Análises COVID-19

Primeiramente será analisado as séries temporais sobre a contaminação do vírus COVID-19 pelo mundo.

```
import pandas as pd
import numpy as np
from datetime import datetime
import plotly.express as px
import plotly.graph_objects as go
```

Depois é feita a importação dos dados. É importante dizer no comando `pd.read_csv` quais são as colunas que serão "parseadas" como datas.

```
url = 'https://github.com/neylsoncrepalde/projeto_eda_covid/blob/master/covid_19_data.csv?raw

df = pd.read_csv(url, parse_dates=['ObservationDate', 'Last Update'])
df
```

	SNo	ObservationDate	Province/State	Country/Region	Last Update	Confirmed
0	1	2020-01-22	Anhui	Mainland China	2020-01-22 17:00:00	1.0
1	2	2020-01-22	Beijing	Mainland China	2020-01-22 17:00:00	14.0
2	3	2020-01-22	Chongqing	Mainland China	2020-01-22 17:00:00	6.0
3	4	2020-01-22	Fujian	Mainland China	2020-01-22 17:00:00	1.0
4	5	2020-01-22	Gansu	Mainland China	2020-01-22 17:00:00	0.0
...	...	...	...	...	...	...

Depois é feita a verificação dos tipos das variáveis e analise das colunas se foram corretamente importadas.

```
df.dtypes
```

```
SNo                        int64
ObservationDate    datetime64[ns]
Province/State             object
Country/Region             object
Last Update        datetime64[ns]
Confirmed                 float64
Deaths                    float64
Recovered                 float64
dtype: object
```

Nomes de colunas não devem ter letras maiúsculas e nem caracteres especiais, logo é utilizada uma função para corrigir os nomes das colunas.

```
import re
def corrige_colunas(col_name):
    return re.sub(r"[/| ]", "", col_name).lower()
```

```
df.columns = [corrige_colunas(col) for col in df.columns]
```

```
df
```

	sno	observationdate	provincestate	countryregion	lastupdate	confirmed	dea
0	1	2020-01-22	Anhui	Mainland China	2020-01-22 17:00:00	1.0	
1	2	2020-01-22	Beijing	Mainland China	2020-01-22 17:00:00	14.0	
2	3	2020-01-22	Chongqing	Mainland China	2020-01-22 17:00:00	6.0	
3	4	2020-01-22	Fujian	Mainland China	2020-01-22 17:00:00	1.0	
4	5	2020-01-22	Gansu	Mainland China	2020-01-22 17:00:00	0.0	
...	...	...	...	...	...	...	
26708	26709	2020-05-19	Wyoming	US	2020-05-20 02:32:19	776.0	
26709	26710	2020-05-19	Xinjiang	Mainland China	2020-05-20 02:32:19	76.0	

Análises

É então investigado as variáveis que temos à disposição. Sabemos que trata-se de séries temporais que estão divididas por estado. Para fazer qualquer análise, portanto, precisa-se dividir os dados esse "grão".

Primeiramente é verificado quantos estados temos informações para o Brasil

```
df.loc[df.countryregion == 'Brazil']
```

	sno	observationdate	provincestate	countryregion	lastupdate	confirme
82	83	2020-01-23	NaN	Brazil	2020-01-23 17:00:00	0.(
2455	2456	2020-02-26	NaN	Brazil	2020-02-26 23:53:02	1.(
2559	2560	2020-02-27	NaN	Brazil	2020-02-26 23:53:02	1.(
2668	2669	2020-02-28	NaN	Brazil	2020-02-26 23:53:02	1.(
2776	2777	2020-02-29	NaN	Brazil	2020-02-29 21:03:05	2.(
...	...	...	...	...	...	..
24850	24851	2020-05-15	NaN	Brazil	2020-05-16 02:32:19	220291.(
25227	25228	2020-05-16	NaN	Brazil	2020-05-17 02:32:32	233511.(

