In [7]:
```
# Installs
!pip install pycountry_convert
!pip install folium
!pip install calmap
!pip install plotly==4.6.0
!pip install keras
```

```
Requirement already satisfied: pycountry_convert in c:\users\jp4514\appdata\l
ocal\continuum\anacondanew\lib\site-packages (0.7.2)
Requirement already satisfied: repoze.lru>=0.7 in c:\users\jp4514\appdata\loc
al\continuum\anacondanew\lib\site-packages (from pycountry_convert) (0.7)
Requirement already satisfied: pytest-mock>=1.6.3 in c:\users\jp4514\appdata
\local\continuum\anacondanew\lib\site-packages (from pycountry_convert) (3.0.
0)
Requirement already satisfied: pycountry>=16.11.27.1 in c:\users\jp4514\appda
ta\local\continuum\anacondanew\lib\site-packages (from pycountry_convert) (1
9.8.18)
Requirement already satisfied: wheel>=0.30.0 in c:\users\jp4514\appdata\local
\continuum\anacondanew\lib\site-packages (from pycountry_convert) (0.33.4)
Requirement already satisfied: pytest>=3.4.0 in c:\users\jp4514\appdata\local
\continuum\anacondanew\lib\site-packages (from pycountry_convert) (5.0.1)
Requirement already satisfied: pytest-cov>=2.5.1 in c:\users\jp4514\appdata\l
ocal\continuum\anacondanew\lib\site-packages (from pycountry_convert) (2.8.1)
Requirement already satisfied: pprintpp>=0.3.0 in c:\users\jp4514\appdata\loc
al\continuum\anacondanew\lib\site-packages (from pycountry_convert) (0.4.0)
Requirement already satisfied: py>=1.5.0 in c:\users\jp4514\appdata\local\con
tinuum\anacondanew\lib\site-packages (from pytest>=3.4.0->pycountry_convert)
(1.8.0)
Requirement already satisfied: packaging in c:\users\jp4514\appdata\local\con
tinuum\anacondanew\lib\site-packages (from pytest>=3.4.0->pycountry_convert)
(19.0)
Requirement already satisfied: attrs>=17.4.0 in c:\users\jp4514\appdata\local
\continuum\anacondanew\lib\site-packages (from pytest>=3.4.0->pycountry_conve
rt) (19.1.0)
Requirement already satisfied: more-itertools>=4.0.0 in c:\users\jp4514\appda
ta\local\continuum\anacondanew\lib\site-packages (from pytest>=3.4.0->pycount
ry_convert) (7.0.0)
Requirement already satisfied: atomicwrites>=1.0 in c:\users\jp4514\appdata\l
ocal\continuum\anacondanew\lib\site-packages (from pytest>=3.4.0->pycountry_c
onvert) (1.3.0)
Requirement already satisfied: pluggy<1.0,>=0.12 in c:\users\jp4514\appdata\l
ocal\continuum\anacondanew\lib\site-packages (from pytest>=3.4.0->pycountry_c
onvert) (0.12.0)
Requirement already satisfied: importlib-metadata>=0.12 in c:\users\jp4514\ap
pdata\local\continuum\anacondanew\lib\site-packages (from pytest>=3.4.0->pyco
untry_convert) (1.4.0)
Requirement already satisfied: wcwidth in c:\users\jp4514\appdata\local\conti
nuum\anacondanew\lib\site-packages (from pytest>=3.4.0->pycountry_convert)
(0.1.7)
Requirement already satisfied: colorama in c:\users\jp4514\appdata\local\cont
inuum\anacondanew\lib\site-packages (from pytest>=3.4.0->pycountry_convert)
(0.4.1)
Requirement already satisfied: coverage>=4.4 in c:\users\jp4514\appdata\local
\continuum\anacondanew\lib\site-packages (from pytest-cov>=2.5.1->pycountry_c
onvert) (5.0.4)
Requirement already satisfied: six in c:\users\jp4514\appdata\local\continuum
\anacondanew\lib\site-packages (from packaging->pytest>=3.4.0->pycountry_conv
ert) (1.12.0)
Requirement already satisfied: pyparsing>=2.0.2 in c:\users\jp4514\appdata\lo
cal\continuum\anacondanew\lib\site-packages (from packaging->pytest>=3.4.0->p
ycountry_convert) (2.4.0)
Requirement already satisfied: zipp>=0.5 in c:\users\jp4514\appdata\local\con
tinuum\anacondanew\lib\site-packages (from importlib-metadata>=0.12->pytest>=
3.4.0->pycountry_convert) (0.5.1)
```

```
Requirement already satisfied: folium in c:\users\jp4514\appdata\local\contin
uum\anacondanew\lib\site-packages (0.10.1)
Requirement already satisfied: requests in c:\users\jp4514\appdata\local\cont
inuum\anacondanew\lib\site-packages (from folium) (2.22.0)
Requirement already satisfied: jinja2>=2.9 in c:\users\jp4514\appdata\local\c
ontinuum\anacondanew\lib\site-packages (from folium) (2.10.1)
Requirement already satisfied: numpy in c:\users\jp4514\appdata\local\continu
um\anacondanew\lib\site-packages (from folium) (1.17.3)
Requirement already satisfied: branca>=0.3.0 in c:\users\jp4514\appdata\local
\continuum\anacondanew\lib\site-packages (from folium) (0.4.0)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
c:\users\jp4514\appdata\local\continuum\anacondanew\lib\site-packages (from r
equests->folium) (1.24.2)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in c:\users\jp4514\appda
ta\local\continuum\anacondanew\lib\site-packages (from requests->folium) (3.
0.4)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\jp4514\appdata
\local\continuum\anacondanew\lib\site-packages (from requests->folium) (2019.
11.28)
Requirement already satisfied: idna<2.9,>=2.5 in c:\users\jp4514\appdata\loca
l\continuum\anacondanew\lib\site-packages (from requests->folium) (2.8)
Requirement already satisfied: MarkupSafe>=0.23 in c:\users\jp4514\appdata\lo
cal\continuum\anacondanew\lib\site-packages (from jinja2>=2.9->folium) (1.1.
1)
Requirement already satisfied: six in c:\users\jp4514\appdata\local\continuum
\anacondanew\lib\site-packages (from branca>=0.3.0->folium) (1.12.0)
Requirement already satisfied: calmap in c:\users\jp4514\appdata\local\contin
uum\anacondanew\lib\site-packages (0.0.7)
Requirement already satisfied: matplotlib in c:\users\jp4514\appdata\local\co
ntinuum\anacondanew\lib\site-packages (from calmap) (3.1.0)
Requirement already satisfied: pandas in c:\users\jp4514\appdata\local\contin
uum\anacondanew\lib\site-packages (from calmap) (0.24.2)
Requirement already satisfied: numpy in c:\users\jp4514\appdata\local\continu
um\anacondanew\lib\site-packages (from calmap) (1.17.3)
Requirement already satisfied: cycler>=0.10 in c:\users\jp4514\appdata\local
\continuum\anacondanew\lib\site-packages (from matplotlib->calmap) (0.10.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\jp4514\appdata\l
ocal\continuum\anacondanew\lib\site-packages (from matplotlib->calmap) (1.1.
0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in
c:\users\jp4514\appdata\local\continuum\anacondanew\lib\site-packages (from m
atplotlib->calmap) (2.4.0)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\jp4514\appdat
a\local\continuum\anacondanew\lib\site-packages (from matplotlib->calmap) (2.
8.0)
Requirement already satisfied: pytz>=2011k in c:\users\jp4514\appdata\local\c
ontinuum\anacondanew\lib\site-packages (from pandas->calmap) (2019.1)
Requirement already satisfied: six in c:\users\jp4514\appdata\local\continuum
\anacondanew\lib\site-packages (from cycler>=0.10->matplotlib->calmap) (1.12.
0)
Requirement already satisfied: setuptools in c:\users\jp4514\appdata\local\co
ntinuum\anacondanew\lib\site-packages (from kiwisolver>=1.0.1->matplotlib->ca
lmap) (41.0.1)
```

In [60]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import ticker
import pycountry_convert as pc
import folium
import branca
from datetime import datetime, timedelta,date
from scipy.interpolate import make_interp_spline, BSpline
import plotly.express as px
import json, requests
import calmap

from keras.layers import Input, Dense, Activation, LeakyReLU
from keras import models
from keras.optimizers import RMSprop, Adam

import warnings
warnings.filterwarnings('ignore')

%matplotlib inline
```

In [69]:
```
# Retriving Dataset
df_confirmed = pd.read_csv('https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/time_series_covid19_confirmed_global.csv')
df_deaths = pd.read_csv('https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/time_series_covid19_deaths_global.csv')

