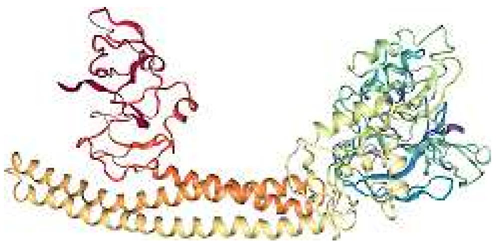
NSP2 Analysis



In [1]:
```
import os
```

In [2]:
```
import datetime

x = datetime.datetime.now()
print("Analysis time "+x.strftime("%c"))
```

```
Analysis time Mon Feb 15 14:25:52 2021
```

In [3]:
```
import socket
print("Analysis computer="+socket.gethostname()+" "+os.environ['HPC_SYSTEM'])
```

```
Analysis computer=r219n11 m100
```

In [4]:
```
# Python module imports
```

In [5]:
```
import numpy as np
```

In [6]:
```
import matplotlib.pyplot as plt
```

In [7]:
```
%matplotlib inline
```

In [8]:
```
from subprocess import Popen, PIPE, STDOUT
```

In [9]:
```
import os.path
from os import path
```

In [10]:
```
import pandas as pd
```

MD Analysis modules

In [11]:
```
import MDAnalysis as mda
```

In [12]:
```
from MDAnalysis.analysis import rms, align, pca
```

In [13]:
```
import nglview as nv
```

Library Routines

In [14]:
```
from EX4Cutils import *
```

In [15]:
```
datadir="/m100_scratch/userinternal/aemerson/Exscalate4Cov/nsp2/"    # end with a /
```

In [16]:
```
%cd /m100_scratch/userinternal/aemerson/Exscalate4Cov/nsp2
```

```
/m100_scratch/userinternal/aemerson/Exscalate4Cov/nsp2
```

In [17]:
```
# pre-processing if necessary
# gmx trjconv -s tpr -f xto.xtc -o xtc -center -pbc mol -ur compact
# gmx trjconv -s tpr -f xtc -o start.pdb -dump 0
```

In [18]:
```
xtc=datadir+'nsp2-10.xtc' # I have used a reduced xtc with trjconv -skip 10
tpr=datadir+'topol.tpr'
pdb=datadir+'nsp2.pdb'
```

In [19]:
```
index=datadir+'index.ndx'  # must exist
```

In [20]:
```
# check files
filelist=[xtc,tpr,pdb,index]  # add index.ndx if necessary
test_files(filelist)
```

protein structure

In [21]:
```
u = mda.Universe(pdb)
protein = u.select_atoms('protein')
domain_view = nv.show_mdanalysis(u)
domain_view.color_by('chainID')
domain_view
```

In [22]:
```
domain_view.render_image()
domain_view.download_image(filename='my_image.png', factor=4, trim=True)
```

Load and show trajectory

In [23]:
```
u = mda.Universe(pdb,xtc,in_memory=True )
```

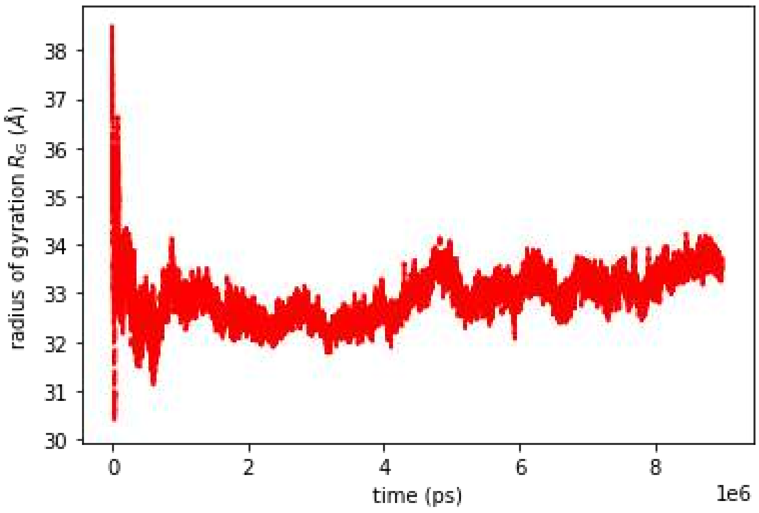
In [24]:
```
print("Trajectory length="+str(len(u.trajectory))+" frames.")
```

```
Trajectory length=18000 frames.
```

