Prática1_respostas

July 15, 2020

1 Prática 1

Aprendizado Dinâmico

por **Cibele Russo** (ICMC/USP - São Carlos SP)

MBA em Ciências de Dados

Considere os dados de COVID-19 para a cidade de São Paulo. Nesta prática, aplicaremos os conhecimentos adquiridos na Aula 1, fazendo:

1. Visualização de dados completos e parciais
2. Construiremos gráficos da média móvel simples e exponencialmente ponderada, para casos e para mortes separadamente.
3. Faremos a decomposição em tendência e sazonalidade

1.1 Exercício 1:

1. Leia os dados de COVID-19 da base covid_caso.csv.
2. Considere os dados de casos e mortes diárias (diferenças). Salve os dados da cidade de São Paulo num arquivo covidSaoPaulo.csv.
3. Produza um gráfico da séries de casos e mortes diários da cidade de São Paulo desde o primeiro caso.
4. Em seguida, considere o gráfico do log de casos a partir somente do centésimo caso.

[1]:
```
# Bibliotecas

import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

%matplotlib inline
```

1. Faça a leitura dos dados a partir do arquivo covid_caso.csv disponível no material do curso.

[2]:
```
pkgdir = '/home/cibele/CibelePython/AprendizadoDinamico/Data'
```

```
# COVID - Leitura dos dados
covid = pd.read_csv(f'{pkgdir}/covid_caso.csv', index_col='date',␣
 ↪parse_dates=True)
covid.head()
```

[2]:

date	state	city	place_type	confirmed	deaths	order_for_place	is_last	estimated_population_2019	city_ibge_code	confirmed_per_100k_inhabitants	death_rate
2020-07-11	AP	NaN	state	31279	473	113	True	845731.0	16.0	3698.45731	0.0151
2020-07-10	AP	NaN	state	31080	470	112	False	845731.0	16.0	3674.92737	0.0151
2020-07-09	AP	NaN	state	30763	467	111	False	845731.0	16.0	3637.44500	0.0152
2020-07-08	AP	NaN	state	30524	462	110	False	845731.0	16.0	3609.18543	0.0151
2020-07-07	AP	NaN	state	30294	455	109	False	845731.0	16.0	3581.99002	0.0150

Considere apenas os dados de casos e óbitos diários da cidade de São Paulo. Tome as diferenças para obter dados diários.

[3]:
```
covid = pd.read_csv(f'{pkgdir}/covid_caso.csv', index_col=0, parse_dates=True)

covid = covid.loc[(covid['city']=='São␣
 ↪Paulo')&(covid['place_type']=='city'),['confirmed','deaths']]

covid = covid.sort_values(by=['date'])

covid.head()
```

[3]:

date	confirmed	deaths
2020-02-25	1	0
2020-02-26	1	0
2020-02-27	1	0
2020-02-28	2	0

```
2020-02-29          2       0
```

[4]:
```
y1 = list(np.diff(covid['confirmed']))
y2 = list(np.diff(covid['deaths']))
x = covid.index[1:] # Note o subconjunto de dados em x iniciando em 1 pois␣
 ↪interessa as diferenças

covid = pd.DataFrame({'date':x,'confirmed':y1,'deaths':y2})

covid = covid.set_index('date') # Estabelece o índice

covid.head()
```

[4]:
```
            confirmed  deaths
date
2020-02-26          0       0
2020-02-27          0       0
2020-02-28          1       0
2020-02-29          0       0
2020-03-01          0       0
```

Salve os dados de São Paulo num arquivo covidSaoPaulo.csv.

