Madrid_Pain_Graphs

August 3, 2020

1 Informes de la comunidad de Madrid

Actualizado diariamente, este documento se visualiza mejor aquí.

Datos de la situación de la infección por coronavirus en la Comunidad de Madrid.

Nos descargamos los datos, agrupamos, y calculamos :

- Gráfico de seguimiento.
- Muertes medias diarias, últimos 7 días.
- Muertes medias diarias desde que la comunidad de Madrid publica datos.

[1]:
```
# Miramos si hay nuevos datos a descargar.

!# cd ../data/; FILELIST=" 200509 200508 200507 200506 200505 200504 200503␣
↪200502 200501 200430 200429 200428 200427 200426 200425 200424 200423 200422␣
↪200510 200511 200512 200513 200514 200515 200516 200517 200518 200519 200520␣
↪200521 200522 200523 200524 200525 200526 200527 200528 200529 200530 200609␣
↪200608 200607 200606 200605 200604 200603 200602 200601 200610 200611 200612␣
↪200613 200614 200615 200616 200617 200618 200619 200620 200621 200622 200623␣
↪200624 200625 200626 200627 200628 200629 200630 " ; for fecha in `echo␣
↪$FILELIST`  ; do  FILE=${fecha}_cam_covid19.pdf ;  [ ! -f ../data/${FILE} ]␣
↪&& echo $FILE::::: && wget https://www.comunidad.madrid/sites/default/
↪files/doc/sanidad/$FILE 1>/dev/null  2>/dev/null && ls -altr $FILE ; done

# Miramos solo hoy y los ultimos diez dias
! cd ../data/; FILELIST=`seq -w 0 10 | while read i ; do date +%y%m%d -d "$i␣
↪day ago" ; done` ; for fecha in `echo $FILELIST`  ; do ␣
↪FILE=${fecha}_cam_covid19.pdf ;  [ ! -f ../data/${FILE} ] && echo $FILE:::::␣
↪   && wget https://www.comunidad.madrid/sites/default/files/doc/sanidad/
↪$FILE 1>/dev/null  2>/dev/null && ls -altr $FILE ; done
```

```
200803_cam_covid19.pdf:::::
200802_cam_covid19.pdf:::::
200801_cam_covid19.pdf:::::
200726_cam_covid19.pdf:::::
200725_cam_covid19.pdf:::::
```

[ ]:
```
https://www.comunidad.madrid/sites/default/files/doc/sanidad/200803cam_covid19.
↪pdf
https://www.comunidad.madrid/sites/default/files/doc/sanidad/200803_cam_covid19.
↪pdf
```