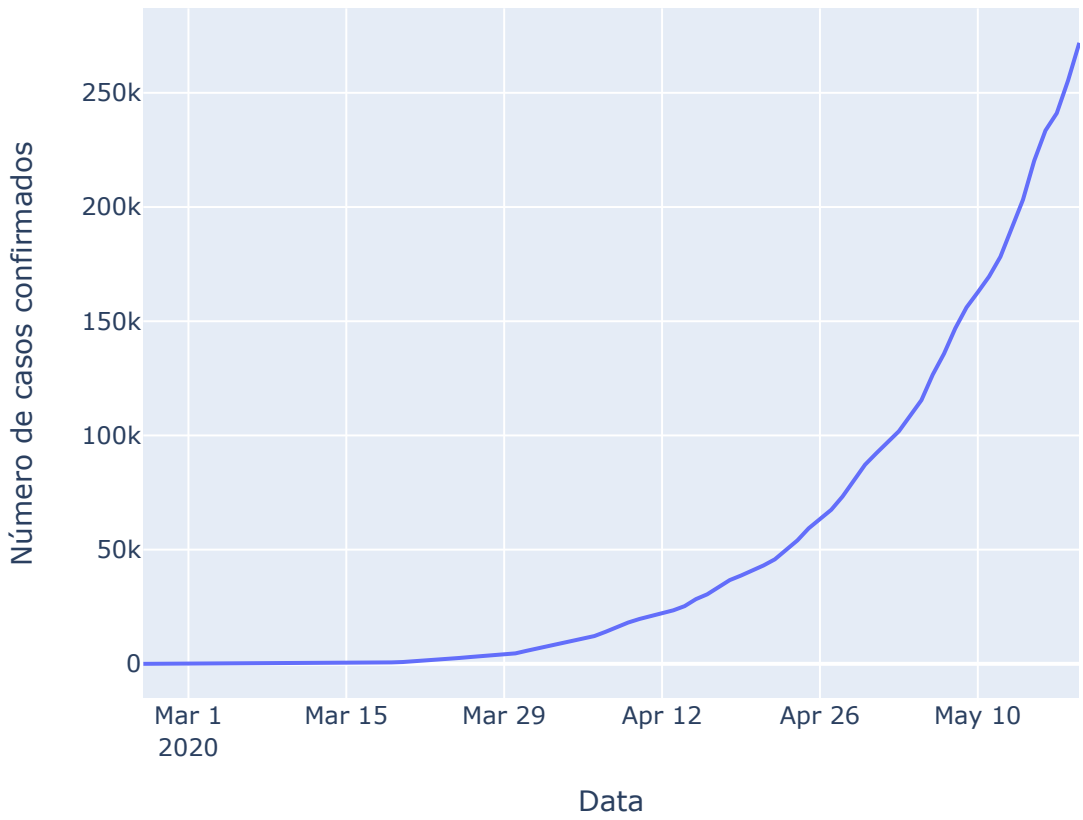
No caso do Brasil, não temos informação a nível de estado, apenas a nível do país. Será verificado assim, o comportamento dos casos confirmados no Brasil desde o primeiro caso confirmado em 26 de fevereiro.

Casos confirmados

```
brasil = df.loc[(df.countryregion == 'Brazil') & (df.confirmed > 0)]
```

```
px.line?
```

```
px.line(brasil, 'observationdate', 'confirmed',
        labels={'observationdate':'Data', 'confirmed':'Número de casos confirmados'},
       title='Casos confirmados no Brasil')
```



Número de novos casos por dia

```
# Vamos implementar uma função para fazer a contagem de novos casos
brasil['novoscasos'] = list(map(
    lambda x: 0 if (x==0) else brasil['confirmed'].iloc[x] - brasil['confirmed'].iloc[x-1],
    np.arange(brasil.shape[0])
))
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:4: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user
```

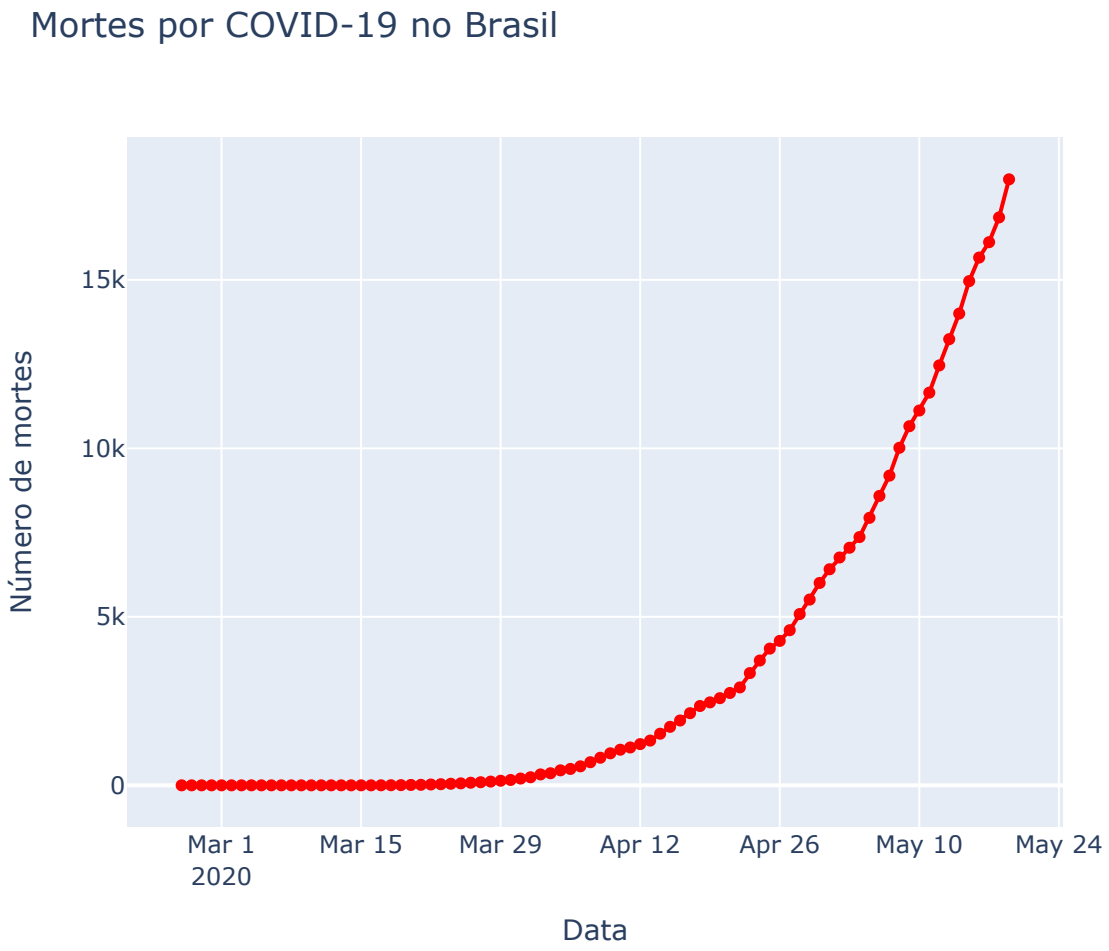
```
# Visualizando
```

```
px.line(brasil, x='observationdate', y='novoscasos', title='Novos casos por dia',
       labels={'observationdate': 'Data', 'novoscasos': 'Novos casos'})
```