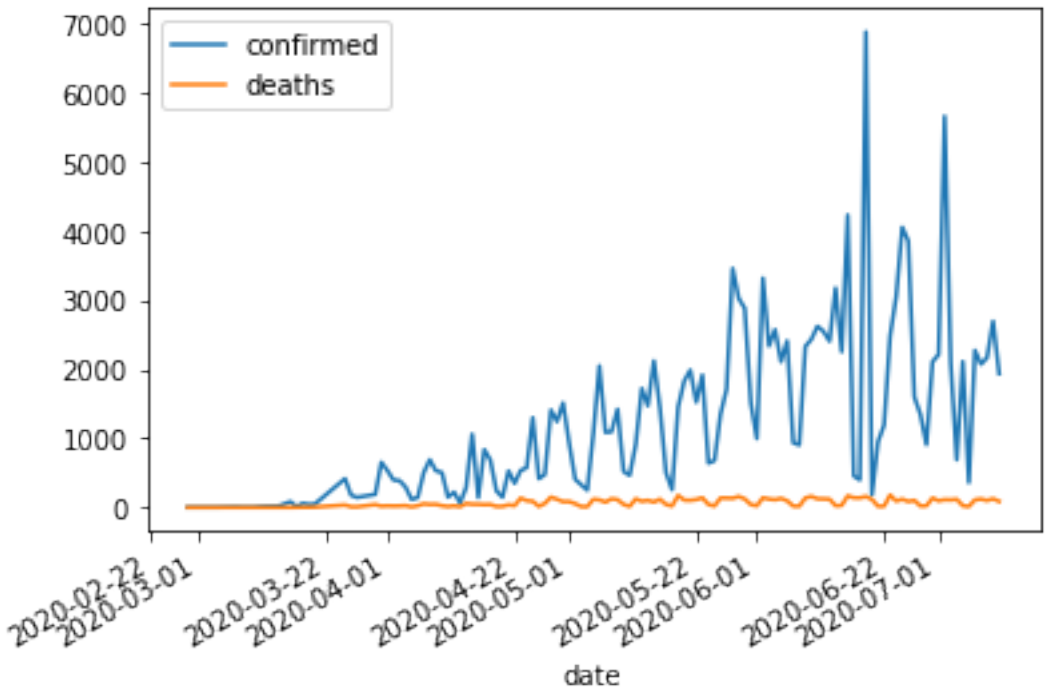
[5]:
```
covid.to_csv('covidSaoPaulo.csv')
```

Produza um gráfico da séries de casos e mortes diários da cidade de São Paulo desde o primeiro caso.

[6]:
```
covid[covid['confirmed']>0].head(3)
```

[6]:
```
            confirmed  deaths
date
2020-02-28          1       0
2020-03-04          1       0
2020-03-05          3       0
```

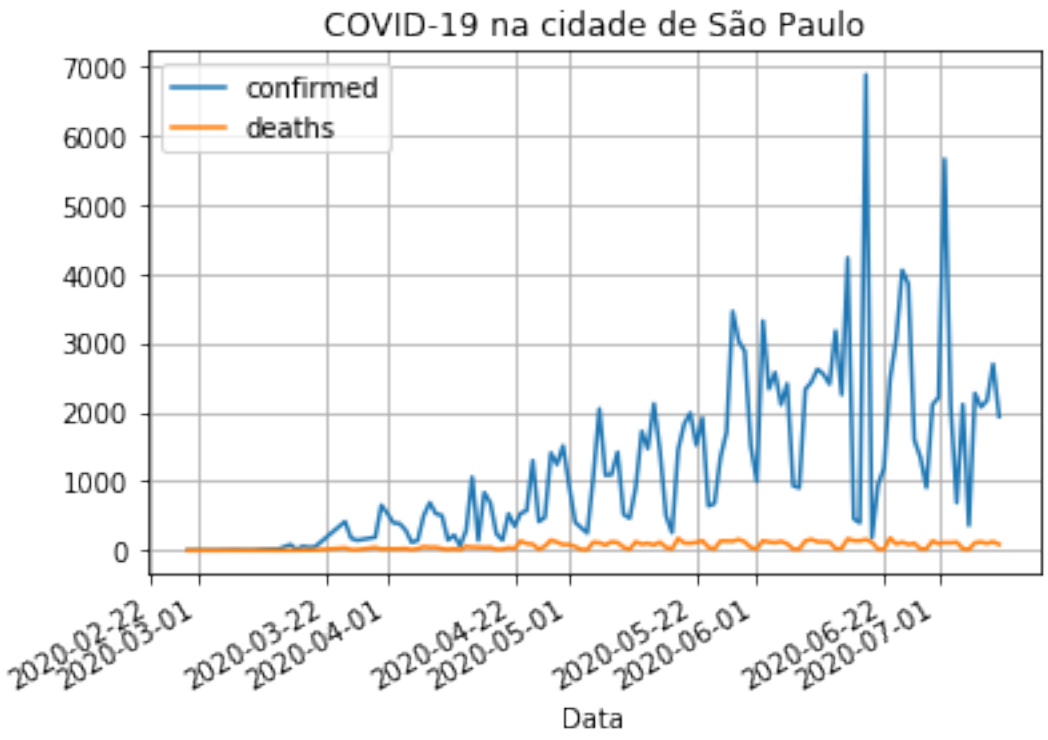
[7]:
```
covid[covid['confirmed']>0].plot();
```



