

First step: we import data and play with it a bit

We want to solve the following questions:

- (1) Bring out a comparative picture of people migrating to neighboring states and the rest of the states from a state across India.
- (2) Give a list of the top 10 employing states.
- (3) Bring out a clear picture of the gender ratio of workers migrating to Maharashtra from states all over India.

```
In [ ]: # Import and excel file using pandas
import pandas as pd
# Read Excel file
df = pd.read_excel('MigrationData.xlsx')
# print(df)

df1 = df
df1 = df1.rename({'Migrant Persons' : 'Migrant Persons X 1e6'}, axis=1)
df1 = df1.rename({'Migrant Males' : 'Migrant Males %'}, axis=1)
df1['Migrant Males %'] = df1['Migrant Males %']/df1['Migrant Persons X 1e
print(df1.head()) # to test that all is properly imported
```

	Home State Name	Home State ID	Current State \
0	DAMAN & DIU	25	JAMMU & KASHMIR
1	DADRA & NAGAR HAVELI	26	JAMMU & KASHMIR
2	PUDUCHERRY	34	JAMMU & KASHMIR
3	ANDAMAN & NICOBAR ISLANDS	35	JAMMU & KASHMIR
4	MIZORAM	15	JAMMU & KASHMIR

	Current State ID	Migrant Persons X 1e6	Migrant Males %	Migrant Females
0	1	6	50.000000	
3				
1	1	17	23.529412	
13				
2	1	18	33.333333	
12				
3	1	51	49.019608	
26				
4	1	57	38.596491	
35				

```
In [ ]: print(df['Current State'].unique())
print(df['Current State ID'].unique())

['JAMMU & KASHMIR' 'HIMACHAL PRADESH' 'PUNJAB' 'CHANDIGARH' 'UTTARAKHAND'
'HARYANA' 'NCT OF DELHI' 'RAJASTHAN' 'UTTAR PRADESH' 'BIHAR' 'SIKKIM'
'ARUNACHAL PRADESH' 'NAGALAND' 'MANIPUR' 'MIZORAM' 'TRIPURA' 'MEGHALAYA'
'ASSAM' 'WEST BENGAL' 'JHARKHAND' 'ODISHA' 'CHHATTISGARH'
'MADHYA PRADESH' 'GUJARAT' 'DAMAN & DIU' 'DADRA & NAGAR HAVELI'
'MAHARASHTRA' 'ANDHRA PRADESH' 'KARNATAKA' 'GOA' 'LAKSHADWEEP' 'KERALA'
'TAMIL NADU' 'PUDUCHERRY' 'ANDAMAN & NICOBAR ISLANDS']
[ 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
 25 26 27 28 29 30 31 32 33 34 35]
```

Let us make an ID file. We will use this dataframe throughout the notebook.

```
In [ ]: # initialize list elements
data = {'ID': df['Current State ID'].unique(),
        'State': df['Current State'].unique()}

IDs = pd.DataFrame(data)

# print dataframe.
print(IDs)
```

	ID	State
0	1	JAMMU & KASHMIR
1	2	HIMACHAL PRADESH
2	3	PUNJAB
3	4	CHANDIGARH
4	5	UTTARAKHAND
5	6	HARYANA
6	7	NCT OF DELHI
7	8	RAJASTHAN
8	9	UTTAR PRADESH
9	10	BIHAR
10	11	SIKKIM
11	12	ARUNACHAL PRADESH
12	13	NAGALAND
13	14	MANIPUR
14	15	MIZORAM
15	16	TRIPURA
16	17	MEGHALAYA
17	18	ASSAM
18	19	WEST BENGAL
19	20	JHARKHAND
20	21	ODISHA
21	22	CHHATTISGARH
22	23	MADHYA PRADESH
23	24	GUJARAT
24	25	DAMAN & DIU
25	26	DADRA & NAGAR HAVELI
26	27	MAHARASHTRA
27	28	ANDHRA PRADESH
28	29	KARNATAKA
29	30	GOA
30	31	LAKSHADWEEP
31	32	KERALA
32	33	TAMIL NADU
33	34	PUDUCHERRY
34	35	ANDAMAN & NICOBAR ISLANDS

Plotting the global data

Here, we will plot how many workers migrated from a given state and to which state.

```

In [ ]: import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
import matplotlib.ticker as ticker

sns.set_theme(style="whitegrid")

# Load the example planets dataset
# planets = sns.load_dataset("planets")
fig, (ax1, ax2) = plt.subplots(1, 2)
fig.set_size_inches(25, 10)

xmin, xmax = min(IDs['ID'])-0.5, max(IDs['ID'])+0.5
ax1.plot([xmin, xmax], [xmin, xmin], 'k-', lw=3)
ax1.plot([xmax, xmax], [xmin, xmax], 'k-', lw=3)
ax1.plot([xmin, xmax], [xmax, xmax], 'k-', lw=3)
ax1.plot([xmin, xmin], [xmin, xmax], 'k-', lw=3)

cmap = sns.cubehelix_palette(rot=-.2, as_cmap=True)

g = sns.scatterplot(
    data=df1,
    x="Home State ID", y="Current State ID",
    size="Migrant Persons X 1e6", hue="Migrant Males %",
    palette=cmap, sizes=(100, 500), ax=ax1
)

ax1.xaxis.grid(True, "minor", linewidth=.25)
ax1.yaxis.grid(True, "minor", linewidth=.25)
ax1.set_aspect('equal')
ax1.set_xlim(0.5, 35.5)
ax1.set_ylim(0.5, 35.5)
ax1.legend(bbox_to_anchor=(0.85, 0, 0.5, 0.5))

start, end = ax1.get_xlim()
ax1.xaxis.set_ticks(np.arange(start+0.5, end-0.5+1, 1))
ax1.xaxis.set_major_formatter(ticker.FormatStrFormatter('%d'))
ax1.yaxis.set_ticks(np.arange(start+0.5, end-0.5+1, 1))
ax1.yaxis.set_major_formatter(ticker.FormatStrFormatter('%d'))

ax2.axis('off')
ax2.table(cellText=IDs.values, colLabels=IDs.columns, loc='center', colWi

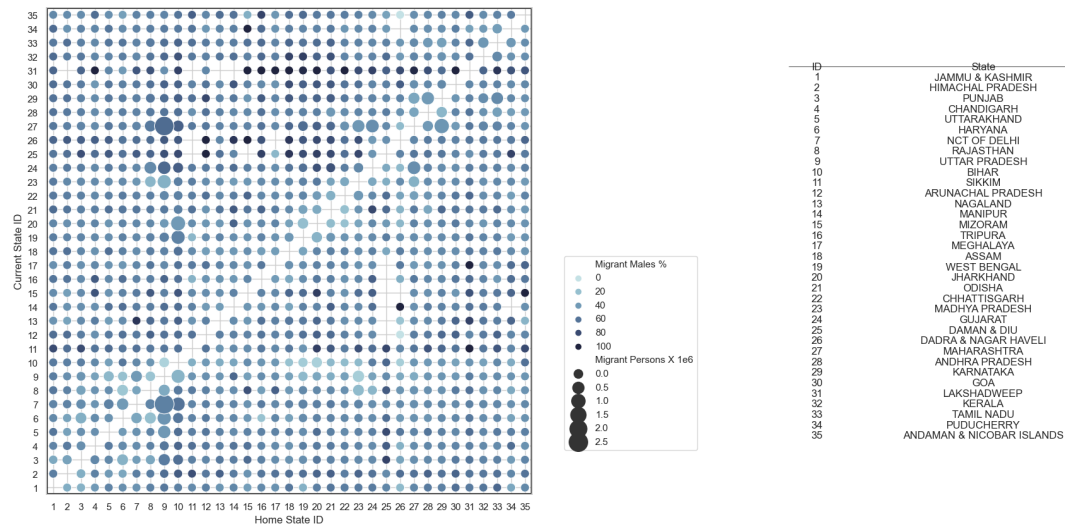
import matplotlib.lines as lines
yL = 0.88
line = lines.Line2D([0.2, 0.7], [yL, yL],
                    lw=0.5, color='black', axes=ax2)
ax2.add_line(line)

```

```

Out[ ]: <matplotlib.lines.Line2D at 0x185d4dea0>

```



- As expected, the diagonal is empty. No one migrated to the state they were already in.
- The most common migration is from Uttar Pradesh to Delhi.
- In fact the number of people migrating from UP to all other states is higher in general.
- UP is followed very closely by Bihar.