O número de novos casos pareceu ser um excelente caso para modelagem.

Mortes

```
fig = go.Figure()

fig.add_trace(
    go.Scatter(x=brasil.observationdate, y=brasil.deaths, name='Mortes', mode='lines+markers'
               line=dict(color='red'))
)
#Edita o layout
fig.update_layout(title='Mortes por COVID-19 no Brasil',
                  xaxis_title='Data',
                  yaxis_title='Número de mortes')
fig.show()
```



Taxa de crescimento

Cálculo da taxa de crescimento do COVID desde o primeiro caso.

```
def taxa_crescimento(data, variable, data_inicio=None, data_fim=None):
    # Se data_inicio for None, define como a primeira data disponível no dataset
    if data_inicio == None:
        data_inicio = data.observationdate.loc[data[variable] > 0].min()
    else:
        data_inicio = pd.to_datetime(data_inicio)

    if data_fim == None:
        data_fim = data.observationdate.iloc[-1]
    else:
        data_fim = pd.to_datetime(data_fim)

    # Define os valores de presente e passado
    passado = data.loc[data.observationdate == data_inicio, variable].values[0]
    presente = data.loc[data.observationdate == data_fim, variable].values[0]

    # Define o número de pontos no tempo q vamos avaliar
```

```
    n = (data_fim - data_inicio).days

    # Calcula a taxa
    taxa = (presente/passado)**(1/n) - 1

    return taxa*100


cresc_medio = taxa_crescimento(brasil, 'confirmed')
print(f"O crescimento médio do COVID no Brasil no período avaliado foi de {cresc_medio.round(
```

```
 O crescimento médio do COVID no Brasil no período avaliado foi de 16.27%.
```