[8]:
```
title = 'COVID-19 na cidade de São Paulo'
ylabel = ''
xlabel = 'Data'

ax=covid[covid['confirmed']>0].plot(title=title);
ax.autoscale(axis='both');
ax.set(xlabel=xlabel,ylabel=ylabel);

ax.xaxis.grid(True) # Com grades
ax.yaxis.grid(True)
```

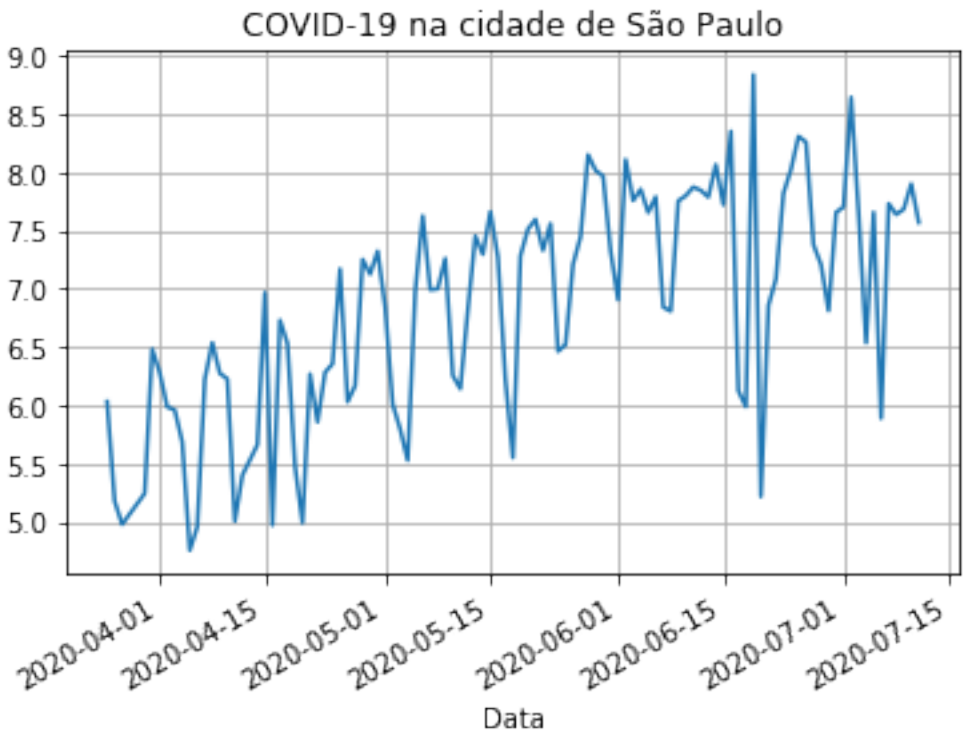


4. Considere o gráfico do log de casos a partir somente do centésimo caso.

[9]:
```
title = 'COVID-19 na cidade de São Paulo'
ylabel = ''
xlabel = 'Data'

ax=np.log(covid[covid['confirmed']>100]['confirmed']).plot(title=title);
ax.autoscale(axis='both');
ax.set(xlabel=xlabel,ylabel=ylabel);

ax.xaxis.grid(True) # Com grades
ax.yaxis.grid(True)
```