# Depricated
#df_recovered = pd.read_csv('https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/time_series_19-covid-Recovered.csv')
df_covid19 = pd.read_csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/web-data/data/cases_country.csv")
df_table = pd.read_csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/web-data/data/cases_time.csv",parse_dates=['Last_Update'])
```

In [70]:
```
# new dataset
df_covid19.head(2)
df_confirmed.head(2)
```

Out[70]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1
0	NaN	Afghanistan	33.0000	65.0000	0	0	0	0	0	
1	NaN	Albania	41.1533	20.1683	0	0	0	0	0	

2 rows × 78 columns

In [71]:
```
df_confirmed = df_confirmed.rename(columns={"Province/State":"state","Country/
Region": "country"})
df_deaths = df_deaths.rename(columns={"Province/State":"state","Country/Regio
n": "country"})
df_covid19 = df_covid19.rename(columns={"Country_Region": "country"})
df_covid19["Active"] = df_covid19["Confirmed"]-df_covid19["Recovered"]-df_covi
d19["Deaths"]
```

In [72]:
```
# Changing the conuntry names as required by pycountry_convert Lib
df_confirmed.loc[df_confirmed['country'] == "US", "country"] = "USA"
df_deaths.loc[df_deaths['country'] == "US", "country"] = "USA"
df_covid19.loc[df_covid19['country'] == "US", "country"] = "USA"
df_table.loc[df_table['Country_Region'] == "US", "Country_Region"] = "USA"
# df_recovered.loc[df_recovered['country'] == "US", "country"] = "USA"


df_confirmed.loc[df_confirmed['country'] == 'Korea, South', "country"] = 'Sout
h Korea'
df_deaths.loc[df_deaths['country'] == 'Korea, South', "country"] = 'South Kore
a'
df_covid19.loc[df_covid19['country'] == "Korea, South", "country"] = "South Ko
rea"
df_table.loc[df_table['Country_Region'] == "Korea, South", "Country_Region"] =
"South Korea"
# df_recovered.loc[df_recovered['country'] == 'Korea, South', "country"] = 'So
uth Korea'

df_confirmed.loc[df_confirmed['country'] == 'Taiwan*', "country"] = 'Taiwan'
df_deaths.loc[df_deaths['country'] == 'Taiwan*', "country"] = 'Taiwan'
df_covid19.loc[df_covid19['country'] == "Taiwan*", "country"] = "Taiwan"
df_table.loc[df_table['Country_Region'] == "Taiwan*", "Country_Region"] = "Tai
wan"
# df_recovered.loc[df_recovered['country'] == 'Taiwan*', "country"] = 'Taiwan'

df_confirmed.loc[df_confirmed['country'] == 'Congo (Kinshasa)', "country"] =
'Democratic Republic of the Congo'
df_deaths.loc[df_deaths['country'] == 'Congo (Kinshasa)', "country"] = 'Democr
atic Republic of the Congo'
df_covid19.loc[df_covid19['country'] == "Congo (Kinshasa)", "country"] = "Demo
cratic Republic of the Congo"
df_table.loc[df_table['Country_Region'] == "Congo (Kinshasa)", "Country_Regio
n"] = "Democratic Republic of the Congo"
# df_recovered.loc[df_recovered['country'] == 'Congo (Kinshasa)', "country"] =
'Democratic Republic of the Congo'

df_confirmed.loc[df_confirmed['country'] == "Cote d'Ivoire", "country"] = "Côt
e d'Ivoire"
df_deaths.loc[df_deaths['country'] == "Cote d'Ivoire", "country"] = "Côte d'Iv
oire"
df_covid19.loc[df_covid19['country'] == "Cote d'Ivoire", "country"] = "Côte
 d'Ivoire"
df_table.loc[df_table['Country_Region'] == "Cote d'Ivoire", "Country_Region"]
= "Côte d'Ivoire"
# df_recovered.loc[df_recovered['country'] == "Cote d'Ivoire", "country"] = "C
ôte d'Ivoire"

df_confirmed.loc[df_confirmed['country'] == "Reunion", "country"] = "Réunion"
df_deaths.loc[df_deaths['country'] == "Reunion", "country"] = "Réunion"
df_covid19.loc[df_covid19['country'] == "Reunion", "country"] = "Réunion"
df_table.loc[df_table['Country_Region'] == "Reunion", "Country_Region"] = "Réu
nion"
# df_recovered.loc[df_recovered['country'] == "Reunion", "country"] = "Réunio
n"
```

```
df_confirmed.loc[df_confirmed['country'] == 'Congo (Brazzaville)', "country"]
= 'Republic of the Congo'
df_deaths.loc[df_deaths['country'] == 'Congo (Brazzaville)', "country"] = 'Rep
ublic of the Congo'
df_covid19.loc[df_covid19['country'] == "Congo (Brazzaville)", "country"] = "R
epublic of the Congo"
df_table.loc[df_table['Country_Region'] == "Congo (Brazzaville)", "Country_Reg
ion"] = "Republic of the Congo"
# df_recovered.loc[df_recovered['country'] == 'Congo (Brazzaville)', "countr
y"] = 'Republic of the Congo'

df_confirmed.loc[df_confirmed['country'] == 'Bahamas, The', "country"] = 'Baha
mas'
df_deaths.loc[df_deaths['country'] == 'Bahamas, The', "country"] = 'Bahamas'
df_covid19.loc[df_covid19['country'] == "Bahamas, The", "country"] = "Bahamas"
df_table.loc[df_table['Country_Region'] == "Bahamas, The", "Country_Region"] =
"Bahamas"
# df_recovered.loc[df_recovered['country'] == 'Bahamas, The', "country"] = 'Ba
hamas'

df_confirmed.loc[df_confirmed['country'] == 'Gambia, The', "country"] = 'Gambi
a'
df_deaths.loc[df_deaths['country'] == 'Gambia, The', "country"] = 'Gambia'
df_covid19.loc[df_covid19['country'] == "Gambia, The", "country"] = "Gambia"
df_table.loc[df_table['Country_Region'] == "Gambia", "Country_Region"] = "Gamb
ia"
# df_recovered.loc[df_recovered['country'] == 'Gambia, The', "country"] = 'Gam
bia'

# getting all countries
countries = np.asarray(df_confirmed["country"])
countries1 = np.asarray(df_covid19["country"])
# Continent_code to Continent_names
continents = {
    'NA': 'North America',
    'SA': 'South America',
    'AS': 'Asia',
    'OC': 'Australia',
    'AF': 'Africa',
    'EU' : 'Europe',
    'na' : 'Others'
}

# Defininng Function for getting continent code for country.
def country_to_continent_code(country):
    try:
        return pc.country_alpha2_to_continent_code(pc.country_name_to_country_
alpha2(country))
    except :
        return 'na'

#Collecting Continent Information
df_confirmed.insert(2,"continent", [continents[country_to_continent_code(count
ry)] for country in countries[:]])
df_deaths.insert(2,"continent",  [continents[country_to_continent_code(country
)] for country in countries[:]])
df_covid19.insert(1,"continent",  [continents[country_to_continent_code(countr
```

```
y)] for country in countries1[:]])
df_table.insert(1,"continent",  [continents[country_to_continent_code(country
)] for country in df_table["Country_Region"].values])
# df_recovered.insert(2,"continent",  [continents[country_to_continent_code(co
untry)] for country in countries[:]] )
```

In [73]:
```
df_table = df_table[df_table["continent"] != "Others"]
```

In [74]:
```
df_deaths[df_deaths["continent" ]== 'Others']
```

Out[74]:

	state	country	continent	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20
88	NaN	Diamond Princess	Others	0.000000	0.000000	0	0	0	0	0
127	NaN	Holy See	Others	41.902900	12.453400	0	0	0	0	0
236	NaN	Timor-Leste	Others	-8.874217	125.727539	0	0	0	0	0
241	NaN	West Bank and Gaza	Others	31.952200	35.233200	0	0	0	0	0
247	NaN	Kosovo	Others	42.602636	20.902977	0	0	0	0	0
248	NaN	Burma	Others	21.916200	95.956000	0	0	0	0	0
252	NaN	MS Zaandam	Others	0.000000	0.000000	0	0	0	0	0

7 rows × 79 columns

In [76]:
```
df_confirmed = df_confirmed.replace(np.nan, '', regex=True)
df_deaths = df_deaths.replace(np.nan, '', regex=True)
df_confirmed.to_excel("Confirmed.xlsx")
df_deaths.to_excel("Deaths.xlsx")
df_covid19.to_excel("COVID19.xlsx")
df_table.to_excel("table.xlsx")
```

In [25]:
```
def plot_params(ax,axis_label= None, plt_title = None,label_size=15, axis_fsiz
e = 15, title_fsize = 20, scale = 'linear' ):
    # Tick-Parameters
    ax.xaxis.set_minor_locator(ticker.AutoMinorLocator())
    ax.yaxis.set_minor_locator(ticker.AutoMinorLocator())
    ax.tick_params(which='both', width=1,labelsize=label_size)
    ax.tick_params(which='major', length=6)
    ax.tick_params(which='minor', length=3, color='0.8')

    # Grid
    plt.grid(lw = 1, ls = '-', c = "0.7", which = 'major')
    plt.grid(lw = 1, ls = '-', c = "0.9", which = 'minor')

