Dynamic Functional Connectivity

A tutorial

Contents

Introduction **1**

Fig 1 **1**

Fig 2 **1**

Fig 3 **2**

Fig 4 **2**

Fig 5 **2**

Fig 6 **2**

Fig 7 **2**

Sliding windows **2**

Cofluctuations **8**

Package References **20**

Introduction

This document was prepared on 2021-02-15.

Fig 1

Fig 2

Adapted from: https://en.wikipedia.org/wiki/Voxel#/media/File:Voxels.svg

Fig 3

Fig 4

Fig 5

Fig 6

Fig 7

Sliding windows

```
import sys
sys.path.insert(1, 'libs/')

import dynfc as dyn
import numpy as np
from numpy.random import seed, rand
import scipy as sc
from scipy import io
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib.cbook as cbook
import matplotlib.cm as cm
import matplotlib.patches as patches

font = {'weight' : 'regular',
        'size'   : 24}
plt.rc('font', **font)

ts = sc.io.loadmat('data/ts.mat')['ts']
ts = ts.transpose()

corr_mats, idx = dyn.corr_slide(ts,300,50)

idx.shape
```

```
> (24,)
```

```
a = [1,1,1,1,1.6]

fig,ax = plt.subplots(1)
fig.set_figheight(10)
fig.set_figwidth(20)
plt.style.use('tableau-colorblind10')
for i in range(5):

    plt.plot(2*i*a[i] + ts[i,]/1.4)
```

```
> [<matplotlib.lines.Line2D object at 0x7f9c5fd194a8>]
> [<matplotlib.lines.Line2D object at 0x7f9b4b64e438>]
> [<matplotlib.lines.Line2D object at 0x7f9aea98cc50>]
> [<matplotlib.lines.Line2D object at 0x7f9aea98ca20>]
> [<matplotlib.lines.Line2D object at 0x7f9b4b64e898>]
```

```
plt.xlabel('Time [TRs]')
```

```
> Text(0.5, 0, 'Time [TRs]')
```

```
plt.ylabel('BOLD')
```

```
> Text(0, 0.5, 'BOLD')
```

```
ax.tick_params(left=False)
ax.set_yticklabels([])
```

```
> [Text(0, -2.0, ''), Text(0, 0.0, ''), Text(0, 2.0, ''), Text(0, 4.0, ''), Text(0, 6.0, ''), Text(0, 8
```

```
ax.set_ylim(-3,15.4)
```

```
> (-3.0, 15.4)
```

```
ax.set_xlim(0,1200)
```

```
> (0.0, 1200.0)
```

```
rect = patches.Rectangle((idx[0],-2.8),300,18,linewidth=4,edgecolor='#595959',facecolor='none')
ax.add_patch(rect)
```

```
> <matplotlib.patches.Rectangle object at 0x7f9b0d19d908>
```

```
rect = patches.Rectangle((idx[5],-2.8),300,18,linewidth=4,edgecolor='#A56B6B',facecolor='none')
ax.add_patch(rect)
```

```
> <matplotlib.patches.Rectangle object at 0x7f9b0d19dc88>
```

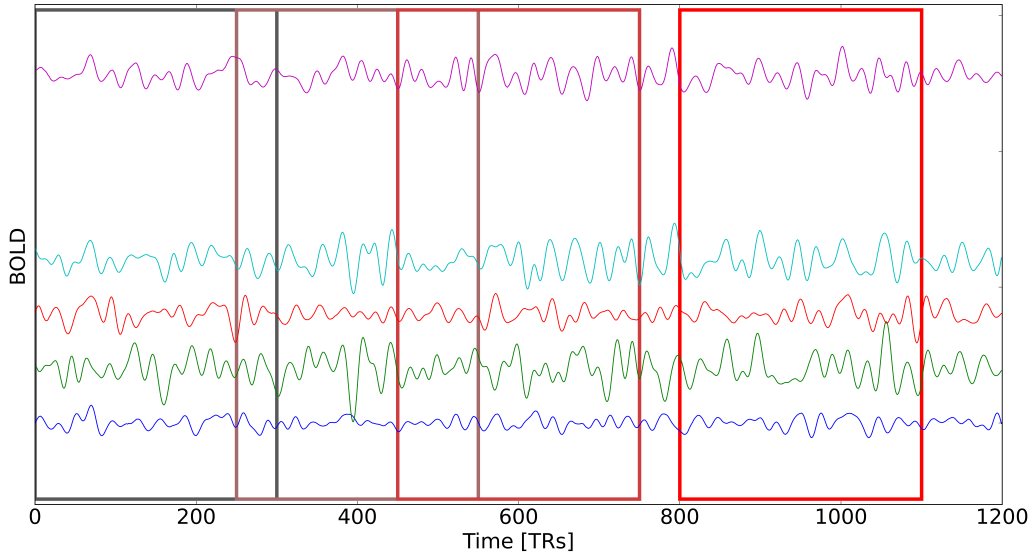
```
rect = patches.Rectangle((idx[9],-2.8),300,18,linewidth=4,edgecolor='#CE3E3E',facecolor='none')
ax.add_patch(rect)
```

```
> <matplotlib.patches.Rectangle object at 0x7f9b0d19df98>
```