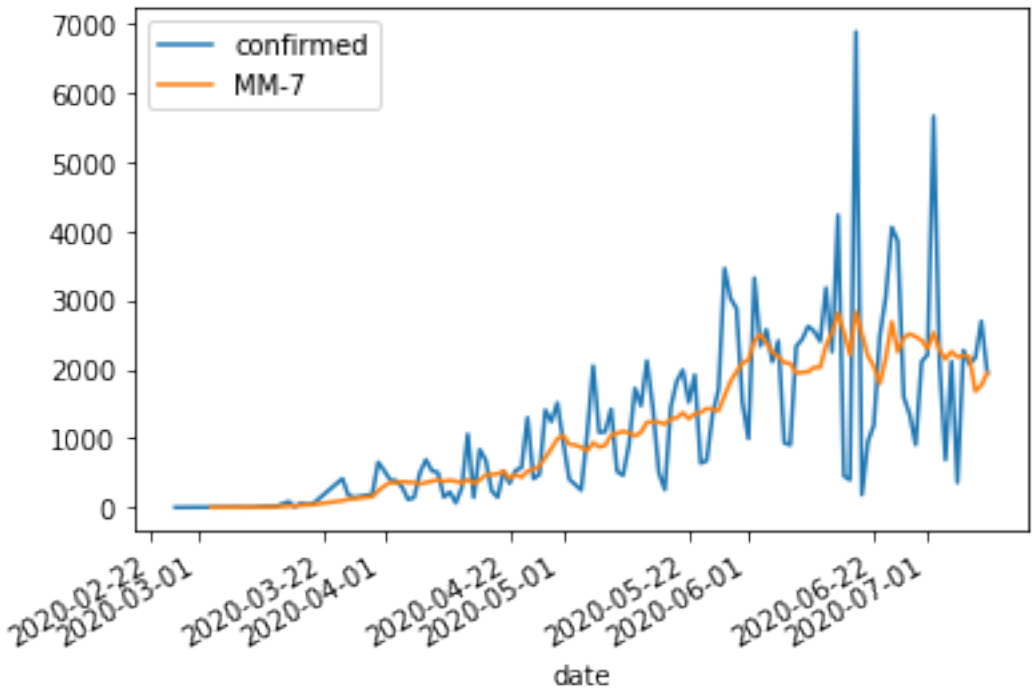
1.2 Exercício 2

1. Construa gráficos da média móvel simples e exponencialmente ponderada para casos. Utilize janela de 7 dias para a MMS e span=7 para MMEP.
2. Construa gráficos da média móvel simples e exponencialmente ponderada para mortes. Utilize janela de 7 dias para a MMS e span=7 para MMEP.

Construa gráficos da média móvel simples e exponencialmente ponderada para a variável casos.

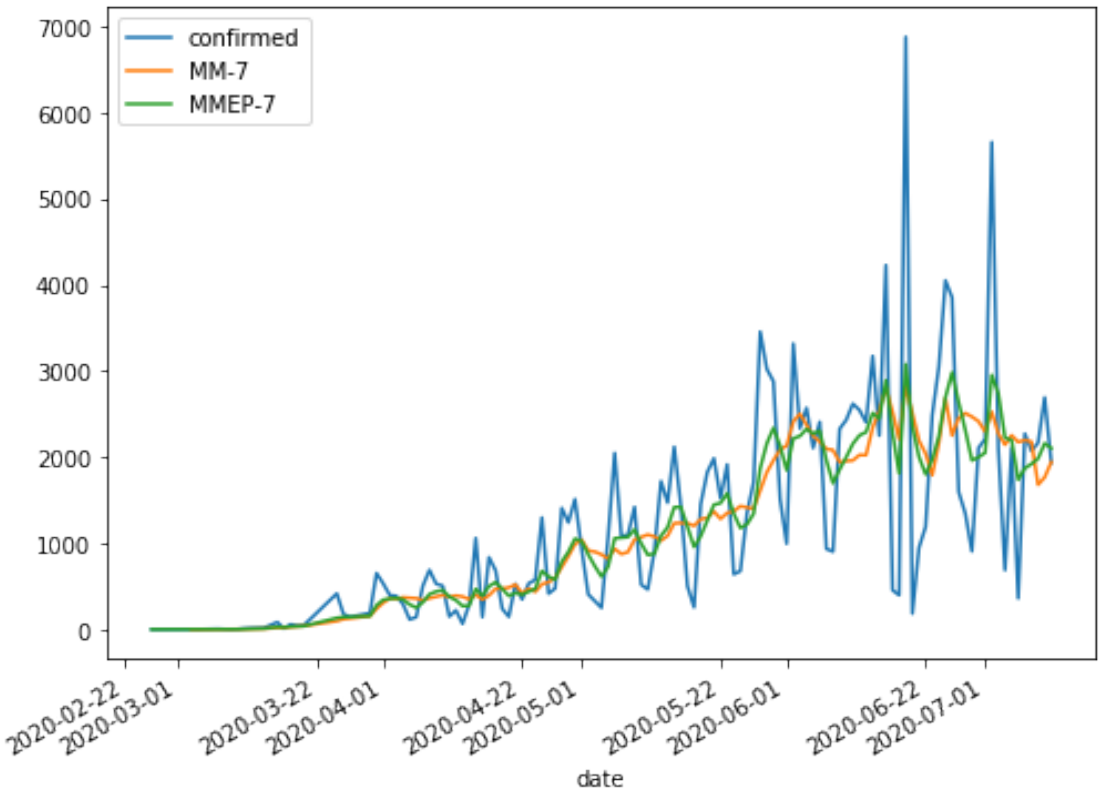
[10]:
```
covid['MM-7'] = covid['confirmed'].rolling(window=7).mean()
```