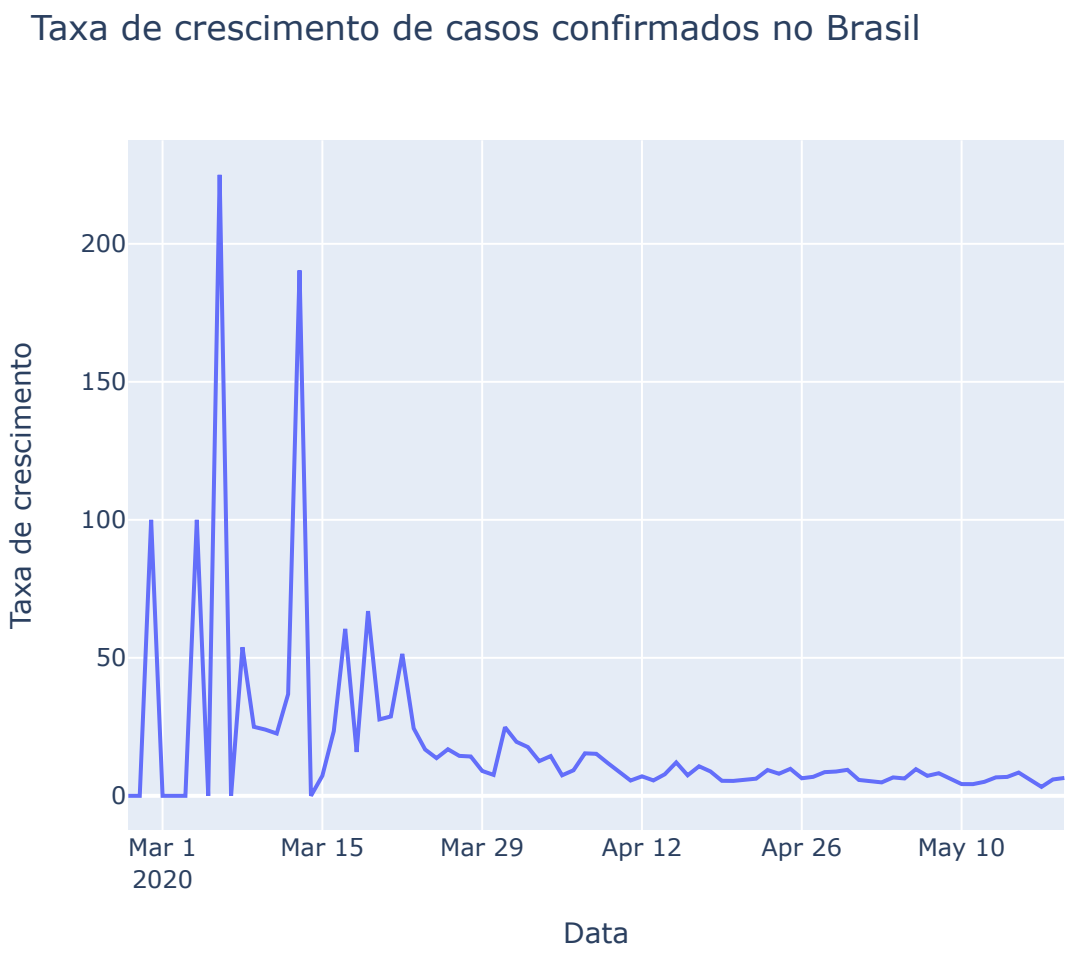
Será analisado o comportamento da **taxa de crescimento no tempo**, definindo uma função para calcular a taxa de crescimento diária.

```
def taxa_crescimento_diaria(data, variable, data_inicio=None):
    if data_inicio == None:
        data_inicio = data.observationdate.loc[data[variable] > 0].min()
    else:
        data_inicio = pd.to_datetime(data_inicio)

    data_fim = data.observationdate.max()
    n = (data_fim - data_inicio).days
    taxas = list(map(
        lambda x: (data[variable].iloc[x] - data[variable].iloc[x-1]) / data[variable].iloc[x
        range(1,n+1)
    ))
    return np.array(taxas)*100


tx_dia = taxa_crescimento_diaria(brasil, 'confirmed')


tx_dia
```

```
 array([  0.        ,   0.        , 100.        ,   0.        ,
          0.        ,   0.        , 100.        ,   0.        ,
        225.        ,   0.        ,  53.84615385,  25.        ,
         24.        ,  22.58064516,  36.84210526, 190.38461538,
          0.        ,   7.28476821,  23.45679012,  60.5       ,
         15.88785047,  66.93548387,  27.69726248,  28.75157629,
         51.4201763 ,  24.45019405,  16.78794179,  13.66266133,
         16.87548943,  14.47236181,  14.25226807,   9.01639344,
          7.58928571,  24.8525879 ,  19.57320273,  17.67115272,
         12.58080557,  14.39929329,   7.43243243,   9.26325247,
         15.40169394,  15.22017956,  11.88620903,   8.54521335,
          5.54537122,   7.06807546,   5.57858688,   7.81903542,
         12.10513815,   7.4329096 ,  10.70501233,   8.83557983,
          5.44492335,   5.4043566 ,   5.73350023,   6.21648599,
          9.35157462,   8.00823407,   9.77184834,   6.36504619,
```

```
6.88748019,   8.58316283,   8.80726429,   9.41456987,
5.75200431,   5.31224919,   4.86714727,   6.67216624,
6.29257964,   9.66263912,   7.23633807,   8.19087742,
6.24055441,   4.25346499,   4.23788714,   5.08272698,
6.69027125,   6.85190152,   8.42960156,   6.00115302,
3.24138906,   5.92666335,   6.4679208 ])
```

```
primeiro_dia = brasil.observationdate.loc[brasil.confirmed > 0].min()
px.line(x=pd.date_range(primeiro_dia, brasil.observationdate.max())[1:],
        y=tx_dia, title='Taxa de crescimento de casos confirmados no Brasil',
       labels={'y':'Taxa de crescimento', 'x':'Data'})
```