[1]:
```
from tabula import read_pdf
from IPython.display import display, HTML
import os
import pandas as pd
import glob
import re
from tqdm.notebook import tqdm
import warnings
warnings.filterwarnings('ignore')


os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.141-1.b16.
↪el7_3.x86_64/jre"

# Auxiliary functions
from datetime import datetime, date, time, timedelta

""" Rellenar dias vacios con interpolacion"""
def interpolate_dataframe(df,freq):
    if freq == 'H':
        rng = pd.date_range(df.index.min(), df.index.max() + pd.Timedelta(23,␣
↪'H'), freq='H')
    elif freq == 'D' :
        rng = pd.date_range(
                        datetime.strptime(str(df.index.min())[:10]+' 00:00:00',␣
↪"%Y-%m-%d %H:%M:%S") ,
                        datetime.strptime(str(df.index.max())[:10]+' 00:00:00',␣
↪"%Y-%m-%d %H:%M:%S"),
                        freq='D')
        df.index = pd.to_datetime(df.index)
    df2 = df.reindex(rng)
    df = df2
    for column in df.columns :
        s = pd.Series(df[column])
        s.interpolate(method="quadratic", inplace =True)
        df[column] = pd.DataFrame([s]).T
    return df

def fet_daily_date_new_format(fecha):
    df_pdf =  read_pdf('../data/'+fecha+'_cam_covid19.pdf',area=(000, 600, 400,␣
↪800) , pages='1')
    df = df_pdf[0]
```

```
    df = df['Unnamed: 0'].astype(str).str.replace(r".", '').replace("(", ' ')
    df = df.T
    df.columns = df.iloc[0]
    df = df.iloc[1:]


    df = pd.DataFrame(data=df)
    df

    dict = {}
    dict['HOSPITALES'] = df[df['Unnamed: 0'].str.contains('Hospitales')].
↪iloc[0]['Unnamed: 0'].split(' ')[0]
    dict['DOMICILIOS'] = df[df['Unnamed: 0'].str.contains('Domicilios')].
↪iloc[0]['Unnamed: 0'].split(' ')[0]
    dict['CENTROS SOCIOSANITARIOS'] = df[df['Unnamed: 0'].str.
↪contains('Centros')].iloc[0]['Unnamed: 0'].split(' ')[0]
    dict['OTROS LUGARES'] = df[df['Unnamed: 0'].str.contains('otros')].
↪iloc[0]['Unnamed: 0'].split(' ')[0]

    cadena_a_parsear = df[df['Unnamed: 0'].str.contains('otal')].
↪iloc[0]['Unnamed: 0']

    dict['FALLECIDOS TOTALES'] = re.search(r'(\d+)', cadena_a_parsear)[0]

    df = pd.DataFrame.from_dict(dict, orient='index').T
    df['Fecha'] =  pd.to_datetime(fecha, format='%y%m%d')
    df.set_index('Fecha', inplace=True, drop=True)
    return df


def get_daily_data(fecha):
    if fecha > '200512' :
        return fet_daily_date_new_format(fecha)

    col2str = {'dtype': str}
    kwargs = {'output_format': 'dataframe',
              'pandas_options': col2str,
              'stream': True}
    df_pdf =  read_pdf('../data/'+fecha+'_cam_covid19.
↪pdf',pages='1',multiple_tables = True,**kwargs)


    df = df_pdf[0]

    df = df[df['Unnamed: 0'].notna()]
```

```
    df = df[(df['Unnamed: 0']=='HOSPITALES') | (df['Unnamed: 0'] ==␣
↪'DOMICILIOS') | (df['Unnamed: 0'] == 'CENTROS SOCIOSANITARIOS') |␣
↪(df['Unnamed: 0'] == 'OTROS LUGARES') | (df['Unnamed: 0'] == 'FALLECIDOS␣
↪TOTALES')]
    df = df[['Unnamed: 0','Unnamed: 2']]
    df['Unnamed: 2'] = df['Unnamed: 2'].astype(str).str.replace(r".", '')
    df = df.T
    df.columns = df.iloc[0]
    df = df.iloc[1:]

    df['Fecha'] =  pd.to_datetime(fecha, format='%y%m%d')
    df = df.rename_axis(None)

    df.set_index('Fecha', inplace=True, drop=True)
    df.index
    df.dropna()
    #df = df.T
    return df

def get_all_data( ):
    #BLACKLIST = ["200429","200422"]
    #BLACKLIST = ["200514",]
    BLACKLIST = []
    df = pd.DataFrame()
    list_df = []

    pdf_list= sorted(glob.glob('../data/*_cam_covid19.pdf'),
                     key=os.path.getmtime,
                     reverse=True )

    #for pdf_file in pdf_list:

    for pdf_file in tqdm(pdf_list,
                         desc="Procesando pdfs diarios"):
        # extract fecha from username , eg : ../data/2200422_cam_covid19.pdf
        fecha = pdf_file.split('/')[2].split('_')[0]
        if fecha not in BLACKLIST:
            #print("processing", fecha)
            df = get_daily_data(fecha)
        list_df.append(df)

    df = pd.concat(list_df)
    df = df.astype(int)
    df = df.drop_duplicates()

    df = df.sort_values(by=['Fecha'], ascending=True)
    ###jaime
```

```
    #df = interpolate_dataframe(df,'D')
    #df.index.name = 'Fecha'

    df['HOSPITALES hoy'] = df['HOSPITALES'] - df['HOSPITALES'].shift(1)
    df['CENTROS SOCIOSANITARIOS hoy'] = df['CENTROS SOCIOSANITARIOS'] -␣
↪df['CENTROS SOCIOSANITARIOS'].shift(1)
    df['FALLECIDOS TOTALES hoy'] = df['FALLECIDOS TOTALES'] - df['FALLECIDOS␣
↪TOTALES'].shift(1)

    df = df.sort_values(by=['Fecha'], ascending=False)

    return df

total =  get_all_data()
total.to_csv('/root/kaggle/covid19-madrid/madrid_results.csv')
```

```
HBox(children=(FloatProgress(value=0.0, description='Procesando pdfs diarios', max=93.0, style=
```

```
Got stderr: ago 03, 2020 5:02:25 PM
org.apache.pdfbox.pdmodel.font.PDCIDFontType2 <init>
INFORMACIÓN: OpenType Layout tables used in font CIDFont+F1 are not implemented
in PDFBox and will be ignored
ago 03, 2020 5:02:25 PM org.apache.pdfbox.pdmodel.font.PDCIDFontType2 <init>
INFORMACIÓN: OpenType Layout tables used in font CIDFont+F2 are not implemented
in PDFBox and will be ignored
ago 03, 2020 5:02:25 PM org.apache.pdfbox.pdmodel.font.PDCIDFontType2 <init>
INFORMACIÓN: OpenType Layout tables used in font CIDFont+F3 are not implemented
in PDFBox and will be ignored
ago 03, 2020 5:02:25 PM org.apache.pdfbox.pdmodel.font.PDCIDFontType2 <init>
INFORMACIÓN: OpenType Layout tables used in font CIDFont+F1 are not implemented
in PDFBox and will be ignored
ago 03, 2020 5:02:25 PM org.apache.pdfbox.pdmodel.font.PDCIDFontType2 <init>
INFORMACIÓN: OpenType Layout tables used in font CIDFont+F2 are not implemented
in PDFBox and will be ignored
ago 03, 2020 5:02:25 PM org.apache.pdfbox.pdmodel.font.PDCIDFontType2 <init>
INFORMACIÓN: OpenType Layout tables used in font CIDFont+F3 are not implemented
in PDFBox and will be ignored
ago 03, 2020 5:02:25 PM org.apache.pdfbox.pdmodel.font.PDCIDFontType2 <init>
INFORMACIÓN: OpenType Layout tables used in font CIDFont+F1 are not implemented
in PDFBox and will be ignored
ago 03, 2020 5:02:26 PM org.apache.pdfbox.pdmodel.font.PDCIDFontType2 <init>
INFORMACIÓN: OpenType Layout tables used in font CIDFont+F2 are not implemented
in PDFBox and will be ignored
ago 03, 2020 5:02:26 PM org.apache.pdfbox.pdmodel.font.PDCIDFontType2 <init>
INFORMACIÓN: OpenType Layout tables used in font CIDFont+F3 are not implemented
in PDFBox and will be ignored
```