[11]:
```
covid[['confirmed','MM-7']].plot();
```



[12]:
```
covid['MMEP-7'] = covid['confirmed'].ewm(span=7,adjust=False).mean()
```

[13]:
```
covid[['confirmed','MM-7','MMEP-7']].plot(figsize=(8,6));
```



Construa gráficos da média móvel simples e exponencialmente ponderada para a variável mortes (deaths).

[ ]:

[ ]:

[ ]:

[ ]:

1.3 Exercício 3

Faça a decomposição em tendência e sazonalidade do número de casos utilizando a função seasonal_decompose do módulo statsmodels.

[14]:
```
covidSP = pd.read_csv('covidSaoPaulo.csv', index_col='date')

covidSP.index = pd.DatetimeIndex(covidSP.index)

covidSP.head()
```

[14]:
```
            confirmed  deaths
date
2020-02-26          0       0
2020-02-27          0       0
2020-02-28          1       0
2020-02-29          0       0
2020-03-01          0       0
```

[15]:
```
covidSP.index
```

[15]:
```
DatetimeIndex(['2020-02-26', '2020-02-27', '2020-02-28', '2020-02-29',
               '2020-03-01', '2020-03-02', '2020-03-03', '2020-03-04',
               '2020-03-05', '2020-03-07',
               ...
               '2020-07-02', '2020-07-03', '2020-07-04', '2020-07-05',
               '2020-07-06', '2020-07-07', '2020-07-08', '2020-07-09',
               '2020-07-10', '2020-07-11'],
              dtype='datetime64[ns]', name='date', length=128, freq=None)
```

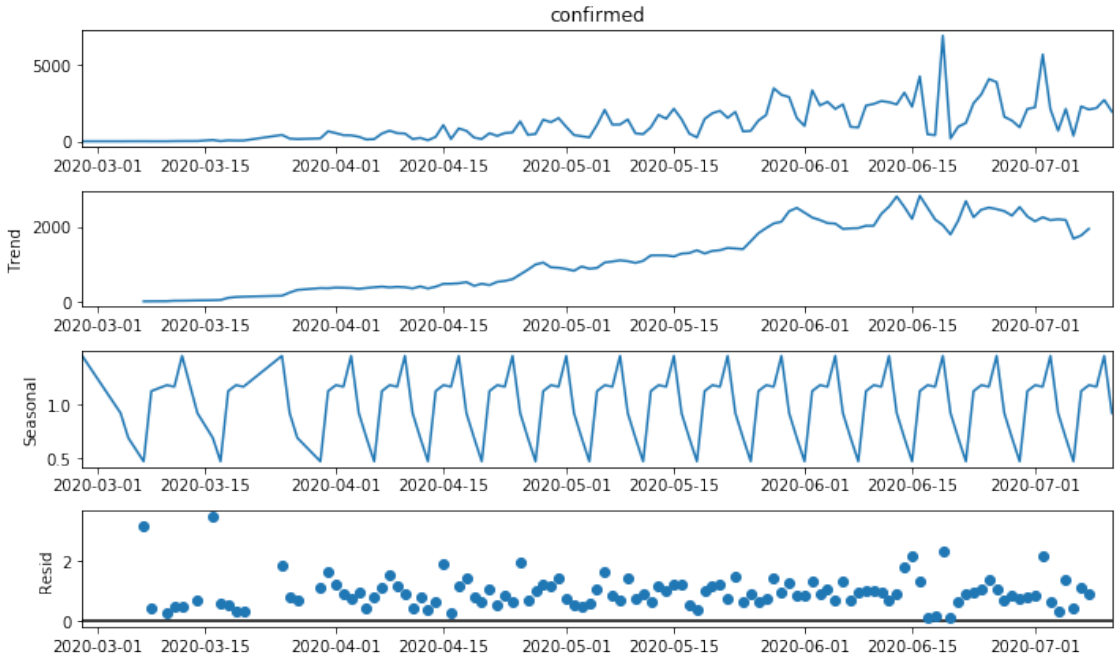
[18]:
```
import pandas as pd
from statsmodels.tsa.seasonal import seasonal_decompose
from pylab import rcParams

result = seasonal_decompose(covidSP[covidSP['confirmed']>0]['confirmed'],␣
 ↪model='multiplicative', period=7)

rcParams['figure.figsize'] = 10, 6

fig = result.plot()

plt.show()
```