    # Plot Title
    plt.title( plt_title,{'fontsize':title_fsize})

    # Yaxis sacle
    plt.yscale(scale)
    plt.minorticks_on()
    # Plot Axes Labels
    xl = plt.xlabel(axis_label[0],fontsize = axis_fsize)
    yl = plt.ylabel(axis_label[1],fontsize = axis_fsize)

def visualize_covid_cases(confirmed, deaths, continent=None , country = None ,
state = None, period = None, figure = None, scale = "linear"):
    x = 0
    if figure == None:
        f = plt.figure(figsize=(10,10))
        # Sub plot
        ax = f.add_subplot(111)
    else :
        f = figure[0]
        # Sub plot
        ax = f.add_subplot(figure[1],figure[2],figure[3])
    ax.set_axisbelow(True)
    plt.tight_layout(pad=10, w_pad=5, h_pad=5)

    stats = [confirmed, deaths]
    label = ["Confirmed", "Deaths"]

    if continent != None:
        params = ["continent",continent]
    elif country != None:
        params = ["country",country]
    else:
        params = ["All", "All"]
    color = ["darkcyan","crimson"]
    marker_style = dict(linewidth=3, linestyle='-', marker='o',markersize=4, m
arkerfacecolor='#ffffff')
    for i,stat in enumerate(stats):
        if params[1] == "All" :
            cases = np.sum(np.asarray(stat.iloc[:,5:]),axis = 0)[x:]
        else :
            cases = np.sum(np.asarray(stat[stat[params[0]] == params[1]].iloc
[:,5:]),axis = 0)[x:]
        date = np.arange(1,cases.shape[0]+1)[x:]
```

```
        plt.plot(date,cases,label = label[i]+" (Total : "+str(cases[-1])+")",c
olor=color[i],**marker_style)
        plt.fill_between(date,cases,color=color[i],alpha=0.3)

    if params[1] == "All" :
        Total_confirmed = np.sum(np.asarray(stats[0].iloc[:,5:]),axis = 0)[x:]
        Total_deaths = np.sum(np.asarray(stats[1].iloc[:,5:]),axis = 0)[x:]
    else :
        Total_confirmed =  np.sum(np.asarray(stats[0][stat[params[0]] == param
s[1]].iloc[:,5:]),axis = 0)[x:]
        Total_deaths = np.sum(np.asarray(stats[1][stat[params[0]] == params[1
]].iloc[:,5:]),axis = 0)[x:]

    text = "From "+stats[0].columns[5]+" to "+stats[0].columns[-1]+"\n"
    text += "Mortality rate : "+ str(int(Total_deaths[-1]/(Total_confirmed[-1
])*10000)/100)+"\n"
    text += "Last 5 Days:\n"
    text += "Confirmed : " + str(Total_confirmed[-1] - Total_confirmed[-6])+"
\n"
    text += "Deaths : " + str(Total_deaths[-1] - Total_deaths[-6])+"\n"
    text += "Last 24 Hours:\n"
    text += "Confirmed : " + str(Total_confirmed[-1] - Total_confirmed[-2])+"
\n"
    text += "Deaths : " + str(Total_deaths[-1] - Total_deaths[-2])+"\n"

    plt.text(0.02, 0.78, text, fontsize=15, horizontalalignment='left', vertic
alalignment='top', transform=ax.transAxes,bbox=dict(facecolor='white', alpha=
0.4))

    # Plot Axes Labels
    axis_label = ["Days ("+df_confirmed.columns[5]+" - "+df_confirmed.columns[
-1]+")","No of Cases"]

    # Plot Parameters
    plot_params(ax,axis_label,scale = scale)

    # Plot Title
    if params[1] == "All" :
        plt.title("COVID-19 Cases World",{'fontsize':25})
    else:
        plt.title("COVID-19 Cases for "+params[1] ,{'fontsize':25})

    # Legend Location
    l = plt.legend(loc= "best",fontsize = 15)

    if figure == None:
        plt.show()

def get_total_cases(cases, country = "All"):
    if(country == "All") :
        return np.sum(np.asarray(cases.iloc[:,5:]),axis = 0)[-1]
    else :
        return np.sum(np.asarray(cases[cases["country"] == country].iloc[:,5
:]),axis = 0)[-1]

def get_mortality_rate(confirmed,deaths, continent = None, country = None):
    if continent != None:
```

```
        params = ["continent",continent]
    elif country != None:
        params = ["country",country]
    else :
        params = ["All", "All"]

    if params[1] == "All" :
        Total_confirmed = np.sum(np.asarray(confirmed.iloc[:,5:]),axis = 0)
        Total_deaths = np.sum(np.asarray(deaths.iloc[:,5:]),axis = 0)
        mortality_rate = np.round((Total_deaths/(Total_confirmed+1.01))*100,2)
    else :
        Total_confirmed =  np.sum(np.asarray(confirmed[confirmed[params[0]] ==
params[1]].iloc[:,5:]),axis = 0)
        Total_deaths = np.sum(np.asarray(deaths[deaths[params[0]] == params[1
]].iloc[:,5:]),axis = 0)
        mortality_rate = np.round((Total_deaths/(Total_confirmed+1.01))*100,2)

    return np.nan_to_num(mortality_rate)
def dd(date1,date2):
    return (datetime.strptime(date1,'%m/%d/%y') - datetime.strptime(date2,'%m/
%d/%y')).days


out = ""#+"output/"
```

In [26]:
```
df_countries_cases = df_covid19.copy().drop(['Lat','Long_','continent','Last_U
pdate'],axis =1)
df_countries_cases.index = df_countries_cases["country"]
df_countries_cases = df_countries_cases.drop(['country'],axis=1)

