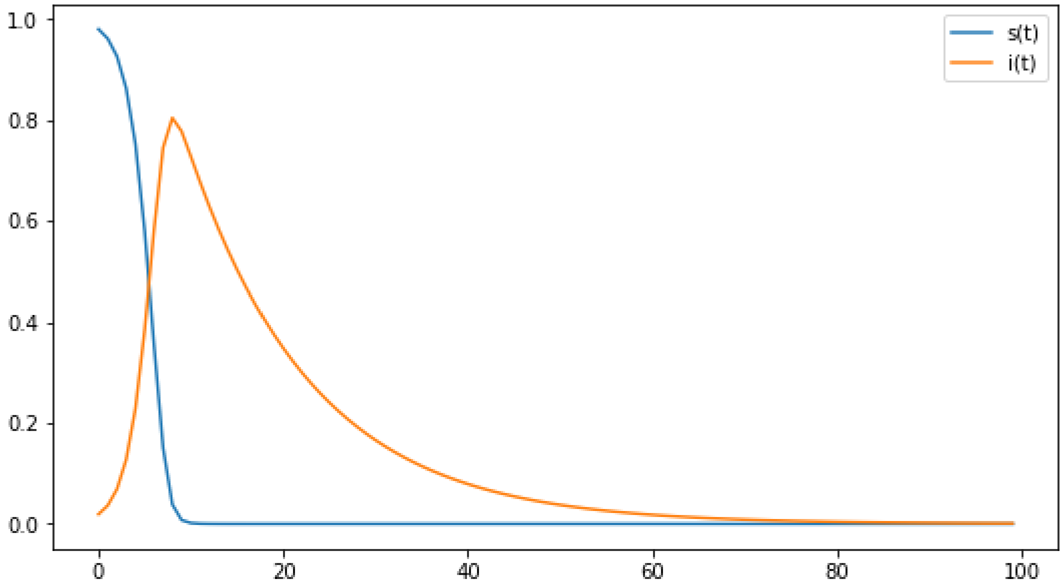
```
def predizer(b = 0.998, k = 1/14, i = 0.001, dt = 1, steps = 100):
    s = 1 - i

    dt = 1

    dados = []

    for t in range(steps):
        st, it = -b*s*i, (b*s - k)*i

        s, i = s + st*dt, i + it*dt

        dados.append((s,i))

    dados = np.array(dados)

    return dados
```

In [235…

```
def montar_grafico(b = 0.998, k = 1/14, i = 0.001, steps = 100):
    dados = predizer(b, k, i, steps=steps)

    plt.figure(figsize=(9, 5))

    plt.plot(dados[:,0], label='s(t)')
    plt.plot(dados[:,1], label='i(t)')

    plt.legend();
```

In [236…

```
montar_grafico(i = 0.01)
```



In [237…

```
a["i"]
```

Out[237…

```
epi_week
11     0.000004
12     0.000022
13     0.000060
14     0.000111
15     0.000226
         ...   
143    0.976771
144    0.981271
145    0.992491
146    1.000000
147    1.000000
Name: i, Length: 90, dtype: float64
```

In [242…

```
dados = predizer(b = 0.998, k = 1/14, i = 0.000004, steps = 90)
```

In [ ]:

In [243…

```
# ∑t(i(t)−dsi(t))2

eq = np.sum((a["i"] - dados[:,1])**2)
eq
```

Out[243…

```
26.452089577392563
```

In [244…

```
def calc_erro_quadratico(b, k, i):
    dados = predizer(b, k, i, steps = 90)
    eq = np.sum((a["i"] - dados[:,1])**2)

    return eq
```

In [ ]:

In [245…

```
calc_erro_quadratico(b = 0.998, k = 1/14, i = 0.000004)
```

```
Out[245… 26.452089577392563

In [247… 1/15, 1/5 # k

Out[247… (0.06666666666666667, 0.2)

In [248… 0, 5 # b

Out[248… (0, 5)

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:
```