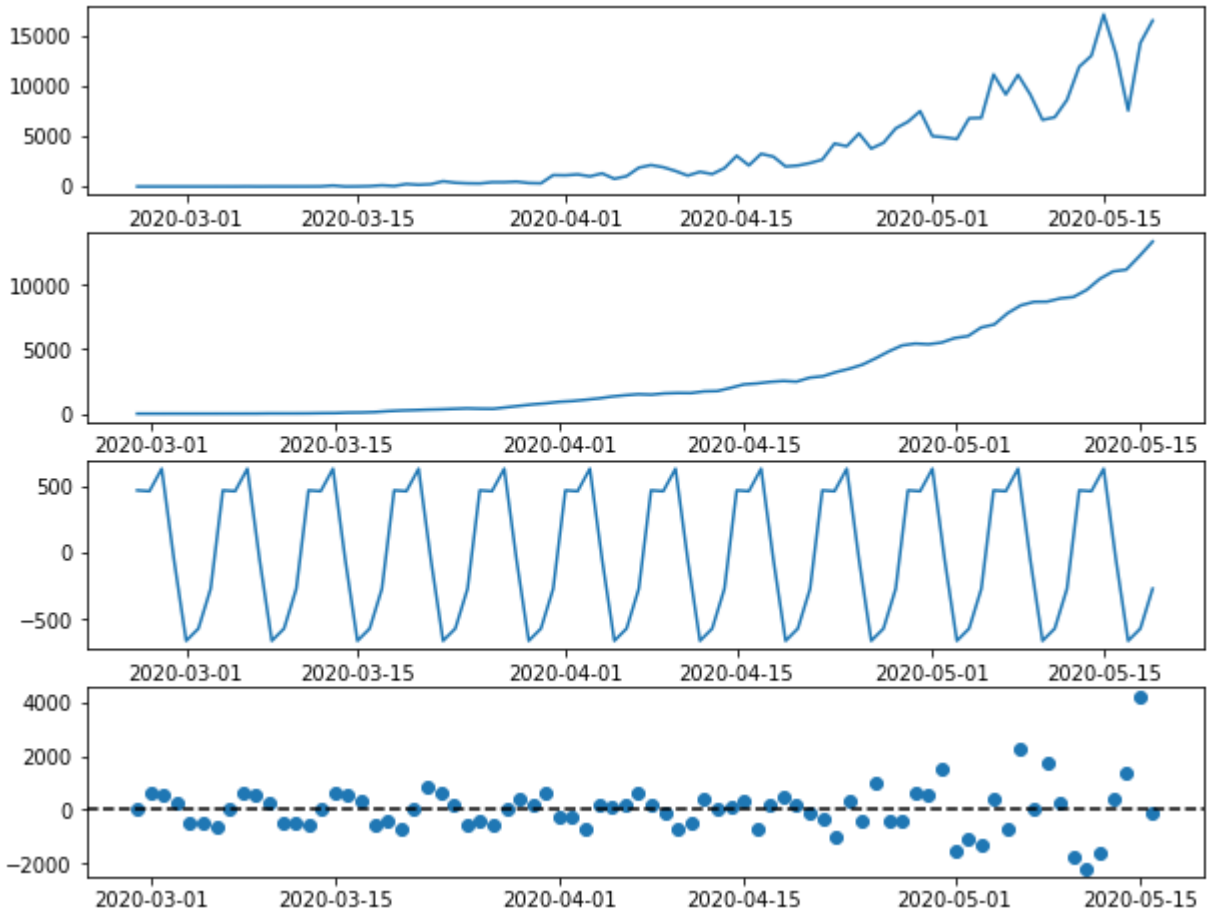


Predições

Será construído um modelo de séries temporais para prever os novos casos.

```
from statsmodels.tsa.seasonal import seasonal_decompose
import matplotlib.pyplot as plt
```

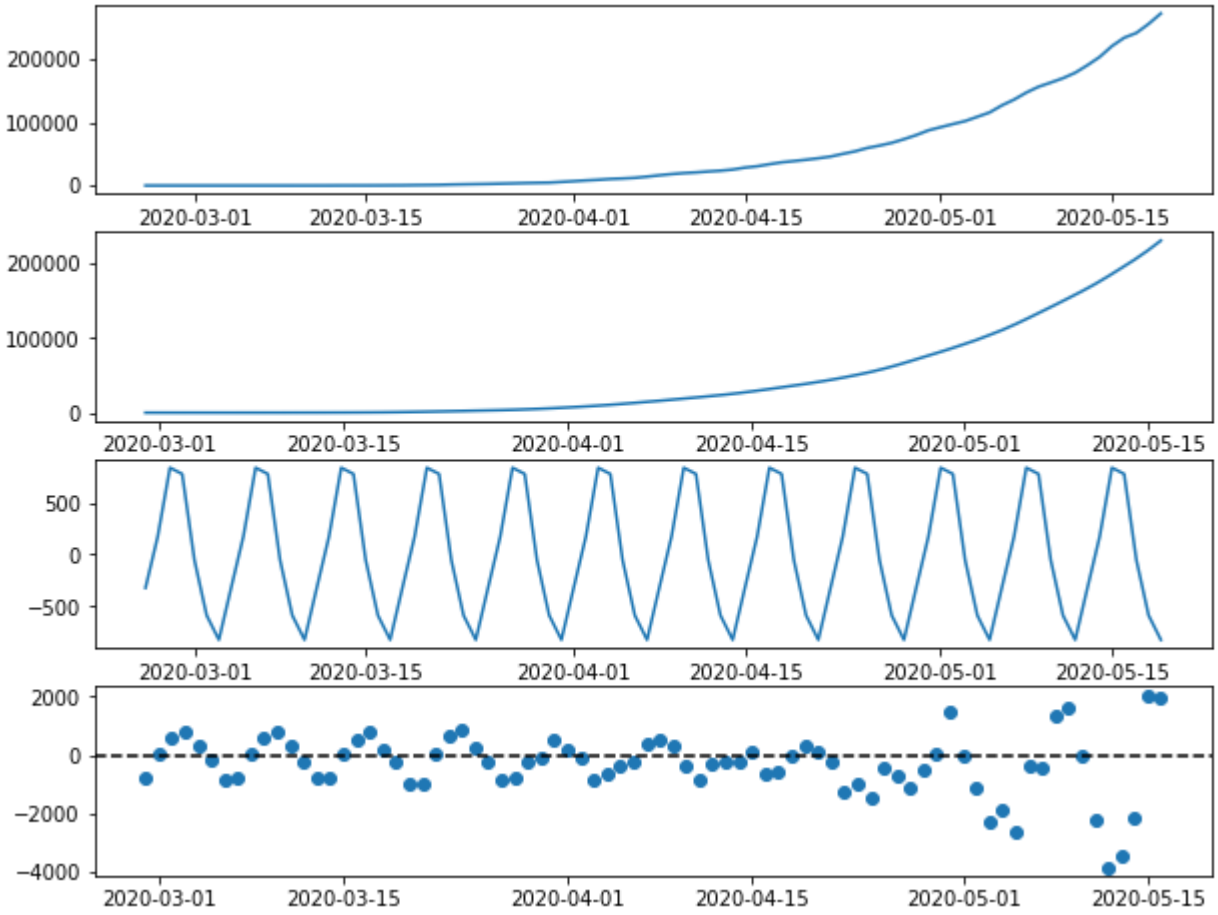
```
novoscasos = brasil.novoscasos
novoscasos.index = brasil.observationdate

res = seasonal_decompose(novoscasos)

fig, (ax1,ax2,ax3, ax4) = plt.subplots(4, 1,figsize=(10,8))
ax1.plot(res.observed)
ax2.plot(res.trend)
ax3.plot(res.seasonal)
ax4.scatter(novoscasos.index, res.resid)
ax4.axhline(0, linestyle='dashed', c='black')
plt.show()
```



