On importe la librairie pandas pour charger les données, ainsi que la classe CA du package fanalysis. Les données sont transformées en matrice de type numpy.ndarray.

Les données doivent se présenter sous forme de tableau croisé, avec des fréquences absolues (effectifs de chaque cellule).

In [1]:

```
import pandas as pd
from fanalysis.ca import CA
import matplotlib as plt
%matplotlib inline
```

In [2]:

```
df = pd.read_table(r"C:\Users\Bilel Khms\Desktop\db.txt", header=0, index_col=0, delimiter="\t", encoding="utf-8"
)
```

In [3]:

```
print(df)
```

```
         total_deaths  new_deaths  total_cases  new_cases  total_recavred
usa             12841        1970       400335      33331             21674
spain           14045         704       141942       5267             43208
italy           17127         604       135586       3039             24392
germany          2016         206       107663       4288             36081
France          10328        1417       109069      11059             19337
china            3331           0        81740         32             77167
iran             3872         133        62589       2089             27039
uk               6159         786        55242       3634               135
turkey            725          76        34109       3892              1582
tunisia            23           1          623         22                25
egypt              94           9         1450        128               276
libya               1           0           20          1                 1
chad                0           0           10          1                 2
```

In [4]:

```
X = df.values
```

In [5]:

```
print(X)
print("\n")
```

```
[[ 12841   1970 400335  33331  21674]
 [ 14045    704 141942   5267  43208]
 [ 17127    604 135586   3039  24392]
 [  2016    206 107663   4288  36081]
 [ 10328   1417 109069  11059  19337]
 [  3331      0  81740     32  77167]
 [  3872    133  62589   2089  27039]
 [  6159    786  55242   3634    135]
 [   725     76  34109   3892   1582]
 [    23      1    623     22     25]
 [    94      9   1450    128    276]
 [     1      0     20      1      1]
 [     0      0     10      1      2]]
```

On crée une instance de la classe CA, en lui passant ici des étiquettes pour les lignes et les colonnes.

In [6]:

```
my_ca = CA(row_labels=df.index.values, col_labels=df.columns.values)
```

On estime le modèle en appliquant la méthode fit de la classe CA sur le jeu de données.

In [7]:

```
print(my_ca.fit(X))
```

```
CA(col_labels=array(['total_deaths', 'new_deaths', 'total_cases', 'new_cases',
       'total_recavred '], dtype=object),
   n_components=None,
   row_labels=array(['usa', 'spain', 'italy', 'germany', 'France', 'china', 'iran',
       'uk', 'turkey', 'tunisia', 'egypt', 'libya', 'chad'], dtype=object),
   stats=True)
```

Analyse des valeurs propres

L'attribut my_ca.eig_ contient :

- en 1ère ligne : les valeurs propres en valeur absolue
- en 2ème ligne : les valeurs propres en pourcentage de la variance totale
- en 3ème ligne : les valeurs propres en pourcentage cumulé de la variance totale

In [8]:

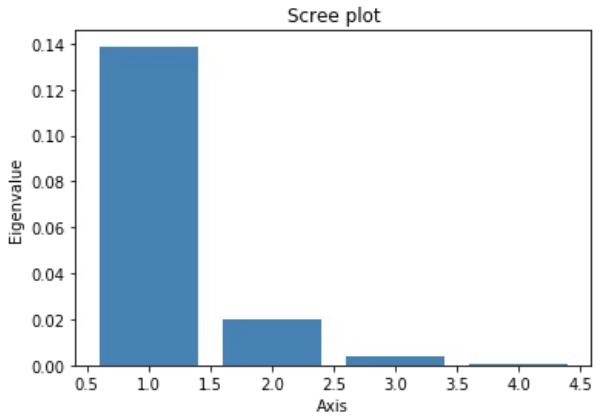
```
print(my_ca.eig_)
```

```
[[1.38808207e-01 1.97886163e-02 3.87875563e-03 3.72806822e-04]
 [8.52376929e+01 1.21515582e+01 2.38182013e+00 2.28928780e-01]
 [8.52376929e+01 9.73892511e+01 9.97710712e+01 1.00000000e+02]]
```

Les valeurs propres peuvent être représentées graphiquement (par défaut : représentation en valeur absolue).

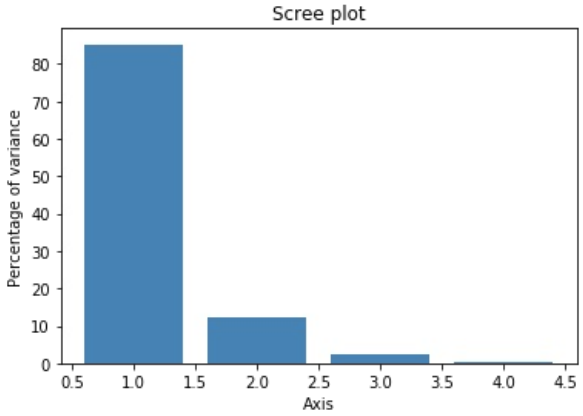
In [9]:

```
my_ca.plot_eigenvalues()
```



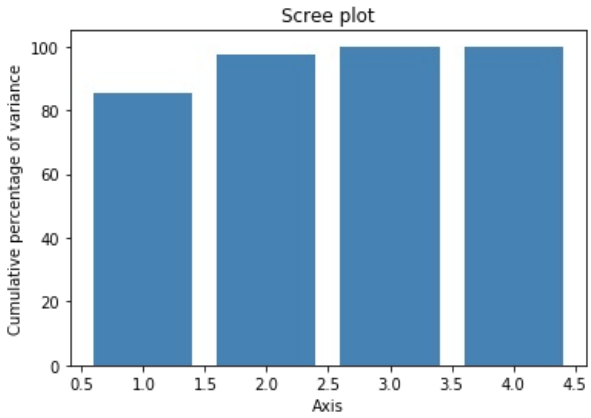
In [10]:

```
my_ca.plot_eigenvalues(type="percentage")
```



In [11]:

```
my_ca.plot_eigenvalues(type="cumulative")
```



Quand l'objet my_ca a été instancié, son paramètre stats a reçu la valeur True par défaut. En conséquence, lors de l'exécution de la méthode my_ca.fit(X), les statistiques suivantes ont été calculées :

- my_ca.row_contrib_ : contributions des points lignes à la variance de l'axe
- my_ca.col_contrib_ : contributions des points colonnes à la variance de l'axe
- my_ca.row_cos2_ : cosinus carrés des points lignes
- my_ca.col_cos2_ : cosinus carrés des points colonnes
  Par défaut, les coordonnées des points lignes et colonnes, leurs contributions et cosinus carrés sont calculés sur l'ensemble des axes extraits de l'analyse.