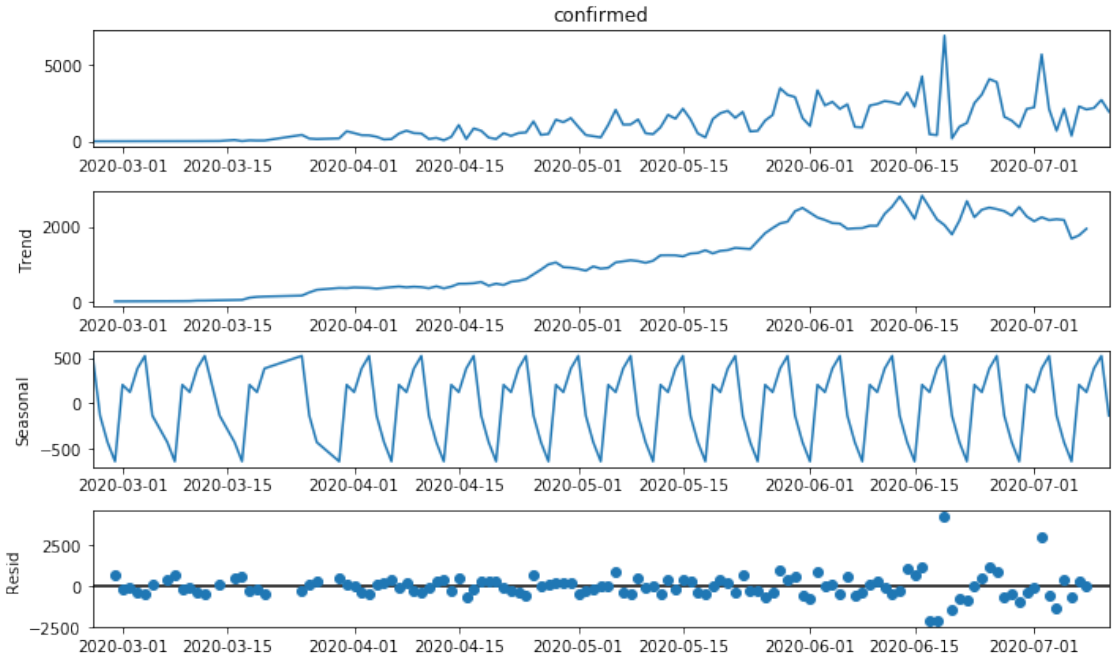


[19]:
```
result = seasonal_decompose(covidSP['confirmed'], model='additive', period=7)

rcParams['figure.figsize'] = 10,6

fig = result.plot()

plt.show()
```



Na sua opinião, qual o tipo de sazonalidade mais adequada, aditiva ou multiplicativa? Justifique com base nos resíduos.

R: Embora haja pontos atípicos nos gráficos de resíduos de ambos os casos, a sazonalidade multiplicativa parece mais adequada a esses dados.