Decompondo a série de confirmados

```
confirmados = brasil.confirmed
confirmados.index = brasil.observationdate


res2 = seasonal_decompose(confirmados)

fig, (ax1,ax2,ax3, ax4) = plt.subplots(4, 1,figsize=(10,8))
ax1.plot(res2.observed)
ax2.plot(res2.trend)
```

```
ax3.plot(res2.seasonal)
ax4.scatter(confirmados.index, res2.resid)
ax4.axhline(0, linestyle='dashed', c='black')
plt.show()
```



Predizendo o número de casos confirmados com um AUTO-ARIMA

```
!pip install pmdarima
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheel
Collecting pmdarima
  Downloading pmdarima-2.0.1-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64
     |████████████████████████████████| 1.8 MB 11.9 MB/s
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.7/dist-packa
Collecting statsmodels>=0.13.2
  Downloading statsmodels-0.13.2-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x8
     |████████████████████████████████| 9.8 MB 43.0 MB/s
Requirement already satisfied: Cython!=0.29.18,!=0.29.31,>=0.29 in /usr/local/lib/
Requirement already satisfied: numpy>=1.21 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: pandas>=0.19 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: setuptools!=50.0.0,>=38.6.0 in /usr/local/lib/pytho
Requirement already satisfied: urllib3 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: scikit-learn>=0.22 in /usr/local/lib/python3.7/dist
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.7/
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/di
Requirement already satisfied: patsy>=0.5.2 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: packaging>=21.3 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python3.
```

