==========

PROTEOMICS EXERCICE EXAMPLE Cellular time-response on mouse dentritic cells to a treatment. The script performs the following steps:

* Imports the cvs data.
* Computes the relative variation of expression of proteins (normalization)
* Quality control selection based on mean coverage
* Identifies main variations of protein expression
* Identifies which proteins/functional groups are affected
* Plots the time course variations

==========

In [1]:
```
# Import cvs data set
import os
from tkinter import *
from tkinter import filedialog
from tkinter.filedialog import askdirectory
import sys


Tk().withdraw()
path = askdirectory(title ="Chose folder where the proteomics data set is located" )
DATAfile = open (path + "/proteins_time_course.csv", "r")
Dataset = []
Features = []
k = 0
for line in DATAfile:
    data = line.split(",")
    if k == 0 :
        Features = data
    if k > 0:
        if [2][0] != "#":
            Dataset.append(data)
    k = k + 1
DATAfile.close()


print("                 DATA SET SUMMARY                    ")
print("=====================================================")
print( "total number of protein hits =   ", len(Dataset) )
print( "total number of colum features  =   ",  len(Features))

for i, col in enumerate(Features):
    print("col index ", i," name : ", col  )
print("=====================================================")
```

```
                DATA SET SUMMARY
=====================================================
total number of protein hits =    4149
total number of colum features  =    31
col index  0  name :  Protein Group
col index  1  name :  Protein ID
col index  2  name :  Accession
col index  3  name :  -10lgP
col index  4  name :  #Peptides
col index  5  name :  #Unique
col index  6  name :  PTM
col index  7  name :  Avg. Mass
col index  8  name :  Description
col index  9  name :  0 hr Sum Area
col index  10  name :  0.5 hr Sum Area
col index  11  name :  1 hr Sum Area
col index  12  name :  2 hr Sum Area
col index  13  name :  3 hr Sum Area
col index  14  name :  4 hr Sum Area
col index  15  name :  5 hr Sum Area
col index  16  name :  6 hr Sum Area
col index  17  name :  9 hr Sum Area
col index  18  name :  12 hr Sum Area
col index  19  name :  24 hr Sum Area
col index  20  name :  0 hr Mean Coverage
col index  21  name :  0.5 hr Mean Coverage
col index  22  name :  1 hr Mean Coverage
col index  23  name :  2 hr Mean Coverage
col index  24  name :  3 hr Mean Coverage
col index  25  name :  4 hr Mean Coverage
col index  26  name :  5 hr Mean Coverage
col index  27  name :  6 hr Mean Coverage
col index  28  name :  9 hr Mean Coverage
col index  29  name :  12 hr Mean Coverage
col index  30  name :  24 hr Mean Coverage

=====================================================
```

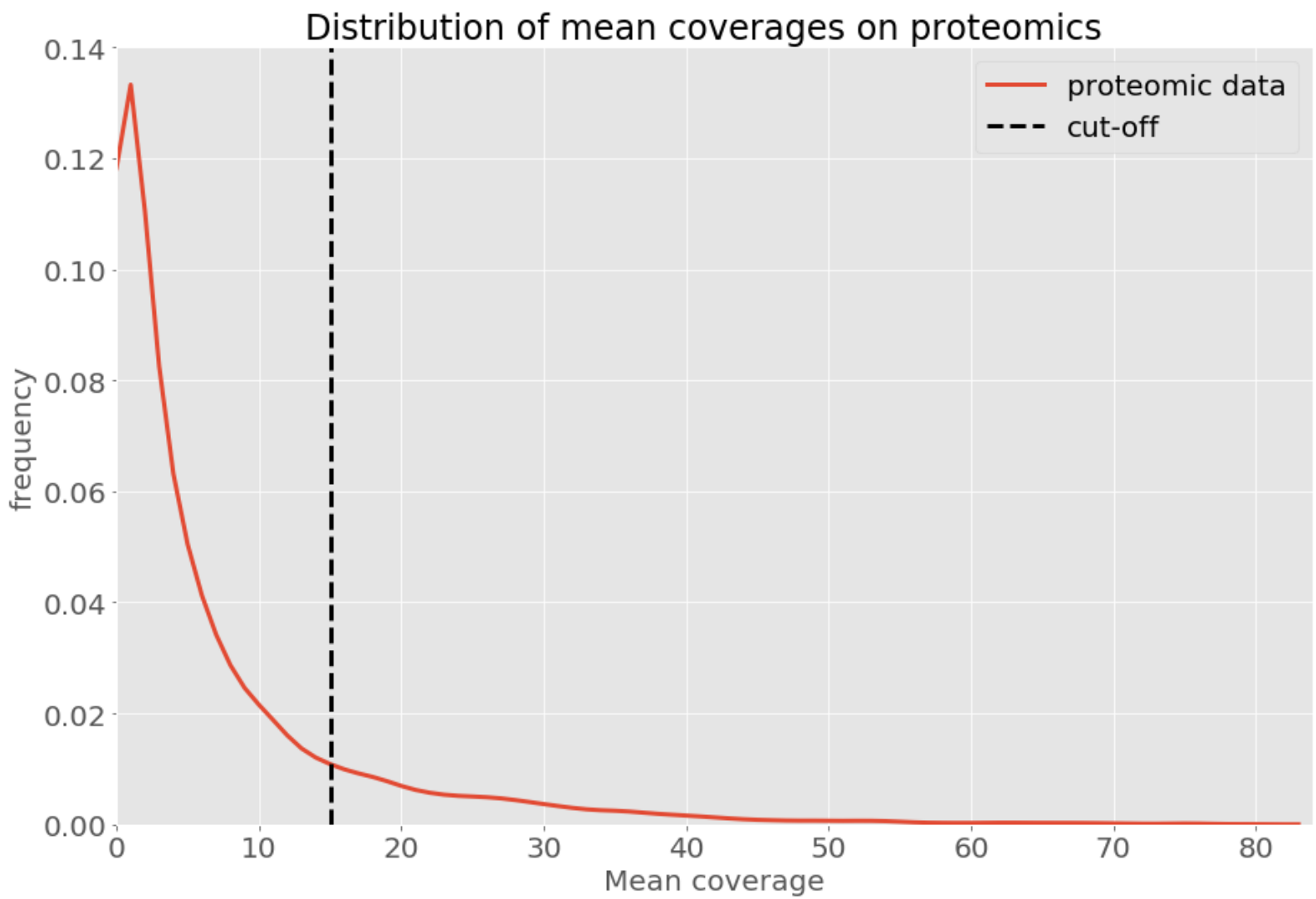
In [155]:
```
# Computing the main relative changes in expression for each protein
import numpy as np
import matplotlib.pyplot as plt
```

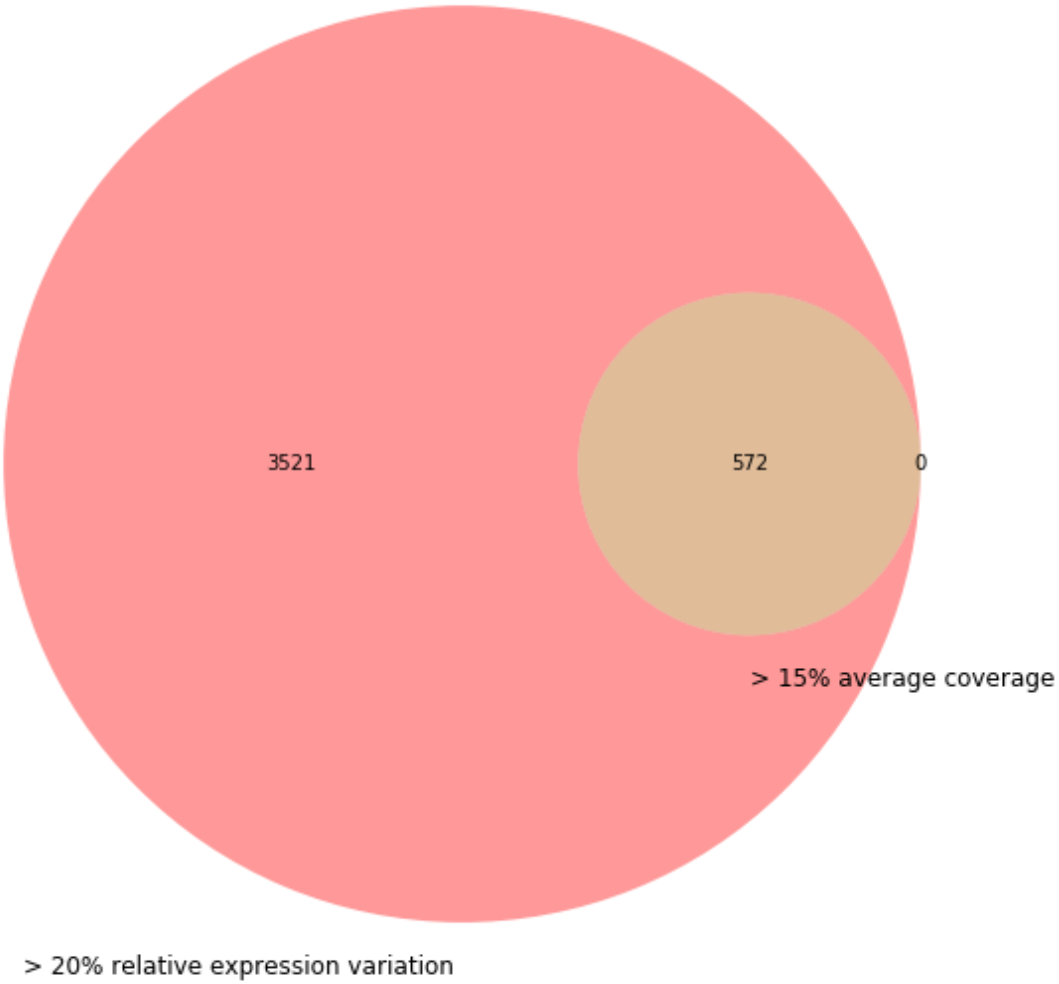
```
from scipy import stats
plt.style.use('ggplot')
plt.rcParams['figure.figsize'] = (15, 10)

#Cutoffs setting
minCov = 15   #  minimum coverage threshold
minSel = 20   # minimal % variation of expression for selectig proteins


PVAR= []  # list with max relatve expression of protein
DREG = []  # Lis for dowregulations
ProtGroups = []
ProtFunc = []
COVERAGEData = []
countHighVar = 0
countHighCov = 0
# parsing the data set and make calulation and removing low expression/bad ones
for data in Dataset:
    ID = data[1]
    Group = data[0]
    Function = data[6]
    Descrip = data[8]
    Areas = [float(A) for A in data[9:20] ]
    Coverages = [float(C) for C in data[20: len(data)] ]
    for cov in Coverages:
        COVERAGEData.append(cov)
    TotalAreas = sum(Areas)
    if TotalAreas >0:
        maxVARp = (max(Areas) - min(Areas))/ sum(Areas) * 100
        Dowreg = (Areas[0] - min(Areas))/ sum(Areas) * 100
    if TotalAreas == 0:
        maxVARp = 0
        Dowreg = 0
```

```
    AvCoverage = sum(Coverages)/11
    if AvCoverage >= minCov:
        countHighCov = countHighCov + 1
    if maxVARp >= minSel:
        countHighVar = countHighVar +1
    if AvCoverage >= minCov and maxVARp >= minSel:
        PVAR.append([maxVARp, ID, Group, Descrip, Function, AvCoverage ])
        if Function not in ProtFunc:
            ProtFunc.append(Function)
        if Group not in ProtGroups:
            ProtGroups.append(Group)
    if AvCoverage > minCov and Dowreg < 0:
        DREG.append([Dowreg, ID, Group, Descrip, Function, AvCoverage ])


# Ploting the distribution of average coverage
density = stats.kde.gaussian_kde(COVERAGEData )
x = np.arange(min(COVERAGEData), max(COVERAGEData), 1)
plt.plot(x, density(x), label = "proteomic data",linewidth = 3 )
Y= [k/1000 for k in range (0,1000) ]
plt.plot([minCov for x in Y],Y ,"k--" , linewidth = 3, label = "cut-off" )
plt.xlabel('Mean coverage' , size = 20)
plt.title("Distribution of mean coverages on proteomics  ", size =24)
plt.ylabel('frequency' , size = 20)
plt.tick_params(axis='both', labelsize = 20) # set the font of axis values
plt.legend( fontsize = 20)
plt.axis([min(COVERAGEData), max(COVERAGEData), 0, 0.14]  )
plt.show()


# Computing Venn diagram with selected data hits

from matplotlib_venn import venn2
```

```
A = countHighVar - len(PVAR)
B = countHighCov - len(PVAR)
L1 = "> " +str(minSel) + "% relative expression variation "
L2 = "> " +str(minCov) + "% average coverage"
C = len(PVAR)
venn2(subsets = (A, B , C), set_labels = (L1, L2))
plt.show()
print ("\n=====================================================================" )
print("% proteome with resonable coverage                    ",
      int( countHighCov/len(Dataset)*100), " %" )
print("% proteome with high expression                       ",
      int(countHighVar/len(Dataset)*100), " %" )
print("% proteome with expression reduction  =               ",
      int(len(DREG)/len(Dataset)*100), " %  " )
print("% proteome with high expression + resonable coverage  ",
      int(len(PVAR)/len(Dataset)*100), " % " )

print ("=====================================================================" )



#Ordering lists for best candidates
PVAR = sorted(PVAR, reverse=True )
DREG = sorted(DREG, reverse=False )

# computed percentils Expression increase
P99 =  np.percentile(np.array([x[0] for x in PVAR ]), 99)
P95 =  np.percentile(np.array([x[0] for x in PVAR ]), 95)
P90 =  np.percentile(np.array([x[0] for x in PVAR ]), 90)
P75 =  np.percentile(np.array([x[0] for x in PVAR ]), 75)
P50 =  np.percentile(np.array([x[0] for x in PVAR ]), 50)
P25 =  np.percentile(np.array([x[0] for x in PVAR ]), 25)
```

```
"""
P99d =  np.percentile(np.array([x[0] for x in DREG ]), 99)
P95d =  np.percentile(np.array([x[0] for x in DREG ]), 95)
P90d =  np.percentile(np.array([x[0] for x in DREG ]), 90)
P75d =  np.percentile(np.array([x[0] for x in DREG ]), 75)
P50d =  np.percentile(np.array([x[0] for x in DREG ]), 50)
P25d =  np.percentile(np.array([x[0] for x in DREG ]), 25)
"""
print("\n\nPercentils for selecting the threshold of highly response proteins: ")
print("\t99 percentil = ", int(P99), "%")
print("\t95 percentil = ", int(P95), "%")
print("\t90 percentil = ", int(P90), "%")
print("\t75 percentil = ", int(P75), "%")
print("\t50 percentil = ", int(P50), "%")
print("\t25 percentil = ", int(P25), "%")
#print("\t99 percentil = ", int(P99d), "%")
#print("\t95 percentil = ", int(P95d), "%")
#print("\t90 percentil = ", int(P90d), "%")
#print("\t75 percentil = ", int(P75d), "%")
#print("\t50 percentil = ", int(P50d), "%")
#print("\t25 percentil = ", int(P25d), "%")

print("\n        TOP Protein > 55% variation   (percentil 95%)                ")
print ("% VAR   Prot ID       GroupID           Desription     ")

PS_IDS = []
for P in PVAR:
    if P[0] > 55:
        PS_IDS.append(P[1])
        print (int(P[0]),"    \t" , P[1],"   \t" ,  P[2], "    " , P[3][0:70] )
print ("=================================================================" )
print ("Total proteins selected = ", len(PS_IDS))
```


Distribution of mean coverages on proteomics
proteomic data
cut-off
frequency
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
0
10
20
30
40
50
60
70
80
Mean coverage

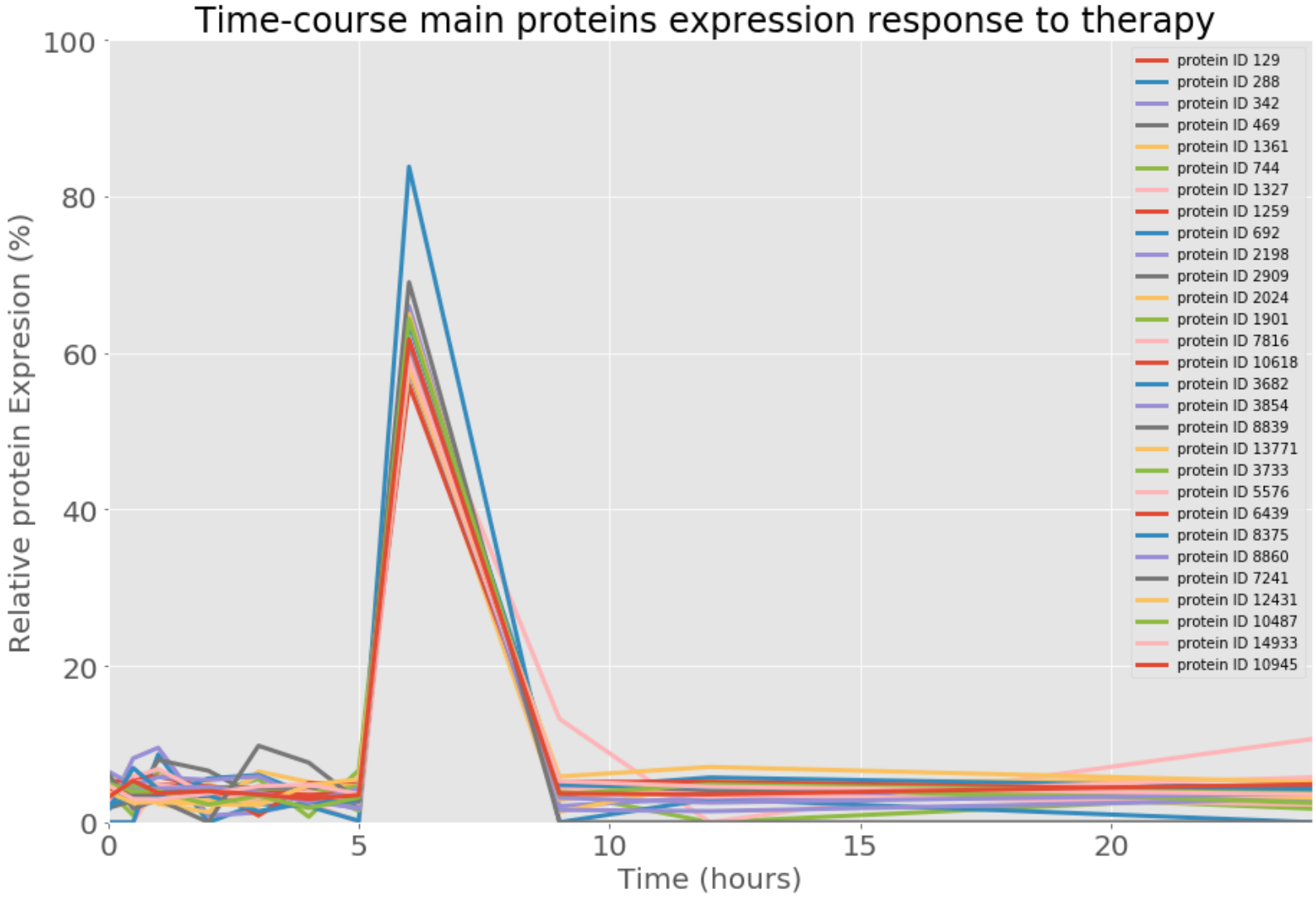

3521
572
0
> 15% average coverage
> 20% relative expression variation

```
======================================================================
% proteome with resonable coverage                    13  %
% proteome with high expression                       98  %
% proteome with expression reduction  =               0  %
% proteome with high expression + resonable coverage   13  %
======================================================================


Percentils for selecting the threshold of highly response proteins:
        99 percentil =  62 %
        95 percentil =  54 %
        90 percentil =  52 %
        75 percentil =  50 %
        50 percentil =  47 %
        25 percentil =  45 %

        TOP Protein > 55% variation   (percentil 95%)
% VAR   Prot ID       GroupID           Desription
83       692             870     Spliceosome RNA helicase Ddx39b OS=Mus musculus GN=Ddx39b PE=1 SV=1
69       7241            744     Cofilin-2 OS=Mus musculus GN=Cfl2 PE=1 SV=1
65       2198            1552     Serine/threonine-protein phosphatase 2A catalytic subunit beta isoform
64       744             969     Calcium/calmodulin-dependent protein kinase type II subunit delta OS=M
62       10487           3333     Protein BRICK1 OS=Mus musculus GN=Brk1 PE=1 SV=1
62       13771           3013     Brain acid soluble protein 1 OS=Mus musculus GN=Basp1 PE=1 SV=3
62       8375            2647     Small nuclear ribonucleoprotein Sm D2 OS=Mus musculus GN=Snrpd2 PE=1 S
60       1327            539     F-actin-capping protein subunit beta OS=Mus musculus GN=Capzb PE=1 SV=
60       1901            1645     Calcyclin-binding protein OS=Mus musculus GN=Cacybp PE=1 SV=1
59       3854            804     Serine/arginine-rich splicing factor 1 OS=Mus musculus GN=Srsf1 PE=1 S
59       8860            2314     Elongin-C OS=Mus musculus GN=Eloc PE=1 SV=1
58       2909            842     Ras-related protein Rab-1B OS=Mus musculus GN=Rab1b PE=1 SV=1
58       288             366     Protein disulfide-isomerase A4 OS=Mus musculus GN=Pdia4 PE=1 SV=3
58       10945           3563     NADH dehydrogenase [ubiquinone] 1 alpha subcomplex subunit 2 OS=Mus mu
58       7816            1356     Interleukin-1 receptor antagonist protein OS=Mus musculus GN=Il1rn PE=
58       1361            797     Polypyrimidine tract-binding protein 1 OS=Mus musculus GN=Ptbp1 PE=1 S
57       129             148     Heat shock protein HSP 90-beta OS=Mus musculus GN=Hsp90ab1 PE=1 SV=3
57       469             773     Aspartate aminotransferase  mitochondrial OS=Mus musculus GN=Got2 PE=1
57       1259            768     60S ribosomal protein L3 OS=Mus musculus GN=Rpl3 PE=1 SV=3
56       3733            1648     Proteasome subunit beta type-2 OS=Mus musculus GN=Psmb2 PE=1 SV=1
56       2024            1338     ADP/ATP translocase 1 OS=Mus musculus GN=Slc25a4 PE=1 SV=4
56       14933           4289     Ubiquitin-fold modifier 1 OS=Mus musculus GN=Ufm1 PE=1 SV=1
56       12431           2573     C-C motif chemokine 5 OS=Mus musculus GN=Ccl5 PE=2 SV=2
55       342             272     Glucose-6-phosphate 1-dehydrogenase X OS=Mus musculus GN=G6pdx PE=1 SV
55       3682            1125     Pyruvate dehydrogenase E1 component subunit beta  mitochondrial OS=Mus
```

```
55       5576            1733     Glia maturation factor beta OS=Mus musculus GN=Gmfb PE=1 SV=3
55       6439            1782     60S ribosomal protein L18a OS=Mus musculus GN=Rpl18a PE=1 SV=1
55       8839            1819     Superoxide dismutase [Cu-Zn] OS=Mus musculus GN=Sod1 PE=1 SV=2
55       10618           2699     Eukaryotic translation initiation factor 1 OS=Mus musculus GN=Eif1 PE=
======================================================================
Total proteins selected =  29
```

In [141]:

```
#
# MAKING A PLOT with main proteins  THAT COMPOSE CANCER DIAGNOSTIC MODELS with tags



# get time times series of selected proteins
Time = [0, 0.5, 1, 2, 3, 4, 5, 6, 9, 12, 24]
for data in Dataset:
    ID = data[1]
    if ID in PS_IDS:
        Areas = [float(A) for A in data[9:20] ]
        TotalAreas = sum(Areas)
        RelativeAreas =  [A/TotalAreas*100 for A in Areas ]
        plt.plot(Time, RelativeAreas, linewidth = 3, label = "protein ID " + ID  )
plt.xlabel('Time (hours)' , size = 20)
plt.title("Time-course main proteins expression response to therapy ", size =24)
plt.ylabel('Relative protein Expresion (%)' , size = 20)
plt.tick_params(axis='both', labelsize = 20) # set the font of axis values
plt.legend( fontsize = 10)
plt.axis([0, 24, 0, 100,]  )
plt.show()
```



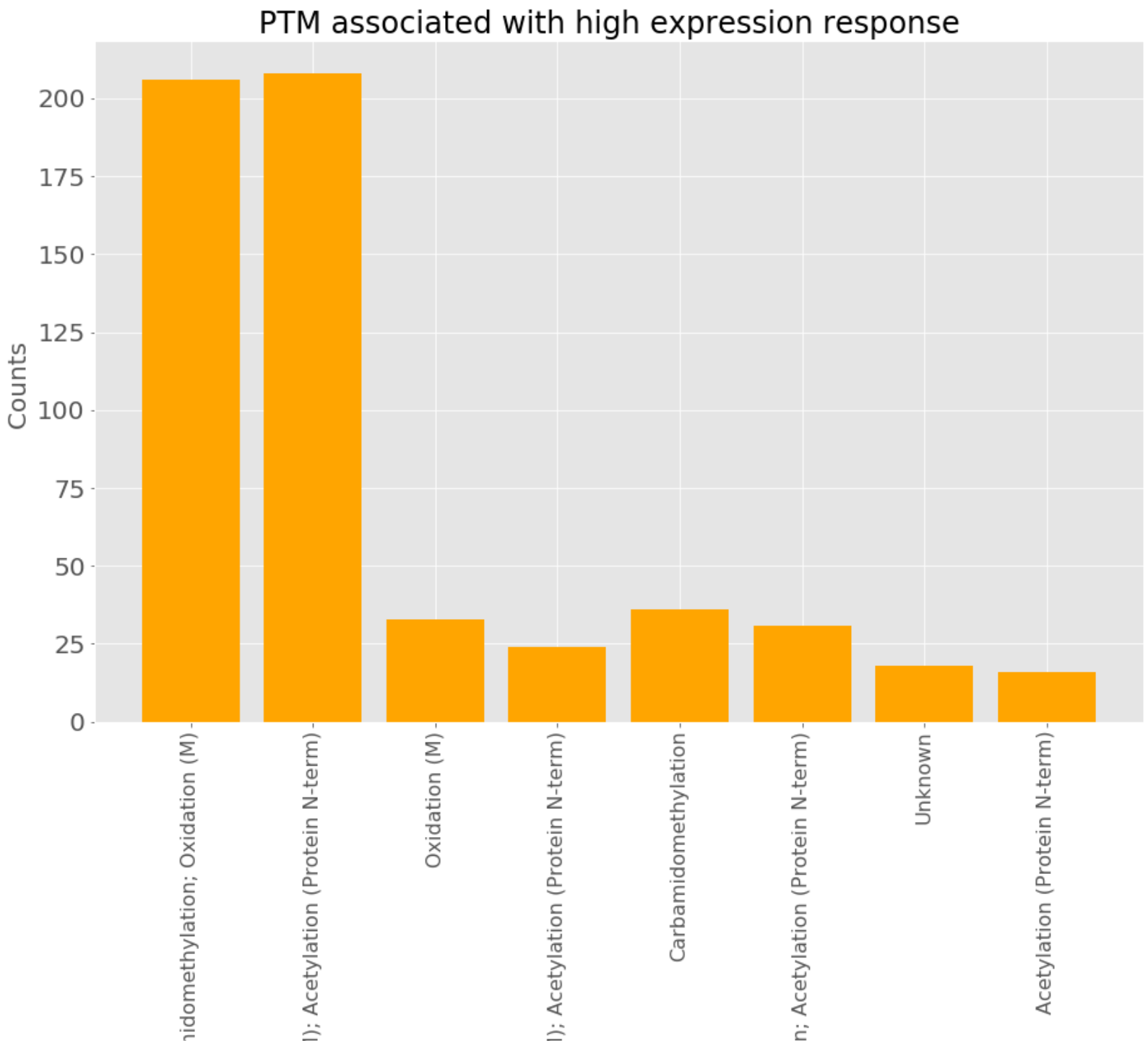
In [104]:
```
len(ProtGroups)
```

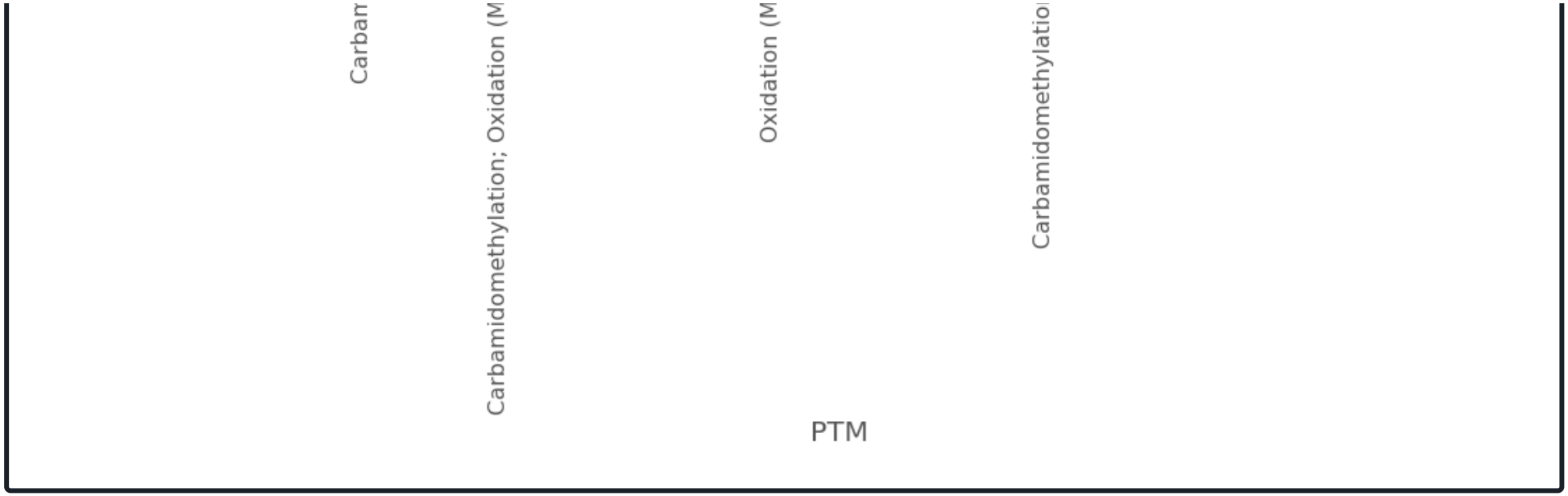
```
374
```

In [151]:
```
# Analyzing groups and functions frequencies in changed expression proteins
```

```
print ("PROTEIN GROUPS DETECTED WITH COUNTS > 1")
print("================================================")
GrupCounts = []
GroupFreq = [ F[2] for F in PVAR]
for G in  ProtGroups:
    c = GroupFreq.count(G)
    GrupCounts.append(c)
    if c > 1:
        print ("protein group ", G, " Count  = ", c )
print("================================================")



FunctCounts = []
FunctFreq = [ F[4] for F in PVAR]
for F in  ProtFunc:
    c = FunctFreq.count(F)
    FunctCounts.append(c)
PlotNames = []
for name in ProtFunc:
    if name == "":
        name = "Unknown"
    PlotNames.append(name)

plt.bar( np.arange(len(ProtFunc)), FunctCounts , color = "orange")
plt.xlabel('PTM' , size = 20)
plt.title("PTM associated with high expression response  ", size =24)
plt.ylabel('Counts ' , size = 20)
plt.tick_params(axis='both', labelsize = 20) # set the font of axis values
plt.xticks(np.arange(len(ProtFunc)), PlotNames , fontsize=16, rotation = 90)

plt.show()
```

```
PROTEIN GROUPS DETECTED WITH COUNTS > 1
================================================
protein group  401  Count  =  4
protein group  754  Count  =  2
protein group  2709  Count  =  2
================================================
```


PTM associated with high expression response
Counts
0
25
50
75
100
125
150
175
200
idomethylation; Oxidation (M)
l); Acetylation (Protein N-term)
Oxidation (M)
l); Acetylation (Protein N-term)
Carbamidomethylation
n; Acetylation (Protein N-term)
Unknown
Acetylation (Protein N-term)


Carban
Carbamidomethylation; Oxidation (M
Oxidation (M
Carbamidomethylatio
PTM