Extraction des statistiques sur les points lignes

Export de la totalité des données lignes vers une DataFrame pandas

On peut simplement envoyer vers une Dataframe : les coordonnées, les contributions et les cos2 de chacun des points lignes, pour tous les axes factoriels (identifiés par les suffixes dim1, dim2, etc.).

In [12]:

```
df_rows = my_ca.row_topandas()
```

In [13]:

```
print(df_rows)
```

```
         row_coord_dim1  row_coord_dim2  row_coord_dim3  row_coord_dim4  \
usa            0.335622       -0.115010        0.017820        0.000319
spain         -0.140095        0.120882        0.005445       -0.009715
italy          0.042482        0.256524        0.071279       -0.014726
germany       -0.210056       -0.131526        0.062148        0.022496
France         0.128998        0.064647       -0.160711        0.000304
china         -0.847519       -0.078963       -0.028584        0.003462
iran          -0.328427       -0.006230        0.013429       -0.007924
uk             0.433203        0.219577       -0.030615        0.060766
turkey         0.379538       -0.200806       -0.050419       -0.066755
tunisia        0.316364       -0.032138        0.199898        0.029587
egypt          0.084286       -0.023087       -0.076036       -0.028393
libya          0.307667       -0.009360        0.140613       -0.027920
chad           0.063692       -0.262529       -0.031562       -0.054437

         row_contrib_dim1  row_contrib_dim2  row_contrib_dim3  \
usa             25.025391         20.613629          2.524754
spain            1.902818          9.937343          0.102861
italy            0.154147         39.425224         15.529612
germany          3.132862          8.615758          9.814012
France           1.189028          2.094690         66.045073
china           55.078360          3.353771          2.242026
iran             4.879032          0.012317          0.291911
uk               5.848985         10.540783          1.045447
turkey           2.748932          5.397675          1.736092
tunisia          0.032823          0.002376          0.468969
egypt            0.006570          0.003457          0.191333
libya            0.001029          0.000007          0.007690
chad             0.000025          0.002970          0.000219

         row_contrib_dim4  row_cos2_dim1  row_cos2_dim2  row_cos2_dim3  \
usa              0.008400       0.892659       0.104824       0.002517
spain            3.406868       0.571156       0.425234       0.000863
italy            6.896463       0.024754       0.902584       0.069687
germany         13.378594       0.670662       0.262940       0.058706
France           0.002465       0.356726       0.089591       0.553682
china            0.342190       0.990261       0.008596       0.001126
iran             1.057583       0.997393       0.000359       0.001667
uk              42.850180       0.780282       0.200468       0.003897
turkey          31.662845       0.752724       0.210707       0.013284
tunisia          0.106890       0.705062       0.007276       0.281496
egypt            0.277589       0.499423       0.037469       0.406433
libya            0.003154       0.820994       0.000760       0.171485
chad             0.006778       0.052727       0.895809       0.012948

         row_cos2_dim4
usa       8.047709e-07
spain     2.746514e-03
italy     2.974462e-03
germany   7.692040e-03
France    1.986032e-06
china     1.652362e-05
iran      5.806516e-04
uk        1.535297e-02
turkey    2.328575e-02
tunisia   6.166751e-03
egypt     5.667502e-02
libya     6.760780e-03
chad      3.851666e-02
```

Statistiques pour les points lignes

In [14]:

```
# Coordonnées des points lignes
print(my_ca.row_coord_)
```

```
[[ 3.35621578e-01 -1.15010321e-01  1.78199962e-02  3.18671497e-04]
 [-1.40095445e-01  1.20881741e-01  5.44489749e-03 -9.71487966e-03]
 [ 4.24823886e-02  2.56524351e-01  7.12787888e-02 -1.47261399e-02]
 [-2.10056296e-01 -1.31526118e-01  6.21479806e-02  2.24959688e-02]
 [ 1.28998253e-01  6.46469292e-02 -1.60711412e-01  3.04375537e-04]
 [-8.47519124e-01 -7.89634058e-02 -2.85836705e-02  3.46200131e-03]
 [-3.28427154e-01 -6.23044904e-03  1.34287654e-02 -7.92435149e-03]
 [ 4.33202773e-01  2.19577349e-01 -3.06154441e-02  6.07661397e-02]
 [ 3.79538153e-01 -2.00806133e-01 -5.04194965e-02 -6.67548432e-02]
 [ 3.16363883e-01 -3.21375449e-02  1.99898360e-01  2.95870408e-02]
 [ 8.42861877e-02 -2.30865900e-02 -7.60355666e-02 -2.83934482e-02]
 [ 3.07666836e-01 -9.36006693e-03  1.40612539e-01 -2.79195930e-02]
 [ 6.36919311e-02 -2.62528792e-01 -3.15620369e-02 -5.44369229e-02]]
```

In [15]:

```
# Contributions des points lignes
print(my_ca.row_contrib_)
```

```
[[2.50253909e+01 2.06136294e+01 2.52475394e+00 8.40037844e-03]
 [1.90281768e+00 9.93734284e+00 1.02861129e-01 3.40686785e+00]
 [1.54147011e-01 3.94252236e+01 1.55296119e+01 6.89646268e+00]
 [3.13286154e+00 8.61575842e+00 9.81401152e+00 1.33785944e+01]
 [1.18902756e+00 2.09469011e+00 6.60450731e+01 2.46476234e-03]
 [5.50783604e+01 3.35377134e+00 2.24202633e+00 3.42190334e-01]
 [4.87903162e+00 1.23166877e-02 2.91910969e-01 1.05758254e+00]
 [5.84898516e+00 1.05407825e+01 1.04544675e+00 4.28501800e+01]
 [2.74893181e+00 5.39767518e+00 1.73609234e+00 3.16628454e+01]
 [3.28229053e-02 2.37590096e-03 4.68969296e-01 1.06890124e-01]
 [6.56972818e-03 3.45743382e-03 1.91333439e-01 2.77588933e-01]
 [1.02880475e-03 6.67927330e-06 7.69028463e-03 3.15443111e-03]
 [2.49204534e-05 2.96989232e-03 2.18997863e-04 6.77806197e-03]]
```