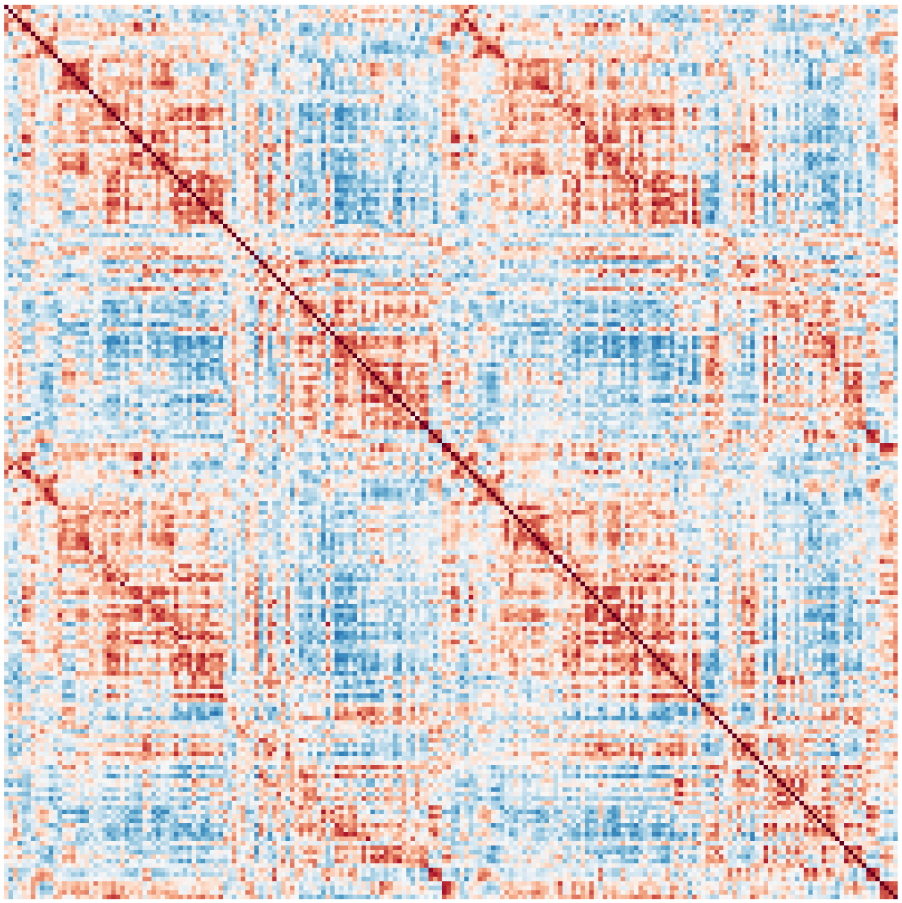
```
rect = patches.Rectangle((idx[16],-2.8),300,18,linewidth=4,edgecolor='#FF0000',facecolor='none')
ax.add_patch(rect)
```

```
> <matplotlib.patches.Rectangle object at 0x7f9b0d1a42e8>
```