[3]:
```
interpolate_dataframe(total,'D')
```

[3]:
```
Unnamed: 0   HOSPITALES  DOMICILIOS  CENTROS SOCIOSANITARIOS  OTROS LUGARES  \
2020-04-22  7144.000000  761.000000              3932.000000           15.0
2020-04-23  7271.000000  769.000000              3996.000000           20.0
2020-04-24  7388.000000  775.000000              4068.000000           21.0
2020-04-25  7633.000000  788.000000              4170.000000           21.0
2020-04-26  7800.000000  798.000000              4236.000000           21.0
…                 …         …                       …             …
2020-07-27  9419.757452  922.426096              4827.749304           28.0
2020-07-28  9420.000000  922.000000              4828.000000           28.0
2020-07-29  9420.040425  921.428984              4828.541783           28.0
2020-07-30  9420.000000  922.000000              4829.000000           28.0
2020-07-31  9420.000000  925.000000              4829.000000           28.0

Unnamed: 0  FALLECIDOS TOTALES  HOSPITALES hoy  CENTROS SOCIOSANITARIOS hoy  \
2020-04-22        11852.000000             NaN                          NaN
2020-04-23        12056.000000      127.000000                    64.000000
2020-04-24        12252.000000      117.000000                    72.000000
2020-04-25        12612.000000      245.000000                   102.000000
2020-04-26        12855.000000      167.000000                    66.000000
…                          …               …                            …
2020-07-27        15197.932851        6.466553                    -0.323286
2020-07-28        15198.000000        5.000000                     0.000000
2020-07-29        15198.011192        2.255575                     0.720548
2020-07-30        15199.000000        0.000000                     1.000000
2020-07-31        15202.000000        0.000000                     0.000000

Unnamed: 0  FALLECIDOS TOTALES hoy
2020-04-22                     NaN
2020-04-23              204.000000
2020-04-24              196.000000
2020-04-25              360.000000
2020-04-26              243.000000
…                              …
2020-07-27                6.764807
2020-07-28                5.000000
2020-07-29                2.372532
2020-07-30                1.000000
2020-07-31                3.000000

[101 rows x 8 columns]
```

[2]:
```
    df_pdf =  read_pdf('../data/'+fecha+'_cam_covid19.pdf',area=(000, 600, 400,␣
↪800) , pages='1')
```

```
    df = df_pdf[0]
    df = df['Unnamed: 0'].astype(str).str.replace(r".", '').replace("(", ' ')
    df = df.T
    df.columns = df.iloc[0]
    df = df.iloc[1:]


    df = pd.DataFrame(data=df)
    df

    dict = {}
    dict['HOSPITALES'] = df[df['Unnamed: 0'].str.contains('Hospitales')].
↪iloc[0]['Unnamed: 0'].split(' ')[0]
    dict['DOMICILIOS'] = df[df['Unnamed: 0'].str.contains('Domicilios')].
↪iloc[0]['Unnamed: 0'].split(' ')[0]
    dict['CENTROS SOCIOSANITARIOS'] = df[df['Unnamed: 0'].str.
↪contains('Centros')].iloc[0]['Unnamed: 0'].split(' ')[0]
    dict['OTROS LUGARES'] = df[df['Unnamed: 0'].str.contains('otros')].
↪iloc[0]['Unnamed: 0'].split(' ')[0]

    cadena_a_parsear = df[df['Unnamed: 0'].str.contains('otal')].
↪iloc[0]['Unnamed: 0']

    dict['FALLECIDOS TOTALES'] = re.search(r'(\d+)', cadena_a_parsear)[0]

    df = pd.DataFrame.from_dict(dict, orient='index').T
    df['Fecha'] =  pd.to_datetime(fecha, format='%y%m%d')
    df.set_index('Fecha', inplace=True, drop=True)
```