df_continents_cases = df_covid19.copy().drop(['Lat','Long_','country','Last_Up
date'],axis =1)
df_continents_cases = df_continents_cases.groupby(["continent"]).sum()
```

In [27]:
```
df_t = pd.DataFrame(pd.to_numeric(df_countries_cases.sum()),dtype=np.float64).
transpose()
df_t["Mortality Rate (per 100)"] = np.round(100*df_t["Deaths"]/df_t["Confirme
d"],2)
df_t.style.background_gradient(cmap='Wistia',axis=1).format("{:.0f}",subset=[
"Confirmed"])
```

Out[27]:

	Confirmed	Deaths	Recovered	Active	Mortality Rate (per 100)
0	1203099	64774	246893	891432	5.38

In [28]:
```
df_continents_cases["Mortality Rate (per 100)"] = np.round(100*df_continents_cases["Deaths"]/df_continents_cases["Confirmed"],2)
df_continents_cases.style.background_gradient(cmap='Blues',subset=["Confirmed"])\
                         .background_gradient(cmap='Reds',subset=["Deaths"])\
                         .background_gradient(cmap='Greens',subset=["Recovered"])\
                         .background_gradient(cmap='Purples',subset=["Active"])\
                         .background_gradient(cmap='YlOrBr',subset=["Mortality Rate (per 100)"])
```

Out[28]:

continent	Confirmed	Deaths	Recovered	Active	Mortality Rate (per 100)
Africa	8592	383	809	7400	4.46
Asia	214446	8333	109714	96399	3.89
Australia	6602	31	857	5714	0.47
Europe	616034	46205	114356	455473	7.5
North America	332884	8984	18309	305591	2.7
Others	1102	16	656	430	1.45
South America	23439	822	2192	20425	3.51

In [29]:
```
# df_countries_cases.sort_values('Confirmed', ascending= False).style.backgrou
nd_gradient(cmap='Wistia')
df_countries_cases["Mortality Rate (per 100)"] = np.round(100*df_countries_cas
es["Deaths"]/df_countries_cases["Confirmed"],2)
df_countries_cases.sort_values('Confirmed', ascending= False).style.background
_gradient(cmap='Blues',subset=["Confirmed"])\
                        .background_gradient(cmap='Reds',subset=["Deaths"])\
                        .background_gradient(cmap='Greens',subset=["Recovered"
])\
                        .background_gradient(cmap='Purples',subset=["Active"])
\
                        .background_gradient(cmap='YlOrBr',subset=["Mortality
 Rate (per 100)"])
```

Out[29]:

country	Confirmed	Deaths	Recovered	Active	Mortality Rate (per 100)
USA	312146	8499	14997	288650	2.72
Spain	126168	11947	34219	80002	9.47
Italy	124632	15362	20996	88274	12.33
Germany	96092	1444	26400	68248	1.5
France	90848	7574	15572	67702	8.34
China	82574	3333	77160	2081	4.04
Iran	55743	3452	19736	32555	6.19
United Kingdom	42479	4320	215	37944	10.17
Turkey	23934	501	786	22647	2.09
Switzerland	20505	666	6415	13424	3.25
Belgium	18431	1283	3247	13901	6.96
Netherlands	16727	1656	262	14809	9.9
Canada	14022	234	2587	11201	1.67
Austria	11781	186	2507	9088	1.58
Portugal	10524	266	75	10183	2.53
Brazil	10360	445	127	9788	4.3
South Korea	10237	183	6463	3591	1.79
Israel	7851	44	427	7380	0.56
Sweden	6443	373	205	5865	5.79
Norway	5550	62	32	5456	1.12
Australia	5550	30	701	4819	0.54
Russia	4731	43	333	4355	0.91
Ireland	4604	137	25	4442	2.98
Czechia	4472	59	78	4335	1.32
Denmark	4269	161	1379	2729	3.77
Chile	4161	27	528	3606	0.65
Poland	3627	79	116	3432	2.18
Romania	3613	146	329	3138	4.04
India	3588	99	229	3260	2.76
Malaysia	3483	57	915	2511	1.64
Ecuador	3465	172	100	3193	4.96
Japan	3139	77	514	2548	2.45
Philippines	3094	144	57	2893	4.65
Pakistan	2818	41	131	2646	1.45

country	Confirmed	Deaths	Recovered	Active	Mortality Rate (per 100)
Luxembourg	2729	31	500	2198	1.14
Saudi Arabia	2370	29	420	1921	1.22
Indonesia	2092	191	150	1751	9.13
Thailand	2067	20	674	1373	0.97
Mexico	1890	79	633	1178	4.18
Finland	1882	25	300	1557	1.33
Panama	1801	46	13	1742	2.55
Peru	1746	73	914	759	4.18
Greece	1673	68	78	1527	4.06
Serbia	1624	44	0	1580	2.71
South Africa	1585	9	95	1481	0.57
Dominican Republic	1578	77	17	1484	4.88
United Arab Emirates	1505	10	125	1370	0.66
Argentina	1451	43	279	1129	2.96
Iceland	1417	4	396	1017	0.28
Colombia	1406	32	85	1289	2.28
Qatar	1325	3	109	1213	0.23
Algeria	1251	130	90	1031	10.39
Ukraine	1225	32	25	1168	2.61
Singapore	1189	6	297	886	0.5
Croatia	1126	12	119	995	1.07
Egypt	1070	71	241	758	6.64
New Zealand	1039	1	156	882	0.1
Estonia	1039	13	59	967	1.25
Slovenia	977	22	79	876	2.25
Morocco	919	59	66	794	6.42
Iraq	878	56	259	563	6.38
Lithuania	771	11	7	753	1.43
Armenia	770	7	43	720	0.91
Moldova	752	12	29	711	1.6
Diamond Princess	712	11	619	82	1.54
Bahrain	688	4	423	261	0.58
Hungary	678	32	58	588	4.72
Bosnia and Herzegovina	624	21	30	573	3.37
Cameroon	555	9	17	529	1.62

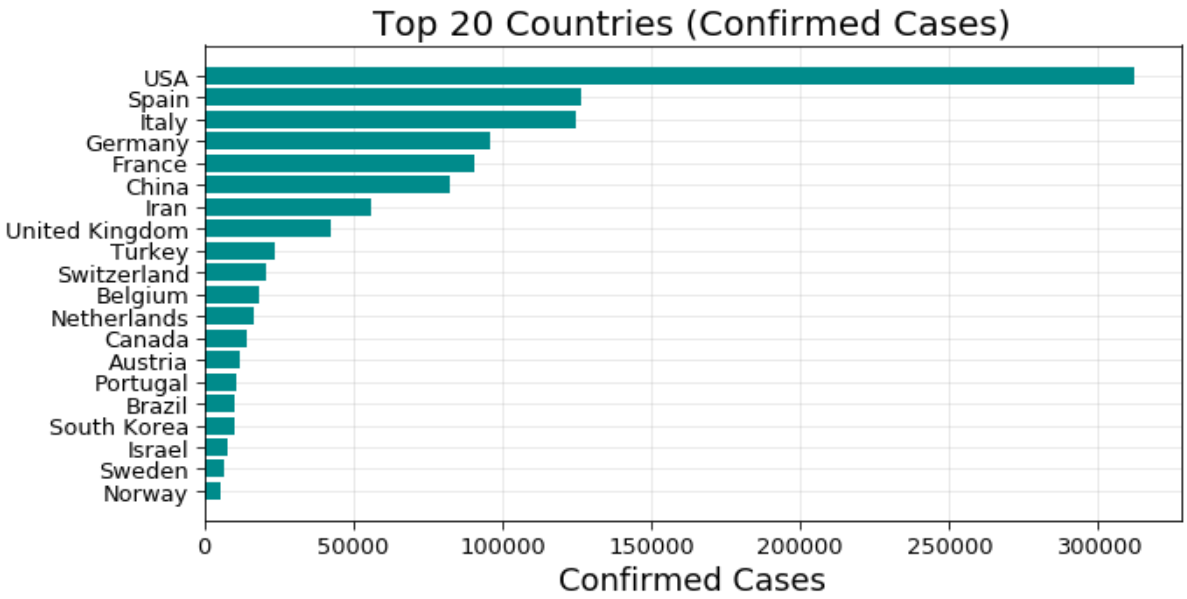
country	Confirmed	Deaths	Recovered	Active	Mortality Rate (per 100)
Tunisia	553	18	5	530	3.25
Kazakhstan	531	5	36	490	0.94
Azerbaijan	521	5	32	484	0.96
Lebanon	520	17	54	449	3.27
Latvia	509	1	1	507	0.2
Bulgaria	503	17	34	452	3.38
North Macedonia	483	17	20	446	3.52
Kuwait	479	1	93	385	0.21
Slovakia	471	1	10	460	0.21
Andorra	466	17	21	428	3.65
Belarus	440	5	53	382	1.14
Costa Rica	435	2	13	420	0.46
Cyprus	426	9	33	384	2.11
Uruguay	400	5	93	302	1.25
Taiwan	355	5	50	300	1.41
Albania	333	20	99	214	6.01
Jordan	323	5	74	244	1.55
Burkina Faso	318	16	66	236	5.03
Afghanistan	299	7	10	282	2.34
Cuba	288	6	15	267	2.08
Oman	277	2	61	214	0.72
Honduras	268	22	6	240	8.21
Uzbekistan	266	2	25	239	0.75
San Marino	259	32	27	200	12.36
Côte d'Ivoire	245	1	25	219	0.41
Vietnam	240	0	90	150	0
Senegal	219	2	72	145	0.91
West Bank and Gaza	217	1	21	195	0.46
Nigeria	214	4	25	185	1.87
Malta	213	0	2	211	0
Ghana	205	5	31	169	2.44
Montenegro	201	2	1	198	1
Mauritius	196	7	7	182	3.57
Sri Lanka	166	5	27	134	3.01
Georgia	162	1	36	125	0.62

country	Confirmed	Deaths	Recovered	Active	Mortality Rate (per 100)
Bolivia	157	10	2	145	6.37
Venezuela	155	7	52	96	4.52
Democratic Republic of the Congo	154	18	3	133	11.69
Niger	144	8	0	136	5.56
Kyrgyzstan	144	1	9	134	0.69
Kosovo	135	1	16	118	0.74
Brunei	135	1	66	68	0.74
Kenya	126	4	4	118	3.17
Cambodia	114	0	50	64	0
Guinea	111	0	5	106	0
Paraguay	104	3	12	89	2.88
Trinidad and Tobago	103	6	1	96	5.83
Rwanda	102	0	0	102	0
Liechtenstein	77	1	0	76	1.3
Bangladesh	70	8	30	32	11.43
Madagascar	70	0	0	70	0
Monaco	66	1	3	62	1.52
Guatemala	61	2	15	44	3.28
El Salvador	56	3	2	51	5.36
Jamaica	55	3	7	45	5.45
Barbados	52	0	0	52	0
Djibouti	50	0	8	42	0
Uganda	48	0	0	48	0
Togo	41	3	17	21	7.32
Mali	41	3	1	37	7.32
Zambia	39	1	2	36	2.56
Ethiopia	38	0	4	34	0
Eritrea	29	0	0	29	0
Bahamas	28	4	0	24	14.29
Guyana	24	4	0	20	16.67
Republic of the Congo	22	2	2	18	9.09
Gabon	21	1	1	19	4.76
Haiti	21	0	1	20	0
Burma	21	1	0	20	4.76

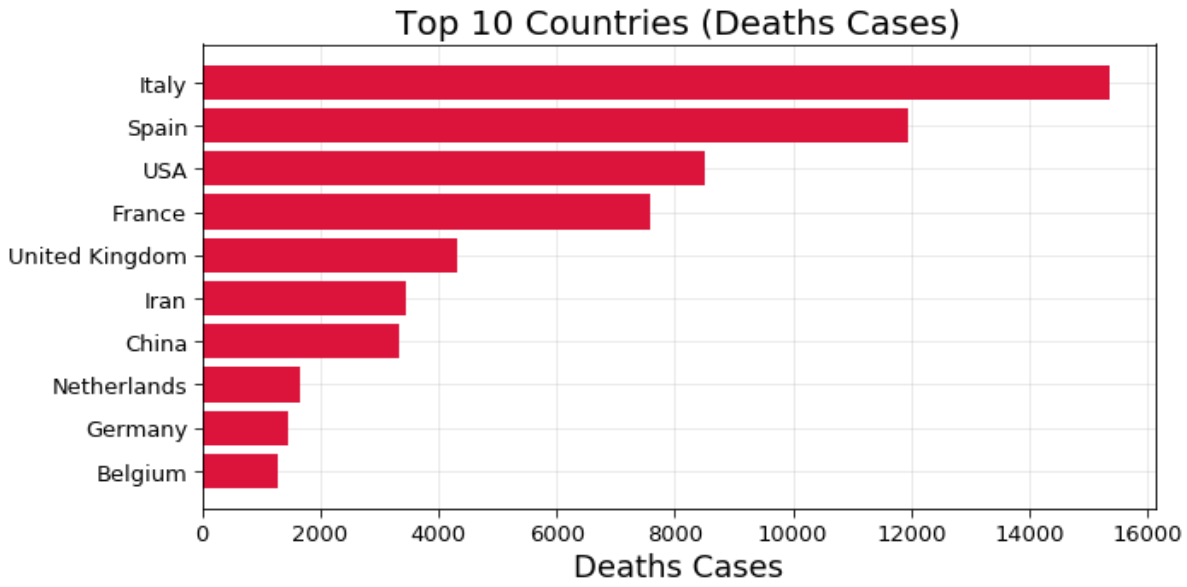
country	Confirmed	Deaths	Recovered	Active	Mortality Rate (per 100)
Tanzania	20	1	3	16	5
Maldives	19	0	13	6	0
Guinea-Bissau	18	0	0	18	0
Libya	18	1	0	17	5.56
Benin	16	0	2	14	0
Syria	16	2	2	12	12.5
Equatorial Guinea	16	0	1	15	0
Antigua and Barbuda	15	0	0	15	0
Mongolia	14	0	2	12	0
Namibia	14	0	3	11	0
Saint Lucia	14	0	1	13	0
Dominica	14	0	0	14	0
Fiji	12	0	0	12	0
Grenada	12	0	0	12	0
Liberia	10	1	3	6	10
Laos	10	0	0	10	0
Angola	10	2	2	6	20
Seychelles	10	0	0	10	0
Mozambique	10	0	1	9	0
Suriname	10	1	0	9	10
Sudan	10	2	2	6	20
Zimbabwe	9	1	0	8	11.11
Nepal	9	0	1	8	0
MS Zaandam	9	2	0	7	22.22
Chad	9	0	0	9	0
Eswatini	9	0	0	9	0
Saint Kitts and Nevis	9	0	0	9	0
Central African Republic	8	0	0	8	0
Holy See	7	0	0	7	0
Cabo Verde	7	1	0	6	14.29
Saint Vincent and the Grenadines	7	0	1	6	0
Somalia	7	0	1	6	0
Mauritania	6	1	2	3	16.67
Nicaragua	5	1	0	4	20
Bhutan	5	0	2	3	0

country	Confirmed	Deaths	Recovered	Active	Mortality Rate (per 100)
Malawi	4	0	0	4	0
Botswana	4	1	0	3	25
Gambia	4	1	2	1	25
Sierra Leone	4	0	0	4	0
Belize	4	0	0	4	0
Burundi	3	0	0	3	0
Timor-Leste	1	0	0	1	0
Papua New Guinea	1	0	0	1	0

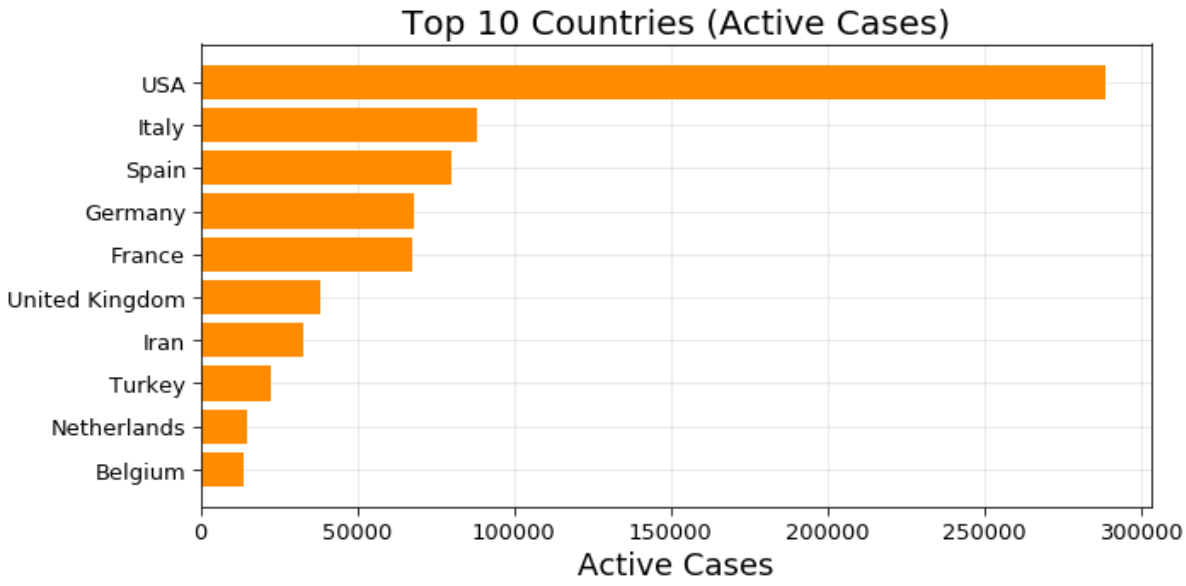
In [39]:
```
f = plt.figure(figsize=(10,5))
f.add_subplot(111)