In [16]:

```
# Cos2 des points lignes
print(my_ca.row_cos2_)
```

```
[[8.92658978e-01 1.04823693e-01 2.51652380e-03 8.04770906e-07]
 [5.71156340e-01 4.25234393e-01 8.62752818e-04 2.74651403e-03]
 [2.47541775e-02 9.02584440e-01 6.96869200e-02 2.97446245e-03]
 [6.70661770e-01 2.62939690e-01 5.87064999e-02 7.69204020e-03]
 [3.56725694e-01 8.95906189e-02 5.53681701e-01 1.98603193e-06]
 [9.90260973e-01 8.59611849e-03 1.12638445e-03 1.65236164e-05]
 [9.97392925e-01 3.58944138e-04 1.66747888e-03 5.80651604e-04]
 [7.80282190e-01 2.00467656e-01 3.89717911e-03 1.53529747e-02]
 [7.52723637e-01 2.10706832e-01 1.32837803e-02 2.32857502e-02]
 [7.05061748e-01 7.27576647e-03 2.81495735e-01 6.16675110e-03]
 [4.99422679e-01 3.74692516e-02 4.06433046e-01 5.66750242e-02]
 [8.20994105e-01 7.59865238e-04 1.71485249e-01 6.76078046e-03]
 [5.27266732e-02 8.95808993e-01 1.29476688e-02 3.85166648e-02]]
```

Extraction des statistiques sur les points colonnes

Export de la totalité des données colonnes vers une DataFrame pandas

On peut envoyer vers une Dataframe : les coordonnées, les contributions et les cos2 de chacun des points colonnes, pour tous les axes factoriels (identifiés par les suffixes dim1, dim2, etc.).

In [17]:

```
df_cols = my_ca.col_topandas()
```

In [18]:

```
print(df_cols)
```

```
                col_coord_dim1  col_coord_dim2  col_coord_dim3  \
total_deaths          0.108124        0.597785       -0.063529
new_deaths            0.479214        0.281941       -0.433710
total_cases           0.140238       -0.017816        0.026186
new_cases             0.540969       -0.241553       -0.230125
total_recavred       -0.817434       -0.030190       -0.028643

                col_coord_dim4  col_contrib_dim1  col_contrib_dim2  \
total_deaths         -0.023032          0.389816         83.580254
new_deaths            0.275078          0.640910          1.556161
total_cases           0.002301         10.505142          1.189326
new_cases            -0.034096          9.235381         12.916198
total_recavred       -0.001288         79.228752          0.758061

                col_contrib_dim3  col_contrib_dim4  col_cos2_dim1  \
total_deaths            4.816007          6.585535       0.031292
new_deaths             18.787016         78.628520       0.400843
total_cases            13.107541          1.052868       0.951226
new_cases              59.808106         13.659804       0.722381
total_recavred          3.481330          0.073274       0.997412

                col_cos2_dim2  col_cos2_dim3  col_cos2_dim4
total_deaths         0.956485       0.010803       0.001420
new_deaths           0.138750       0.328331       0.132077
total_cases          0.015353       0.033165       0.000256
new_cases            0.144028       0.130722       0.002870
total_recavred       0.001360       0.001225       0.000002
```

Statistiques pour les points colonnes

In [19]:

```
# Coordonnées des points colonnes
print(my_ca.col_coord_)
```

```
[[ 0.10812414  0.59778453 -0.06352941 -0.02303152]
 [ 0.47921447  0.28194136 -0.43370951  0.27507797]
 [ 0.14023849 -0.01781627  0.02618578  0.00230085]
 [ 0.54096922 -0.24155312 -0.23012501 -0.03409587]
 [-0.81743411 -0.03019003 -0.02864327 -0.00128831]]
```

In [20]:

```
# Contributions des points colonnes
print(my_ca.col_contrib_)
```

```
[[3.89816302e-01 8.35802544e+01 4.81600652e+00 6.58553457e+00]
 [6.40909997e-01 1.55616092e+00 1.87870160e+01 7.86285196e+01]
 [1.05051416e+01 1.18932596e+00 1.31075410e+01 1.05286781e+00]
 [9.23538052e+00 1.29161978e+01 5.98081063e+01 1.36598039e+01]
 [7.92287516e+01 7.58060919e-01 3.48133025e+00 7.32741112e-02]]
```

In [21]:

```
# Cos2 des points colonnes
print(my_ca.col_cos2_)
```

```
[[3.12920706e-02 9.56485250e-01 1.08028575e-02 1.41982247e-03]
 [4.00842722e-01 1.38749602e-01 3.28331111e-01 1.32076564e-01]
 [9.51226276e-01 1.53526416e-02 3.31650325e-02 2.56049825e-04]
 [7.22380620e-01 1.44027779e-01 1.30721981e-01 2.86962049e-03]
 [9.97412373e-01 1.36049306e-03 1.22465691e-03 2.47748598e-06]]
```

Graphiques

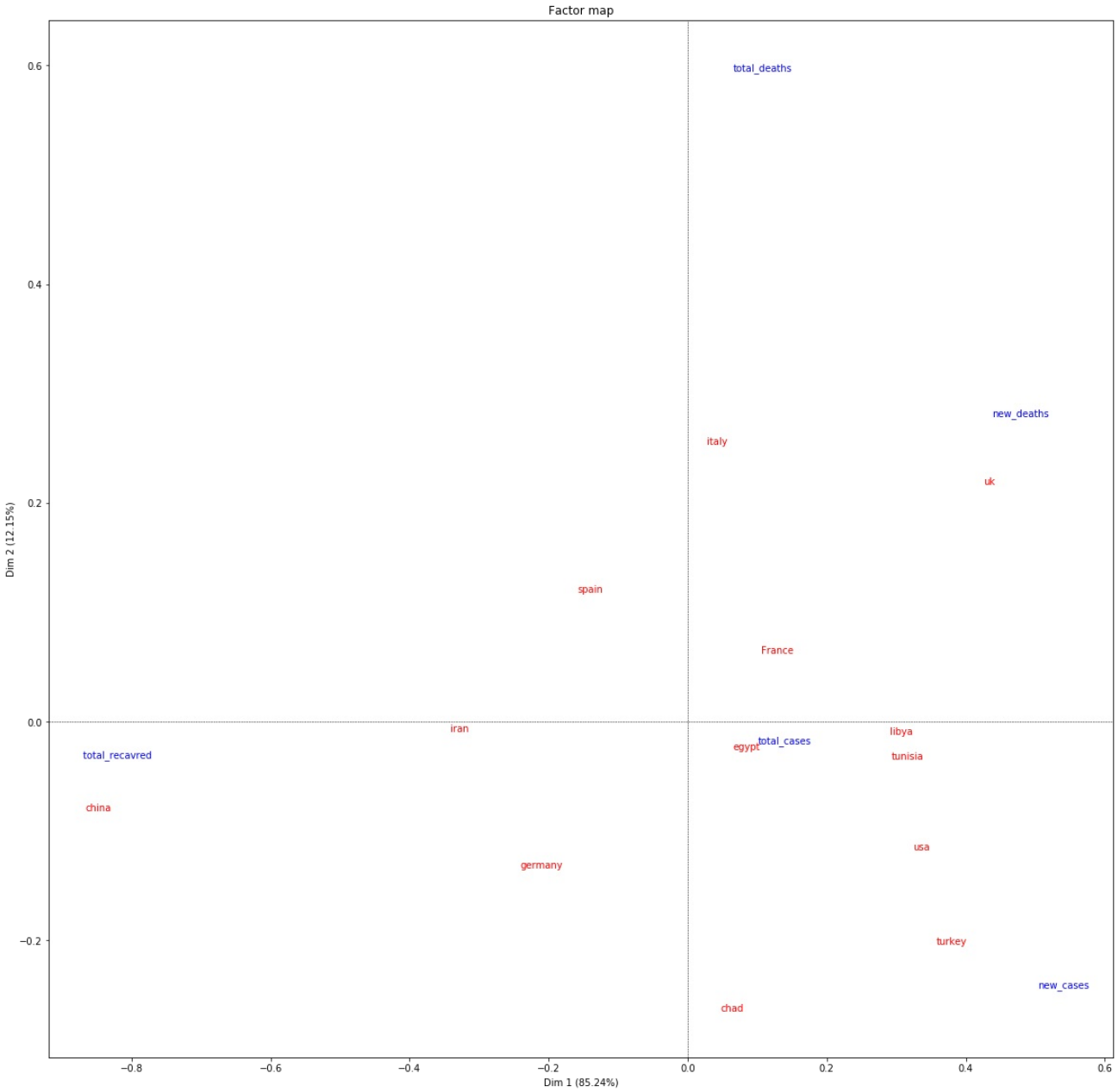
2 types de graphiques peuvent être réalisés :

- Les mapping classiques qui représentent les points lignes et colonnes sur un plan factoriel
- Des graphiques qui permettent d'interpréter rapidement les axes : on choisit un axe factoriel (le 1er axe dans notre exemple) et on observe quels sont les points lignes et colonnes qui présentent les plus fortes contributions et cos2 pour cet axe

Graphiques factoriels

In [25]:

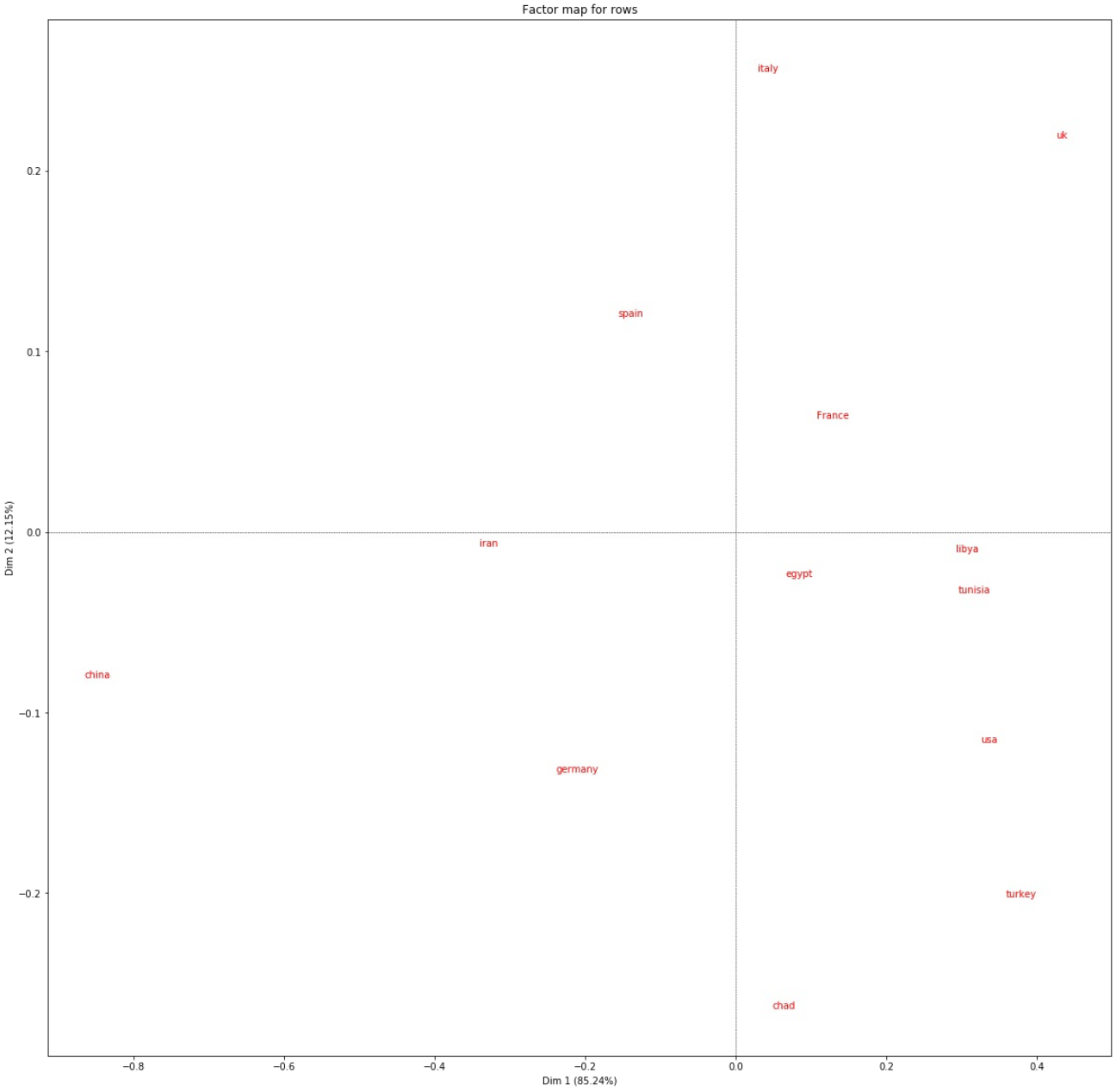
```
# Mapping simultané des points lignes et colonnes
# Les paramètres de la méthode mapping indiquent que ce sont les axes 1 et 2 qui sont ici représentés
my_ca.mapping(num_x_axis=1, num_y_axis=2)
plt.rcParams["figure.figsize"]=20,20
```



Ce graphe montre que La Chine est le pays qui a le plus grand nombre des cas rétablis dans le monde car 'Total_recovered' et 'China' sont dans le même nuage, ainsi que la Turkey a un nombre important de 'new_cases' et 'UK' et 'New_deaths' se trouvent dans le même nuage ce qui prouve que le UK a un nombre imprtant de 'new_deaths'

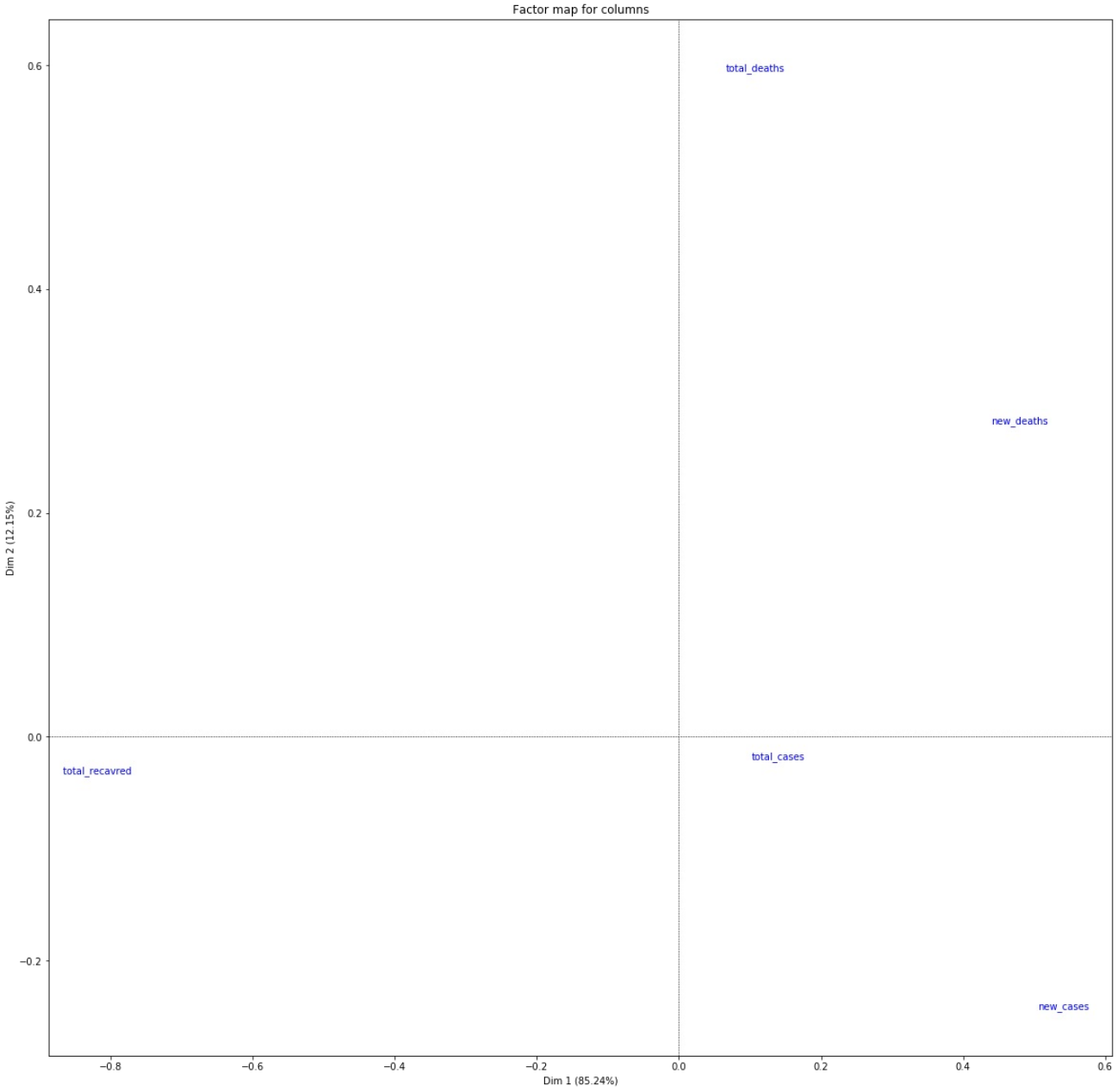
In [26]:

```
# Mapping des points lignes
my_ca.mapping_row(num_x_axis=1, num_y_axis=2)
plt.rcParams["figure.figsize"]=20,20
```



In [27]:

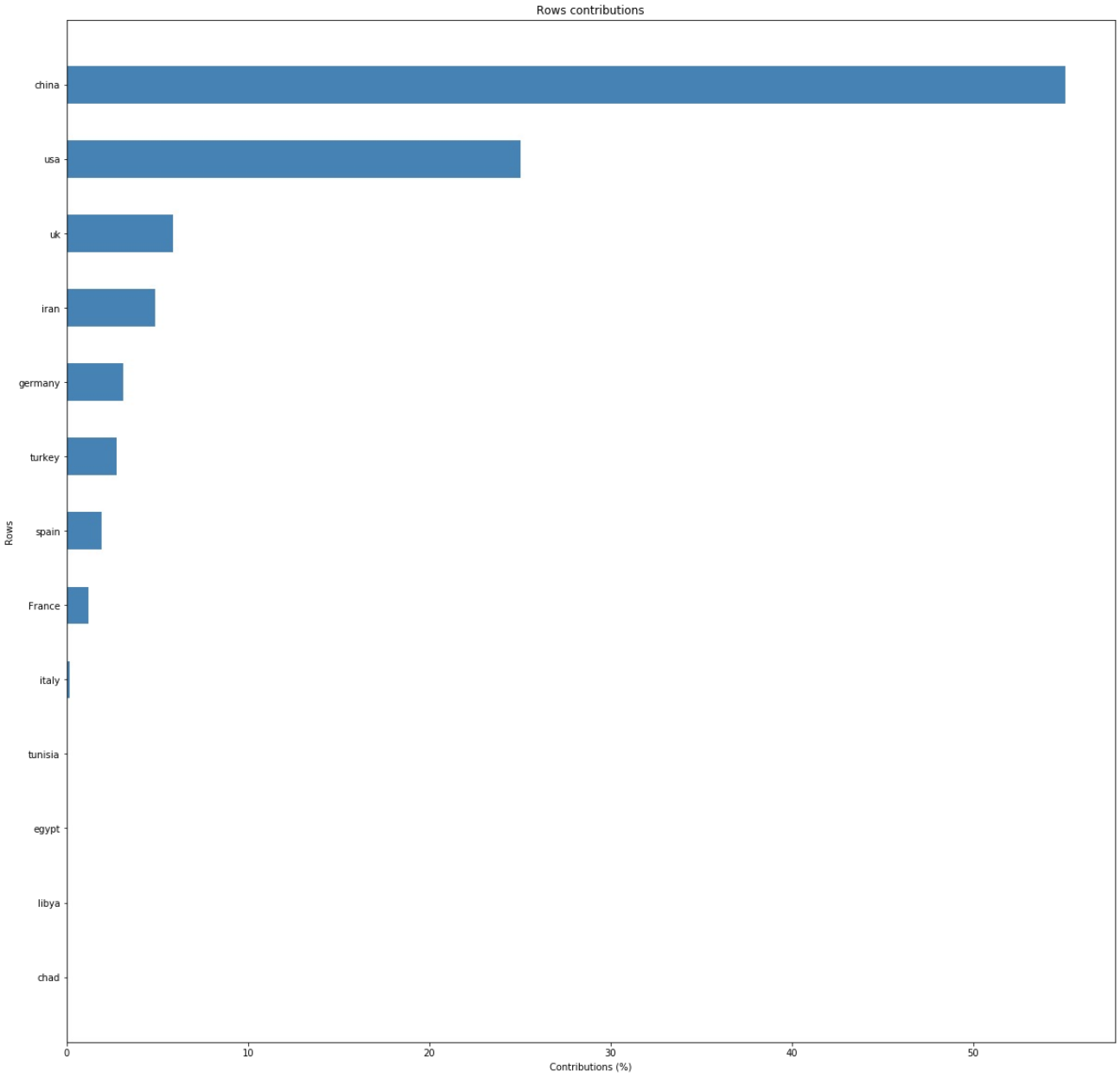
```
# Mapping des points colonnes
my_ca.mapping_col(num_x_axis=1, num_y_axis=2)
plt.rcParams["figure.figsize"]=20,20
```



Analyse du 1er axe - Points lignes

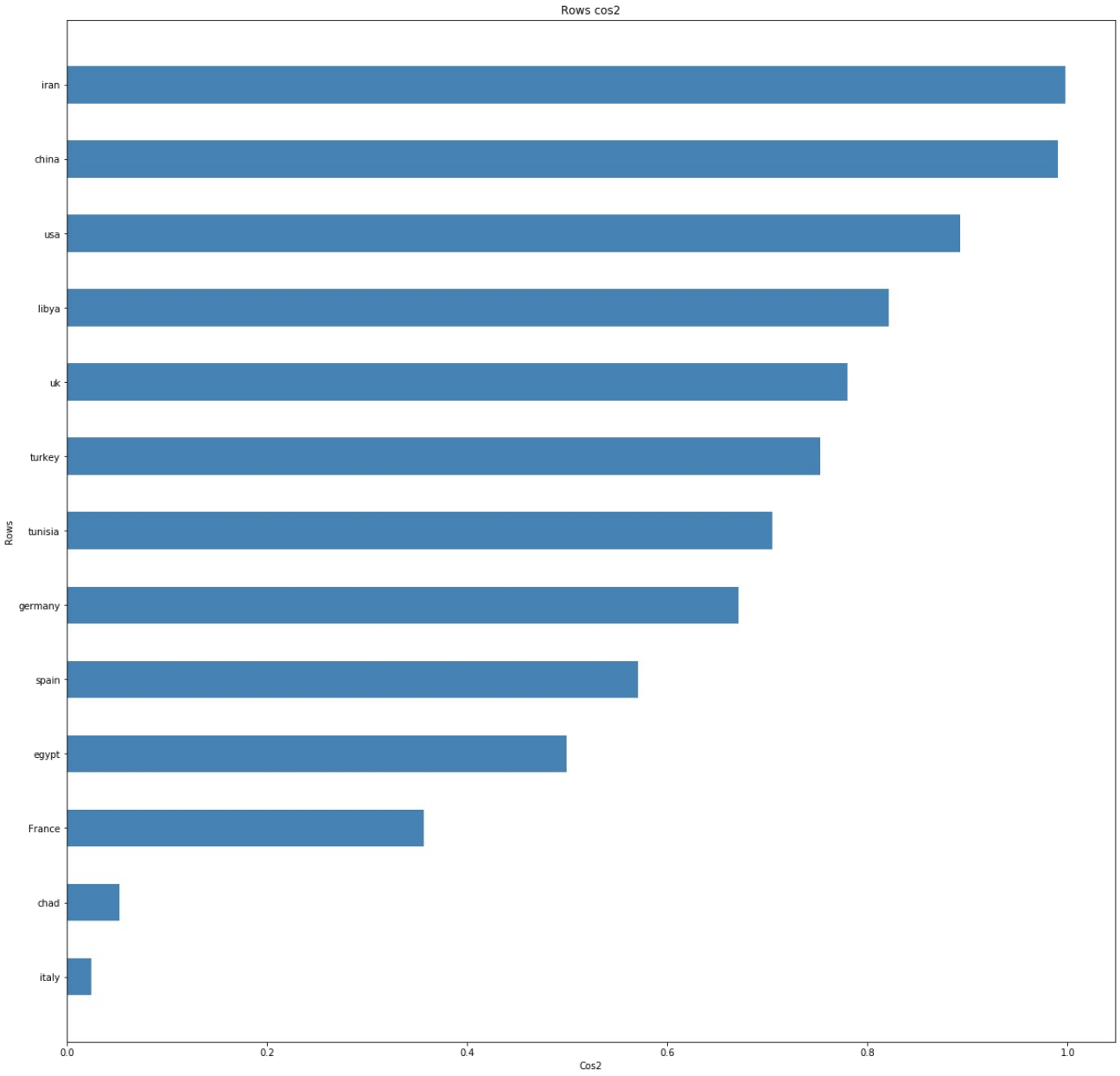
In [28]:

```
# Classement des points lignes en fonction de leur contribution au 1er axe
# Le paramètre de la méthode plot_row_contrib indique que c'est pour l'axe numéro 1 que les contributions sont ic
i  représentées
my_ca.plot_row_contrib(num_axis=1)
```



In [29]:

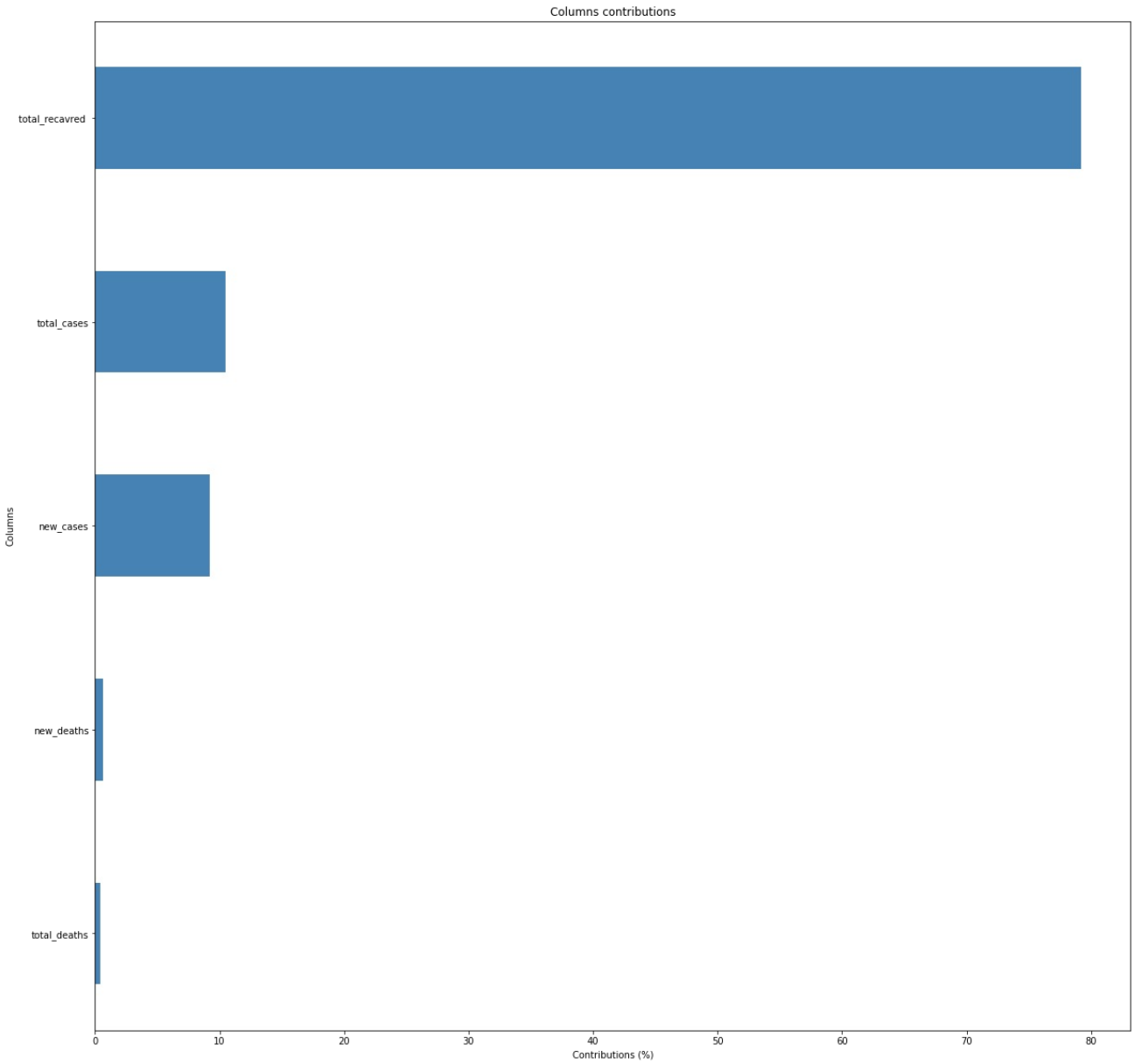
```
# Classement des points lignes en fonction de leur cos2 sur le 1er axe
my_ca.plot_row_cos2(num_axis=1)
```



Analyse du 1er axe - Points colonnes

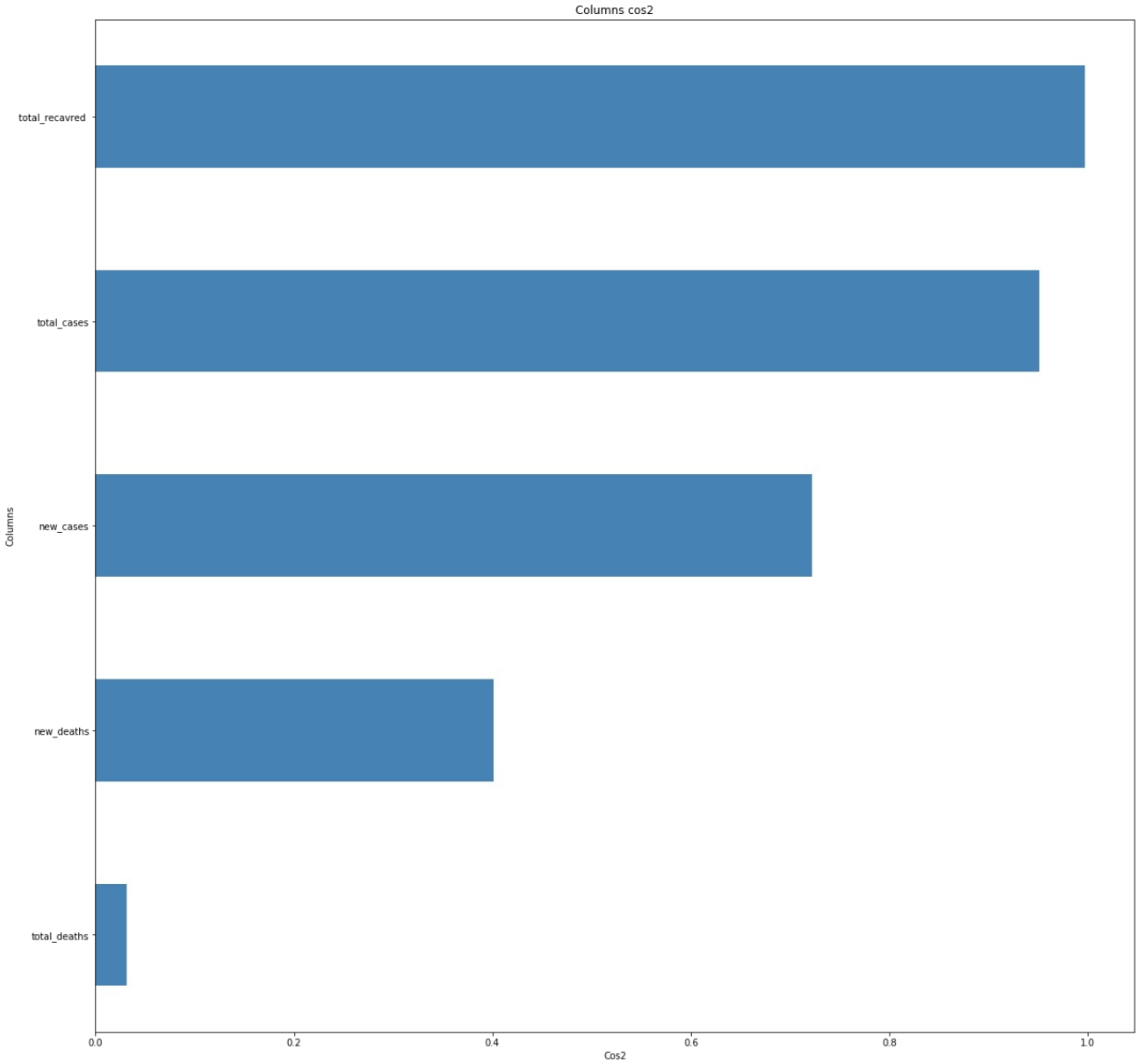
In [30]:

```
# Classement des points colonnes en fonction de leur contribution au 1er axe
my_ca.plot_col_contrib(num_axis=1)
```



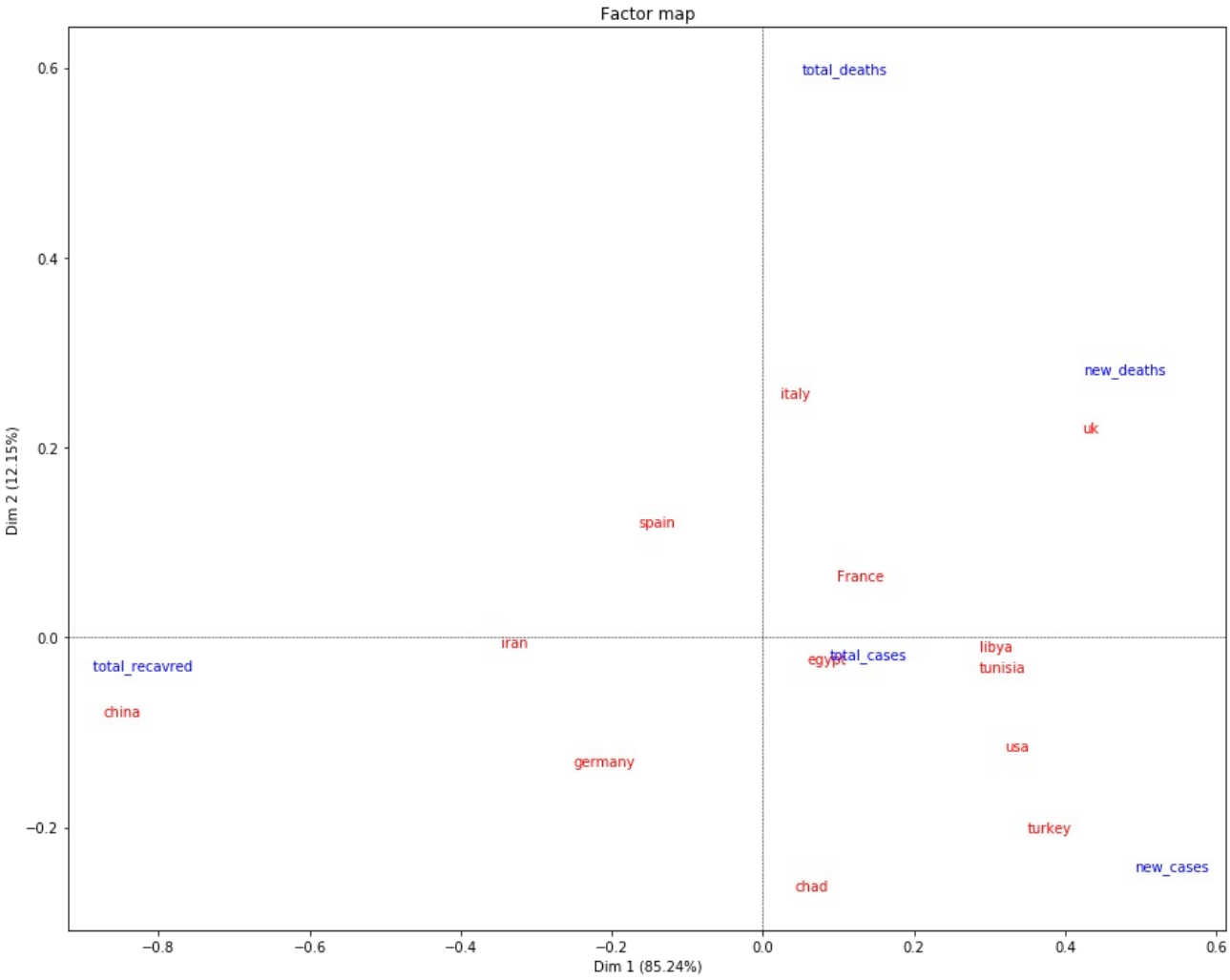
In [31]:

```
# Classement des points colonnes en fonction de leur cos2 sur le 1er axe
my_ca.plot_col_cos2(num_axis=1)
```



In [34]:

```
my_ca.mapping(1, 2, figsize=(15, 12))
```



In [ ]:

```
```