[2]:

Unnamed: 0	HOSPITALES	DOMICILIOS	CENTROS SOCIOSANITARIOS	OTROS LUGARES	\
Fecha					
2020-07-31	9420	925	4829	28	
2020-07-30	9420	922	4829	28	
2020-07-28	9420	922	4828	28	
2020-07-24	9415	922	4828	28	
2020-07-23	9411	921	4828	28	
2020-07-21	9409	921	4828	28	
2020-07-17	9403	920	4827	28	
2020-07-16	9401	920	4827	28	
2020-07-14	9394	918	4825	28	
2020-07-10	9390	917	4825	28	
2020-07-09	9384	916	4823	28	
2020-07-07	9382	916	4823	28	
2020-07-03	9378	913	4823	28	
2020-07-02	9369	913	4819	28	
2020-07-01	9367	911	4816	28	
2020-06-30	9357	911	4815	28	

2020-06-28	9351	910	4815	28
2020-06-25	9349	910	4815	28
2020-06-23	9337	906	4813	28
2020-06-21	9319	905	4808	28
2020-06-18	9297	905	4804	28
2020-06-16	9270	902	4801	28
2020-06-14	9249	900	4795	28
2020-06-11	9240	898	4789	28
2020-06-09	9205	897	4781	27
2020-06-07	9184	892	4775	27
2020-06-04	9165	888	4768	27
2020-06-02	9098	881	4747	27
2020-05-31	9074	878	4739	27
2020-05-28	9044	876	4724	27
2020-05-26	8988	870	4696	27
2020-05-24	8907	860	4623	27
2020-05-21	8820	848	4554	24
2020-05-19	8748	847	4525	24
2020-05-17	8640	844	4510	24
2020-05-14	8573	843	4491	24
2020-05-12	8521	840	4472	24
2020-05-11	8404	838	4438	24
2020-05-08	8321	835	4405	24
2020-05-06	8266	834	4377	24
2020-05-04	8203	827	4355	24
2020-04-30	8136	823	4338	24
2020-04-29	7958	806	4295	21
2020-04-27	7881	801	4273	21
2020-04-26	7800	798	4236	21
2020-04-25	7633	788	4170	21
2020-04-24	7388	775	4068	21
2020-04-23	7271	769	3996	20
2020-04-22	7144	761	3932	15

Unnamed: 0	FALLECIDOS TOTALES	HOSPITALES hoy	CENTROS SOCIOSANITARIOS hoy	\
Fecha				
2020-07-31	15202	0.0	0.0	
2020-07-30	15199	0.0	1.0	
2020-07-28	15198	5.0	0.0	
2020-07-24	15193	4.0	0.0	
2020-07-23	15188	2.0	0.0	
2020-07-21	15186	6.0	1.0	
2020-07-17	15178	2.0	0.0	
2020-07-16	15176	7.0	2.0	
2020-07-14	15165	4.0	0.0	
2020-07-10	15160	6.0	2.0	
2020-07-09	15151	2.0	0.0	

2020-07-07	15149	4.0	0.0
2020-07-03	15142	9.0	4.0
2020-07-02	15129	2.0	3.0
2020-07-01	15122	10.0	1.0
2020-06-30	15111	6.0	0.0
2020-06-28	15104	2.0	0.0
2020-06-25	15102	12.0	2.0
2020-06-23	15084	18.0	5.0
2020-06-21	15060	22.0	4.0
2020-06-18	15034	27.0	3.0
2020-06-16	15001	21.0	6.0
2020-06-14	14972	9.0	6.0
2020-06-11	14955	35.0	8.0
2020-06-09	14910	21.0	6.0
2020-06-07	14878	19.0	7.0
2020-06-04	14848	67.0	21.0
2020-06-02	14753	24.0	8.0
2020-05-31	14718	30.0	15.0
2020-05-28	14671	56.0	28.0
2020-05-26	14581	81.0	73.0
2020-05-24	14417	87.0	69.0
2020-05-21	14246	72.0	29.0
2020-05-19	14144	108.0	15.0
2020-05-17	14018	67.0	19.0
2020-05-14	13931	52.0	19.0
2020-05-12	13857	117.0	34.0
2020-05-11	13704	83.0	33.0
2020-05-08	13585	55.0	28.0
2020-05-06	13501	63.0	22.0
2020-05-04	13409	67.0	17.0
2020-04-30	13321	178.0	43.0
2020-04-29	13080	77.0	22.0
2020-04-27	12976	81.0	37.0
2020-04-26	12855	167.0	66.0
2020-04-25	12612	245.0	102.0
2020-04-24	12252	117.0	72.0
2020-04-23	12056	127.0	64.0
2020-04-22	11852	NaN	NaN