plt.axes(axisbelow=True)
plt.barh(df_countries_cases.sort_values('Confirmed')["Confirmed"].index[-20:],
df_countries_cases.sort_values('Confirmed')["Confirmed"].values[-20:],color="d
arkcyan")
plt.tick_params(size=5,labelsize = 13)
plt.xlabel("Confirmed Cases",fontsize=18)
plt.title("Top 20 Countries (Confirmed Cases)",fontsize=20)
plt.grid(alpha=0.3)
plt.savefig(out+'Top 20 Countries (Confirmed Cases).png')
```



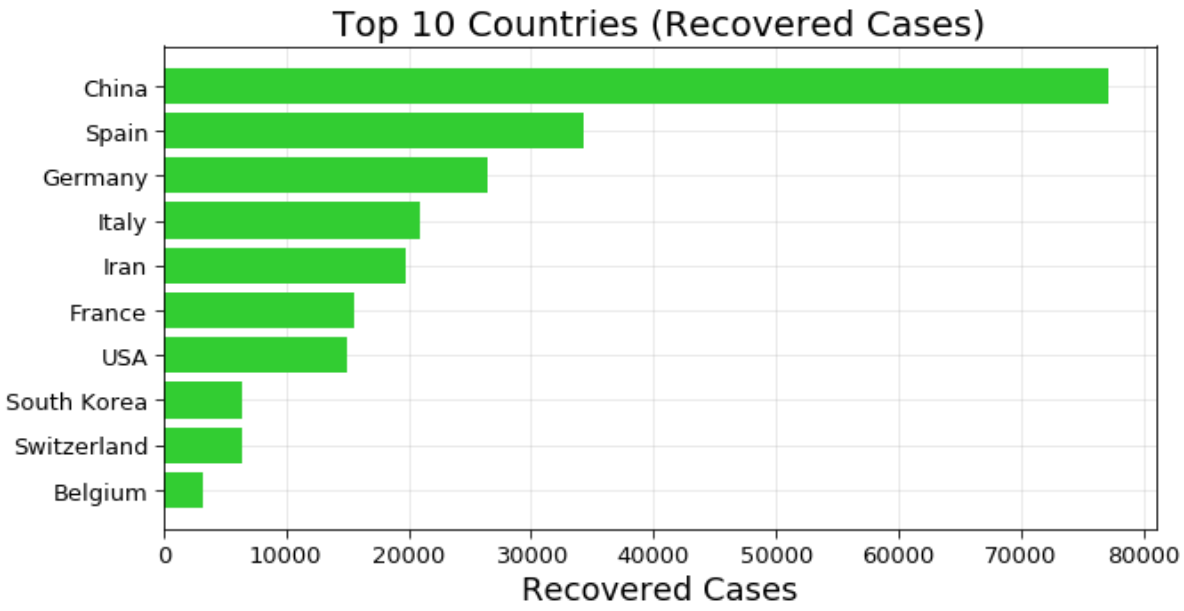
In [40]:
```
f = plt.figure(figsize=(10,5))
f.add_subplot(111)

plt.axes(axisbelow=True)
plt.barh(df_countries_cases.sort_values('Deaths')["Deaths"].index[-10:],df_cou
ntries_cases.sort_values('Deaths')["Deaths"].values[-10:],color="crimson")
plt.tick_params(size=5,labelsize = 13)
plt.xlabel("Deaths Cases",fontsize=18)
plt.title("Top 10 Countries (Deaths Cases)",fontsize=20)
plt.grid(alpha=0.3,which='both')
plt.savefig(out+'Top 10 Countries (Deaths Cases).png')
```



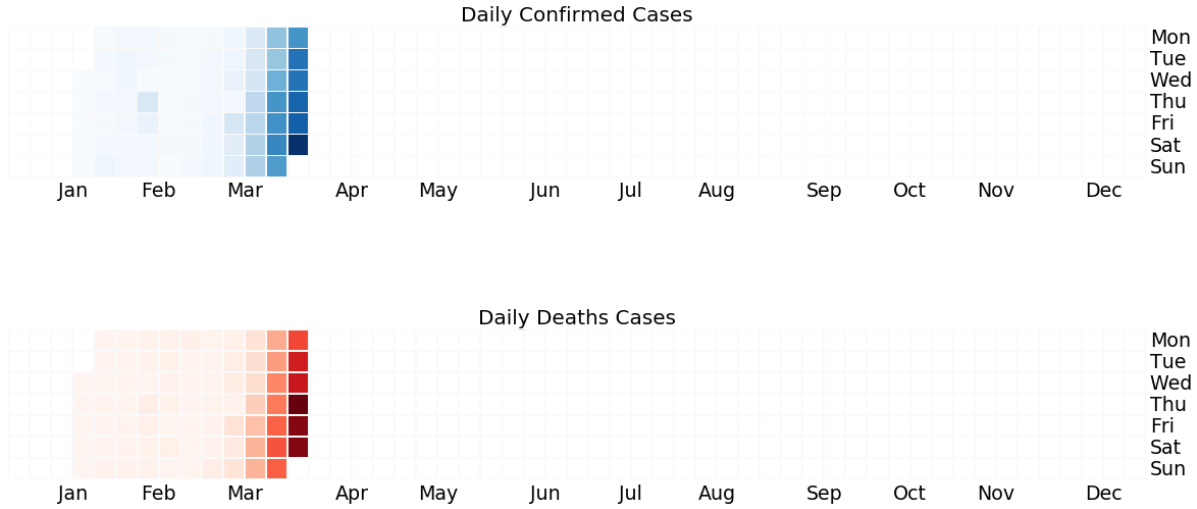
In [41]:
```
f = plt.figure(figsize=(10,5))
f.add_subplot(111)