In [25]:
```
calpha=u.select_atoms("name CA")
protein = u.select_atoms('protein')

w = nv.show_mdanalysis(protein)
w
```

Radius of gyration

In [28]:
```
Rgyr = []
protein = u.select_atoms("protein")
for ts in u.trajectory:
   Rgyr.append((u.trajectory.time, protein.radius_of_gyration()))
Rgyr = np.array(Rgyr)

# plot
ax = plt.subplot(111)
ax.plot(Rgyr[:,0], Rgyr[:,1], 'r--', lw=2, label=r"$R_G$")
ax.set_xlabel("time (ps)")
ax.set_ylabel(r"radius of gyration $R_G$ ($\AA$)")
plt.draw()
```



RMSD of trajectory

```
In [28]: r=rms.RMSD(u,u,select="backbone",ref_frame=0)
```

```
In [29]: r.run()
```

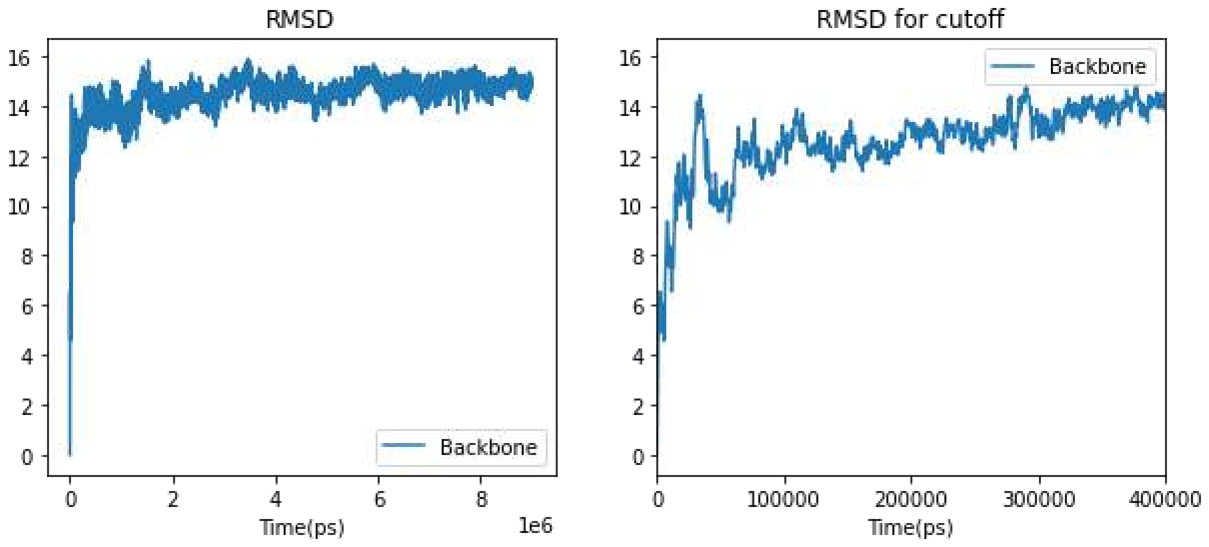
```
Out[29]: <MDAnalysis.analysis.rms.RMSD at 0x7fff64f93a30>
```

```
In [31]: df=pd.DataFrame(r.rmsd,columns=['Frame','Time(ps)','Backbone'])
```

```
In [32]: df=df[:-1] # remove last row (empty)
```

```
In [53]: # adjust the xlim in the plots so as to estimate the frames to discard for analysis
         fig, (ax1, ax2) = plt.subplots(1, 2,figsize=(10, 4))
         df.plot(ax=ax2,x='Time(ps)',y='Backbone',kind='line',title='RMSD for cutoff',xlim=([
                 xticks=([0,100000,200000,300000,400000]))
         df.plot(ax=ax1,x='Time(ps)',y='Backbone',kind='line',title='RMSD')
```

```
Out[53]: <AxesSubplot:title={'center':'RMSD'}, xlabel='Time(ps)'>
```



```
In [54]: # equilibration cutoffs from above
         start='400000' # 200ns
```

```
In [58]: backbone=u.select_atoms('backbone')
         calpha=u.select_atoms('name CA')
```

```
In [59]: print(len(backbone))
```