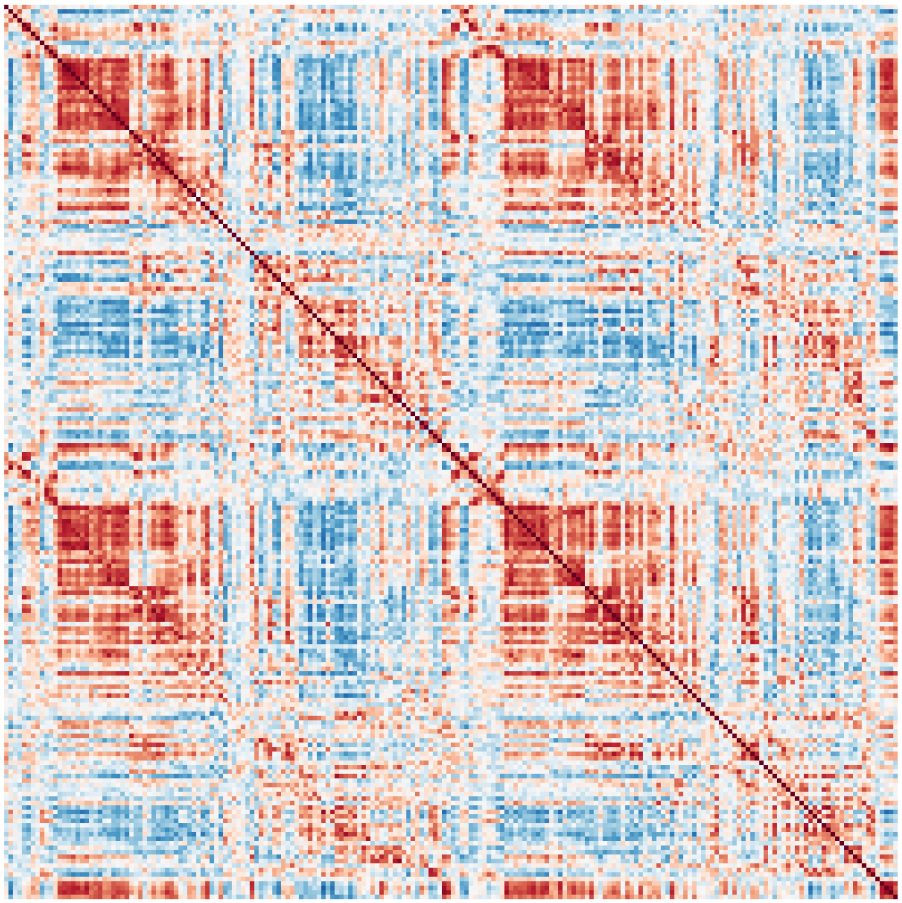
```
plt.show()
```



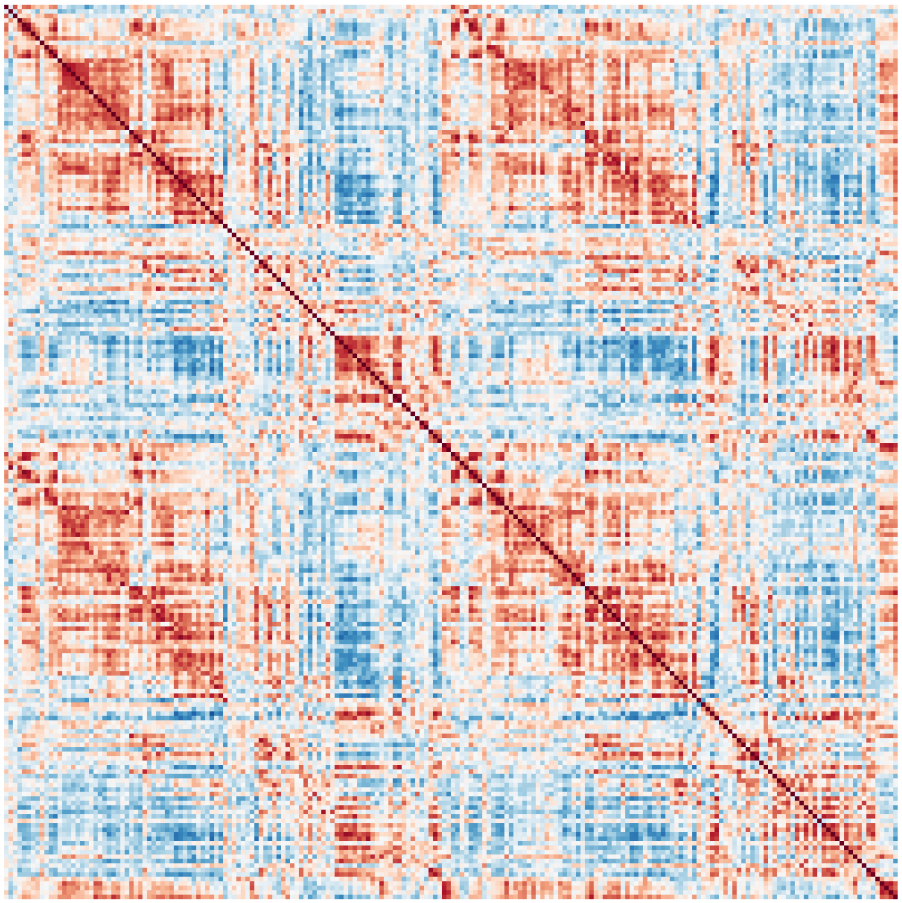

```
> (0.0, 200.0, 200.0, 0.0)
```



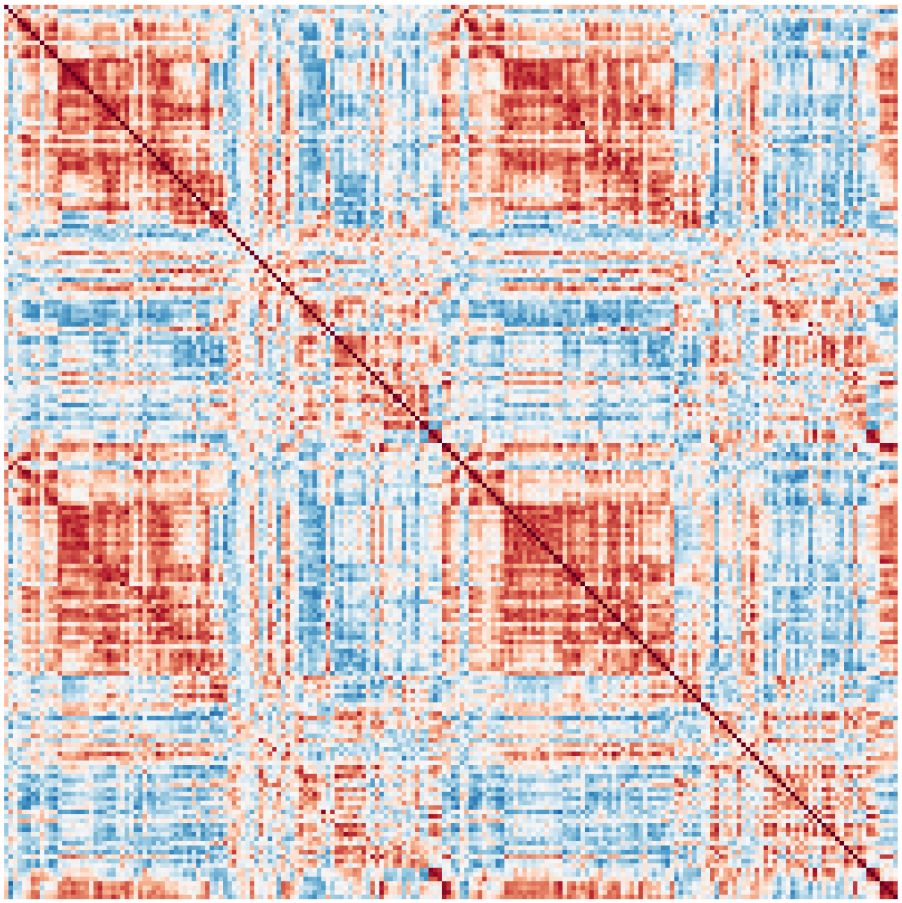
```
> (0.0, 200.0, 200.0, 0.0)
```



```
> (0.0, 200.0, 200.0, 0.0)
```



```
> (0.0, 200.0, 200.0, 0.0)
```



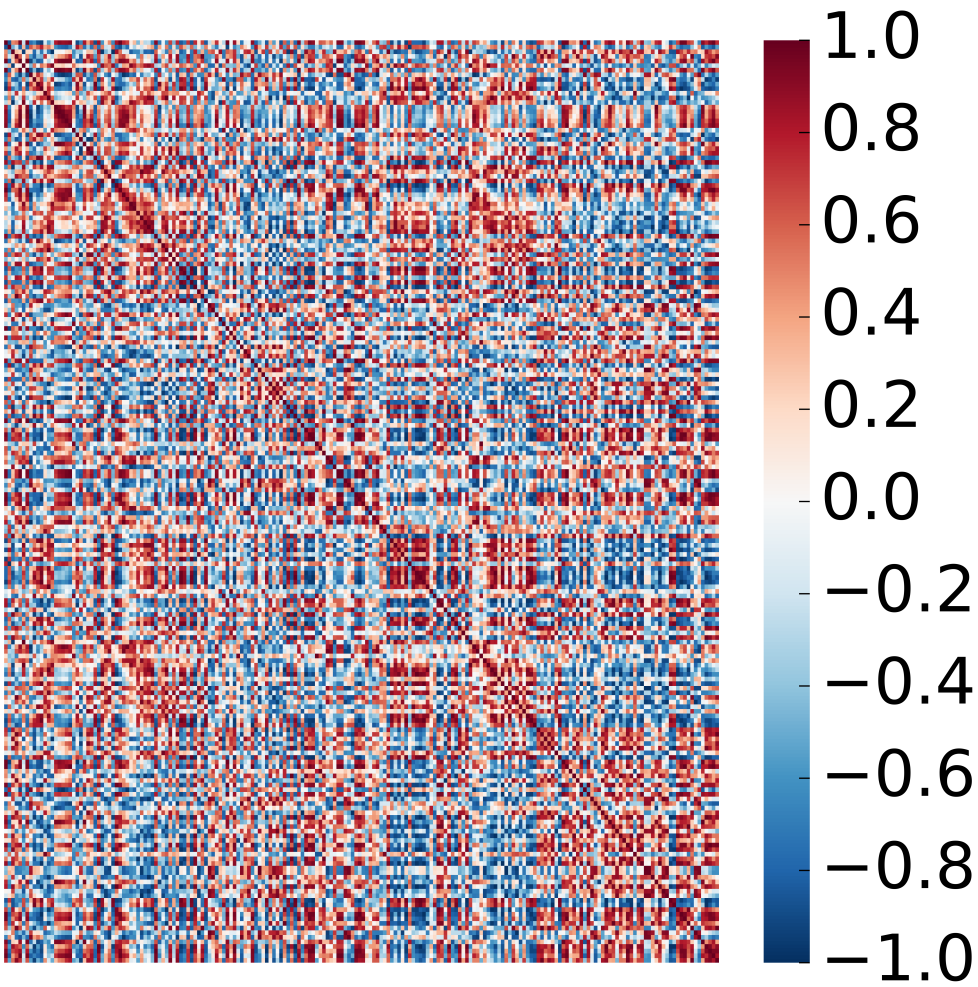
Cofluctuations

```
import sys
sys.path.insert(1, 'libs/')

import numpy as np
import scipy as sc
from scipy import io
import dynfc as dyn
import seaborn as sns
import matplotlib.pyplot as plt

ts = sc.io.loadmat('data/ts.mat')['ts']
ts = ts.transpose()
```

```
mat1, rss = dyn.corr_slide(ts,24)
mat1 = mat1[:,:,0]

aa = plt.figure(figsize = [6,6])
ax = sns.heatmap(mat1,
                 cmap = "RdBu_r",
                 vmin = -1,
                 vmax = 1,
                 square = False,
                 cbar = True)
ax.axis('off')
```

```
> (0.0, 200.0, 200.0, 0.0)
```

```
ax.tick_params(left=False, bottom=False)
plt.show()
```



```
upt = np.triu_indices(mat1.shape[0], k = 1)
vec = (mat1[upt])

toPlot = np.zeros((vec.shape[0],1))
toPlot[:,0] = vec

aa = plt.figure(figsize = (12/50,4))
ax = sns.heatmap(toPlot[:,0:1],
                 cmap = "RdBu_r",
                 vmin = -1,
                 vmax = 1,
                 square = False,
                 cbar = False)
ax.axis('off')
```

```
> (0.0, 1.0, 19900.0, 0.0)
```

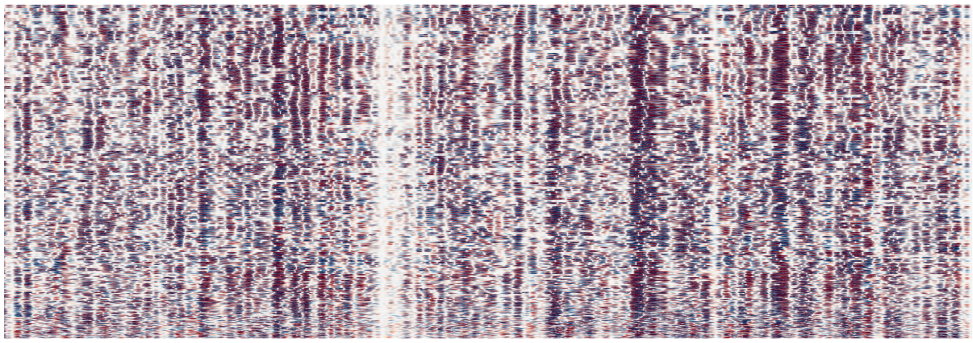
```
plt.show()
```

```
edges_series, corr_mats, rss = dyn.cofluct(ts, 24)

aa = plt.figure(figsize = (12,4))
ax = sns.heatmap(edges_series,
                 cmap = "RdBu_r",
                 vmin = -1,
                 vmax = 1,
                 square = False,
```

```
                  cbar = False)
ax.axis('off')
```

```
> (0.0, 1200.0, 19900.0, 0.0)
```

```
plt.show()
```



#Phase difference

```
import sys
sys.path.insert(1, 'libs/')

import numpy as np
import matplotlib.pyplot as plt
import matplotlib.patches as patches
from mpl_toolkits.mplot3d import Axes3D

theta = np.linspace(0*np.pi, 8*np.pi, 100)
time = np.linspace(0, 8, 100)
y1 = np.sin(theta)
x1 = np.cos(theta)
y2 = np.sin(theta + np.pi/2)
x2 = np.cos(theta + np.pi/2)
r = time/np.max(time)
ones = np.ones(time.shape[0])

fig = plt.figure()
ax = fig.add_subplot(projection='3d')
plt.style.use('classic')

ax.plot(time, x1, y1, color = '#67001f', linewidth=2)
```

```
> [<mpl_toolkits.mplot3d.art3d.Line3D object at 0x7f9c606c1a20>]
```

```
ax.plot(time,ones,y1, linestyle = '--', color = '#0f0f0f')
```

```
> [<mpl_toolkits.mplot3d.art3d.Line3D object at 0x7f9b4b55bdd8>]
```

```
ax.plot(time,x1,-ones, linestyle = '--', color = '#0f0f0f')
```

```
> [<mpl_toolkits.mplot3d.art3d.Line3D object at 0x7f9b4b55b630>]
```

```
ax.quiver(time,0,0,0, 0.99*x1, 0.99*y1,
          length = 0.9,
          normalize = False,
          arrow_length_ratio = 0.1,
          alpha = 0.4,
          color = '#053061')
```

```
> <mpl_toolkits.mplot3d.art3d.Line3DCollection object at 0x7f9bdc905748>
```

```
ax.text(0.5, 1, 0.5, r'$\sin(\theta)$', fontsize = 20)
```

```
> Text(0.5, 1, '$\\sin(\\theta)$')
```

```
ax.text(0.5, -1, -1, r'$\cos(\theta)$', fontsize = 20)
```

```
> Text(0.5, -1, '$\\cos(\\theta)$')
```

```
ax.set_xlabel('Time')
```

```
> Text(0.5, 0, 'Time')
```

```
ax.set_ylabel('Re')
```

```
> Text(0.5, 0, 'Re')
```

```
ax.set_zlabel('Im')
```

```
> Text(0.5, 0, 'Im')
```

```
ax.set_xlim(0,8)
```

```
> (0.0, 8.0)
```

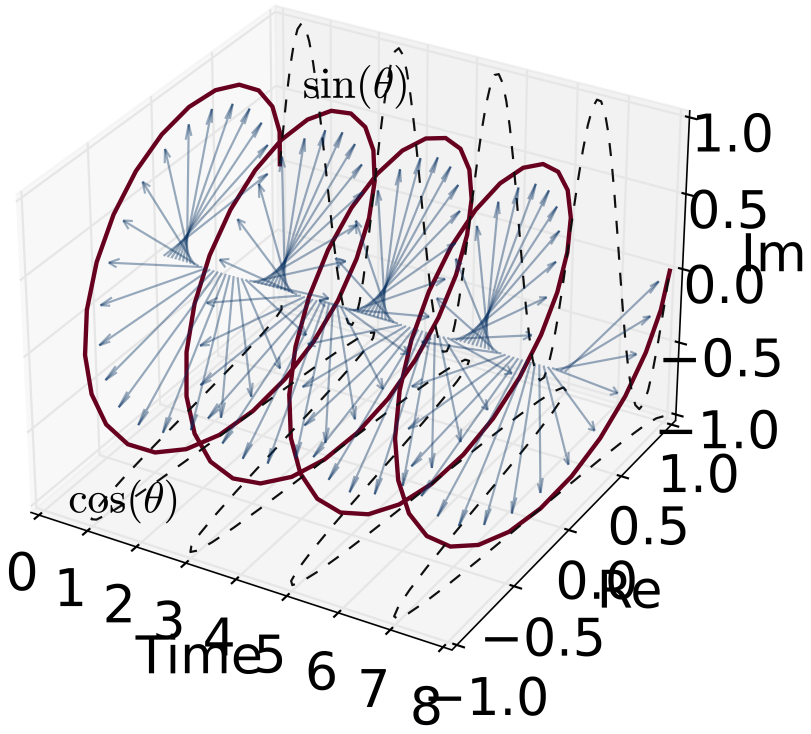
```
ax.set_ylim(-1,1)
```

```
> (-1.0, 1.0)
```

```
ax.set_zlim(-1,1)
```

```
> (-1.0, 1.0)
```

```
plt.show()
```




```
colors1 = plt.cm.Reds(np.arange(0,1,0.01))
colors1[:,-1] = np.arange(0,1,0.01)

colors2 = plt.cm.Blues(np.arange(0,1,0.01))
colors2[:,-1] = np.arange(0,1,0.01)

fig = plt.figure(figsize = (12,8))
ax = fig.add_subplot(projection='3d')
plt.style.use('classic')

ax.scatter(time,ones,r*y1, color = colors1, marker = ",", s = 3)
```

```
> <mpl_toolkits.mplot3d.art3d.Path3DCollection object at 0x7f9b0d20b4a8>
```

```
ax.scatter(time,r*x1,-ones, color = colors1, marker = ",", s = 3)
```

```
> <mpl_toolkits.mplot3d.art3d.Path3DCollection object at 0x7f9aead6f240>
```

```
ax.scatter(time,ones,r*y2, color = colors2, marker = ",", s = 3)
```

```
> <mpl_toolkits.mplot3d.art3d.Path3DCollection object at 0x7f9aead6fa90>
```

```
ax.scatter(time,r*x2,-ones, color = colors2, marker = ",", s = 3)
#ax.plot(time,r*x2,-ones, linestyle = '--', color = colors2[-10,:])
```

```
> <mpl_toolkits.mplot3d.art3d.Path3DCollection object at 0x7f9aead6fe80>
```

```
ax.quiver(time,0,0,0, 0.99*r*x1, 0.99*r*y1,
          length = 0.9,
          normalize = False,
          arrow_length_ratio = 0.1,
          alpha = 1,
          color = colors1)
```

```
> <mpl_toolkits.mplot3d.art3d.Line3DCollection object at 0x7f9aead6a908>
```

```
ax.quiver(time,0,0,0, 0.99*r*x2, 0.99*r*y2,
          length = 0.9,
          normalize = False,
          arrow_length_ratio = 0.1,
          alpha = 1,
          color = colors2)
```

```
> <mpl_toolkits.mplot3d.art3d.Line3DCollection object at 0x7f9aead6af28>
```

```
ax.text(0.5, 1, 0.4, r'$A(t)$ $\sin(\theta)$', fontsize = 20)
```

```
> Text(0.5, 1, '$A(t)$ $\\sin(\\theta)$')
```

```
ax.text(0.5, -0.9, -1, r'$A(t)$ $\cos(\theta)$', fontsize = 20)
```

```
> Text(0.5, -0.9, '$A(t)$ $\\cos(\\theta)$')
```

```
ax.set_xlabel('Time')
```

```
> Text(0.5, 0, 'Time')
```

```
ax.set_ylabel('Re')
```

```
> Text(0.5, 0, 'Re')
```

```
ax.set_zlabel('Im')
```

```
> Text(0.5, 0, 'Im')
```

```
ax.set_xlim(0,8)
```

```
> (0.0, 8.0)
```

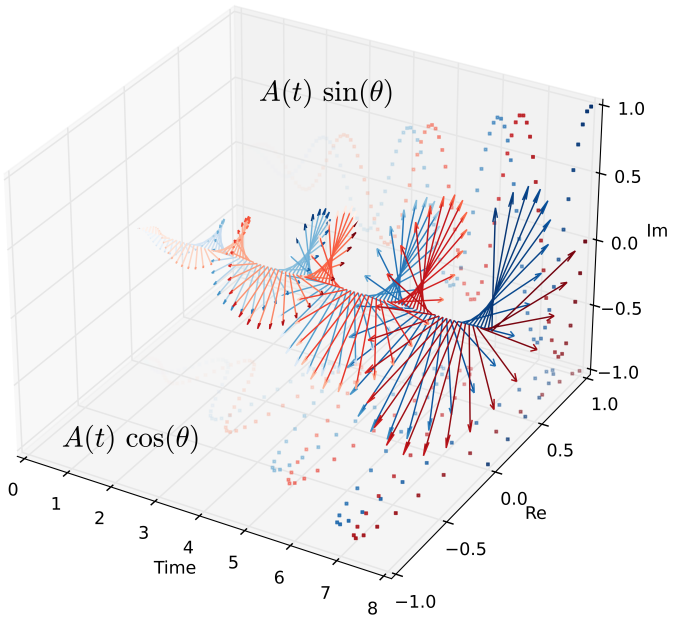
```
ax.set_ylim(-1,1)
```

```
> (-1.0, 1.0)
```

```
ax.set_zlim(-1,1)
```

```
> (-1.0, 1.0)
```

```
plt.show()
```




```
x1_vec = x1
x1_vec = x1_vec[99]
y1_vec = y1
y1_vec = y1_vec[99]

x2_vec = x2
x2_vec = x2_vec[99]
y2_vec = y2
y2_vec = y2_vec[99]
```

```
x, y = 0.0, 0.0

fig,ax = plt.subplots(1)
fig.set_figheight(6)
fig.set_figwidth(7)
ax.scatter(r*x1,r*y1, color = colors1, marker = ".", s = 100)
```

```
> <matplotlib.collections.PathCollection object at 0x7f9a710a7cc0>
```

```
ax.scatter(r*x2,r*y2, color = colors2, marker = ".", s = 100)
```

```
> <matplotlib.collections.PathCollection object at 0x7f9a710e1908>
```

```
ax.arrow(0,0,x1_vec,y1_vec, width = 0.015, color = colors1[-1,:], head_width = 0.1)
```

```
> <matplotlib.patches.FancyArrow object at 0x7f9a710e14a8>
```

```
ax.arrow(0,0,x2_vec,y2_vec, width = 0.015, color = colors2[-1,:], head_width = 0.1)
```

```
> <matplotlib.patches.FancyArrow object at 0x7f9a710a7240>
```

```
ax.add_patch(patches.Arc((x,y), 2.3, 2.3, theta1=0.0, theta2=90.0, linestyle = '--'))
```

```
> <matplotlib.patches.Arc object at 0x7f9a710e1780>
```

```
ax.text(0.7,1, r'$\mathcal{\Delta\varphi = \frac{\pi}{2}}$', fontsize = 24)
```

```
> Text(0.7, 1, '$\\mathcal{\\Delta\\varphi = \\frac{\\pi}{2}}$')
```

```
ax.text(1.18*x1_vec,y1_vec, r'$\mathcal{\varphi}_{1}$', fontsize = 20)
```

```
> Text(1.18, -9.797174393178826e-16, '$\\mathcal{\\varphi}_{1}$')
```

```
ax.text(x2_vec,1.22*y2_vec, r'$\mathcal{\varphi}_{2}$', fontsize = 20)
```

```
> Text(-7.354070601250002e-16, 1.22, '$\\mathcal{\\varphi}_{2}$')
```

```
ax.set_xlim(-1.35,1.35)
```

```
> (-1.35, 1.35)
```

```
ax.set_ylim(-1.35,1.35)
```

```
> (-1.35, 1.35)
```

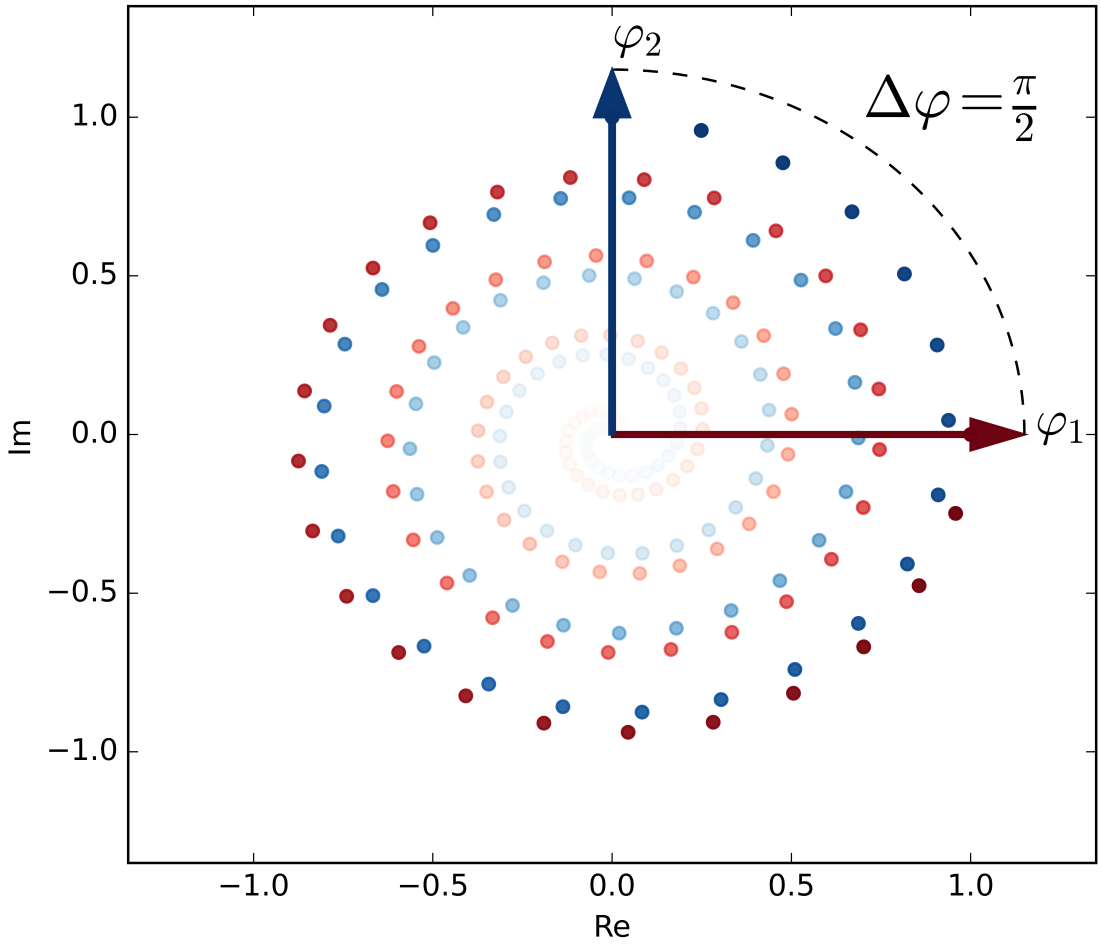
```
ax.set_xlabel('Re')
```

```
> Text(0.5, 0, 'Re')
```

```
ax.set_ylabel('Im')
```

```
> Text(0, 0.5, 'Im')
```

```
plt.show()
```




```
import sys
sys.path.insert(1, 'libs/')

import numpy as np
import scipy as sc
from scipy import io
import dynfc as dyn
```

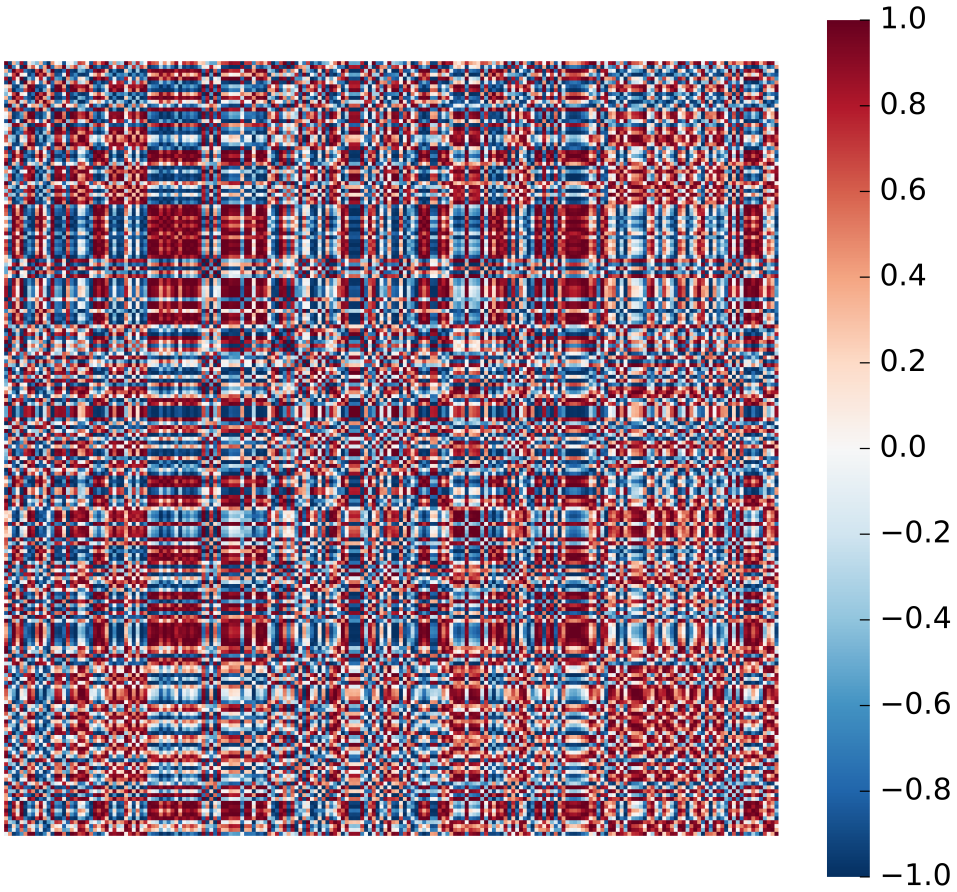
```
import seaborn as sns
import matplotlib.pyplot as plt

ts = sc.io.loadmat('data/ts.mat')['ts']
ts = ts

RSsig = np.zeros((ts.shape[0],ts.shape[1],1))
RSsig[:,:,0] = ts

Phases, syncConn, leidaArray = dyn.run_multiPat(RSsig)
```

```
> Signal filtered.
> Phases obtained.
> Matrices obtained.
> Routine finished for patient no. 1.
```

```
mat1 = syncConn[:,:,0,0]

aa = plt.figure(figsize = [7,6])
ax = sns.heatmap(mat1,
                 cmap = "RdBu_r",
                 vmin = -1,
                 vmax = 1,
                 square = True,
                 cbar = True)
ax.axis('off')
```

```
> (0.0, 200.0, 200.0, 0.0)
```

```
ax.tick_params(left=False, bottom=False)
plt.show()
```




```
aa = plt.figure(figsize = (12/50,4))
ax = sns.heatmap(leidaArray[:,0,]/max(abs(leidaArray[:,0,])),
                 cmap = "RdBu_r",
                 vmin = -1,
                 vmax = 1,
                 square = False,
                 cbar = False)
ax.axis('off')
```

```
> (0.0, 1.0, 1180.0, 0.0)
```

```
plt.show()
```

Package References

```
report::cite_packages(sessionInfo())
```

- JJ Allaire and Yihui Xie and Jonathan McPherson and Javier Luraschi and Kevin Ushey and Aron Atkins and Hadley Wickham and Joe Cheng and Winston Chang and Richard Iannone (2020). rmarkdown: Dynamic Documents for R. R package version 2.6. URL https://rmarkdown.rstudio.com.
- Kevin Ushey, JJ Allaire and Yuan Tang (2021). reticulate: Interface to 'Python'. R package version 1.18-9007. https://github.com/rstudio/reticulate
- R Core Team (2020). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.