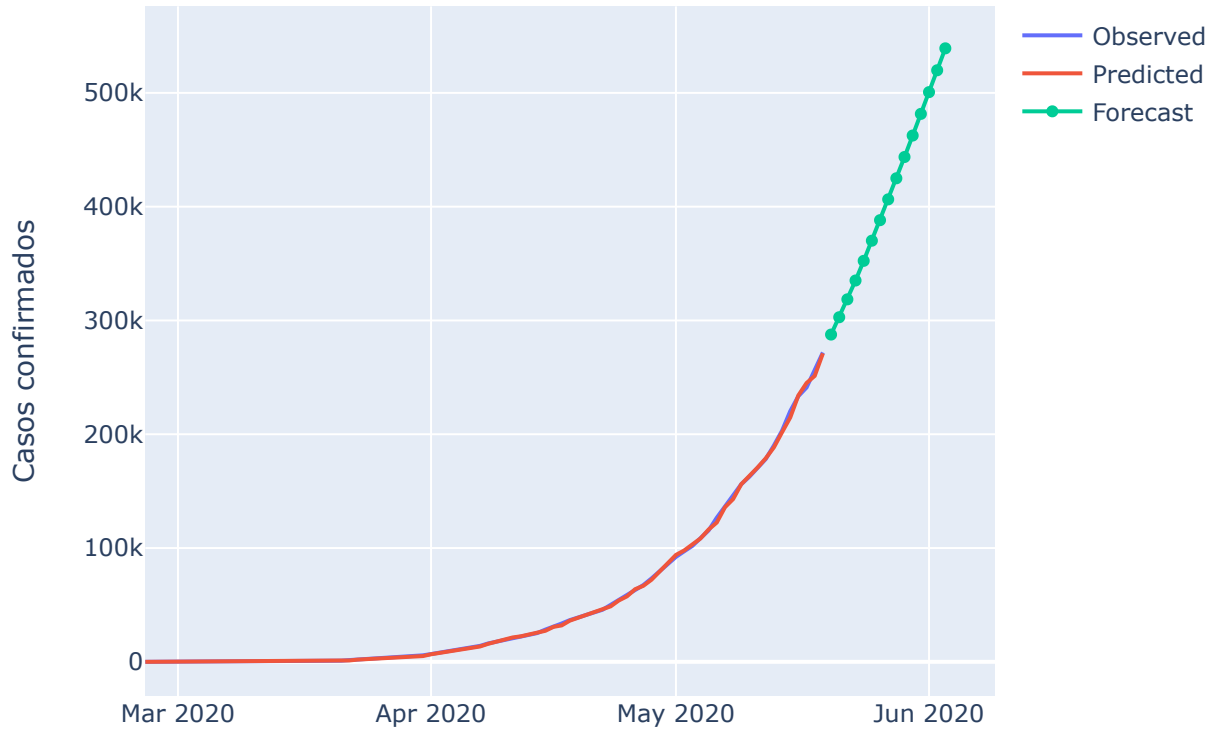
```
from pmdarima.arima import auto_arima
```

```
modelo = auto_arima(confirmados)
```

```
pd.date_range('2020-05-01', '2020-05-19')
```

```
DatetimeIndex(['2020-05-01', '2020-05-02', '2020-05-03', '2020-05-04',
               '2020-05-05', '2020-05-06', '2020-05-07', '2020-05-08',
               '2020-05-09', '2020-05-10', '2020-05-11', '2020-05-12',
               '2020-05-13', '2020-05-14', '2020-05-15', '2020-05-16',
               '2020-05-17', '2020-05-18', '2020-05-19'],
              dtype='datetime64[ns]', freq='D')
```

```
fig = go.Figure(go.Scatter(
    x=confirmados.index, y=confirmados, name='Observed'
))

fig.add_trace(go.Scatter(x=confirmados.index, y = modelo.predict_in_sample(), name='Predictec

fig.add_trace(go.Scatter(x=pd.date_range('2020-05-20', '2020-06-05'), y=modelo.predict(15), r

fig.update_layout(title='Previsão de casos confirmados para os próximos 15 dias',
                  yaxis_title='Casos confirmados', xaxis_title='Data')
fig.show()
```



Forecasting com Facebook Prophet

```
!pip install pystan~=2.14
!pip install fbprophet
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/pub
Collecting pystan~=2.14
  Downloading pystan-2.19.1.1-cp37-cp37m-manylinux1_x86_64.whl (67.3 MB)
     |████████████████████████████████| 67.3 MB 129 kB/s
Requirement already satisfied: numpy>=1.7 in /usr/local/lib/python3.7/dist-packages (fr
Requirement already satisfied: Cython!=0.25.1,>=0.22 in /usr/local/lib/python3.7/dist-p
Installing collected packages: pystan
  Attempting uninstall: pystan
    Found existing installation: pystan 3.3.0
    Uninstalling pystan-3.3.0:
      Successfully uninstalled pystan-3.3.0
Successfully installed pystan-2.19.1.1
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/pub
Collecting fbprophet
  Downloading fbprophet-0.7.1.tar.gz (64 kB)
     |████████████████████████████████| 64 kB 1.6 MB/s
Requirement already satisfied: Cython>=0.22 in /usr/local/lib/python3.7/dist-packages (
Collecting cmdstanpy==0.9.5
  Downloading cmdstanpy-0.9.5-py3-none-any.whl (37 kB)
Requirement already satisfied: pystan>=2.14 in /usr/local/lib/python3.7/dist-packages (
```

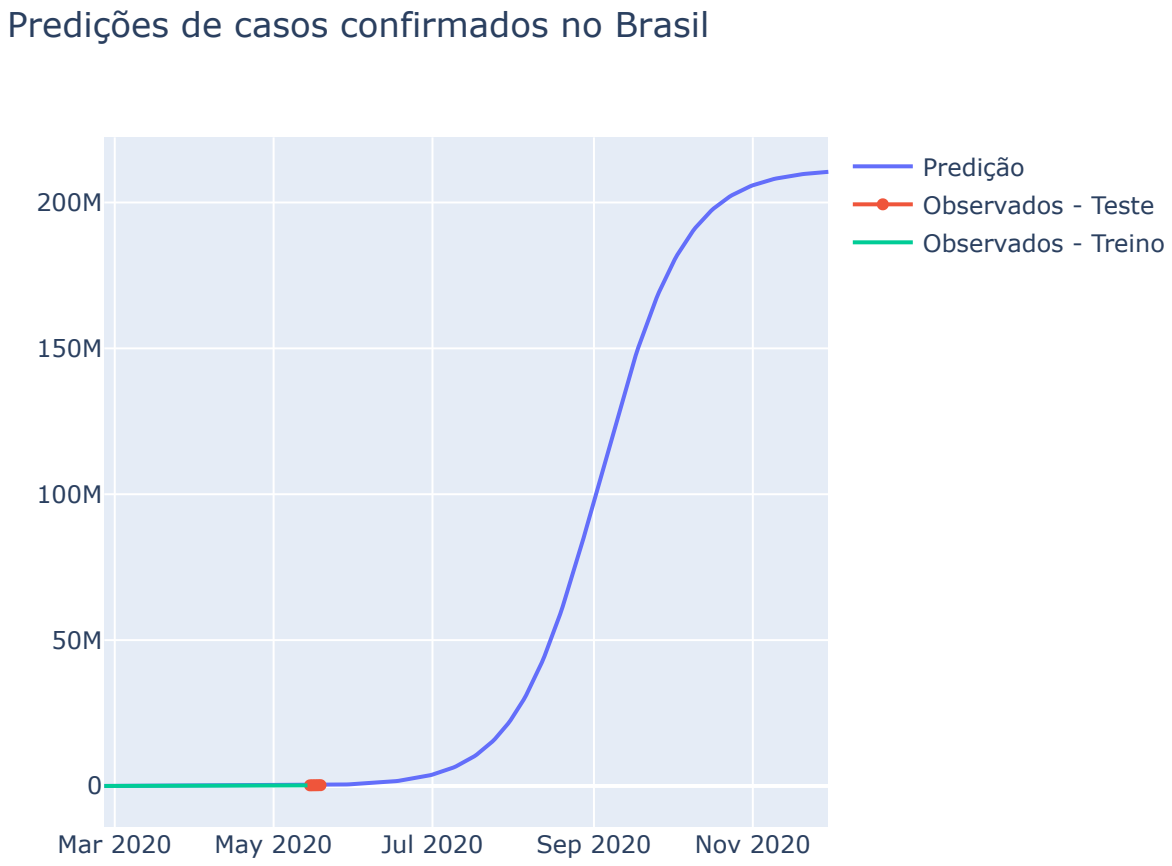
```
Requirement already satisfied: numpy>=1.15.4 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: pandas>=1.0.4 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: matplotlib>=2.0.0 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: LunarCalendar>=0.0.9 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: convertdate>=2.1.2 in /usr/local/lib/python3.7/dist-pack
Requirement already satisfied: holidays>=0.10.2 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: setuptools-git>=1.2 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: python-dateutil>=2.8.0 in /usr/local/lib/python3.7/dist-
Requirement already satisfied: tqdm>=4.36.1 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: pymeeus<=1,>=0.3.13 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: korean-lunar-calendar in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: hijri-converter in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: ephem>=3.7.5.3 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: pytz in /usr/local/lib/python3.7/dist-packages (from Lun
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local/l
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (from
Building wheels for collected packages: fbprophet
  Building wheel for fbprophet (setup.py) ... done
  Created wheel for fbprophet: filename=fbprophet-0.7.1-py3-none-any.whl size=6638822 s
  Stored in directory: /root/.cache/pip/wheels/cd/a1/12/db63ff624de492fe6cccf676091a086
Successfully built fbprophet
Installing collected packages: cmdstanpy, fbprophet
  Attempting uninstall: cmdstanpy
    Found existing installation: cmdstanpy 1.0.7
    Uninstalling cmdstanpy-1.0.7:
      Successfully uninstalled cmdstanpy-1.0.7
ERROR: pip's dependency resolver does not currently take into account all the packages
prophet 1.1.1 requires cmdstanpy>=1.0.4, but you have cmdstanpy 0.9.5 which is incompat
Successfully installed cmdstanpy-0.9.5 fbprophet-0.7.1
```

```
from fbprophet import Prophet


# preparando os dados
train = confirmados.reset_index()[:-5]
test = confirmados.reset_index()[-5:]

# renomeia colunas
train.rename(columns={"observationdate":"ds","confirmed":"y"},inplace=True)
test.rename(columns={"observationdate":"ds","confirmed":"y"},inplace=True)
test = test.set_index("ds")
test = test['y']

profeta = Prophet(growth="logistic", changepoints=['2020-03-21', '2020-03-30', '2020-04-25',

#pop = 1000000
pop = 211463256 #https://www.ibge.gov.br/apps/populacao/projecao/box_popclock.php
train['cap'] = pop

# Treina o modelo
```

```
profeta.fit(train)

# Construindo previsões para o futuro
future_dates = profeta.make_future_dataframe(periods=200)
future_dates['cap'] = pop
forecast =  profeta.predict(future_dates)
```

```
INFO:fbprophet:Disabling yearly seasonality. Run prophet with yearly_seasonality=True t
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to
```

```
fig = go.Figure()

fig.add_trace(go.Scatter(x=forecast.ds, y=forecast.yhat, name='Predição'))
fig.add_trace(go.Scatter(x=test.index, y=test, name='Observados - Teste'))
fig.add_trace(go.Scatter(x=train.ds, y=train.y, name='Observados - Treino'))
fig.update_layout(title='Predições de casos confirmados no Brasil')
fig.show()
```



Produtos pagos do Colab - Cancelar contratos

✓ 0s conclusão: 09:53