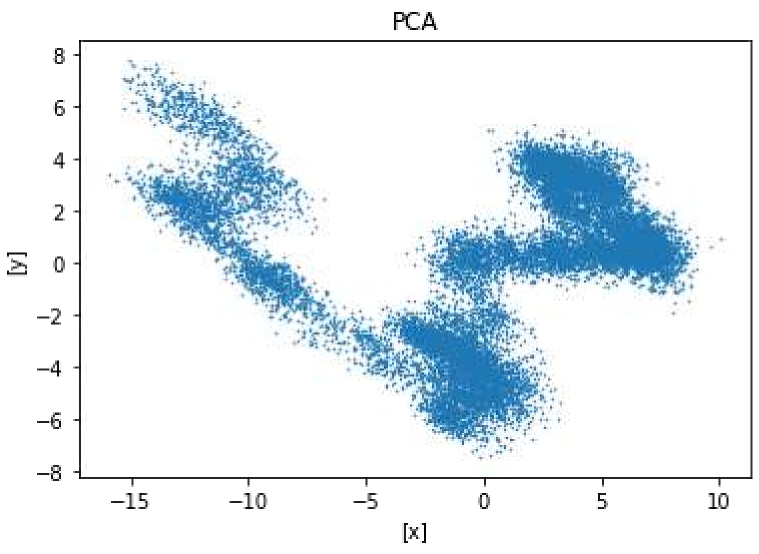
```
1159
```

GROMACS PCA and cluster Analysis

```
In [55]: ##  PCA  via Gromacs
         # 1. find covariance matrix -> eigenvec.trr, eigenval.xvg
         # 2. project onto eigenvec.trr -> Firstplane.xvg

         # May take some time
         options='3\n3'  # 3 =c-alpha, input here twice
         eigenvec='eigenvec.trr'
         gmx_covar(xtc,tpr,start,eigenvec,options)
         df_pca=gmx_anaeig(xtc,tpr,start,eigenvec,options)
```

```
In [57]: df_pca.plot(x=['x'],y=['y'],s=1,marker='.',kind='scatter',title='PCA')
```

```
Out[57]: <AxesSubplot:title={'center':'PCA'}, xlabel='[x]', ylabel='[y]'>
```



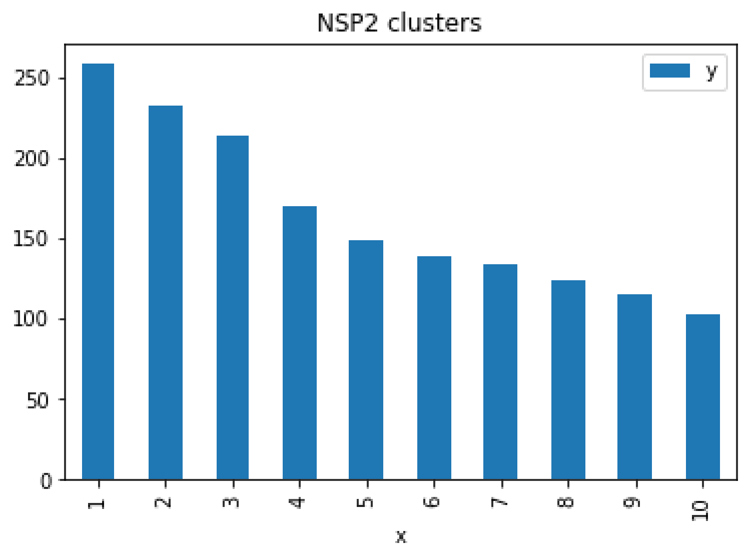
Cluster Analysis

```
In [70]: # cluster analysis
         # outputs:
         # - clindex   index of frames for each cluster
         # - cluster.log
         # This takes some time

         # GMX command line
         # "gmx cluster -n ../newdyn/prt+zn.ndx -cutoff 0.2 -b 400000.0 -f
         #traj.xtc  -s ../topol.tpr -method gromos -o -g -dist -ev -sz -cl
         #cluster.xtc -wcl 3 -clndx"


         options='3\n3'  # c-alpha, c-alpha
         skip='1'  # adjust as necessary
         clindex='clusters.ndx'
         df=gmx_cluster(xtc,tpr,index,start,skip,clindex,options)
         print("Clusters found="+str(len(df.index)))
```

```
Clusters found=136
```

```
In [71]: # plot
         df=df[:10]  # use only first 10
         df.plot(x='x',y='y',kind='bar',title='NSP2 clusters')
```

Out[71]:
```
<AxesSubplot:title={'center':'NSP2 clusters'}, xlabel='x'>
```



In [ ]:
```
#![title](rmsd-clust.png)
```

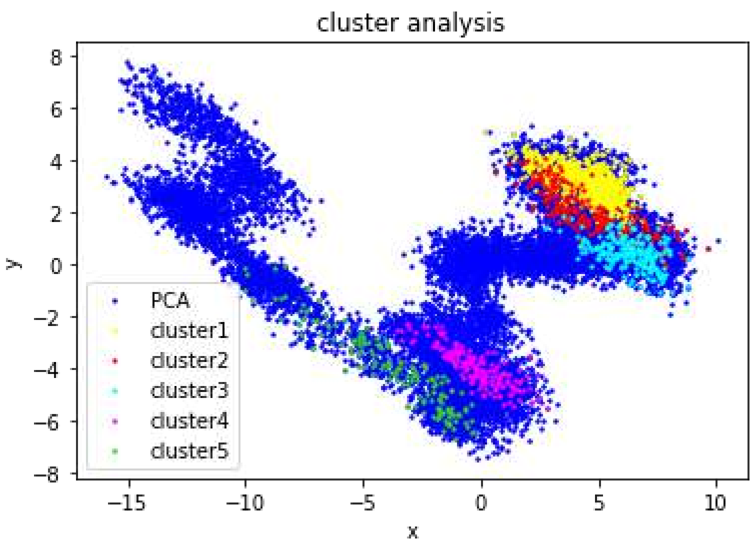
In [72]:
```
# use the index created by gmx cluster to extract frames from original trajectory

# "gmx extract-cluster -f traj.xtc -s topol.tpr -clusters clusters.ndx
# -n ../newdyn/prt+zn.ndx"
#cmd=['gmx','extract-cluster','-f',xtc_apo,'-s',tpr_apo ,'-n',index,'-clusters','clu

options='3\n3'
prefix='trajout'
clindex='clusters.ndx'
gmx_extract_cluster(xtc,tpr,index,clindex,prefix,options)
```

In [73]:
```
# We can now project these on the eigenvectors obtained from the original PCA analys
## loop over selected trajectories


###
prefix='trajout'
ncluster=5   ## adjust as necessary
xtc_list=[]
for i in range(1,ncluster+1):
    xtc_cluster=prefix+"_Cluster_{:0>4}".format(i)+".xtc"
    #print(xtc_cluster)
    df=gmx_anaeig(xtc_cluster,tpr,start,eigenvec,options,xvg='FirstPlane.xvg')
    xtc_list.append(df)
if (len(df.index)==0):
    print("Problem with gmx_anaeig")
```

In [66]:
```
%pwd
```

Out[66]:
```
'/m100_scratch/userinternal/aemerson/Exscalate4Cov/nsp2'
```

In [75]:
```
colors=['yellow','red','cyan','magenta','limegreen','black']

# PCA
plt.scatter(df_pca['x'],df_pca['y'],s=2,marker='o',color='blue',label='PCA')

# clusters
i=0
for df in xtc_list:
    i=i+1
    plt.scatter(df['x'],df['y'],s=2,marker='o', color=colors[i-1], label='cluster'+f
    plt.title("cluster analysis")
    plt.xlabel("x")
    plt.ylabel("y")
    plt.legend()
```



In [76]:
```
x = datetime.datetime.now()
print("Analysis Finish time "+x.strftime("%c"))
```

```
Analysis Finish time Wed Feb 10 12:19:28 2021
```

In [ ]: