In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [ ]:
```
data = pd.read_csv(filepath_or_buffer='./giRNA_clustering_ALL_SAMPLES_v2.txt'
data
# Add UMI + giRNAseq column to control for PCR bias amplification
data['Bias'] = data['UMI'] + '_' + data['giRNAseq']
data
```

Out[ ]:

	Sample_id	Frequency	UMI	giRNAseq	g
0	7773.1	1	AAATAAATAACTT	GCTGGGCCTGCCTGAAAAGT	NUBPL_+_3203
1	7773.1	1	AAATAAATAGCTG	GTGGGAACAGTTGCAGTAGG	PLN_+_11886
2	7773.1	1	AAATAAATCAGTC	GTAGTCTGCTTGGAGAGGAG	RA _160189 ENST000002
3	7773.1	1	AAATAAATCAGTT	GGATGCACTGGGCGGGATCA	SNRPD3_+_2495
4	7773.1	1	AAATAAATCCCTA	GCTTCCTAGGAGCCTTCCTG	DOC2A_-_3002
...	...	...	...	...	
398601	7773.4	2	TTTTTATTAACTC	GAGAGCCTCACAGCTCCGGA	SGCB_+_5290
398602	7773.4	2	TTTTTCATGGATT	GGAAGCAGCGTCAGGACCAT	SARM1_+_2669
398603	7773.4	2	TTTTTGATGAGTG	GTGTACTACCAGATGTTAAA	RGS3_+_11622 ENST000003
398604	7773.4	2	TTTTTTCTAATTG	GAGGGCCCCGGGACTTGCAG	DNAH7_-_19693
398605	7773.4	3	CCGTAGTTCGATT	GGGACAGGCAGCAGTACTTG	SNAP29_-_2121

398606 rows × 8 columns

In [ ]:
```
# Print out some stats about the dataset
total_number_identified_girnas = len(data['giRNAseq'].unique())
total_number_represented_genes = len(data.loc[(data['Gene_name']!='NO') & (da
total_number_annotated_girnas = len(data.loc[(data['Gene_name']!='NO') & (dat
print(f"Total number of identified giRNAs = {total_number_identified_girnas}"
print(f"Total number of identified annotated giRNAs = {total_number_annotated
print(f"Total number of represented genes = {total_number_represented_genes}"
```

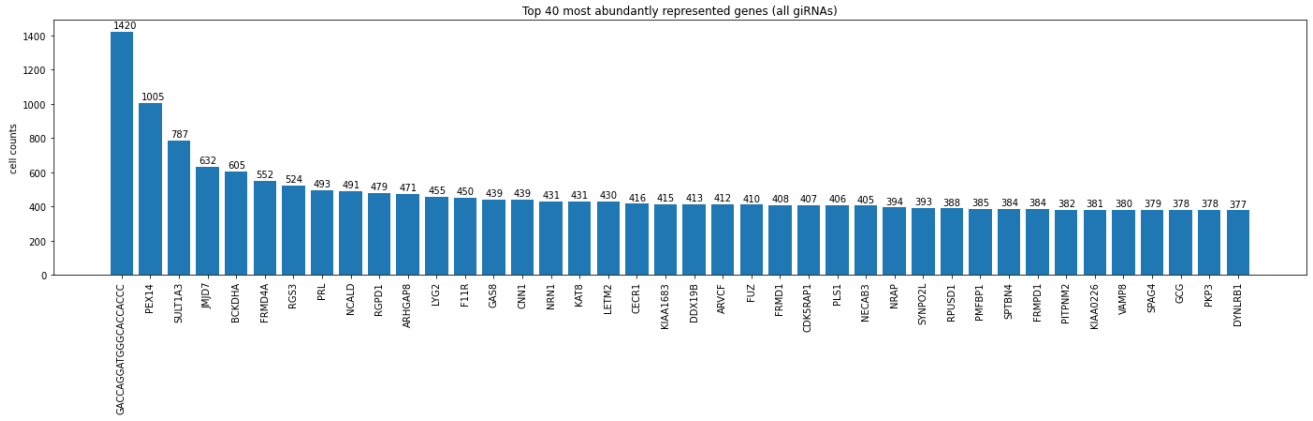
```
Total number of identified giRNAs = 18238
Total number of identified annotated giRNAs = 12390
Total number of represented genes = 2426
```

In [ ]:
```
# Remove rows with PCR bias (same combination of UMI+giRNAseq)
data.drop_duplicates(subset='Bias', keep='first', inplace=True)
```

In [ ]:
```
# Replace gene names with giRNAseqs for Gene_names = 'NO' or 'non-argeting'

my_index1 = list(data.loc[data['Gene_name']=='NO'].index)
my_girnaseq = list(data.loc[data['Gene_name']=='NO']['giRNAseq'])
data.at[my_index1,'Gene_name'] = my_girnaseq

my_index2 = list(data.loc[data['Gene_name']=='non-targeting'].index)
my_girnaseq = list(data.loc[data['Gene_name']=='non-targeting']['giRNAseq'])
data.at[my_index2,'Gene_name'] = my_girnaseq
```

In [ ]:

```
# How many giRNAs were assigned to each gene?
#ATATTCCTCTCTG_GACGAGGCGCTAGGGAACAA
girna_per_gene = data.groupby(['Gene_name']).count().sort_values(by='Bias', a
girna_per_gene.head(n=20)

# Save results
girna_per_gene.to_csv(path_or_buf='giRNAs_per_gene.csv')

# function to add value labels
def addlabels(x,y):
    for i in range(len(x)):
        plt.text(i-0.3,y[i]+20,y[i])


xvals = list(girna_per_gene.head(40).index)
yvals = list(girna_per_gene.head(40).values)

fig, ax = plt.subplots(figsize=(24, 5),facecolor='w')
plt.bar(x=xvals,
        height=yvals,
        log=False)
plt.xticks(rotation=90)

# calling the function to add value labels
addlabels(xvals, yvals)
plt.ylabel('cell counts')
plt.title('Top 40 most abundantly represented genes (all giRNAs)')
plt.savefig('giRNAs_per_gene.pdf')
plt.show()
```

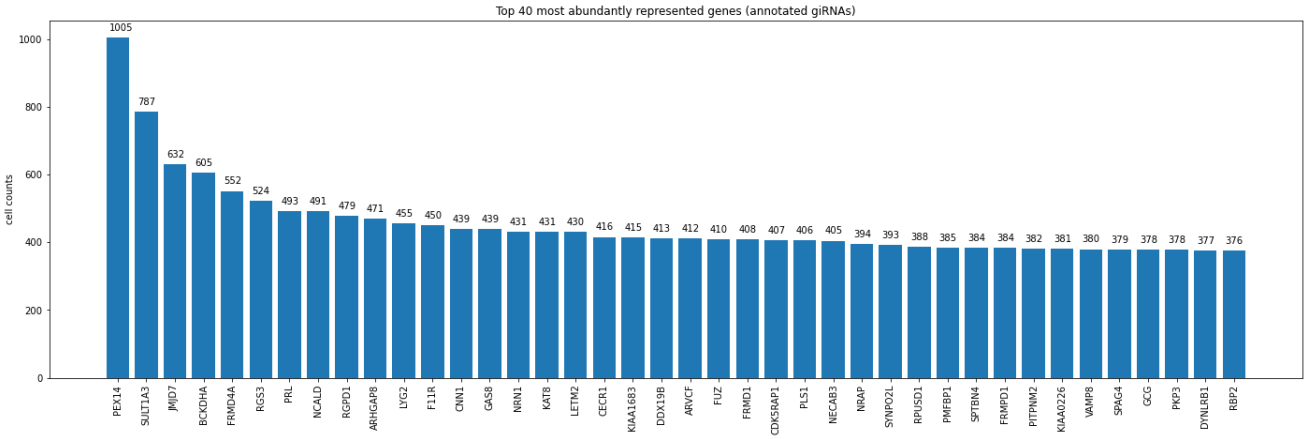


In [ ]:

```
# Drop NO_giRNA_HIT and non-targeting
data.drop(my_index1 + my_index2, inplace=True)
```

In [ ]:

```
# How many giRNAs were assigned to each gene?
#ATATTCCTCTCTG_GACGAGGCGCTAGGGAACAA
girna_per_gene = data.groupby(['Gene_name']).count().sort_values(by='Bias', a
girna_per_gene.head(n=20)

# Save results
girna_per_gene.to_csv(path_or_buf='giRNAs_per_gene.csv')

xvals = list(girna_per_gene.head(40).index)
yvals = list(girna_per_gene.head(40).values)

fig, ax = plt.subplots(figsize=(24, 7),facecolor='w')
plt.bar(x=xvals,
        height=yvals,
        log=False)
plt.xticks(rotation=90)

# calling the function to add value labels
addlabels(xvals, yvals)
plt.ylabel('cell counts')
plt.title('Top 40 most abundantly represented genes (annotated giRNAs)')
plt.savefig('annotated_giRNAs_per_gene.pdf')

plt.show()
```



In [ ]:

```
counts_per_gene_per_girna = pd.DataFrame(data.groupby(['Gene_name','giRNAseq'
counts_per_gene_per_girna.columns = ['perc_giRNA_per_gene']
counts_per_gene_per_girna.sort_values(by='perc_giRNA_per_gene', ascending=Fal
counts_per_gene_per_girna.columns
girna_perc_per_gene = counts_per_gene_per_girna['perc_giRNA_per_gene']/counts
girna_perc_per_gene.to_csv('percentage_giRNAs_per_gene.csv')

biased_giRNAs = girna_perc_per_gene[girna_perc_per_gene>=0.75].sort_values(as
biased_giRNAs.to_csv('percentage_giRNAs_per_gene_0.75cutoff.csv')
biased_giRNAs
```

Out[ ]:

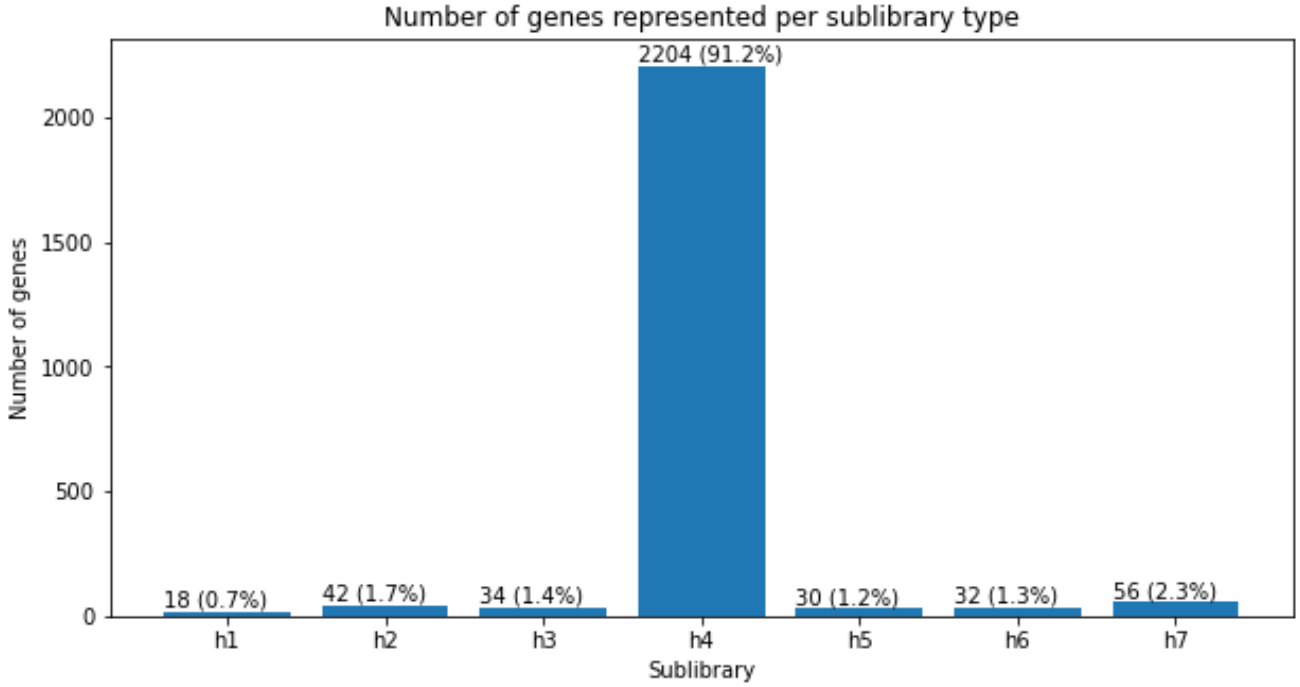
```
Gene_name       giRNAseq
ARSK            GTTCTCTTCTACAAACGCCG    1.000000
FAM204A         GCGCTGCGACGCCCCTTTCG    1.000000
DOLK            GCGCCAGCGGGCCGTGTGTG    1.000000
WDR83OS         GCGAACAAAGTGCTGAGGTG    1.000000
B9D2            GCCGTTAAGTGCGCTAAGGT    1.000000
                                          ...
C15orf38-AP3S2  GGCCACGGTTCTCTCAGCAC    0.770115
ACTL7A          GTTCAGGCCTTGAATCCAGT    0.766082
TRMT44          GGTTGAACTGTGGAATGTGT    0.750000
PFDN2           GTCGCAGGGTCCAATCCGGA    0.750000
RBM18           GTCCCCTCGAGGCCCTGTCA    0.750000
Name: perc_giRNA_per_gene, Length: 172, dtype: float64
```

In [ ]:

```
# Number of different giRNAs per gene (see excel spreadsheet for list of cand
girna_perc_per_gene_group = girna_perc_per_gene.groupby(level=0)
girna_number_per_gene_group = girna_perc_per_gene_group.count().sort_values(a
girna_number_per_gene_group.name = 'giRNA_number_per_gene'
girna_number_per_gene_group.to_csv('giRNA_number_per_gene.csv')
```

In [ ]:

```
# What proportion are h4 genes compared to the total?
gene_names_nr = data.drop_duplicates(subset=['Gene_name'])
count_genes_per_sublibrary = gene_names_nr.groupby(by=['Sublibrary']).count()
count_genes_per_sublibrary.name = 'Number_of_genes'
count_genes_per_sublibrary.to_csv('Number_of_genes_per_sublibrary.csv')


# function to add value labels
def addlabels(x,y, rot=0):
    my_total = sum(y)
    my_perc = y/my_total
    my_perc = [round(z, ndigits= 1) for z in list((y / my_total) * 100)]
    for i in range(len(x)):
        plt.text(i-0.4,y[i]+20, f"{y[i]} ({my_perc[i]}%)", rotation=rot)


# Generate plot
xvals = list(count_genes_per_sublibrary.index)
yvals = list(count_genes_per_sublibrary.values)

fig, ax = plt.subplots(figsize=(10, 5),facecolor='w')
plt.bar(x=xvals,
        height=yvals,
        log=False)
#plt.xticks(rotation=90)

# calling the function to add value labels
addlabels(xvals, yvals)

plt.title('Number of genes represented per sublibrary type')
plt.xlabel('Sublibrary')
plt.ylabel('Number of genes')
plt.savefig('gene_per_sublibrary.pdf')
plt.show()
```

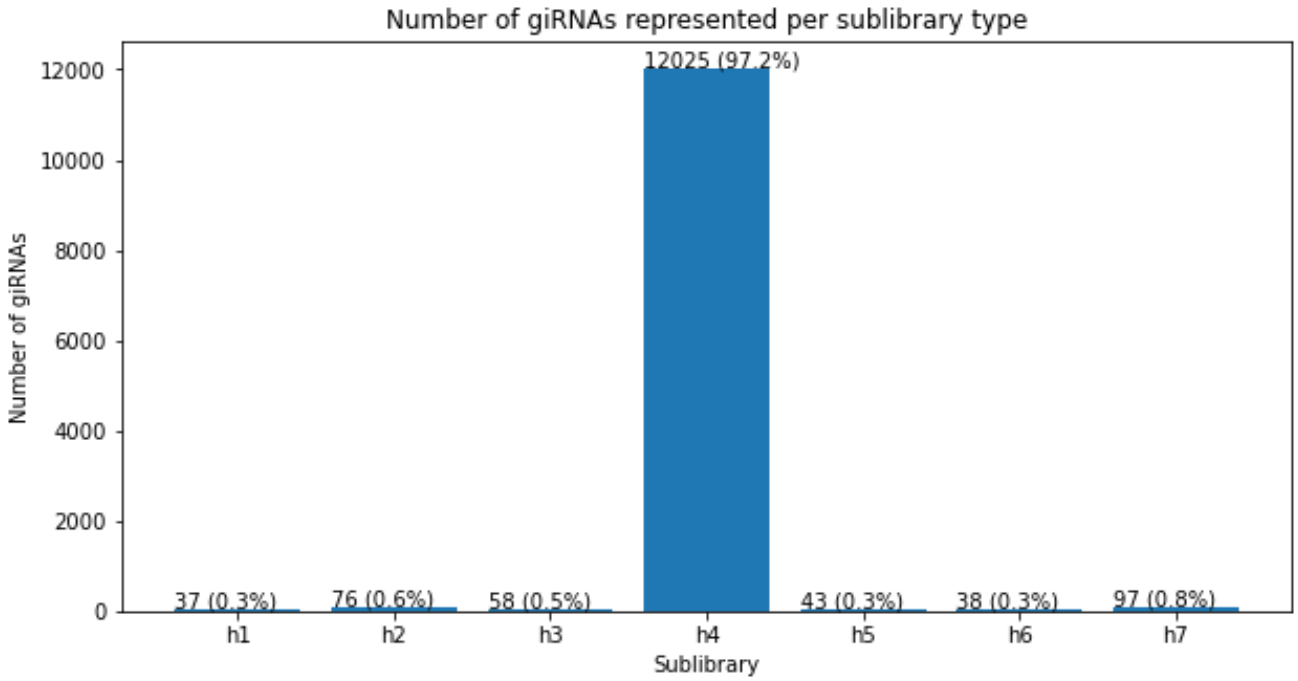


In [ ]:
```
len(gene_names_nr['Gene_name'].unique())
```

Out[ ]:
```
2426
```

In [ ]:

```
# What proportion are h4 giRNAs compared to the total?
girnas_nr = data.drop_duplicates(subset=['giRNAseq'])
count_girnas_per_sublibrary = girnas_nr.groupby(by=['Sublibrary']).count()['B
count_girnas_per_sublibrary.name = 'Number_of_giRNAs'
count_girnas_per_sublibrary.to_csv('Number_of_giRNAs_per_sublibrary_type.csv'

# Generate plot
xvals = list(count_girnas_per_sublibrary.index)
yvals = list(count_girnas_per_sublibrary.values)

fig, ax = plt.subplots(figsize=(10, 5),facecolor='w')
plt.bar(x=xvals,
        height=yvals,
        log=False)

#plt.xticks(rotation=90)

# calling the function to add value labels
addlabels(xvals, yvals)

plt.title('Number of giRNAs represented per sublibrary type')
plt.xlabel('Sublibrary')
plt.ylabel('Number of giRNAs')
plt.savefig('Number_of_giRNAs_per_sublibrary_type.pdf')
plt.show()
```

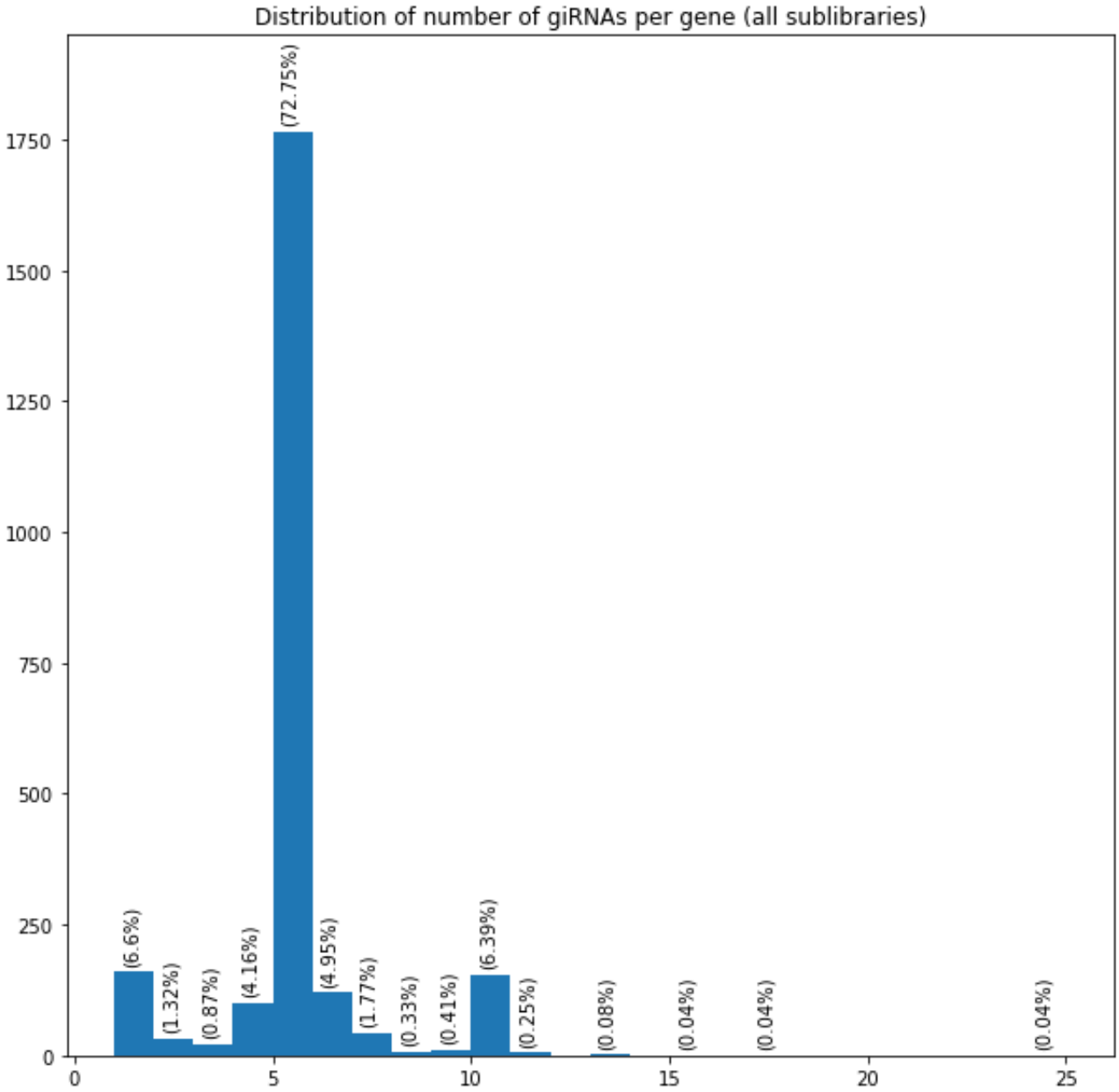


In [ ]:

```
len(girnas_nr['giRNAseq'])
yvals
count_girnas_per_sublibrary
```

Out[ ]:
```
Sublibrary
h1       37
h2       76
h3       58
h4    12025
h5       43
h6       38
h7       97
Name: Number_of_giRNAs, dtype: int64
```

In [ ]:
```
# Distribution of giRNAs per gene
girna_number_per_gene_group

# function to add value labels
def addperc(x,y):
    my_total = sum(x)
    my_perc = x/my_total
    my_perc = [round(z, ndigits= 2) for z in list((x / my_total) * 100)]
    for i in range(int(max(y)-1)):
        if x[i] > 0:
            plt.text(i+1.2,x[i]+ 20, f"({my_perc[i]}%)", rotation='vertical')

# Plot histogram
fig, ax = plt.subplots(figsize=(10, 10),facecolor='w')
h = plt.hist(x=girna_number_per_gene_group, bins=max(girna_number_per_gene_gr
plt.ylim(top=1950)
addperc(list(h[0]),list(h[1]))

plt.title(label='Distribution of number of giRNAs per gene (all sublibraries)
plt.savefig('Distribution_number_giRNAs_per_gene.pdf')
plt.show()
```

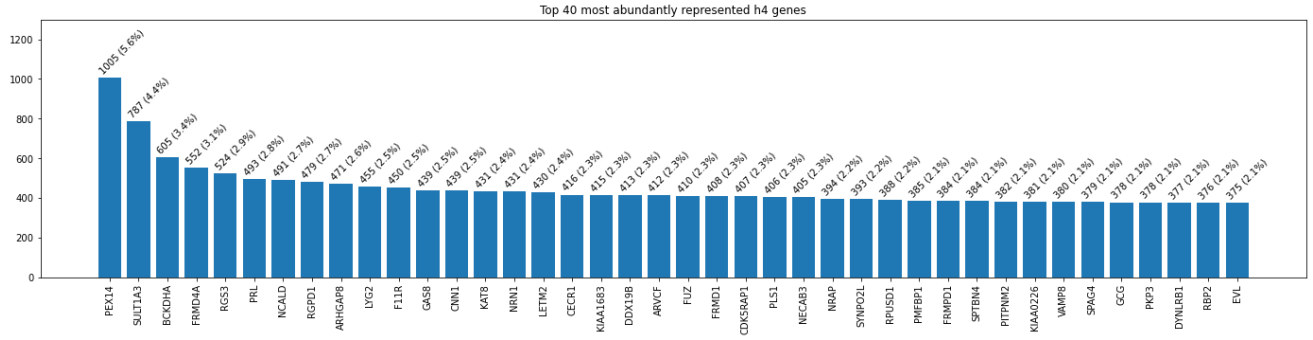


In [ ]:

```
# Keep only h4 giRNAs
data_h4 = data.loc[data['Sublibrary'] == 'h4']
```

In [ ]:

```
# How many giRNAs were assigned to each h4 gene?
girna_per_gene_h4 = data_h4.groupby(['Gene_name']).count().sort_values(by='Bi
girna_per_gene_h4.head(n=20)

# Save results
girna_per_gene_h4.to_csv(path_or_buf='giRNAs_per_gene.csv')

# generate plot
xvals = list(girna_per_gene_h4.head(40).index)
yvals = list(girna_per_gene_h4.head(40).values)

fig, ax = plt.subplots(figsize=(24, 5),facecolor='w')
plt.bar(x=xvals,
        height=yvals,
        log=False)
plt.xticks(rotation=90)

# calling the function to add value labels
addlabels(xvals, yvals, rot=45)
plt.ylim(top=1300)
plt.title('Top 40 most abundantly represented h4 genes')

# save plot
plt.savefig('Top_40_most_abundantly_represented_h4_genes.pdf')

# display plot
plt.show()
```

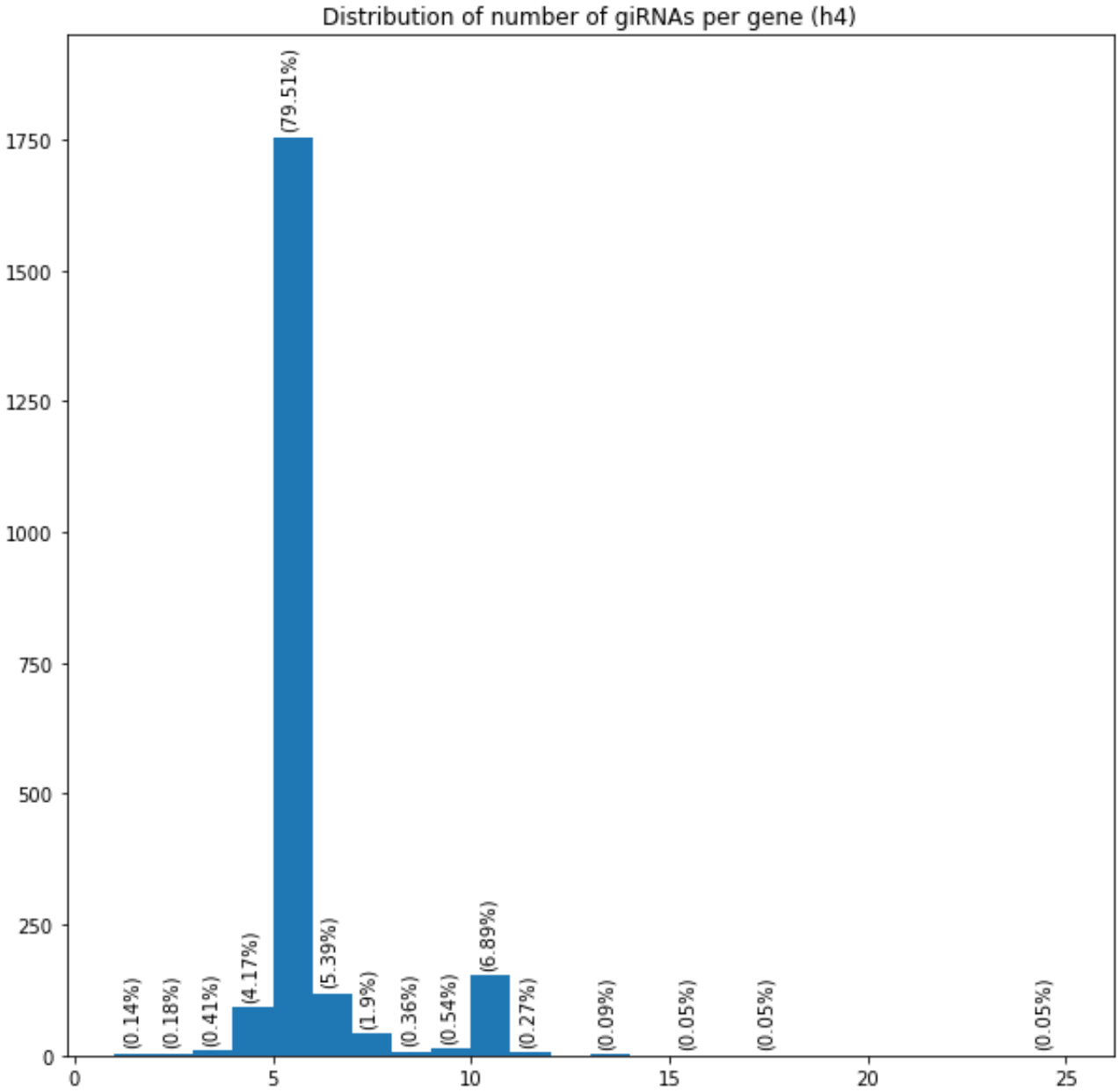


In [ ]:

```
girna_number_per_gene_group_h4 = data_h4.groupby(by=['Gene_name'])['giRNAseq'
girna_number_per_gene_group_h4
```

Out[ ]:
```
Gene_name
1-Mar      [GGATCCGGCTCGGGGCAAGG, GTGGCTTCTCCGCGAGGTGG, G...
14-Sep     [GCTACTATGAGGACTAACAG, GACAACTGGGGACAGTAGAA, G...
2-Mar      [GAGGCTTGGTCACCGCATTA, GGAGCGGGAATGCCTTAATG, G...
8-Sep      [GTGGGCTGGGACGAGCGCAG, GGGCAGGTGCGAAGATAGAG, G...
AAAS       [GCCTCGCCGTTTGTCCCTTG, GACGGCGAGGCGGAACTCAA, G...
                                    ...
ZNF563     [GAGGCTACACAGACGTTCCA, GGCGGGTCCCACTGTGACAG, G...
ZNF607     [GCCGCAGCTCCAGCACCCTA, GTGCTGGAGCTGCGGAGGAG, G...
ZNF692     [GAAGAAGAAACGGTGCCTCT, GAACGCTGCGCGCGCGAGGT, G...
ZNF696     [GCCCGCACGTGTCGGACCCC, GCGCGGCCGAGAGAACGGGG, G...
ZSCAN20    [GGTGAAGTGGGTGTCTCGGT, GTGTCTCGGTGGGTGAGTCC, G...
Name: giRNAseq, Length: 2206, dtype: object
```

In [ ]:
```
my_girna_per_gene_h4 = [len(z) for z in girna_number_per_gene_group_h4]
my_girna_per_gene_h4
my_girna_counts_per_gene_h4 = pd.DataFrame({'genes':girna_number_per_gene_gro
my_girna_counts_per_gene_h4['counts']

#my_girna_counts_per_gene_h4
# Plot histogram
fig, ax = plt.subplots(figsize=(10, 10),facecolor='w')
h = plt.hist(x=my_girna_counts_per_gene_h4['counts'], bins=max(my_girna_count
#h
plt.ylim(top=1950)
addperc(list(h[0]),list(h[1]))

plt.title(label='Distribution of number of giRNAs per gene (h4)')
plt.savefig('Distribution_number_giRNAs_per_h4_gene.pdf')
plt.show()
```



In [ ]:

```
# Load annotation data
annot = pd.read_csv(filepath_or_buffer='./giRNA_library.csv', )

# Keep only h4 giRNAs/genes
annot_h4 = annot.loc[annot['Sublibrary'] == 'h4']
annot_h4
```

Out[ ]:

	gene	transcript	protospacer sequence	selection rank	Sublibrary	
165	ARHGAP8	P1	GCGCGCGGCCAGCACAGACC	3.0	h4	table3
166	ARHGAP8	P1	GCGGCTCCAGGGCCTCCGGG	4.0	h4	table3
167	ARHGAP8	P1	GTAGCCCGCGGACGGCTCAG	5.0	h4	table3
168	ARHGAP8	P1	GCGCCGGGTTAATCATTGCA	6.0	h4	table3
169	ARHGAP8	P1	GACAGACCCGGCGCAAACGG	7.0	h4	table3
...	...	...	...	...	...	
208240	negative_control	na	GTCCACCATCGGAGACAACT	NaN	h4	table3_l
208241	negative_control	na	GGCGTCCCAGGCGAACCAAA	NaN	h4	table3_l
208242	negative_control	na	GCAGGGCAATGCGCCACCAG	NaN	h4	table3_l
208243	negative_control	na	GACCTCTTGACGGCCGGGCT	NaN	h4	table3_l
208244	negative_control	na	GAGGGTAACGCAGAAGAAGG	NaN	h4	table3_l

24600 rows × 7 columns

In [ ]:

```
# Proportion of TableS3_hCRISPERiv2 h4 giRNAs represented in the samples
data_h4_unique_girnas = pd.Series(data_h4['giRNAseq'].unique())
my_h4_girna_list = list(annot_h4['protospacer sequence'].unique())
data_h4_in_my_girna_list = data_h4_unique_girnas[data_h4_unique_girnas.isin(m
print('total h4 library giRNAs =',len(my_h4_girna_list))
print('total data_h4 giRNAs =',len(data_h4_unique_girnas))
my_percentage = round(len(data_h4_unique_girnas) * 100 / len(my_h4_girna_list
print('Percentage of represented h4 giRNAs from TableS3_hCRISPERiv2 file  =',
```

```
total h4 library giRNAs = 24488
total data_h4 giRNAs = 12025
Percentage of represented h4 giRNAs from TableS3_hCRISPERiv2 file  = 49.11 %
```

In [ ]:

```
# Proportion of TableS3_hCRISPERiv2 h4 giRNAs represented in the samples
data_h4_unique_genes = pd.Series(data_h4['Gene_name'].unique())
my_h4_gene_list = list(annot_h4['gene'].unique())
data_h4_in_my_girna_list = data_h4_unique_genes[data_h4_unique_genes.isin(my_
print('total h4 library genes =',len(my_h4_gene_list))
print('total data_h4 genes =',len(data_h4_unique_genes))
my_percentage = round(len(data_h4_unique_genes) * 100 / len(my_h4_gene_list),
print('Percentage of represented h4 genes from TableS3_hCRISPERiv2 file  =',m
```

```
total h4 library genes = 2220
total data_h4 genes = 2206
Percentage of represented h4 genes from TableS3_hCRISPERiv2 file  = 99.37 %
```