Unnamed: 0	FALLECIDOS TOTALES hoy
Fecha	
2020-07-31	3.0
2020-07-30	1.0
2020-07-28	5.0
2020-07-24	5.0
2020-07-23	2.0
2020-07-21	8.0

```
2020-07-17     2.0
2020-07-16    11.0
2020-07-14     5.0
2020-07-10     9.0
2020-07-09     2.0
2020-07-07     7.0
2020-07-03    13.0
2020-07-02     7.0
2020-07-01    11.0
2020-06-30     7.0
2020-06-28     2.0
2020-06-25    18.0
2020-06-23    24.0
2020-06-21    26.0
2020-06-18    33.0
2020-06-16    29.0
2020-06-14    17.0
2020-06-11    45.0
2020-06-09    32.0
2020-06-07    30.0
2020-06-04    95.0
2020-06-02    35.0
2020-05-31    47.0
2020-05-28    90.0
2020-05-26   164.0
2020-05-24   171.0
2020-05-21   102.0
2020-05-19   126.0
2020-05-17    87.0
2020-05-14    74.0
2020-05-12   153.0
2020-05-11   119.0
2020-05-08    84.0
2020-05-06    92.0
2020-05-04    88.0
2020-04-30   241.0
2020-04-29   104.0
2020-04-27   121.0
2020-04-26   243.0
2020-04-25   360.0
2020-04-24   196.0
2020-04-23   204.0
2020-04-22     NaN
```

[4]:
```
total
VENTANA_MEDIA_MOVIL=7
df = interpolate_dataframe(total,'D')
```

```
df.index.name = 'Fecha'
df = df.sort_values(by=['Fecha'], ascending=True)
df['HOSPITALES hoy'] = df['HOSPITALES'] - df['HOSPITALES'].shift(1)
df['CENTROS SOCIOSANITARIOS hoy'] = df['CENTROS SOCIOSANITARIOS'] - df['CENTROS␣
↪SOCIOSANITARIOS'].shift(1)
df['FALLECIDOS TOTALES hoy'] = df['FALLECIDOS TOTALES'] - df['FALLECIDOS␣
↪TOTALES'].shift(1)


df['MA CENTROS SOCIOSANITARIOS hoy'] = df['CENTROS SOCIOSANITARIOS hoy'].
↪rolling(window=VENTANA_MEDIA_MOVIL).mean()
df['MA HOSPITALES hoy'] = df['HOSPITALES hoy'].
↪rolling(window=VENTANA_MEDIA_MOVIL).mean()
df['MA FALLECIDOS TOTALES hoy'] = df['FALLECIDOS TOTALES hoy'].
↪rolling(window=VENTANA_MEDIA_MOVIL).mean()

df = df.sort_index(ascending=False)
df_master = df.copy()
```

[5]:
```
total.head()
```

[5]:
```
Unnamed: 0  HOSPITALES  DOMICILIOS  CENTROS SOCIOSANITARIOS  OTROS LUGARES  \
Fecha
2020-07-31        9420         925                     4829             28
2020-07-30        9420         922                     4829             28
2020-07-28        9420         922                     4828             28
2020-07-24        9415         922                     4828             28
2020-07-23        9411         921                     4828             28

Unnamed: 0  FALLECIDOS TOTALES  HOSPITALES hoy  CENTROS SOCIOSANITARIOS hoy  \
Fecha
2020-07-31               15202             0.0                          0.0
2020-07-30               15199             0.0                          1.0
2020-07-28               15198             5.0                          0.0
2020-07-24               15193             4.0                          0.0
2020-07-23               15188             2.0                          0.0

Unnamed: 0  FALLECIDOS TOTALES hoy
Fecha
2020-07-31                     3.0
2020-07-30                     1.0
2020-07-28                     5.0
2020-07-24                     5.0
2020-07-23                     2.0
```

[6]:
```
# Hacemos lo contrario
# En lugar de sacar el nº de muertos dado el nº de infectados, como lo primero␣
 ↪lo sabemos (en madrid), sacamos lo segundo y extrapolamos al conjunto de␣
 ↪españa
df = df_master

R0_estimada = df['FALLECIDOS TOTALES hoy'].values[0:7].sum()  / df['FALLECIDOS␣
 ↪TOTALES hoy'].values[7:14].sum()
print(df['FALLECIDOS TOTALES hoy'].values[0:7].sum(), df['FALLECIDOS TOTALES␣
 ↪hoy'].values[7:14].sum() )
print(f"""R0_estimada = {R0_estimada}""")
PROPORCION_ENFERMOS_MUERTOS=750000/15000 # Esta es la proporcion enfermos␣
 ↪muertos (15.000 muertos para 750.000 afectados)
RATIO_NO_HEMOS_COLAPSADO=2 # La mitad de los muertos se ha calculado del␣
 ↪colapso. Como ahora no hemos colapsado
PESO_MADRID_MUERTES_TOTALES=1/3
casos_españa_estimados =  df['FALLECIDOS TOTALES hoy'].values[0:5].sum() *␣
 ↪PROPORCION_ENFERMOS_MUERTOS * RATIO_NO_HEMOS_COLAPSADO /␣
 ↪PESO_MADRID_MUERTES_TOTALES
print(f"""casos_españa_estimados = {casos_españa_estimados}""")
```

```
9.0 15.0
R0_estimada = 0.6
casos_españa_estimados = 1257.0767685530882
```

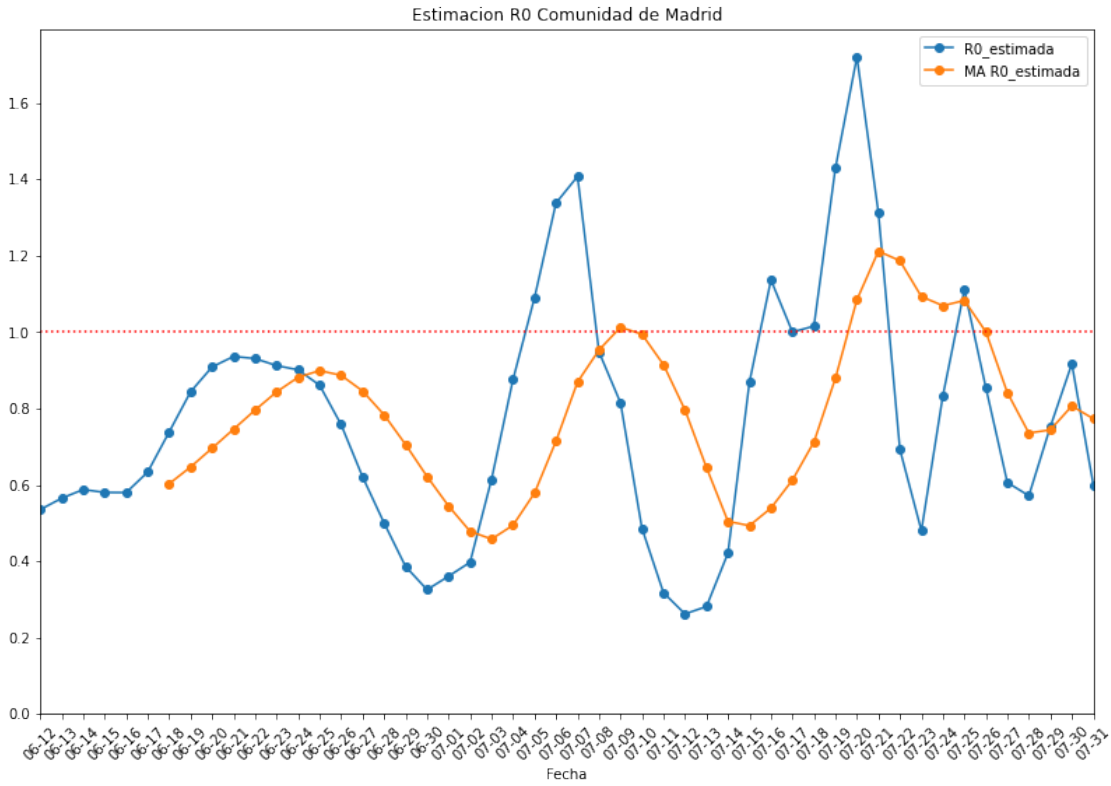
1.1 Gráfico estimacion R0

Considerando solo los datos de Madrid, estimamos el R0 a partir del nº de muertos (considerando que el nº de muertos es una combinacion lineal del nº de enfermos), por lo que es posible calcular el ratio igual.

Para calcular el R0, sacamos la suma de muertos de la última semana, entre la suma de muertos de la semana anterior.

[7]:
```
from datetime import datetime, timedelta
import seaborn as sns
from matplotlib import pyplot as plt
import matplotlib.dates as mdates

df = df_master

def calcular_estimaciones_R0(df):
    def calcular_R0_dia(dia,df):
        dia_semana_anterior = dia - timedelta(days=7)
        return dia,df.loc[dia:dia - timedelta(days=6)]['FALLECIDOS TOTALES␣
 ↪hoy'].sum() / df.loc[dia- timedelta(days=7):dia -␣
 ↪timedelta(days=13)]['FALLECIDOS TOTALES hoy'].sum()
```

```
    VENTANA_MEDIA_MOVIL=7

    df_R0_estimada = pd.DataFrame([calcular_R0_dia(dia,df) for dia in df.
 ↪index[0:50]],columns=['Fecha','R0_estimada'])

    df_R0_estimada = df_R0_estimada.sort_values(by=['Fecha'], ascending=True)
    df_R0_estimada['MA R0_estimada'] = df_R0_estimada['R0_estimada'].
 ↪rolling(window=VENTANA_MEDIA_MOVIL).mean()
    df_R0_estimada = df_R0_estimada.sort_values(by=['Fecha'], ascending=False)
    df_R0_estimada.set_index('Fecha', inplace=True, drop=True)
    return df_R0_estimada

df= calcular_estimaciones_R0(df_master)
#df=df[['R0_estimada']]
df

chart_df=df[df.columns[-3:]]
chart_df.plot(legend=True,figsize=(13.5,9), marker='o')

plt.gca().xaxis.set_major_formatter(mdates.DateFormatter('%m-%d'))
plt.gca().xaxis.set_major_locator(mdates.DayLocator(interval=1))
plt.xticks(rotation=45)

ax = plt.gca()
ax.axhline(1, color='r',linestyle =  ':' )


ax.set_title("Estimacion R0 Comunidad de Madrid")
ax.set_ylim(ymin=0)

plt.show()

df.style.format ({ c : "{:20,.3f}" for c in df.columns }).
 ↪background_gradient(cmap='Wistia', )
```



[7]:
```
<pandas.io.formats.style.Styler at 0x7f1b4f7f7390>
```

[8]:
```
R0_estimada * 1.2
```

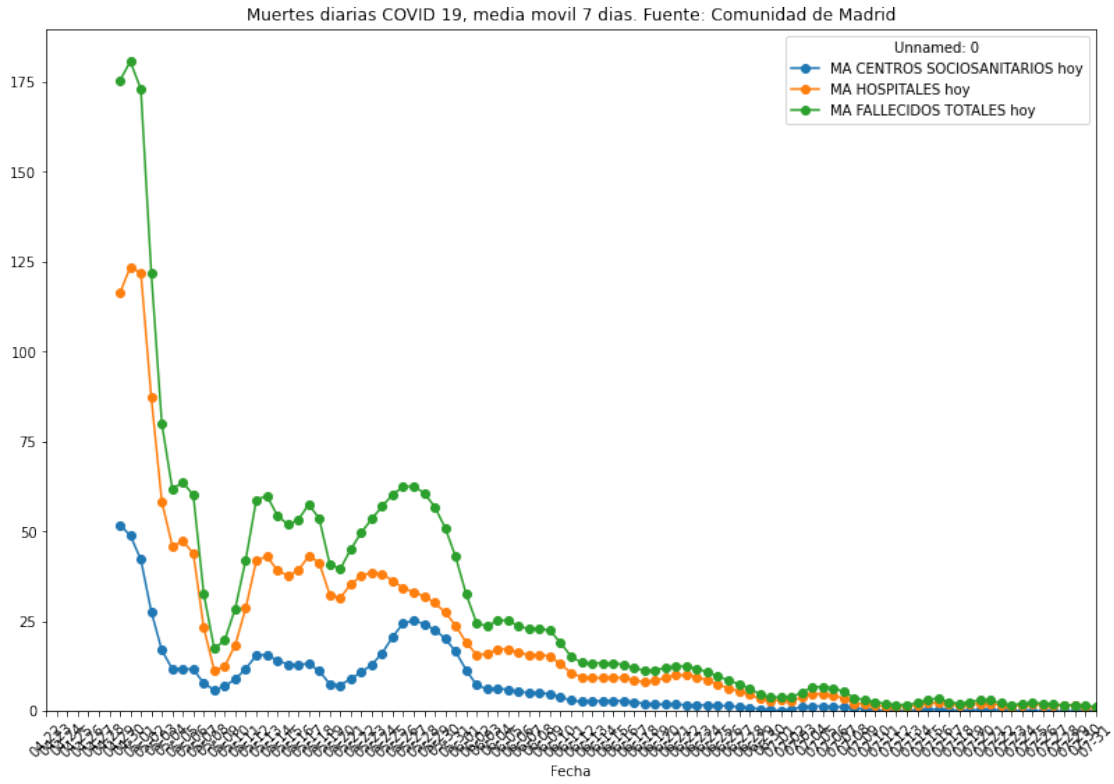
[8]:
```
0.72
```

[9]:
```
HTML("<h2>Gráfico muertes diarias en Madrid, según Comunidad de Madrid </h2>")
```

[9]:
```
<IPython.core.display.HTML object>
```

[10]:
```
import pandas as pd
import io
import matplotlib.dates as mdates
from matplotlib import pyplot as plt

df = df_master
chart_df=df[df.columns[-3:]]
chart_df.plot(legend=True,figsize=(13.5,9), marker='o')

plt.gca().xaxis.set_major_formatter(mdates.DateFormatter('%m-%d'))
plt.gca().xaxis.set_major_locator(mdates.DayLocator(interval=1))
plt.xticks(rotation=45)
```

```
ax = plt.gca()

ax.set_title("Muertes diarias COVID 19, media movil␣
 ↪"+str(VENTANA_MEDIA_MOVIL)+" dias. Fuente: Comunidad de Madrid")
ax.set_ylim(ymin=0)

plt.show()
```



[11]:
```
 from IPython.display import display, HTML
HTML("<h2>Comparamos los datos de hoy, de hace una semana y de un mes </h2>")
```

[11]: <IPython.core.display.HTML object>

[12]:
```
from matplotlib import colors

def background_gradient(s, m, M, cmap='PuBu', low=0, high=0):
    rng = M - m
    norm = colors.Normalize(m - (rng * low),
                            M + (rng * high))
    normed = norm(s.values)
    c = [colors.rgb2hex(x) for x in plt.cm.get_cmap(cmap)(normed)]
```

```
    return ['background-color: %s' % color for color in c]

df = df_master

df.style.format ({ c : "{:20,.0f}" for c in df.columns }).
 ↪background_gradient(cmap='Wistia', subset= df.columns[-3:] )
```

[12]: <pandas.io.formats.style.Styler at 0x7f1b47b999e8>

[13]:
```
df = df_master
pd.concat([df.head(1).tail(1) , df.head(8).tail(1) , df.head(30).tail(1)]).
 ↪astype(int)[['MA HOSPITALES hoy','MA CENTROS SOCIOSANITARIOS hoy','MA␣
 ↪FALLECIDOS TOTALES hoy']].style.format ({ c : "{:20,.0f}" for c in df.
 ↪columns }).background_gradient(cmap='Wistia', subset= df.columns[-3:] )
```

[13]: <pandas.io.formats.style.Styler at 0x7f1b568762b0>

[14]:
```
from IPython.display import display, HTML
HTML("<h2>Muertes medias diarias, últimos 7 días, con datos</h2>")
```

[14]: <IPython.core.display.HTML object>

[15]:
```
from datetime import date

df = df_master
inicio_crisis = df.head(7).index[6]
df=df.head(7)
dia_mas_reciente = df.index[0]
dias_transcurridos_inicio_crisis = dia_mas_reciente - inicio_crisis
df = pd.DataFrame((df.head(1).max(axis=0) - df.tail(1).max(axis=0) ) / ␣
 ↪dias_transcurridos_inicio_crisis.days  ).
 ↪T[['HOSPITALES','DOMICILIOS','CENTROS SOCIOSANITARIOS','OTROS␣
 ↪LUGARES','FALLECIDOS TOTALES']]
df.style.format ({ c : "{:20,.0f}" for c in df.columns }).
 ↪background_gradient(cmap='Wistia'  )
```

[15]: <pandas.io.formats.style.Styler at 0x7f1b47b4fc18>

[16]:
```
HTML("<h2>Muertes medias diarias desde que la comunidad de Madrid publica␣
 ↪datos</h2>")
```

[16]: <IPython.core.display.HTML object>

[17]:
```
# Calculamos los incrementos medios, desde que tenemos fechas
df = df_master
```

```
df = pd.DataFrame((df.head(1).max(axis=0) - df.tail(1).max(axis=0) ) / df.
 ↪shape[0] ).T[['HOSPITALES','DOMICILIOS','CENTROS SOCIOSANITARIOS','OTROS␣
 ↪LUGARES','FALLECIDOS TOTALES']]
df.style.format ({ c : "{:20,.0f}" for c in df.columns }).
 ↪background_gradient(cmap='Wistia'  )
```

[17]:
```
<pandas.io.formats.style.Styler at 0x7f1b568d1a58>
```

[ ]:
```

```