Shortcoming of this plot:

We do not really see any details because of the large numbers included in this case. In the subsequent questions, we will make use of specific questions to understand the data better.

Question 1

Now we do question 1. To keep things simple, we will use brute force.

To start, let us first calculate the total number of workers migrating out from a particular state

```

In [ ]: # Now, we get a sum of migrants from a particular Home State
df2em = df.groupby(['Home State ID']).sum()

# Plot Migrant Persons against Home State ID
# print(df2em.index)
fig, (ax, ax0) = plt.subplots(1,2)
fig.set_size_inches(25, 10)
ax.plot(df2em.index, df2em['Migrant Persons'], 'o-')
ax.xaxis.grid(True, "minor", linewidth=.25)
ax.yaxis.grid(True, "minor", linewidth=.25)
ax.set_xlim(1, 35)
start, end = ax1.get_xlim()
ax.xaxis.set_ticks(np.arange(1, 36, 1))
ax.xaxis.set_major_formatter(ticker.FormatStrFormatter('%d'))
ax.set_xlabel('Home State ID')
ax.set_ylabel('Cumulative out-Migrant Persons')

# set yaxis to log scale
ax.set_yscale('log')

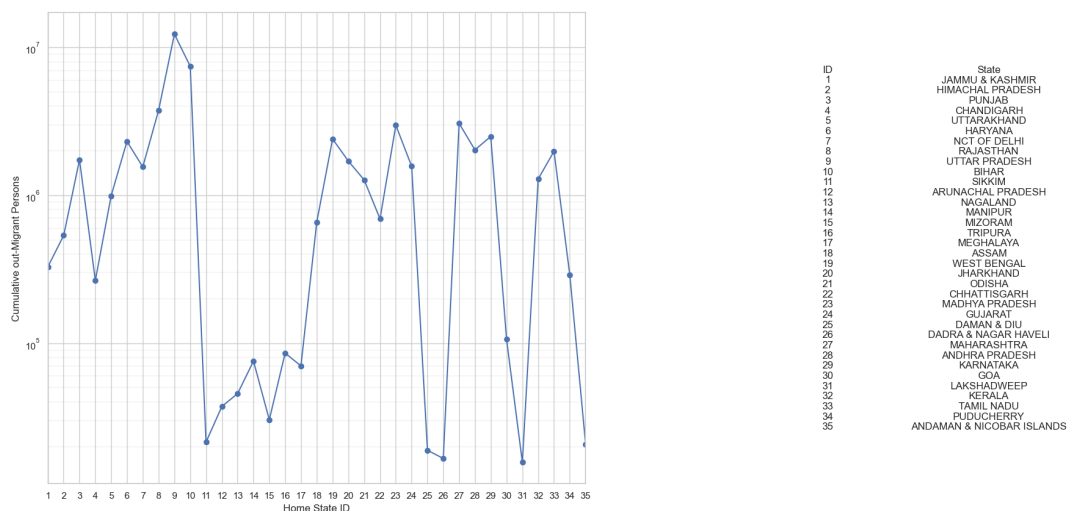
ax0.axis('off')
ax0.table(cellText=IDs.values, colLabels=IDs.columns, loc='center', colWi

```

```

Out[ ]: <matplotlib.table.Table at 0x114b23d00>

```



- Clearly, we see that by far the most workers are migrating out of Uttar Pradesh. This is followed by Bihar and Rajasthan. This is inline with our previous plot.

For better visualization, we will plot the above line graph onto the map of India. This way, we can keep track of the states and their migration patterns.

```
In [ ]: import geopandas as gpd

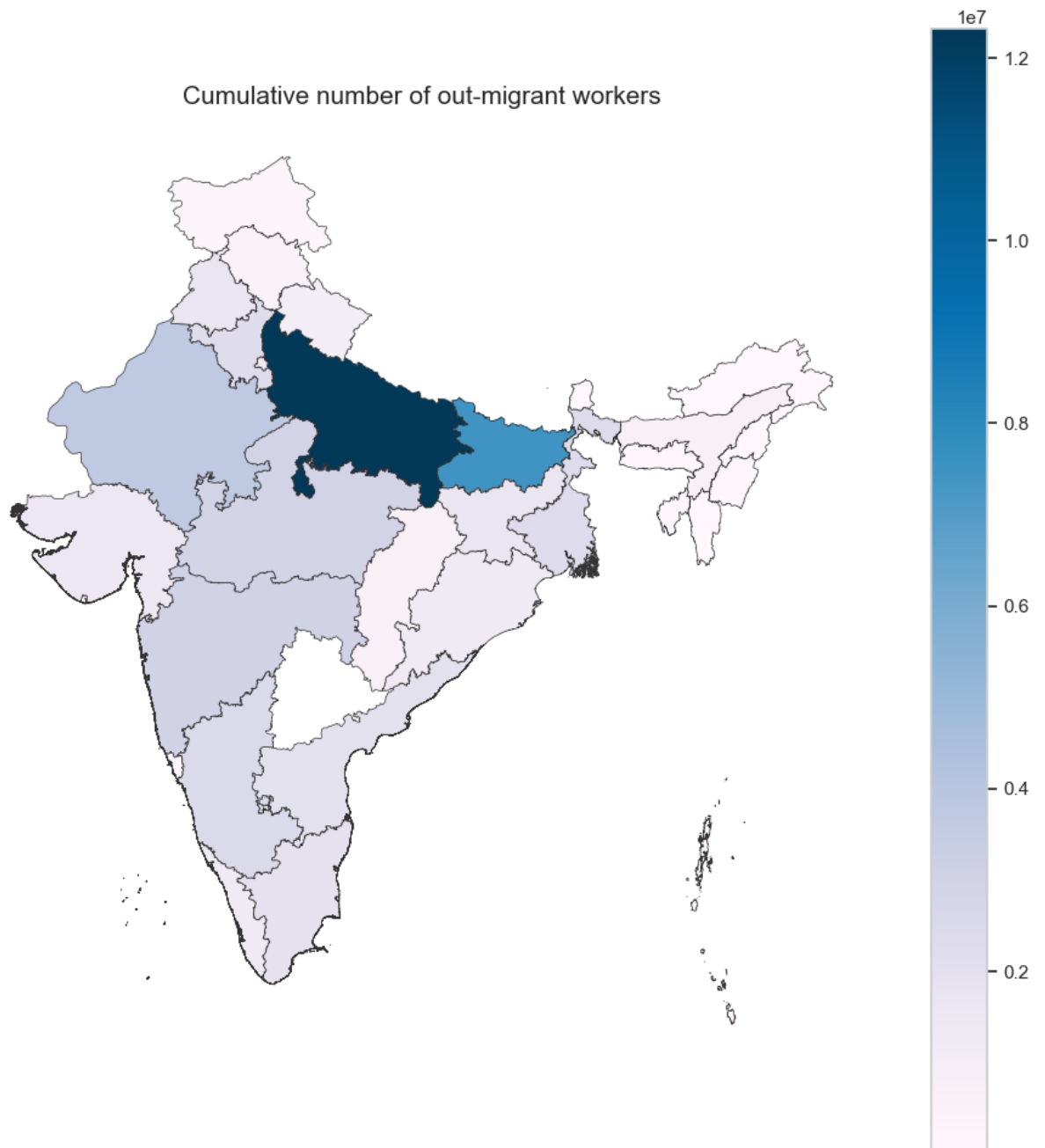
df6id = df2em.index.to_numpy()
df6value = df2em['Migrant Persons'].to_numpy()

data = {'ID': df6id,
        'Migrant persons': df6value}

df6em = pd.DataFrame(data)

df7em = pd.merge(IDs, df6em, on="ID", how="left")
# print(df7)

map_dfem = gpd.read_file('IND_adm/IND_adm1.shp')
# Rename some states
map_dfem = map_dfem.replace({'Andaman and Nicobar': 'ANDAMAN & NICOBAR ISL
# print(map_df)
dfMergeem = pd.merge(map_dfem, df7em, left_on='NAME_1', right_on='State',
# print(dfMerge)
fig, ax = plt.subplots(1, figsize=(12, 12))
ax.axis('off')
ax.set_title('Cumulative number of out-migrant workers',
             fontdict={'fontsize': '15', 'fontweight' : '3'})
fig = dfMergeem.plot(column='Migrant persons', cmap='PuBu', linewidth=0.5
```



Dark blue states are those with the highest number of workers migrating out. Light blue states are those with the lowest number of workers migrating out.

- Clearly, once again UP stands out.
- Unfortunately, there was no Telengana in the datafile. So, we have skipped it.

Finding the neighbours of a given state

Here, we will use geopanda to find the neighbours of a given state. We will use this to find the states that are most likely to be migrated to from a given state.


```

In [ ]: from sre_parse import State

map_dfem2 = gpd.read_file('IND_adm/IND_adm1.shp')
# Rename some states
map_dfem2 = map_dfem2.replace({'Andaman and Nicobar':'ANDAMAN & NICOBAR I

# map_dfem2.plot()
# add NEIGHBORS column
map_dfem2["NEIGHBORS"] = None

for index, State in map_dfem2.iterrows():

    # get 'not disjoint' countries
    neighbors = map_dfem2[~map_dfem2.geometry.disjoint(State.geometry)].N

    # remove own name of the country from the list
    neighbors = [ name for name in neighbors if State.NAME_1 != name ]

    # add names of neighbors as NEIGHBORS value
    map_dfem2.at[index, "NEIGHBORS"] = ", ".join(neighbors)

# print(map_dfem2)

map_dfem3StatesTemp = map_dfem2['NAME_1'].to_numpy()
map_dfem3NeighboursTemp = map_dfem2['NEIGHBORS'].to_numpy()

data = {'Home State Name': map_dfem3StatesTemp,
        'Neighbouring states': map_dfem3NeighboursTemp}

map_dfem3 = pd.DataFrame(data)

# print(map_dfem3)

```

df is our base dataframe. map_dfem3 is the dataframe that has all the states and their corresponding neighbours.

```

In [ ]: print(map_dfem3.head())

```

	Home State Name \	Neighbouring states
0	ANDAMAN & NICOBAR ISLANDS	
1	ANDHRA PRADESH	
2	ARUNACHAL PRADESH	
3	ASSAM	
4	BIHAR	

	Neighbouring states
0	
1	KARNATAKA, ODISHA, PUDUCHERRY, TAMIL NADU, TEL...
2	ASSAM, NAGALAND
3	ARUNACHAL PRADESH, MANIPUR, MEGHALAYA, MIZORAM...
4	JHARKHAND, UTTAR PRADESH, WEST BENGAL

```

In [ ]: print(df.head())

```

	Home State Name	Home State ID	Current State \
0	DAMAN & DIU	25	JAMMU & KASHMIR
1	DADRA & NAGAR HAVELI	26	JAMMU & KASHMIR
2	PUDUCHERRY	34	JAMMU & KASHMIR
3	ANDAMAN & NICOBAR ISLANDS	35	JAMMU & KASHMIR
4	MIZORAM	15	JAMMU & KASHMIR

	Current State ID	Migrant Persons	Migrant Males	Migrant Females
0	1	6	3	3
1	1	17	4	13
2	1	18	6	12
3	1	51	25	26
4	1	57	22	35

The most important part of the code to calculate the number of migrations to the neighbouring states

Here, we loop over all the states to find the percentage of workers migrating to the neighbouring states.

```
In [ ]: # sort based on Home State Name
df9em = df.sort_values(by=['Home State Name'])
# print(df9em)

neighborPercent = []
counter = 0
for state in map_dfem3['Home State Name']:
    # print("Doing %s" % state)
    df10em = df9em[df9em['Home State Name'] == state]
    TotalMigrants = df10em['Migrant Persons'].sum()
    # print(df10em)

    neighbors = map_dfem3['Neighbouring states'][counter]
    neighbors = neighbors.split(", ")
    # print(df10em)
    NeighborMigrant = 0
    for neighbor in neighbors:
        df11em = df10em[df10em['Current State'] == neighbor]
        NeighborMigrant += df11em['Migrant Persons'].sum()
        # df9em = df9em[df9em['Home State Name'] != neighbor]
    # % of migrants going to neighboring states
    if TotalMigrants != 0:
        NeighborMigrant = (NeighborMigrant/TotalMigrants)*100
    else:
        NeighborMigrant = 0
    print("Percentage Migrants from %s to neighbours is %d percent" % (state, NeighborMigrant))
    neighborPercent.append(NeighborMigrant)

    # if counter == 1:
    #     break
    counter = counter + 1
```

Percentage Migrants from ANDAMAN & NICOBAR ISLANDS to neighbours is 0 percent

Percentage Migrants from ANDHRA PRADESH to neighbours is 65 percent

Percentage Migrants from ARUNACHAL PRADESH to neighbours is 54 percent

Percentage Migrants from ASSAM to neighbours is 63 percent

Percentage Migrants from BIHAR to neighbours is 47 percent

Percentage Migrants from CHANDIGARH to neighbours is 83 percent

Percentage Migrants from CHHATTISGARH to neighbours is 81 percent

Percentage Migrants from DADRA & NAGAR HAVELI to neighbours is 83 percent

Percentage Migrants from DAMAN & DIU to neighbours is 76 percent

Percentage Migrants from NCT OF DELHI to neighbours is 66 percent

Percentage Migrants from GOA to neighbours is 85 percent

Percentage Migrants from GUJARAT to neighbours is 84 percent

Percentage Migrants from HARYANA to neighbours is 90 percent

Percentage Migrants from HIMACHAL PRADESH to neighbours is 61 percent

Percentage Migrants from JAMMU & KASHMIR to neighbours is 26 percent

Percentage Migrants from JHARKHAND to neighbours is 75 percent

Percentage Migrants from KARNATAKA to neighbours is 95 percent

Percentage Migrants from KERALA to neighbours is 62 percent

Percentage Migrants from LAKSHADWEEP to neighbours is 0 percent

Percentage Migrants from MADHYA PRADESH to neighbours is 87 percent

Percentage Migrants from MAHARASHTRA to neighbours is 73 percent

Percentage Migrants from MANIPUR to neighbours is 50 percent

Percentage Migrants from MEGHALAYA to neighbours is 57 percent

Percentage Migrants from MIZORAM to neighbours is 74 percent

Percentage Migrants from NAGALAND to neighbours is 57 percent

Percentage Migrants from ODISHA to neighbours is 54 percent

Percentage Migrants from PUDUCHERRY to neighbours is 97 percent

Percentage Migrants from PUNJAB to neighbours is 61 percent

Percentage Migrants from RAJASTHAN to neighbours is 62 percent

Percentage Migrants from SIKKIM to neighbours is 53 percent

Percentage Migrants from TAMIL NADU to neighbours is 79 percent

Percentage Migrants from TELANGANA to neighbours is 0 percent

Percentage Migrants from TRIPURA to neighbours is 53 percent

Percentage Migrants from UTTAR PRADESH to neighbours is 58 percent

Percentage Migrants from UTTARAKHAND to neighbours is 41 percent

Percentage Migrants from WEST BENGAL to neighbours is 42 percent

neighborPercent variable stores the percentage of workers migrating to the neighbouring states.

```
In [ ]: print(neighborPercent)
```

```
[0.0, 65.61292490064059, 54.5546992078784, 63.66300133092009, 47.12418613
692903, 83.37480472058574, 81.38551854585717, 83.22813345356177, 76.70051
835396171, 66.4719322910375, 85.26498173189198, 84.38215314067011, 90.910
28815824414, 61.32043603951305, 26.104907287204448, 75.20950806152179, 95
.22908113446661, 62.201769500319436, 0.0, 87.2134914273977, 73.9331882117
0897, 50.873255798603324, 57.999373825923605, 74.4574345463527, 57.974373
551405954, 54.81578858346295, 97.28286836037309, 61.745085953803745, 62.4
2973916580332, 53.76298988769281, 79.88224608935212, 0, 53.00598635018983
6, 58.42205650966363, 41.89740028382499, 42.19454239038346]
```

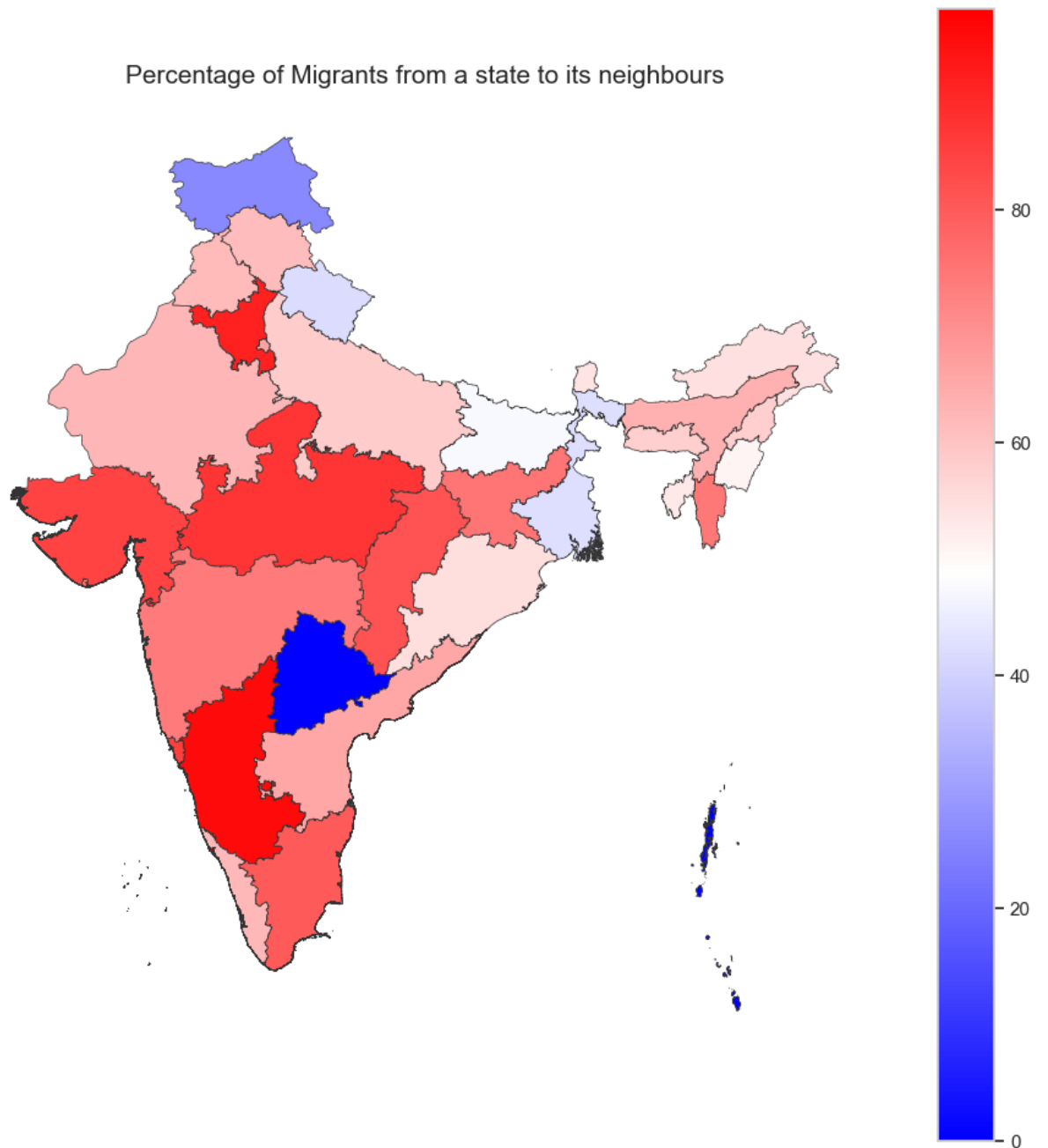
Let us make neighborPercent into a panda dataframe

```
In [ ]: data = {'Home State Name': map_dfem3['Home State Name'],
               'Neighbor Percent migration': neighborPercent}

df12em = pd.DataFrame(data)
print(df12em.head())
```

	Home State Name	Neighbor Percent migration
0	ANDAMAN & NICOBAR ISLANDS	0.000000
1	ANDHRA PRADESH	65.612925
2	ARUNACHAL PRADESH	54.554699
3	ASSAM	63.663001
4	BIHAR	47.124186

```
In [ ]: map_dfNeighbour = gpd.read_file('IND_adm/IND_adm1.shp')
        # Rename some states
        map_dfNeighbour = map_dfNeighbour.replace({'Andaman and Nicobar': 'ANDAMAN'})
        # print(map_df)
        dfMergeem = pd.merge(map_dfNeighbour, df12em, left_on='NAME_1', right_on='Neighbor Percent migration')
        # print(dfMergeem)
        fig, ax = plt.subplots(1, figsize=(12, 12))
        ax.axis('off')
        ax.set_title('Percentage of Migrants from a state to its neighbours',
                    fontdict={'fontsize': '15', 'fontweight' : '3'})
        fig = dfMergeem.plot(column='Neighbor Percent migration', cmap='bwr', lin
```



Red denotes the states with highest number of workers migrating to the neighbouring states. Blue denotes the states with the lowest number of workers migrating to the neighbouring states.

In the above map of India, we have plotted the percentage of workers migrating to the neighbouring states. Clearly, the southern states have the highest percentage of workers migrating to the neighbouring states.

We also see that percentage of migrant workers from North-East to their neighbours is very high.

Not surprising, % of workers migrating from UP, Bihar, and other northern to their neighbours is very low. This is due to the fact that people from UP and Bihar go to Delhi, Maharashtra, and other southern states for jobs.

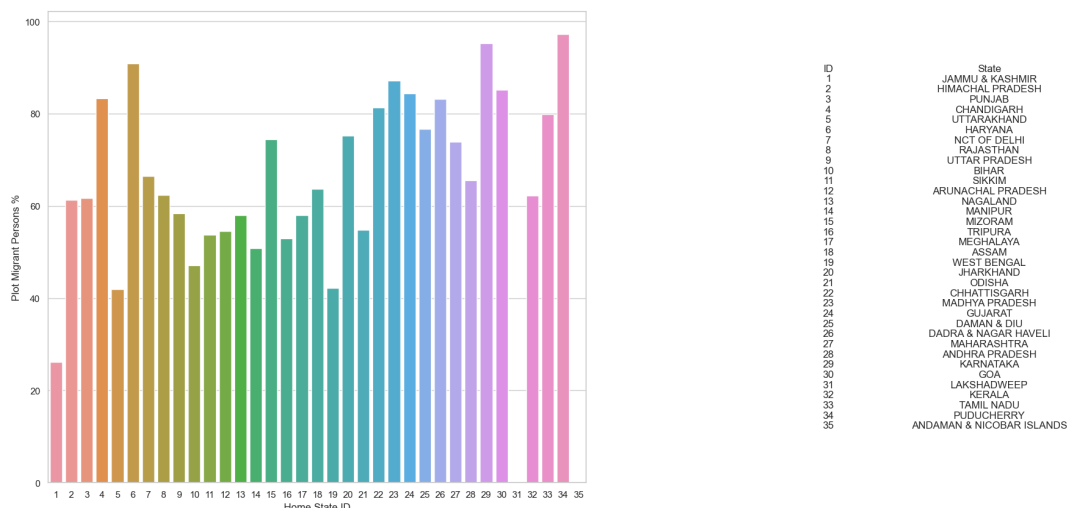
To further understand the data, we can make bar plots of the states with the highest number of workers migrating to the neighbouring states.

```
In [ ]: sns.set_theme(style="whitegrid")

df13em = pd.merge(IDs, df12em, left_on='State', right_on='Home State Name')
# print(df13em)
# Plot Migrant Persons % to neighbour against Home State ID
fig, (ax10, ax12) = plt.subplots(1,2)
fig.set_size_inches(25, 10)
sns.barplot(x="ID", y="Neighbor Percent migration", data=df13em, ax=ax10)
ax10.xaxis.grid(True, "minor", linewidth=.25)
ax10.yaxis.grid(True, "minor", linewidth=.25)
ax10.set_xlabel('Home State ID')
ax10.set_ylabel('Plot Migrant Persons %')

ax12.axis('off')
ax12.table(cellText=IDs.values, colLabels=IDs.columns, loc='center', colw
```

```
Out [ ]: <matplotlib.table.Table at 0x186ce6bf0>
```



End of Question 1

Question (2)

In this question, we need to find the top 10 employing states.

Idea is to first group and sum the data based on this group. Then plot it using a line plot to visualize the data.

After that, we can sort this group based on highest number of migrant persons.

```
In [ ]: # Now, we get a sum of migrants to a particular Current State

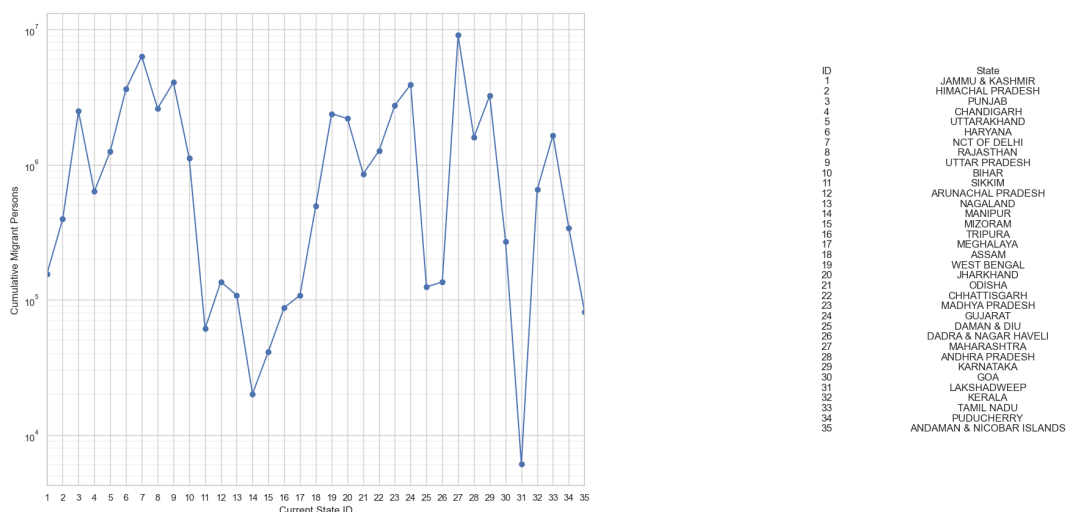
df2 = df.groupby(['Current State ID']).sum()

# Plot Migrant Persons against Current State ID
# print(df2.index)
fig, (ax, ax0) = plt.subplots(1,2)
fig.set_size_inches(25, 10)
ax.plot(df2.index, df2['Migrant Persons'], 'o-')
ax.xaxis.grid(True, "minor", linewidth=.25)
ax.yaxis.grid(True, "minor", linewidth=.25)
ax.set_xlim(1, 35)
start, end = ax1.get_xlim()
ax.xaxis.set_ticks(np.arange(1, 36, 1))
ax.xaxis.set_major_formatter(ticker.FormatStrFormatter('%d'))
ax.set_xlabel('Current State ID')
ax.set_ylabel('Cumulative Migrant Persons')

# set yaxis to log scale
ax.set_yscale('log')

ax0.axis('off')
ax0.table(cellText=IDs.values, colLabels=IDs.columns, loc='center', colWi
```

```
Out [ ]: <matplotlib.table.Table at 0x1858f91b0>
```



As we saw in the previous plots, Maharashtra and Delhi are the top 2 states with the highest number of migrant workers.

Surprisingly, UP is third in this list. This implies that UP does not only have the highest emigration but also has one of the highest immigration.

To further enhance the interpretation of this data, let us plot on the map of India

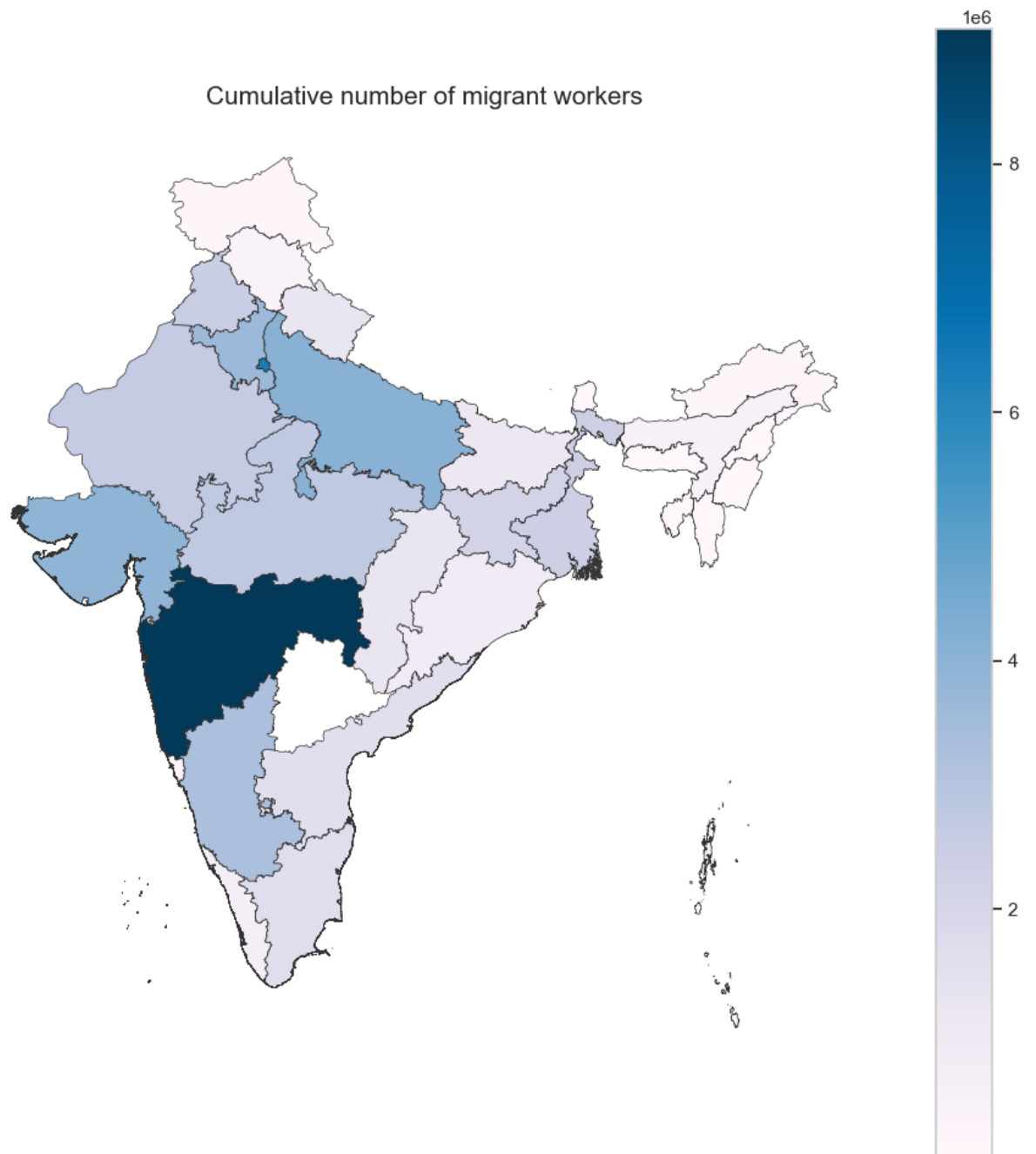
```
In [ ]: df6id = df2.index.to_numpy()
df6value = df2['Migrant Persons'].to_numpy()

data = {'ID': df6id,
        'Migrant persons': df6value}

df6 = pd.DataFrame(data)

df7 = pd.merge(IDs, df6, on="ID", how="left")
# print(df7)

import geopandas as gpd
# Read shapefile using Geopandas
map_df = gpd.read_file('IND_adm/IND_adm1.shp')
# Rename some states
map_df = map_df.replace({'Andaman and Nicobar': 'ANDAMAN & NICOBAR ISLANDS'})
# print(map_df)
dfMerge = pd.merge(map_df, df7, left_on='NAME_1', right_on='State', how='left')
# print(dfMerge)
fig, ax = plt.subplots(1, figsize=(12, 12))
ax.axis('off')
ax.set_title('Cumulative number of migrant workers',
             fontdict={'fontsize': '15', 'fontweight' : '3'})
fig = dfMerge.plot(column='Migrant persons', cmap='PuBu', linewidth=0.5,
```

There is no Telengana in the data. So, I have removed it from the plot.

```
In [ ]: # sort the data frame by Migrant Persons

df3 = df2.sort_values(by=['Migrant Persons'], ascending=False)
# top 10 states with highest number of migrants
df4 = df3.head(10)
# print(df4)

df5id = df4.index.to_numpy()
df5value = df4['Migrant Persons'].to_numpy()

data = {'ID': df5id,
        'Migrant persons': df5value}

df5 = pd.DataFrame(data)

print(df5)
```

	ID	Migrant persons
0	27	9087380
1	7	6330065
2	9	4061933
3	24	3916075
4	6	3626318
5	29	3247660
6	23	2744332
7	8	2604298
8	3	2488299
9	19	2381045

So, we have the id of the top ten employing states. Now, we need to find their names. For this, we will use the IDs dataframe that we created earlier.

```
In [ ]: # find index of top 10 states from IDs
IDs3 = IDs[IDs['ID'].isin(df4.index)]
# print(IDs3)

Tempid = IDs3['ID'].to_numpy()
Tempvalue = IDs3['State'].to_numpy()
Tempdata = {'ID': Tempid,
            'State': Tempvalue}

IDs4 = pd.DataFrame(Tempdata)
print(IDs4)
```

	ID	State
0	3	PUNJAB
1	6	HARYANA
2	7	NCT OF DELHI
3	8	RAJASTHAN
4	9	UTTAR PRADESH
5	19	WEST BENGAL
6	23	MADHYA PRADESH
7	24	GUJARAT
8	27	MAHARASHTRA
9	29	KARNATAKA

Of course, we need to arrange these states based on the migrant population as available in df5

```
In [ ]: IDs4 = IDs4.set_index('ID')
IDs4 = IDs4.reindex(index=df5['ID'])
IDs4 = IDs4.reset_index()
print(IDs4)
```

	ID	State
0	27	MAHARASHTRA
1	7	NCT OF DELHI
2	9	UTTAR PRADESH
3	24	GUJARAT
4	6	HARYANA
5	29	KARNATAKA
6	23	MADHYA PRADESH
7	8	RAJASTHAN
8	3	PUNJAB
9	19	WEST BENGAL

Finally we can merge the two datasets to get the top 10 employing states

```
In [ ]: pd.merge(df5, IDs4, on="ID", how="left")
```

```
Out[ ]:
```

	ID	Migrant persons	State
0	27	9087380	MAHARASHTRA
1	7	6330065	NCT OF DELHI
2	9	4061933	UTTAR PRADESH
3	24	3916075	GUJARAT
4	6	3626318	HARYANA
5	29	3247660	KARNATAKA
6	23	2744332	MADHYA PRADESH
7	8	2604298	RAJASTHAN
8	3	2488299	PUNJAB
9	19	2381045	WEST BENGAL

Above, we have plotted the top ten states with the highest number of migrant workers.

End of Question (2)

Question 3

Here, we want to look at the gender ratio of workers migrating to Maharashtra from all states over India.

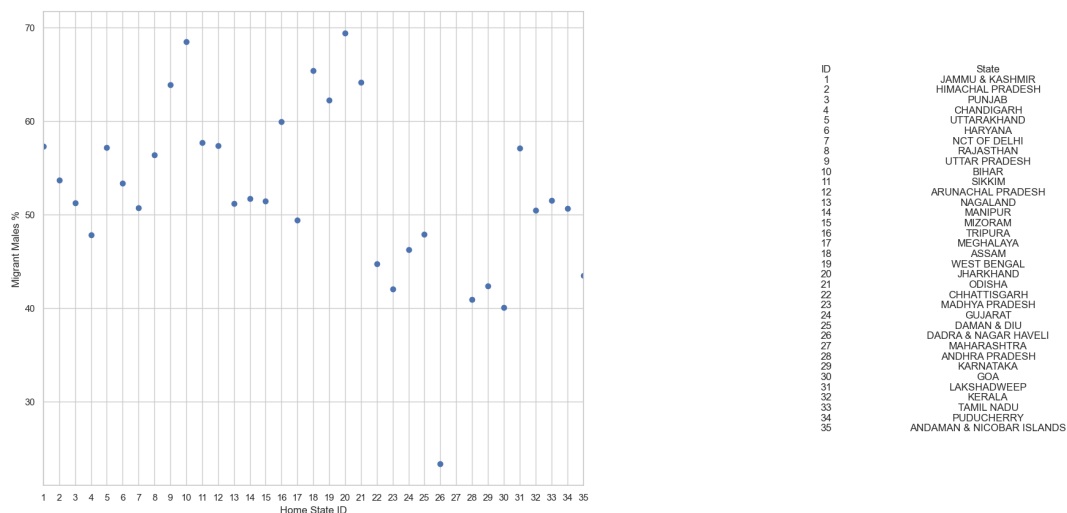
To do so, we will categorize data into dfMaharashtra

```
In [ ]: dfMaharashtra = df[df['Current State ID'] == 27]
dfMaharashtra = dfMaharashtra.rename({'Migrant Males' : 'Migrant Males %'})
dfMaharashtra['Migrant Males %'] = dfMaharashtra['Migrant Males %']/dfMah
# print(dfMaharashtra)

# plot Migrant Males % against Home State ID
fig, (ax3, ax4) = plt.subplots(1,2)
fig.set_size_inches(25, 10)
ax3.plot(dfMaharashtra['Home State ID'], dfMaharashtra['Migrant Males %'])
ax3.xaxis.grid(True, "minor", linewidth=.25)
ax3.yaxis.grid(True, "minor", linewidth=.25)
ax3.set_xlim(1, 35)
start, end = ax1.get_xlim()
ax3.xaxis.set_ticks(np.arange(1, 36, 1))
ax3.xaxis.set_major_formatter(ticker.FormatStrFormatter('%d'))
ax3.set_xlabel('Home State ID')
ax3.set_ylabel('Migrant Males %')

ax4.axis('off')
ax4.table(cellText=IDs.values, colLabels=IDs.columns, loc='center', colWi
```

```
Out [ ]: <matplotlib.table.Table at 0x187131a50>
```



To further understand this data about percentage of males migrating to Maharashtra, we once again plot it on the map of India

```
In [ ]: map_dfNeighbour = gpd.read_file('IND_adm/IND_adm1.shp')
# Rename some states
map_dfNeighbour = map_dfNeighbour.replace({'Andaman and Nicobar':'ANDAMAN'})
# print(map_df)
dfMergeem = pd.merge(map_dfNeighbour, dfMaharashtra, left_on='NAME_1', right_on='NAME_1')
print(dfMergeem)
fig, ax = plt.subplots(1, figsize=(12, 12))
ax.axis('off')
ax.set_title('Percentage of Males Migrants from other states to Maharashtra')
fontdict={'fontsize': '15', 'fontweight' : '3'}
fig = dfMergeem.plot(column='Migrant Males %', cmap='hot_r', linewidth=0.)
```

	ID_0	ISO	NAME_0	ID_1	NAME_1	TYPE_1	\
0	105	IND	India	1	ANDAMAN & NICOBAR ISLANDS	Union Territory	
1	105	IND	India	2	ANDHRA PRADESH	State	
2	105	IND	India	3	ARUNACHAL PRADESH	State	
3	105	IND	India	4	ASSAM	State	
4	105	IND	India	5	BIHAR	State	
5	105	IND	India	6	CHANDIGARH	Union Territory	
6	105	IND	India	7	CHHATTISGARH	State	
7	105	IND	India	8	DADRA & NAGAR HAVELI	Union Territory	
8	105	IND	India	9	DAMAN & DIU	Union Territory	
9	105	IND	India	10	NCT OF DELHI	Union Territory	
10	105	IND	India	11	GOA	State	
11	105	IND	India	12	GUJARAT	State	
12	105	IND	India	13	HARYANA	State	
13	105	IND	India	14	HIMACHAL PRADESH	Union Territory	
14	105	IND	India	15	JAMMU & KASHMIR	State	
15	105	IND	India	16	JHARKHAND	State	
16	105	IND	India	17	KARNATAKA	State	
17	105	IND	India	18	KERALA	State	
18	105	IND	India	19	LAKSHADWEEP	Union Territory	
19	105	IND	India	20	MADHYA PRADESH	State	
20	105	IND	India	21	MAHARASHTRA	State	
21	105	IND	India	22	MANIPUR	State	
22	105	IND	India	23	MEGHALAYA	State	
23	105	IND	India	24	MIZORAM	State	
24	105	IND	India	25	NAGALAND	State	
25	105	IND	India	26	ODISHA	State	
26	105	IND	India	27	PUDUCHERRY	Union Territory	
27	105	IND	India	28	PUNJAB	State	
28	105	IND	India	29	RAJASTHAN	State	
29	105	IND	India	30	SIKKIM	State	
30	105	IND	India	31	TAMIL NADU	State	
31	105	IND	India	32	TELANGANA	State	
32	105	IND	India	33	TRIPURA	State	
33	105	IND	India	34	UTTAR PRADESH	State	
34	105	IND	India	35	UTTARAKHAND	State	
35	105	IND	India	36	WEST BENGAL	State	

	ENGTYPE_1	NL_NAME_1	\
0	Union Territory	None	
1	State	None	
2	State	None	
3	State	None	
4	State	None	

5	Union Territory	None
6	State	None
7	Union Territory	None
8	Union Territory	None
9	Union Territory	None
10	State	None
11	State	None
12	State	None
13	Union Territory	None
14	State	None
15	State	None
16	State	None
17	State	None
18	Union Territory	None
19	State	None
20	State	None
21	State	None
22	State	None
23	State	None
24	State	None
25	State	None
26	Union Territory	None
27	State	None
28	State	None
29	State	None
30	State	None
31	State	None
32	State	None
33	State	None
34	State	None
35	State	None

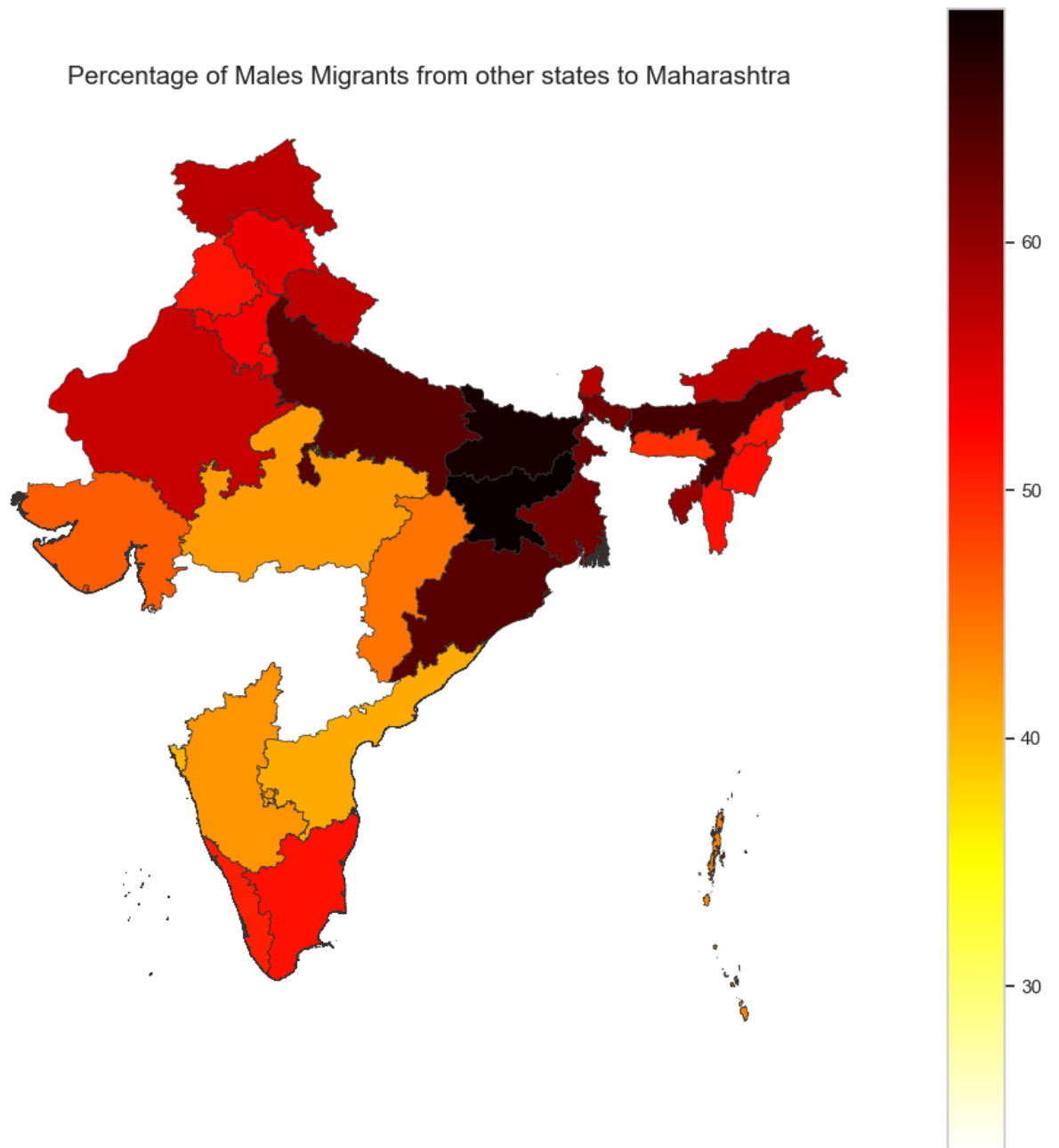
	VARNAME_1 \
0	Andaman & Nicobar Islands Andaman et Nicobar I...
1	None
2	Agence de la Frontière du Nord-Est(French-obso...
3	None
4	None
5	None
6	None
7	Dadra et Nagar Haveli Dadra e Nagar Haveli
8	None
9	None
10	Gôa
11	Goudjerate Gujerat Gujerate
12	None
13	None
14	None
15	Vananchal
16	Maisur Mysore
17	None
18	Îles Laquedives Laccadive Minicoy and Amindivi...
19	None
20	None
21	None
22	None
23	None

24 None
 25 None
 26 Pondicherry|Puduchcheri|Pondichéry
 27 None
 28 Greater Rajasthan|Rajputana
 29 None
 30 Madras|Tamilnad
 31 None
 32 None
 33 United Provinces
 34 UTTARAKHAND
 35 Bangla|Bengala Occidentale|Bengala Ocidental|B...

	geometry	ID \
0	MULTIPOLYGON (((93.78773 6.85264, 93.78849 6.8...	35.0
1	MULTIPOLYGON (((80.27458 13.45958, 80.27458 13...	28.0
2	POLYGON ((96.15778 29.38310, 96.16380 29.37668...	12.0
3	MULTIPOLYGON (((89.87145 25.53730, 89.87118 25...	18.0
4	MULTIPOLYGON (((88.10548 26.53904, 88.10505 26...	10.0
5	POLYGON ((76.80293 30.67548, 76.79437 30.66932...	4.0
6	POLYGON ((83.32760 24.09965, 83.34575 24.09707...	22.0
7	POLYGON ((73.02468 20.09630, 73.01955 20.10502...	26.0
8	MULTIPOLYGON (((72.86014 20.47096, 72.86340 20...	25.0
9	POLYGON ((77.32713 28.68516, 77.32539 28.68250...	7.0
10	MULTIPOLYGON (((73.78181 15.35569, 73.78181 15...	30.0
11	MULTIPOLYGON (((70.86097 20.75292, 70.86097 20...	24.0
12	POLYGON ((76.83715 30.87887, 76.85243 30.87069...	6.0
13	POLYGON ((76.80276 33.23666, 76.80630 33.23623...	2.0
14	POLYGON ((77.89957 35.42789, 77.90297 35.42759...	1.0
15	POLYGON ((87.59989 25.31466, 87.60688 25.31138...	20.0
16	MULTIPOLYGON (((74.67097 13.19986, 74.67097 13...	29.0
17	MULTIPOLYGON (((76.46736 9.54097, 76.46736 9.5...	32.0
18	MULTIPOLYGON (((73.01014 8.28042, 73.01014 8.2...	31.0
19	POLYGON ((78.36465 26.86884, 78.36688 26.86259...	23.0
20	MULTIPOLYGON (((73.45597 15.88986, 73.45597 15...	27.0
21	POLYGON ((94.57723 25.64833, 94.57609 25.64470...	14.0
22	POLYGON ((91.85384 26.10479, 91.86470 26.10035...	17.0
23	POLYGON ((92.80080 24.41905, 92.80370 24.41879...	15.0
24	POLYGON ((95.21445 26.93695, 95.21706 26.93420...	13.0
25	MULTIPOLYGON (((84.76986 19.10597, 84.76986 19...	21.0
26	MULTIPOLYGON (((79.84486 10.82653, 79.84486 10...	34.0
27	POLYGON ((75.86877 32.48868, 75.88712 32.47203...	3.0
28	POLYGON ((73.88944 29.97761, 73.89118 29.97007...	8.0
29	POLYGON ((88.64526 28.09912, 88.65411 28.08984...	11.0
30	MULTIPOLYGON (((77.55596 8.07903, 77.55596 8.0...	33.0
31	POLYGON ((78.33625 19.88319, 78.34669 19.88140...	NaN
32	POLYGON ((92.18520 24.52287, 92.18896 24.52019...	16.0
33	POLYGON ((77.58468 30.40878, 77.58639 30.40801...	9.0
34	POLYGON ((79.19478 31.35362, 79.19817 31.35196...	5.0
35	MULTIPOLYGON (((88.01861 21.57278, 88.01889 21...	19.0

	State	Migrant persons
0	ANDAMAN & NICOBAR ISLANDS	81267.0
1	ANDHRA PRADESH	1591890.0
2	ARUNACHAL PRADESH	136010.0
3	ASSAM	495699.0
4	BIHAR	1111954.0

5	CHANDIGARH	633966.0
6	CHHATTISGARH	1267668.0
7	DADRA & NAGAR HAVELI	135602.0
8	DAMAN & DIU	124522.0
9	NCT OF DELHI	6330065.0
10	GOA	269689.0
11	GUJARAT	3916075.0
12	HARYANA	3626318.0
13	HIMACHAL PRADESH	395504.0
14	JAMMU & KASHMIR	155187.0
15	JHARKHAND	2195521.0
16	KARNATAKA	3247660.0
17	KERALA	654423.0
18	LAKSHADWEEP	6077.0
19	MADHYA PRADESH	2744332.0
20	MAHARASHTRA	9087380.0
21	MANIPUR	20100.0
22	MEGHALAYA	107915.0
23	MIZORAM	41380.0
24	NAGALAND	108020.0
25	ODISHA	855096.0
26	PUDUCHERRY	339967.0
27	PUNJAB	2488299.0
28	RAJASTHAN	2604298.0
29	SIKKIM	61163.0
30	TAMIL NADU	1650771.0
31	NaN	NaN
32	TRIPURA	87378.0
33	UTTAR PRADESH	4061933.0
34	UTTARAKHAND	1250575.0
35	WEST BENGAL	2381045.0



We notice that Bihar and Jharkhand present one of the highest % of migrant males to Maharashtra. Clearly, the trend implies a mass gendered biased migration of people from these states to Maharashtra.

End of Question 3