plt.axes(axisbelow=True)
plt.barh(df_countries_cases.sort_values('Active')["Active"].index[-10:],df_cou
ntries_cases.sort_values('Active')["Active"].values[-10:],color="darkorange")
plt.tick_params(size=5,labelsize = 13)
plt.xlabel("Active Cases",fontsize=18)
plt.title("Top 10 Countries (Active Cases)",fontsize=20)
plt.grid(alpha=0.3,which='both')
plt.savefig(out+'Top 10 Countries (Active Cases).png')
```



In [42]:
```
f = plt.figure(figsize=(10,5))
f.add_subplot(111)

plt.axes(axisbelow=True)
plt.barh(df_countries_cases.sort_values('Recovered')["Recovered"].index[-10:],
df_countries_cases.sort_values('Recovered')["Recovered"].values[-10:],color="l
imegreen")
plt.tick_params(size=5,labelsize = 13)
plt.xlabel("Recovered Cases",fontsize=18)
plt.title("Top 10 Countries (Recovered Cases)",fontsize=20)
plt.grid(alpha=0.3,which='both')
plt.savefig(out+'Top 10 Countries (Recovered Cases).png')
```



In [46]:
```
f = plt.figure(figsize=(20,10))
f.add_subplot(2,1,1)
calmap.yearplot(df_table.groupby('Last_Update')['Confirmed'].sum().diff(), fil
lcolor='white', cmap='Blues', linewidth=0.5,linecolor="#fafafa",year=2020,)
plt.title("Daily Confirmed Cases",fontsize=20)
plt.tick_params(labelsize=19)

f.add_subplot(2,1,2)
calmap.yearplot(df_table.groupby('Last_Update')['Deaths'].sum().diff(), fillco
lor='white', cmap='Reds', linewidth=1,linecolor="#fafafa",year=2020,)
plt.title("Daily Deaths Cases",fontsize=20)
plt.tick_params(labelsize=19)
plt.show()
```



In [45]:
```
df_table[df_table['Last_Update']==df_table['Last_Update'].max()]
```

Out[45]:

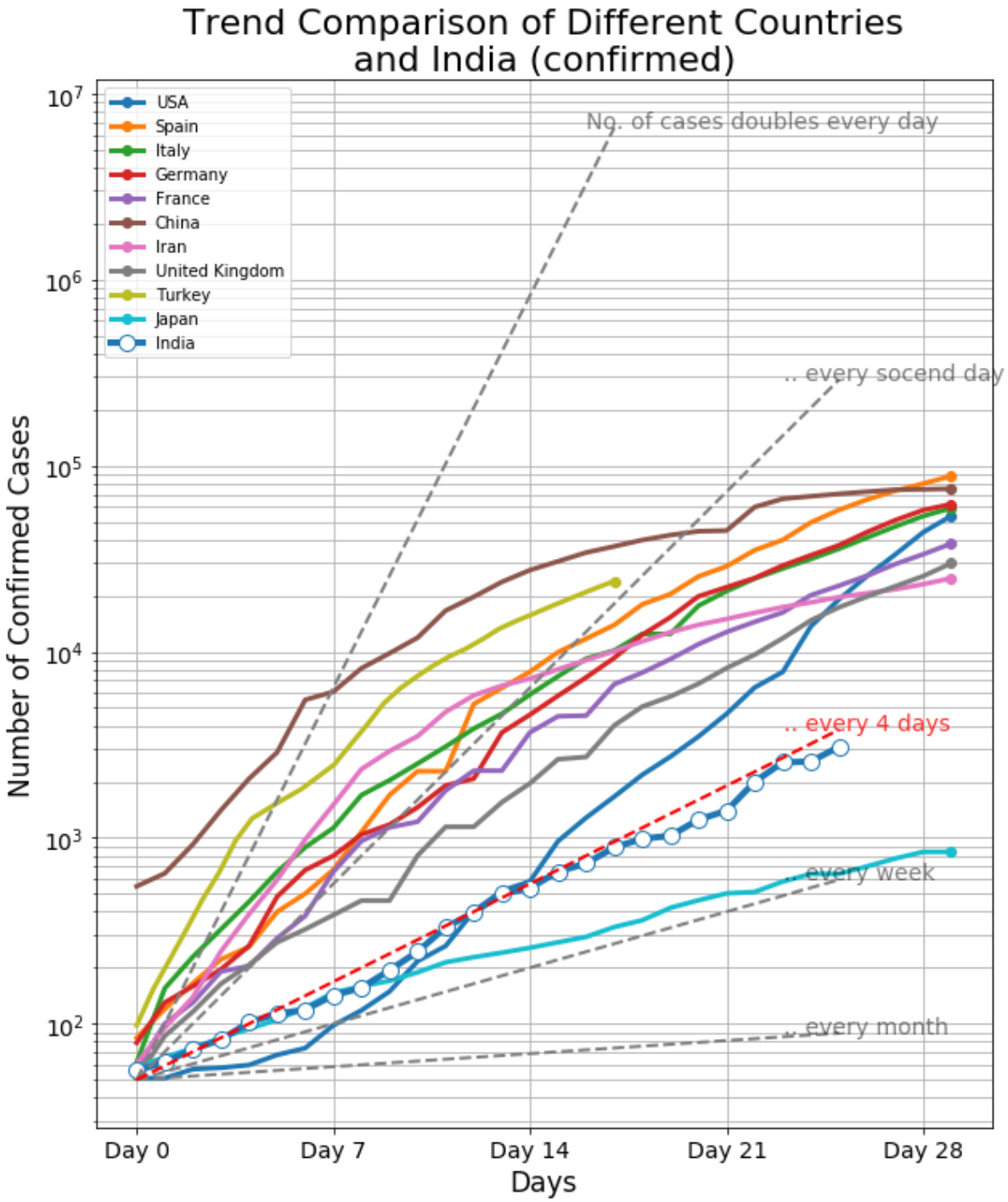
	Country_Region	continent	Last_Update	Confirmed	Deaths	Recovered	Active	Delta_C
73	Afghanistan	Asia	2020-04-04	299	7	NaN	NaN	
147	Albania	Europe	2020-04-04	333	20	NaN	NaN	
221	Algeria	Africa	2020-04-04	1251	130	NaN	NaN	
295	Andorra	Europe	2020-04-04	466	17	NaN	NaN	
369	Angola	Africa	2020-04-04	10	2	NaN	NaN	
443	Antigua and Barbuda	North America	2020-04-04	15	0	NaN	NaN	
517	Argentina	South America	2020-04-04	1451	43	NaN	NaN	
591	Armenia	Asia	2020-04-04	770	7	NaN	NaN	
665	Australia	Australia	2020-04-04	5550	30	NaN	NaN	
739	Austria	Europe	2020-04-04	11781	186	NaN	NaN	
813	Azerbaijan	Asia	2020-04-04	521	5	NaN	NaN	
887	Bahamas	North America	2020-04-04	28	4	NaN	NaN	
961	Bahrain	Asia	2020-04-04	688	4	NaN	NaN	
1035	Bangladesh	Asia	2020-04-04	70	8	NaN	NaN	
1109	Barbados	North America	2020-04-04	52	0	NaN	NaN	
1183	Belarus	Europe	2020-04-04	440	5	NaN	NaN	
1257	Belgium	Europe	2020-04-04	18431	1283	NaN	NaN	
1331	Belize	North America	2020-04-04	4	0	NaN	NaN	
1405	Benin	Africa	2020-04-04	16	0	NaN	NaN	
1479	Bhutan	Asia	2020-04-04	5	0	NaN	NaN	
1553	Bolivia	South America	2020-04-04	139	10	NaN	NaN	
1627	Bosnia and Herzegovina	Europe	2020-04-04	624	21	NaN	NaN	
1701	Botswana	Africa	2020-04-04	4	1	NaN	NaN	
1775	Brazil	South America	2020-04-04	10360	445	NaN	NaN	
1849	Brunei	Asia	2020-04-04	135	1	NaN	NaN	
1923	Bulgaria	Europe	2020-04-04	503	17	NaN	NaN	
1997	Burkina Faso	Africa	2020-04-04	318	16	NaN	NaN	
2145	Burundi	Africa	2020-04-04	3	0	NaN	NaN	
2219	Cabo Verde	Africa	2020-04-04	7	1	NaN	NaN	
2293	Cambodia	Asia	2020-04-04	114	0	NaN	NaN	
...	...	...	...	...	...	...	...	

	Country_Region	continent	Last_Update	Confirmed	Deaths	Recovered	Active	Delta_C
11099	Singapore	Asia	2020-04-04	1189	6	NaN	NaN	
11173	Slovakia	Europe	2020-04-04	471	1	NaN	NaN	
11247	Slovenia	Europe	2020-04-04	977	22	NaN	NaN	
11321	Somalia	Africa	2020-04-04	7	0	NaN	NaN	
11395	South Africa	Africa	2020-04-04	1585	9	NaN	NaN	
11469	Spain	Europe	2020-04-04	126168	11947	NaN	NaN	
11543	Sri Lanka	Asia	2020-04-04	166	5	NaN	NaN	
11617	Sudan	Africa	2020-04-04	10	2	NaN	NaN	
11691	Suriname	South America	2020-04-04	10	1	NaN	NaN	
11765	Sweden	Europe	2020-04-04	6443	373	NaN	NaN	
11839	Switzerland	Europe	2020-04-04	20505	666	NaN	NaN	
11913	Syria	Asia	2020-04-04	16	2	NaN	NaN	
11987	Taiwan	Asia	2020-04-04	355	5	NaN	NaN	
12061	Tanzania	Africa	2020-04-04	20	1	NaN	NaN	
12135	Thailand	Asia	2020-04-04	2067	20	NaN	NaN	
12283	Togo	Africa	2020-04-04	41	3	NaN	NaN	
12357	Trinidad and Tobago	North America	2020-04-04	103	6	NaN	NaN	
12431	Tunisia	Africa	2020-04-04	553	18	NaN	NaN	
12505	Turkey	Asia	2020-04-04	23934	501	NaN	NaN	
12579	USA	North America	2020-04-04	308850	8407	NaN	NaN	
12653	Uganda	Africa	2020-04-04	48	0	NaN	NaN	
12727	Ukraine	Europe	2020-04-04	1225	32	NaN	NaN	
12801	United Arab Emirates	Asia	2020-04-04	1505	10	NaN	NaN	
12875	United Kingdom	Europe	2020-04-04	42477	4320	NaN	NaN	
12949	Uruguay	South America	2020-04-04	400	5	NaN	NaN	
13023	Uzbekistan	Asia	2020-04-04	266	2	NaN	NaN	
13097	Venezuela	South America	2020-04-04	155	7	NaN	NaN	
13171	Vietnam	Asia	2020-04-04	240	0	NaN	NaN	
13319	Zambia	Africa	2020-04-04	39	1	NaN	NaN	
13393	Zimbabwe	Africa	2020-04-04	9	1	NaN	NaN	

174 rows × 9 columns

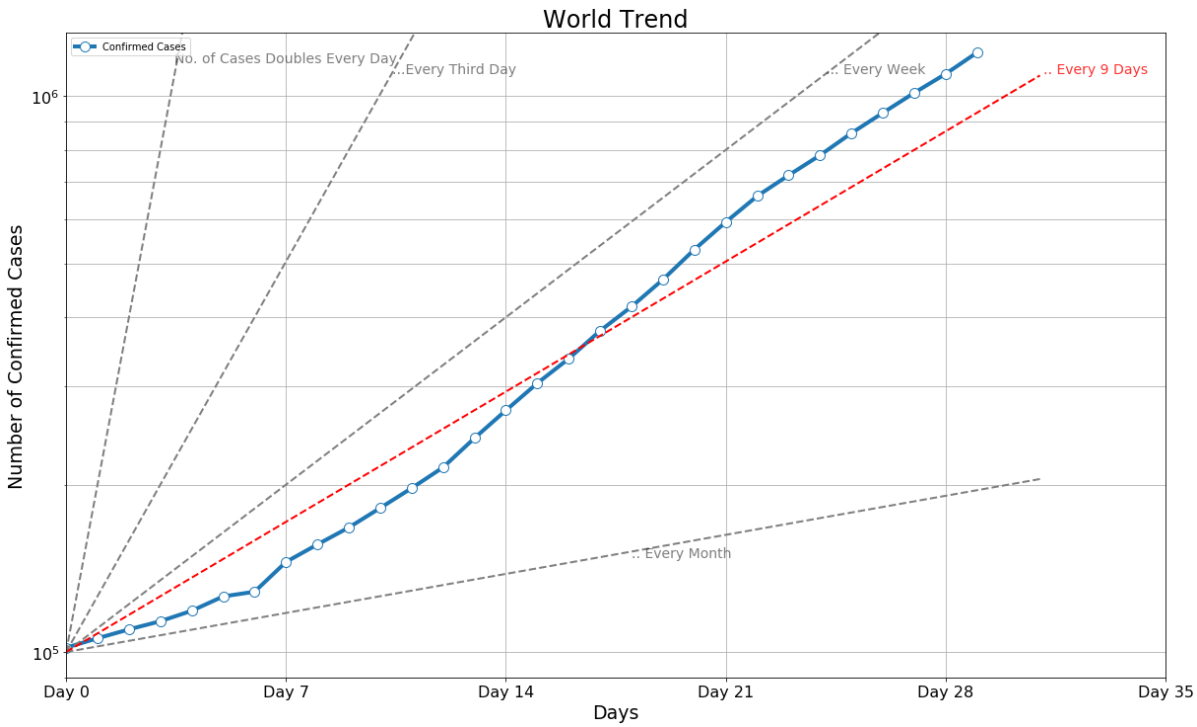
In [49]:
```
temp = df_confirmed.groupby('country').sum().drop(["Lat","Long"],axis =1).sort
_values(df_confirmed.columns[-1], ascending= False)

threshold = 50
f = plt.figure(figsize=(10,12))
ax = f.add_subplot(111)
for i,country in enumerate(temp.index):
    if i >= 9:
        if country != "India" and country != "Japan" :
            continue
    x = 30
    t = temp.loc[temp.index== country].values[0]
    t = t[t>threshold][:x]

    date = np.arange(0,len(t[:x]))
    xnew = np.linspace(date.min(), date.max(), 30)
    spl = make_interp_spline(date, t, k=1)  # type: BSpline
    power_smooth = spl(xnew)
    if country != "India":
        plt.plot(xnew,power_smooth,'-o',label = country,linewidth =3, markever
y=[-1])
    else:
        marker_style = dict(linewidth=4, linestyle='-', marker='o',markersize=
10, markerfacecolor='#ffffff')
        plt.plot(date,t,"-.",label = country,**marker_style)

plt.tick_params(labelsize = 14)
plt.xticks(np.arange(0,30,7),[ "Day "+str(i) for i in range(30)][::7])

# Reference lines
x = np.arange(0,18)
y = 2**(x+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate("No. of cases doubles every day",(x[-2],y[-1]),xycoords="data",fo
ntsize=14,alpha = 0.5)

x = np.arange(0,26)
y = 2**(x/2+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate(".. every socend day",(x[-3],y[-1]),xycoords="data",fontsize=14,a
lpha = 0.5)

x = np.arange(0,26)
y = 2**(x/7+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate(".. every week",(x[-3],y[-1]),xycoords="data",fontsize=14,alpha =
0.5)

x = np.arange(0,26)
y = 2**(x/30+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate(".. every month",(x[-3],y[-1]),xycoords="data",fontsize=14,alpha
= 0.5)


# India is following trend similar to doulbe the cases in 4 days but it may in
```

```
crease the rate
x = np.arange(0,26)
y = 2**(x/4+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "Red")
plt.annotate(".. every 4 days",(x[-3],y[-1]),color="Red",xycoords="data",fonts
ize=14,alpha = 0.8)

# plot Params
plt.xlabel("Days",fontsize=17)
plt.ylabel("Number of Confirmed Cases",fontsize=17)
plt.title("Trend Comparison of Different Countries\n and India (confirmed) ",f
ontsize=22)
plt.legend(loc = "upper left")
plt.yscale("log")
plt.grid(which="both")
plt.savefig(out+'Trend Comparison with India (confirmed).png')
plt.show()
```


Trend Comparison of Different Countries and India (confirmed)
USA
Spain
Italy
Germany
France
China
Iran
United Kingdom
Turkey
Japan
India
No. of cases doubles every day
.. every socend day
.. every 4 days
.. every week
.. every month
Number of Confirmed Cases
Days
Day 0
Day 7
Day 14
Day 21
Day 28

In [50]:
```
temp_data = df_confirmed.iloc[:,5:].sum(axis =0)
f = plt.figure(figsize=(20,12))
f.add_subplot(111)

threshold = 100000

t = temp_data.values
t = t[t >threshold]

date = np.arange(0,len(t[:]))
xnew = np.linspace(date.min(), date.max(), 10)
spl = make_interp_spline(date, t, k=1)  # type: BSpline
power_smooth = spl(xnew)

marker_style = dict(linewidth=4, linestyle='-', marker='o',markersize=10, mark
erfacecolor='#ffffff')
plt.plot(date,t,"-.",label="Confirmed Cases",**marker_style)


# Reference lines
x = np.arange(0,32)
y = 2**(x+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate("No. of Cases Doubles Every Day",(np.log2((t.max()-threshold)/thr
eshold),t.max()-threshold/2),xycoords="data",fontsize=14,alpha = 0.5)

x = np.arange(0,32)
y = 2**(x/3+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate("...Every Third Day",(np.log2((t.max()-threshold)/threshold)*3,t.
max()-threshold),xycoords="data",fontsize=14,alpha = 0.5)

x = np.arange(0,32)
y = 2**(x/7+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate("... Every Week",(np.log2((t.max()-threshold)/threshold)*7,t.max
()-threshold),xycoords="data",fontsize=14,alpha = 0.5)

x = np.arange(0,32)
y = 2**(x/30+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "gray")
plt.annotate(".. Every Month",(18,2**(17/30+np.log2(threshold))),xycoords="dat
a",fontsize=14,alpha = 0.5)

x = np.arange(0,32)
y = 2**(x/9+np.log2(threshold))
plt.plot(x,y,"--",linewidth =2,color = "Red")
plt.annotate(".. Every 9 Days",(np.log2((t.max()-threshold)/threshold)*9,t.max
()-threshold),color="Red",xycoords="data",fontsize=14,alpha = 0.8)


plt.xlim(date[0],date[-1])
plt.ylim(threshold - threshold/10,t.max()+threshold)
# plot Params
# plt.tight_layout()
plt.tick_params(labelsize = 16)
```

```
plt.xticks(np.arange(0,len(t[:])+7,7),[ "Day "+str(i) for i in range(len(t[:])
+7)][::7])
plt.xlabel("Days",fontsize=19)
plt.ylabel("Number of Confirmed Cases",fontsize=19)
plt.title("World Trend",fontsize=24)
plt.legend(loc = "upper left")
plt.yscale("log")
plt.grid(which="both")
plt.savefig(out+"World Trend Confirmed cases.png")
plt.show()
```



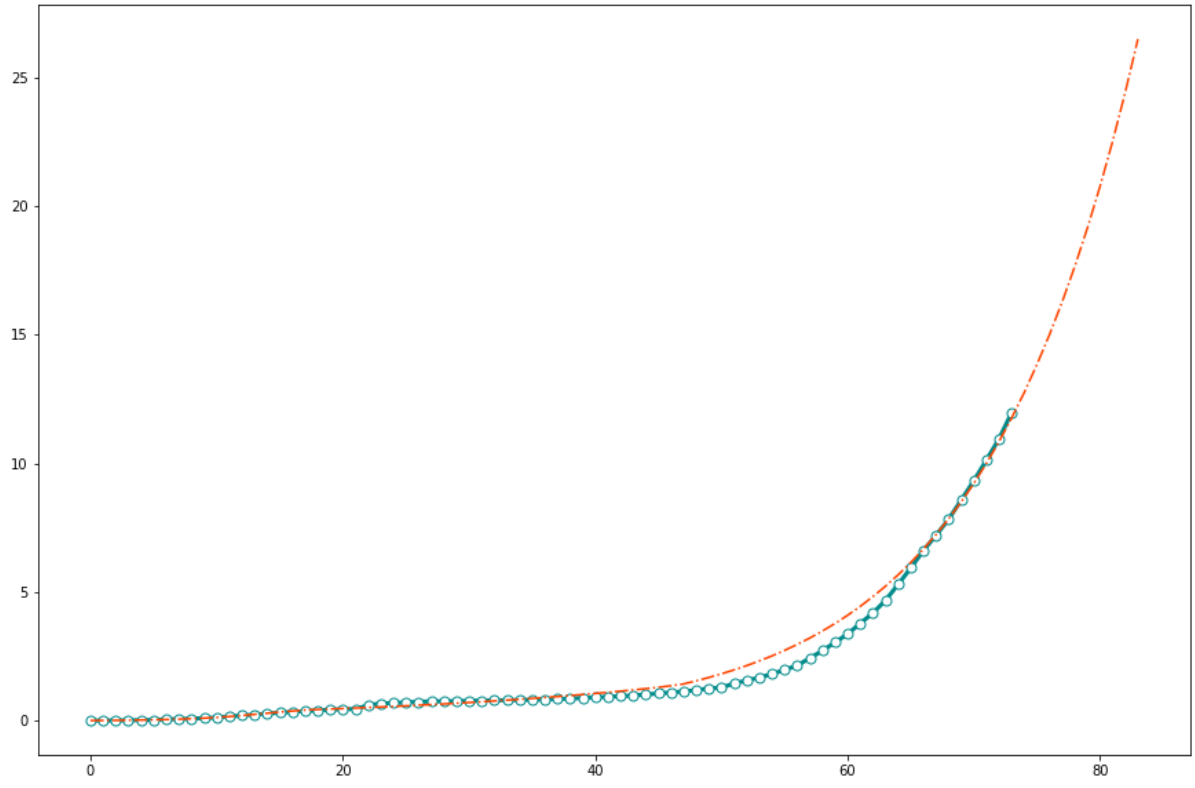
In [57]:
```
data_y = np.log10(np.asarray(df_confirmed.sum()[5:]).astype("float32"))
data_x = np.arange(1,len(data_y)+1)
model = models.load_model("model_confirmed.h5")
model.summary()
```

```
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 1)                 0
_________________________________________________________________
Dense_l1 (Dense)             (None, 80)                160
_________________________________________________________________
LRelu_l1 (LeakyReLU)         (None, 80)                0
_________________________________________________________________
Dense_l2 (Dense)             (None, 80)                6480
_________________________________________________________________
LRelu_l2 (LeakyReLU)         (None, 80)                0
_________________________________________________________________
Dense_l3 (Dense)             (None, 1)                 81
_________________________________________________________________
Output (LeakyReLU)           (None, 1)                 0
=================================================================
Total params: 6,721
Trainable params: 6,721
Non-trainable params: 0
_________________________________________________________________
```

In [58]:
```
lakh = 100000
prediction_days = 10
data = np.power(10,model.predict(np.arange(1,len(data_y)+prediction_days+1)))
temp_data = df_confirmed.iloc[:,5:].sum(axis =0)
f = plt.figure(figsize=(15,10))
ax = f.add_subplot(111)

date = np.arange(0,len(temp_data))

marker_style = dict(linewidth=3, linestyle='-', marker='o',markersize=7, marke
rfacecolor='#ffffff')
plt.plot(date,temp_data/lakh,"-.",color="darkcyan",**marker_style, label="Actu
al Curve")

date = np.arange(0,len(data))
plt.plot(date,data/lakh,"-.",color="orangered",label="Predicted Curve")

nextdays = [(datetime.strptime(d[-1],'%d %b')+timedelta(days=i)).strftime("%d
 %b") for i in range(1,prediction_days+1)]
total = d + nextdays

text = "Prediction for next "+str(prediction_days) +" days:\n"
for i in range(prediction_days):
    text += nextdays[i]+" : "+str(np.round(data[-1*(prediction_days-i)],-3)[0]
/lakh)+" L\n"

plt.text(0.02, 0.78, text, fontsize=17, horizontalalignment='left', verticalal
ignment='top', transform=ax.transAxes,bbox=dict(facecolor='white', alpha=0.4))

# X-axis
plt.xticks(list(np.arange(0,len(total),int(len(total)/5))),d[:-1:int(len(total
)/5)]+[total[-1]])

# Tick-Parameters
ax.xaxis.set_minor_locator(ticker.AutoMinorLocator())
ax.yaxis.set_minor_locator(ticker.AutoMinorLocator())
ax.tick_params(which='both', width=1,labelsize=14)
ax.tick_params(which='major', length=6)
ax.tick_params(which='minor', length=3, color='0.8')

# Grid
plt.grid(lw = 1, ls = '-', c = "0.7", which = 'major')
plt.grid(lw = 1, ls = '-', c = "0.9", which = 'minor')

# Plot Title
plt.title("COVID-19 Next 10 day Prediction Curve-Global Confirmed Cases",{'fon
tsize':22})

# Axis Lable
plt.xlabel("Date",fontsize =18)
plt.ylabel("Number of Confirmed Cases (Lakh)",fontsize =18)

plt.yscale("log")
plt.legend(fontsize =18)
plt.tick_params(labelsize = 13)
```

```
plt.savefig(out+"Prediction Curve-Confirmed.png")
plt.show()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-58-d90dcc563d02> in <module>
     14 plt.plot(date,data/lakh,"-.",color="orangered",label="Predicted Curv
e")
     15
---> 16 nextdays = [(datetime.strptime(d[-1],'%d %b')+timedelta(days=i)).strf
time("%d %b") for i in range(1,prediction_days+1)]
     17 total = d + nextdays
     18

<ipython-input-58-d90dcc563d02> in <listcomp>(.0)
     14 plt.plot(date,data/lakh,"-.",color="orangered",label="Predicted Curv
e")
     15
---> 16 nextdays = [(datetime.strptime(d[-1],'%d %b')+timedelta(days=i)).strf
time("%d %b") for i in range(1,prediction_days+1)]
     17 total = d + nextdays
     18

NameError: name 'd' is not defined
```



In [ ]: