Tutorial 1 - Computational Biology

February 7, 2019

1 Tutorial 1

1.1 Week 1

1.1.1 1) Import the cell cycle dataset excel spreadsheet (using Pandas). You may need to do some tidying of the data such as dropping rows with missing NaN values.

In [2]:
```
# Importing necessary libraries and Cell-Cycle-Set.xlsx dataset
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
df=pd.read_excel('Cell-Cycle-Set.xlsx')
df.head()
```

Out[2]:

	Gene_Name	mean_RNA_G1	mean_RNA_S	mean_RNA_G2	mean_protein_G1	\
0	ATG2A	9.006204	9.022302	9.129524	18.214767	
1	RBM47	10.330107	10.396423	10.677257	24.748020	
2	ADAM9	12.321340	12.203630	12.233293	19.083593	
3	UBA6	10.827333	10.758463	10.685847	24.614467	
4	ESYT2	11.336907	11.184317	11.284027	18.593760	

	mean_protein_S	mean_protein_G2	\
0	14.898980	20.236780	
1	22.426777	24.651200	
2	16.248873	19.281277	
3	21.356450	25.207883	
4	19.151843	19.015813	

	GOBP	\
0	NaN	
1	base conversion or substitution editing;biolog...	
2	activation of MAPKK activity;activation of pro...	
3	catabolic process;cellular catabolic process;c...	
4	NaN	

	GOMF	\
0	binding;protein binding	
1	binding;nucleic acid binding;nucleotide bindin...	

```
2  binding;catalytic activity;cation binding;coll...
3  adenyl nucleotide binding;adenyl ribonucleotid...
4                                                NaN

                                                  GOCC
0                                                 NaN
1  apolipoprotein B mRNA editing enzyme complex;c...
2  cell part;extracellular region part;extracellu...
3            cell part;cytoplasm;intracellular part
4  cell part;integral to membrane;intrinsic to me...
```

In [3]:
```
# Exploring dataset characteristics
df.shape
```

Out[3]:
```
(499, 10)
```

In [4]:
```
df.dtypes
```

Out[4]:
```
Gene_Name           object
mean_RNA_G1        float64
mean_RNA_S         float64
mean_RNA_G2        float64
mean_protein_G1    float64
mean_protein_S     float64
mean_protein_G2    float64
GOBP                object
GOMF                object
GOCC                object
dtype: object
```

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 499 entries, 0 to 498
Data columns (total 10 columns):
Gene_Name          499 non-null object
mean_RNA_G1        499 non-null float64
mean_RNA_S         499 non-null float64
mean_RNA_G2        499 non-null float64
mean_protein_G1    495 non-null float64
mean_protein_S     494 non-null float64
mean_protein_G2    493 non-null float64
GOBP               438 non-null object
GOMF               466 non-null object
GOCC               466 non-null object
dtypes: float64(6), object(4)
memory usage: 39.1+ KB
```

In [6]:
```
#Making each column of the dataset have the same number of rows (deleting not a number

# Alternatively clean = df.dropna()
df.dropna(inplace=True)
df.head()
```

Out[6]:

	Gene_Name	mean_RNA_G1	mean_RNA_S	mean_RNA_G2	mean_protein_G1	\
1	RBM47	10.330107	10.396423	10.677257	24.748020	
2	ADAM9	12.321340	12.203630	12.233293	19.083593	
3	UBA6	10.827333	10.758463	10.685847	24.614467	
5	SHTN1	10.845517	10.824347	10.634980	26.112690	
6	SIL1	9.042438	8.924093	9.035878	22.750520	

	mean_protein_S	mean_protein_G2	\
1	22.426777	24.651200	
2	16.248873	19.281277	
3	21.356450	25.207883	
5	22.905927	26.138843	
6	20.598227	23.093443	

	GOBP	\
1	base conversion or substitution editing;biolog...	
2	activation of MAPKK activity;activation of pro...	
3	catabolic process;cellular catabolic process;c...	
5	axon guidance;chemotaxis;locomotion;response t...	
6	cellular macromolecule metabolic process;cellu...	

	GOMF	\
1	binding;nucleic acid binding;nucleotide bindin...	
2	binding;catalytic activity;cation binding;coll...	
3	adenyl nucleotide binding;adenyl ribonucleotid...	
5	binding;enzyme binding;kinase binding;protein ...	
6	binding;protein binding;unfolded protein binding	

	GOCC
1	apolipoprotein B mRNA editing enzyme complex;c...
2	cell part;extracellular region part;extracellu...
3	cell part;cytoplasm;intracellular part
5	axon;cell part;cell projection;neuron projection
6	cell part;cytoplasmic part;endoplasmic reticul...

In [7]:
```
# Checking dataset properties after having cleaned the original version

df.shape
```

Out[7]: (397, 10)

In [8]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 397 entries, 1 to 498
Data columns (total 10 columns):
Gene_Name          397 non-null object
mean_RNA_G1        397 non-null float64
mean_RNA_S         397 non-null float64
mean_RNA_G2        397 non-null float64
mean_protein_G1    397 non-null float64
mean_protein_S     397 non-null float64
mean_protein_G2    397 non-null float64
GOBP               397 non-null object
GOMF               397 non-null object
GOCC               397 non-null object
dtypes: float64(6), object(4)
memory usage: 34.1+ KB
```

```
In [9]: # Checking for duplicates

        df.duplicated()
```

```
Out[9]: 1      False
        2      False
        3      False
        5      False
        6      False
        9      False
        10     False
        11     False
        14     False
        15     False
        18     False
        19     False
        21     False
        22     False
        26     False
        27     False
        29     False
        30     False
        32     False
        33     False
        34     False
        35     False
        36     False
        37     False
        38     False
        39     False
        40     False
```

```
42     False
43     False
44     False
       ...
464    False
465    False
466    False
467    False
468    False
469    False
470    False
472    False
473    False
475    False
477    False
478    False
479    False
480    False
481    False
482    False
483    False
484    False
486    False
487    False
488    False
489    False
490    False
491    False
492    False
494    False
495    False
496    False
497    False
498    False
Length: 397, dtype: bool
```

1.1.2 2) Perform exploratory analysis of the data, thus:

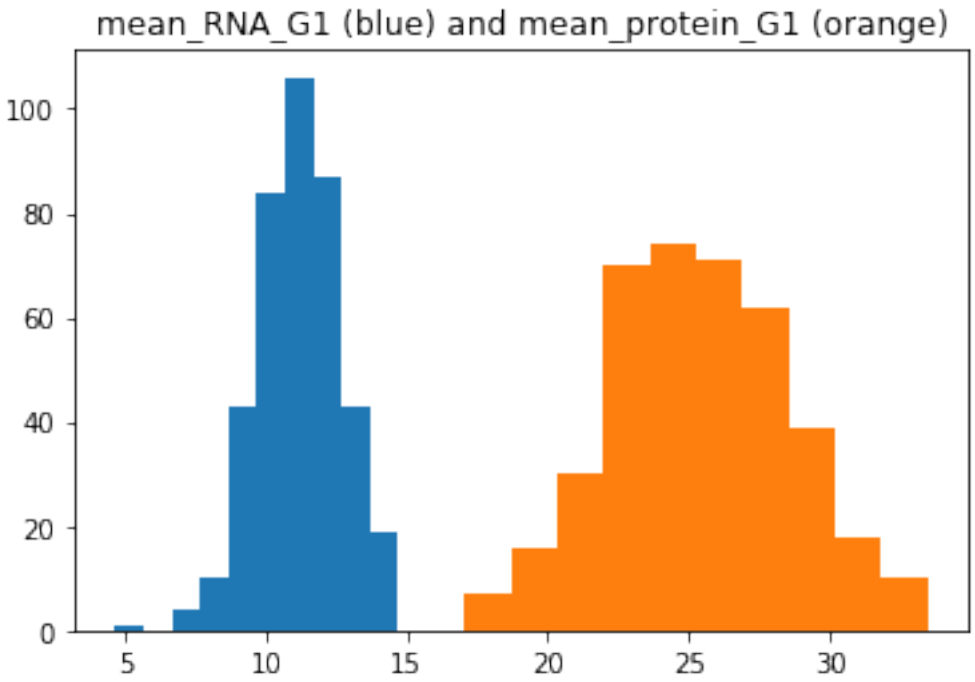
1) Generate a histogram of one of the cell cycle stages of the RNA and protein distribution. Do you notice anything interesting with regards to the mean/variance of the distribution?

Answer =

In all the RNA distributions cases the mean was equal to around 11 and the variance to around 2. Instead in the Protein cases the mean was equal to 25 (apart from the case of the S-stage m=22) and the variance around 10. Both the RNA and protein cases resemble a gaussian distribution. In all cases the Protein mean and variance is greater than in the RNA counterpart. (One RNA can produce multiple protein genes)

In [10]:
```
# mean_RNA_G1 (blue) and mean_protein_G1 (orange) Histogram representation
x = df["mean_RNA_G1"]
plt.hist(x)
mean1 = np.mean(df["mean_RNA_G1"])
print("mean_RNA_G1 =", mean1)
variance1 = np.var(df["mean_RNA_G1"])
print("variance_RNA_G1 =", variance1)
x = df["mean_protein_G1"]
plt.hist(x)
plt.title('mean_RNA_G1 (blue) and mean_protein_G1 (orange)');
mean2 = np.mean(df["mean_protein_G1"])
print("mean_protein_G1= ", mean2)
variance2 = np.var(df["mean_protein_G1"])
print("variance_protein_G1=", variance2)
```

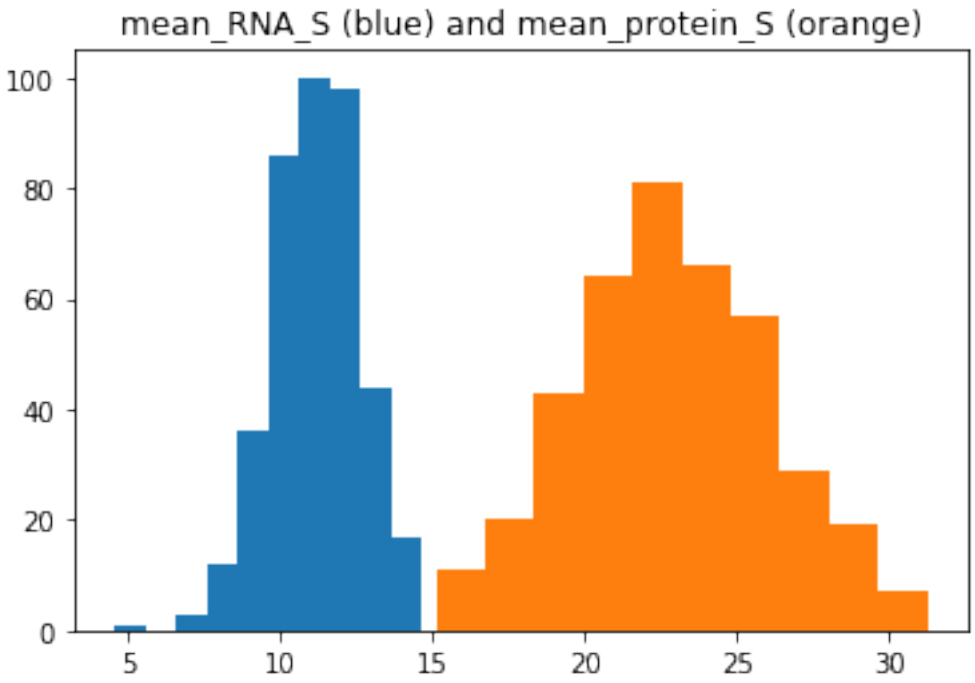
```
mean_RNA_G1 = 11.215627083963062
variance_RNA_G1 = 2.155063534349558
mean_protein_G1=  25.35167163727959
variance_protein_G1= 10.427242156064343
```



In [11]:
```
# mean_RNA_S (blue) and mean_protein_S (orange) histogram representation
x2 = df["mean_RNA_S"]
plt.hist(x2)
mean3 = np.mean(df["mean_RNA_S"])
```

```
print("mean_RNA_S =", mean3)
variance3 = np.var(df["mean_RNA_S"])
print("variance_RNA_S =", variance3)
x2 = df["mean_protein_S"]
plt.hist(x2)
plt.title('mean_RNA_S (blue) and mean_protein_S (orange)');
mean4 = np.mean(df["mean_protein_S"])
print("mean_protein_S = ", mean4)
variance4 = np.var(df["mean_protein_S"])
print("variance_protein_S =", variance4)
```

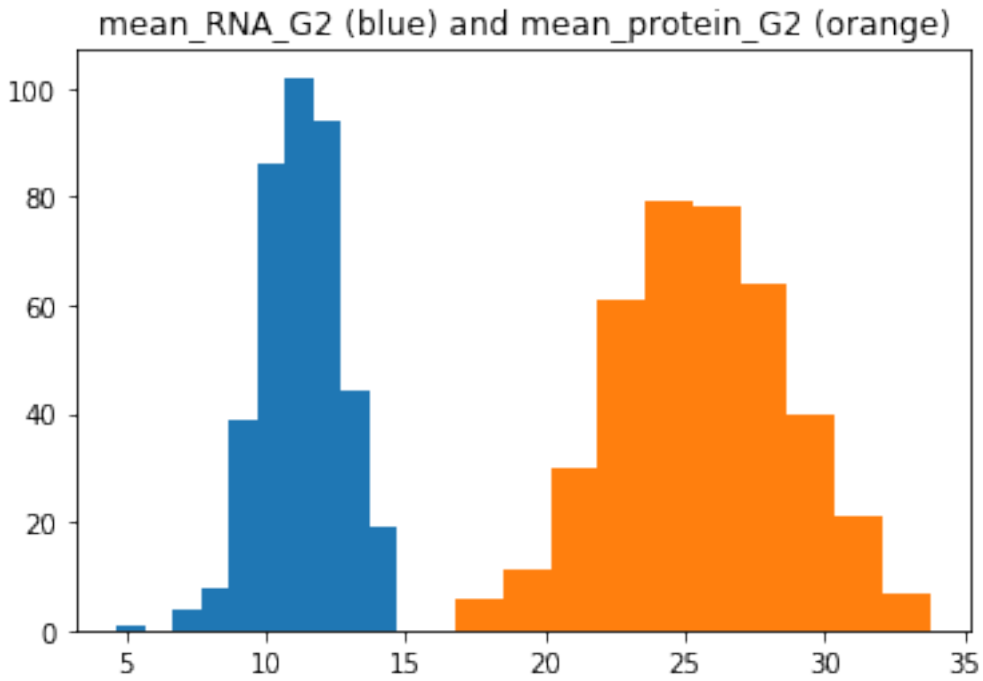
```
mean_RNA_S = 11.186962324937035
variance_RNA_S = 2.1401874564067667
mean_protein_S =  22.84765777917715
variance_protein_S = 10.376814362619672
```



In [12]:
```
# mean_RNA_G2 (blue) and mean_protein_G2 (orange) histogram representation
x3 = df["mean_RNA_G2"]
plt.hist(x3)
mean5 = np.mean(df["mean_RNA_G2"])
print("mean_RNA_G2 =", mean5)
variance5 = np.var(df["mean_RNA_G2"])
print("variance_RNA_G2 =", variance5)
x3 = df["mean_protein_G2"]
```

```
plt.hist(x3)
plt.title('mean_RNA_G2 (blue) and mean_protein_G2 (orange)');
mean6 = np.mean(df["mean_protein_G2"])
print("mean_protein_G2 = ", mean6)
variance6 = np.var(df["mean_protein_G2"])
print("variance_protein_G2 =", variance6)
```

```
mean_RNA_G2 = 11.257939490344251
variance_RNA_G2 = 2.096354817294313
mean_protein_G2 =  25.57355267002515
variance_protein_G2 = 9.99298221673658
```



2) Look at the pairwise correlations between each of the RNA/protein columns (this can be achieved using the corr() function). Does the change in timestep have much effect on the relationship(s) between RNA and protein?

Answer =

The change in timestep does not cause any overall effect. If there is any change, in the grand scheme of things they cancel out. The correlation value considering two times the same element is equal to 1 (table main diagonal, the elements are the same therefore correlation is maximum). Correlations between RNA and RNA values during different cell sphases gives values close to 1 (very high correlation). Correlations between protein and protein values during different cell sphases gives values close to 1 (very high correlation). Correlations between RNA and protein

values (or protein and RNA) during different or same cell sphases gives values close to 0.5 (low correlation).

Correlation between mean_RNA_G1 and mean_protein_G1 = 0.522657733063862 Correlation between mean_RNA_S and mean_protein_S = 0.5361902686743043 Correlation between mean_RNA_G2 and mean_protein_G2 = 0.5325650185250103 There are two different ways to calculate correlation = Springer and Pearson (the difference between the two is that one assumes the data can be fitted with a linear model)

We can use the same model generated at one stage to make prediction on the other stages (there is not much change in correlation between the different stages)

In [13]:
```
# Looking at the table, increasing the time-steps does not cause substantial changes
df.corr()
```

Out[13]:

	mean_RNA_G1	mean_RNA_S	mean_RNA_G2	mean_protein_G1	mean_protein_S	mean_protein_G2
mean_RNA_G1	1.000000	0.991063	0.992023	0.522658	0.541428	0.544206
mean_RNA_S	0.991063	1.000000	0.986836	0.514705	0.536190	0.534322
mean_RNA_G2	0.992023	0.986836	1.000000	0.510364	0.529690	0.532565
mean_protein_G1	0.522658	0.514705	0.510364	1.000000	0.970289	0.977016
mean_protein_S	0.541428	0.536190	0.529690	0.970289	1.000000	0.975964
mean_protein_G2	0.544206	0.534322	0.532565	0.977016	0.975964	1.000000

In [14]:
```
# Calculating
# 1) Correlation between mean_RNA_G1 and mean_protein_G1
# 2) Correlation between mean_RNA_S and mean_protein_S
# 3) Correlation between mean_RNA_G2 and mean_protein_G2

print("Correlation between mean_RNA_G1 and mean_protein_G1 =", df["mean_RNA_G1"].corr
print("Correlation between mean_RNA_G1 and mean_protein_G1 considering just the first
      (df["mean_RNA_G1"].loc[0:50]).corr(df["mean_protein_G1"].loc[0:50]))
print("Correlation between mean_RNA_G1 and mean_protein_G1 considering just the first
      (df["mean_RNA_G1"].loc[100:200]).corr(df["mean_protein_G1"].loc[100:200]))
print("Correlation between mean_RNA_G1 and mean_protein_G1 considering just the first
      (df["mean_RNA_G1"].loc[250:350]).corr(df["mean_protein_G1"].loc[250:350]))

print("\n")

print("Correlation between mean_RNA_S and mean_protein_S =", df["mean_RNA_S"].corr(df
print("Correlation between mean_RNA_S and mean_protein_S considering just the first 5
      (df["mean_RNA_S"].loc[0:50]).corr(df["mean_protein_S"].loc[0:50]))
print("Correlation between mean_RNA_S and mean_protein_S considering just the first 1
```

```
      (df["mean_RNA_S"].loc[100:200]).corr(df["mean_protein_S"].loc[100:200]))
print("Correlation between mean_RNA_S and mean_protein_S considering just the first 25
      (df["mean_RNA_S"].loc[250:350]).corr(df["mean_protein_S"].loc[250:350]))

print("\n")

print("Correlation between mean_RNA_G2 and mean_protein_G2 =", df["mean_RNA_G2"].corr
print("Correlation between mean_RNA_G2 and mean_protein_G2 considering just the first
      (df["mean_RNA_G2"].loc[0:50]).corr(df["mean_protein_G2"].loc[0:50]))
print("Correlation between mean_RNA_G2 and mean_protein_G2 considering just the first
      (df["mean_RNA_G2"].loc[100:200]).corr(df["mean_protein_G2"].loc[100:200]))
print("Correlation between mean_RNA_G2 and mean_protein_G2 considering just the first
      (df["mean_RNA_G2"].loc[250:350]).corr(df["mean_protein_G2"].loc[250:350]))
```

```
Correlation between mean_RNA_G1 and mean_protein_G1 = 0.522657733063862
Correlation between mean_RNA_G1 and mean_protein_G1 considering just the first 50 elements= 0.5
Correlation between mean_RNA_G1 and mean_protein_G1 considering just the first 100-200 index el
Correlation between mean_RNA_G1 and mean_protein_G1 considering just the first 250-350 index el


Correlation between mean_RNA_S and mean_protein_S = 0.5361902686743043
Correlation between mean_RNA_S and mean_protein_S considering just the first 50 elements= 0.553
Correlation between mean_RNA_S and mean_protein_S considering just the first 100-200 index elem
Correlation between mean_RNA_S and mean_protein_S considering just the first 250-350 index elem


Correlation between mean_RNA_G2 and mean_protein_G2 = 0.5325650185250103
Correlation between mean_RNA_G2 and mean_protein_G2 considering just the first 50 elements= 0.5
Correlation between mean_RNA_G2 and mean_protein_G2 considering just the first 100-200 index el
Correlation between mean_RNA_G2 and mean_protein_G2 considering just the first 250-350 index el
```

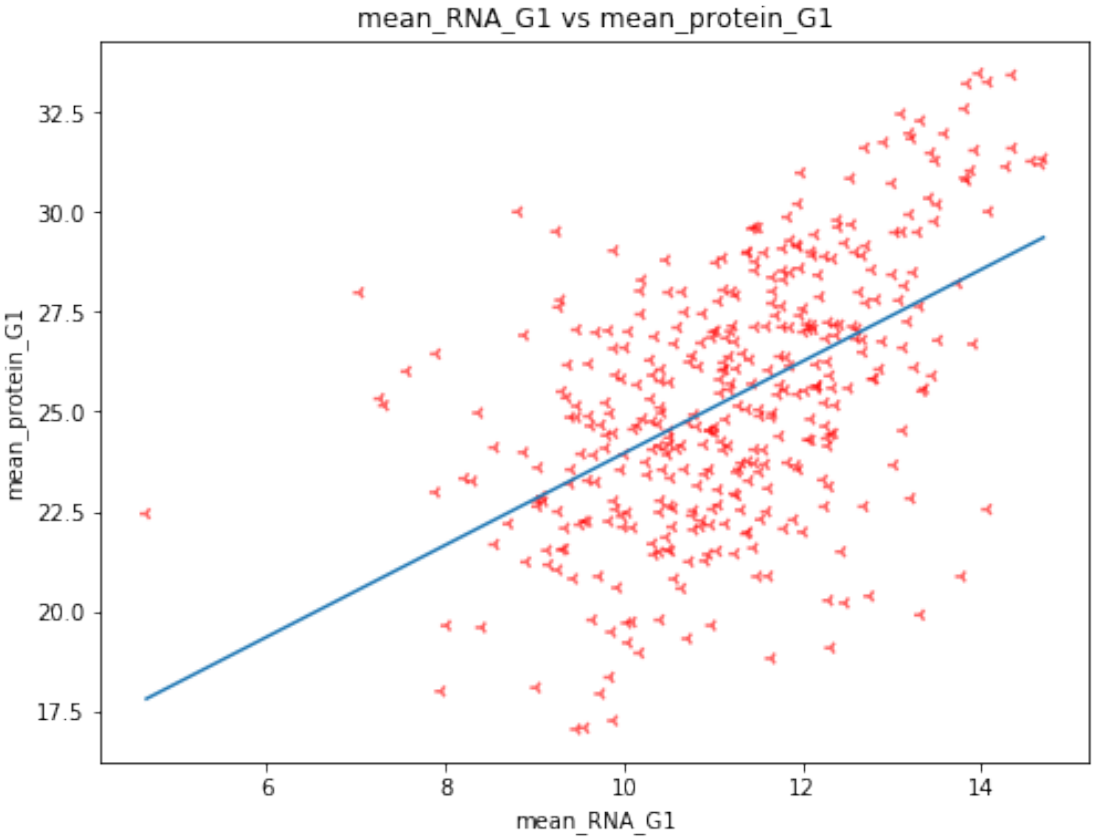
3) Generate a scatterplot of the RNA versus protein for each cell cycle stage. Fit a linear model to the data, can we infer protein concentration from RNA concentration?

Answer =

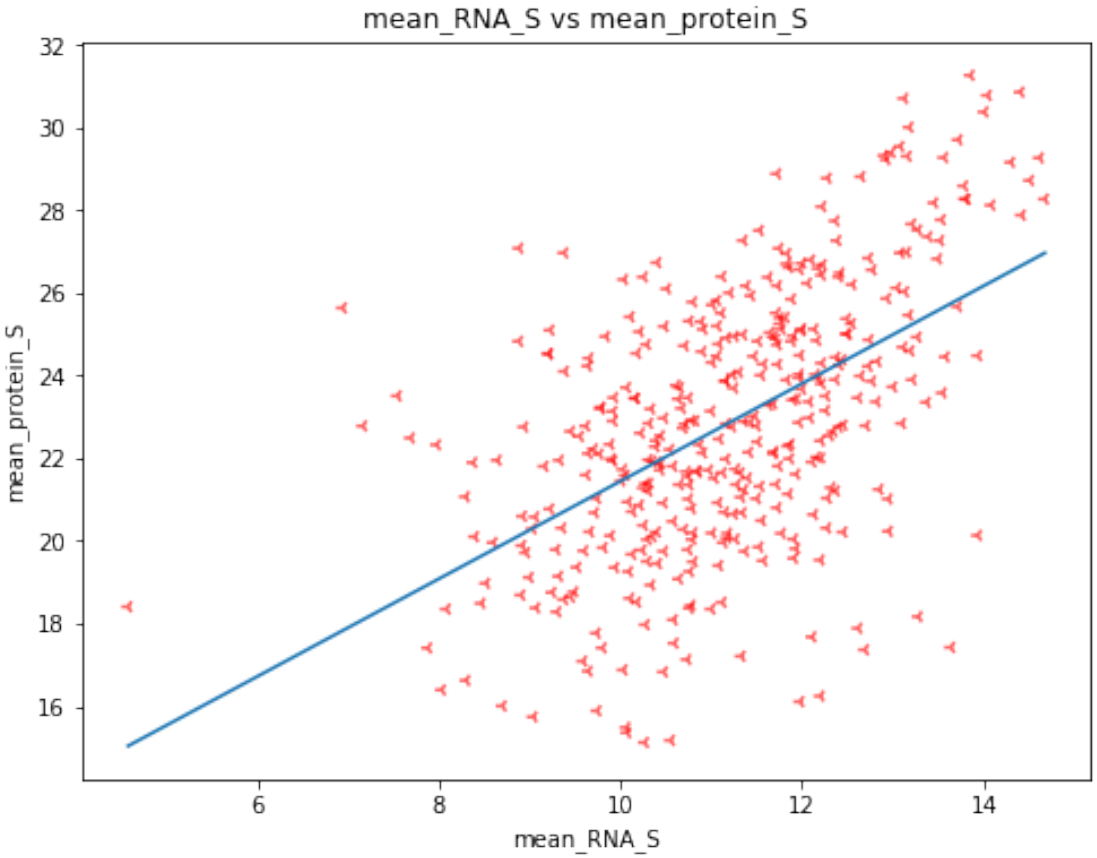
In all the considered cases (G1,S and G2 phases), when trying to fit a linear model to the data we don't get very good results. That because, as seen in the previous section, there is a low correlation between RNA and proteins values (around 0.5). When trying to fit a straight line to the data, we can only get good results if the data variables are all strongly correlated each other (either positively or negatively). If there is a low variables correlation, then there will be some outlier values in the data that can't be captured using a straight line. That's quite interesting to observe, especially considering that historically, RNA concentration was used to infer protein concentration. The outlier values (the values that are further away from the straight line), are particularly important in determining cell phases characteristics.

```
In [15]: # Making a scatter plot of mean_RNA_G1 vs mean_protein_G1 and then finding the best f
         plt.figure(figsize=(8,6))
```

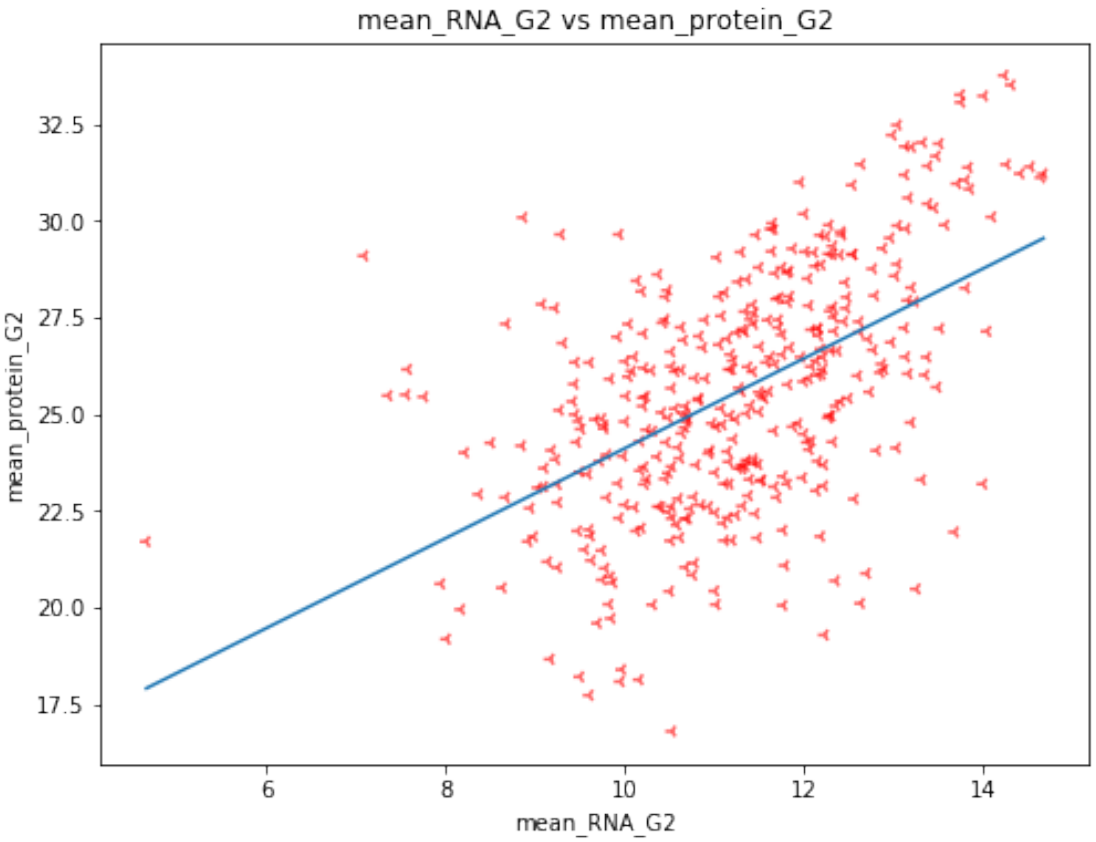
```
plt.scatter(df["mean_RNA_G1"], df["mean_protein_G1"], color='r', linestyle='dashed', m
plt.title('mean_RNA_G1 vs mean_protein_G1')
plt.xlabel('mean_RNA_G1')
plt.ylabel('mean_protein_G1')
# Fit a line to the data
plt.plot(np.unique(df["mean_RNA_G1"]), np.poly1d(np.polyfit(df["mean_RNA_G1"],
                                                            df["mean_protein_G1"], 1))
plt.show()
```



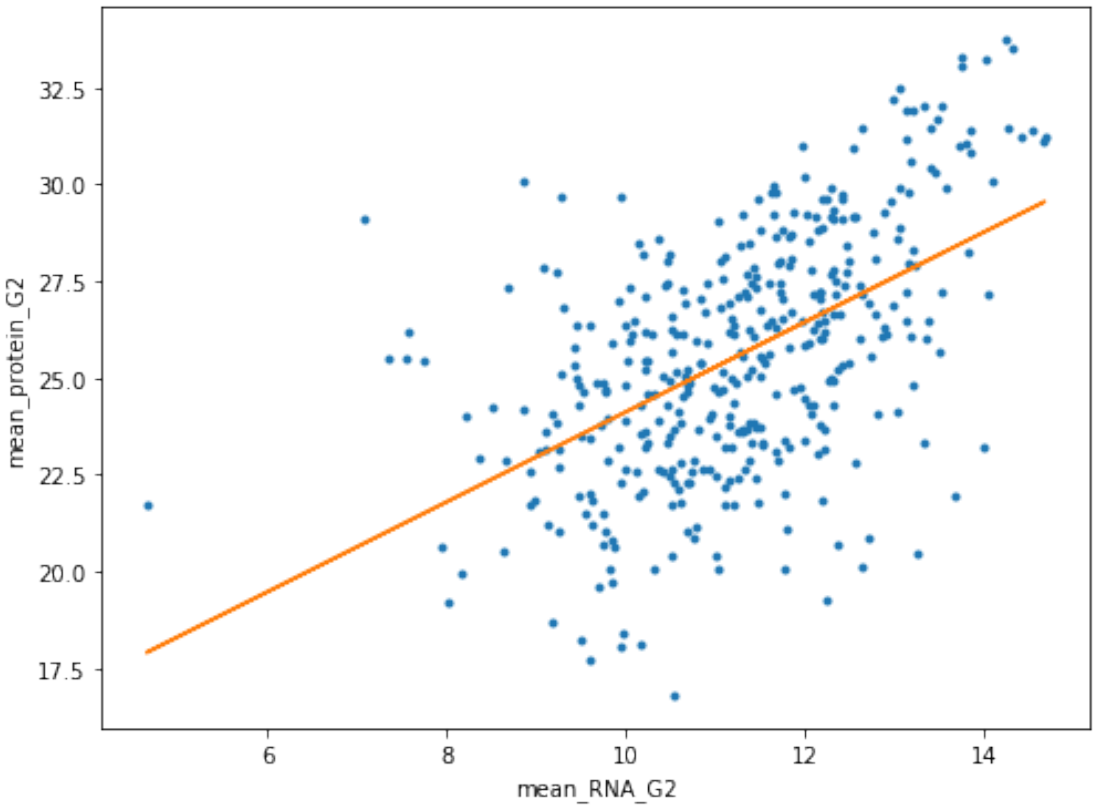
In [16]:
```
# Making a scatter plot of mean_RNA_S vs mean_protein_S and then finding the best fit
plt.figure(figsize=(8,6))
plt.scatter(df["mean_RNA_S"], df["mean_protein_S"], color='r', linestyle='dashed', ma
plt.title('mean_RNA_S vs mean_protein_S')
plt.xlabel('mean_RNA_S')
plt.ylabel('mean_protein_S')
# Fit a line to the data
plt.plot(np.unique(df["mean_RNA_S"]), np.poly1d(np.polyfit(df["mean_RNA_S"],
                                                           df["mean_protein_S"], 1))
plt.show()
```




```
In [17]: # Making a scatter plot of mean_RNA_G2 vs mean_protein_G2 and then finding the best f
         plt.figure(figsize=(8,6))
         plt.scatter(df["mean_RNA_G2"], df["mean_protein_G2"], color='r', linestyle='dashed', m
         plt.title('mean_RNA_G2 vs mean_protein_G2')
         plt.xlabel('mean_RNA_G2')
         plt.ylabel('mean_protein_G2')
         # Fit a line to the data
         plt.plot(np.unique(df["mean_RNA_G2"]), np.poly1d(np.polyfit(df["mean_RNA_G2"],
                                                                     df["mean_protein_G2"], 1))
         plt.show()
```



In [18]:
```
# Alternative way to fit a line to the data
from numpy.polynomial.polynomial import polyfit

# Sample data
x = df["mean_RNA_G2"]
y = df["mean_protein_G2"]

# Fit with polyfit
b, m = polyfit(x, y, 1)

plt.figure(figsize=(8,6))
plt.plot(x, y, '.')
plt.plot(x, b + m * x, '-')
plt.xlabel('mean_RNA_G2')
plt.ylabel('mean_protein_G2')
plt.show()
```



2 Week 2

3 GOBP = biological process

4 GOMF = molecular function

5 GOCC = cell/component

5.1 1) Find all genes that contain 'cell cycle' in their GOBP term and plot them as a scatterplot (with different colour) overlaid across all genes for each cell cycle phase. Is there a stronger/weaker correlation?

Answer =

When superimposing mean_RNA_G1 vs mean_protein_G1 with the elements of mean_RNA_G1 and mean_protein_G1 containing 'cell cycle' in their GOBP term it can be noticed quite a strong positive correlation between the considered features. The same can be obesrved when considering mean_RNA_S vs mean_protein_S and mean_RNA_G2 vs mean_protein_G2. The elements of mean_RNA_G1 and mean_protein_G1 containing 'cell cycle' in their GOBP represents quite well the overall trend of the data contained by mean_RNA_G1 vs mean_protein_G1. (Same for mean_RNA_S vs mean_protein_S and mean_RNA_G2 vs mean_protein_G2). Although, that

doesn't enable us to describe the elements of mean_RNA_G1 and mean_protein_G1 containing 'cell cycle' in their GOBP using a straight line (the data is too sparse, too many outliers) (Same for mean_RNA_S vs mean_protein_S and mean_RNA_G2 vs mean_protein_G2). When comparing the different cell stages (G1,S,G2), it can be noticed that the distance between each of the point is almost the same in all the cases, there is just a scaling factor that varies when comparing the different cell phases. Correlation between mean_RNA_G1 and mean_protein_G1 = 0.522657733063862 Correlation between mean_RNA_S and mean_protein_S = 0.5361902686743043 Correlation between mean_RNA_G2 and mean_protein_G2 = 0.5325650185250103 Correlation between mean_RNA_G1 and mean_protein_G1 containing 'cell cycle' in their GOBP term = 0.43654604979020545 Correlation between mean_RNA_S and mean_protein_S containing 'cell cycle' in their GOBP term = 0.4383882841441922 Correlation between mean_RNA_G2 and mean_protein_G2 containing 'cell cycle' in their GOBP term = 0.45332725500447946

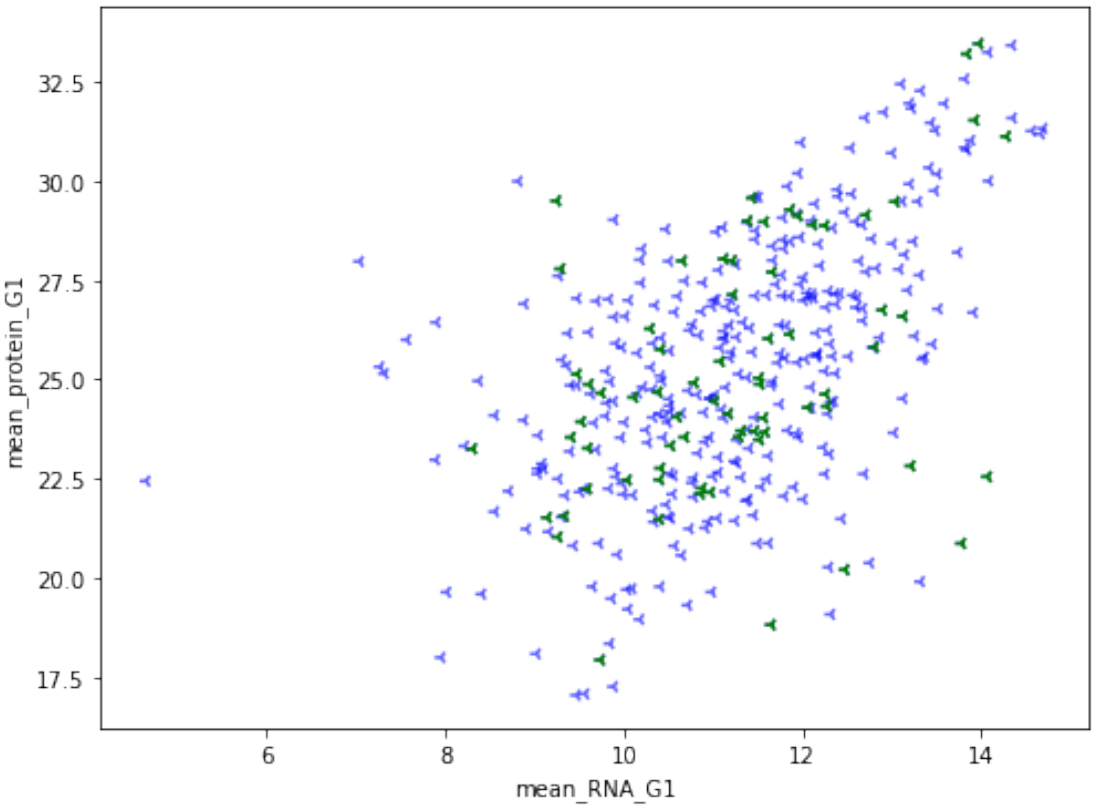
In [19]:
```
# Finding all genes that contain 'cell cycle' in their GOBP term
cellcgobp = df[df.GOBP.str.contains('cell cycle')]
print("Number of genes containging 'cell cycle' in their GOBP term =", len(cellcgobp))
cellcgobp.GOBP.head()
```

```
Number of genes containging 'cell cycle' in their GOBP term = 71
```

Out[19]:
```
10    biological regulation;cell cycle;cell cycle ch...
14    anatomical structure development;biological re...
21    activation of adenylate cyclase activity;activ...
33    biological regulation;blood coagulation;cell c...
52    anaphase-promoting complex-dependent proteasom...
Name: GOBP, dtype: object
```

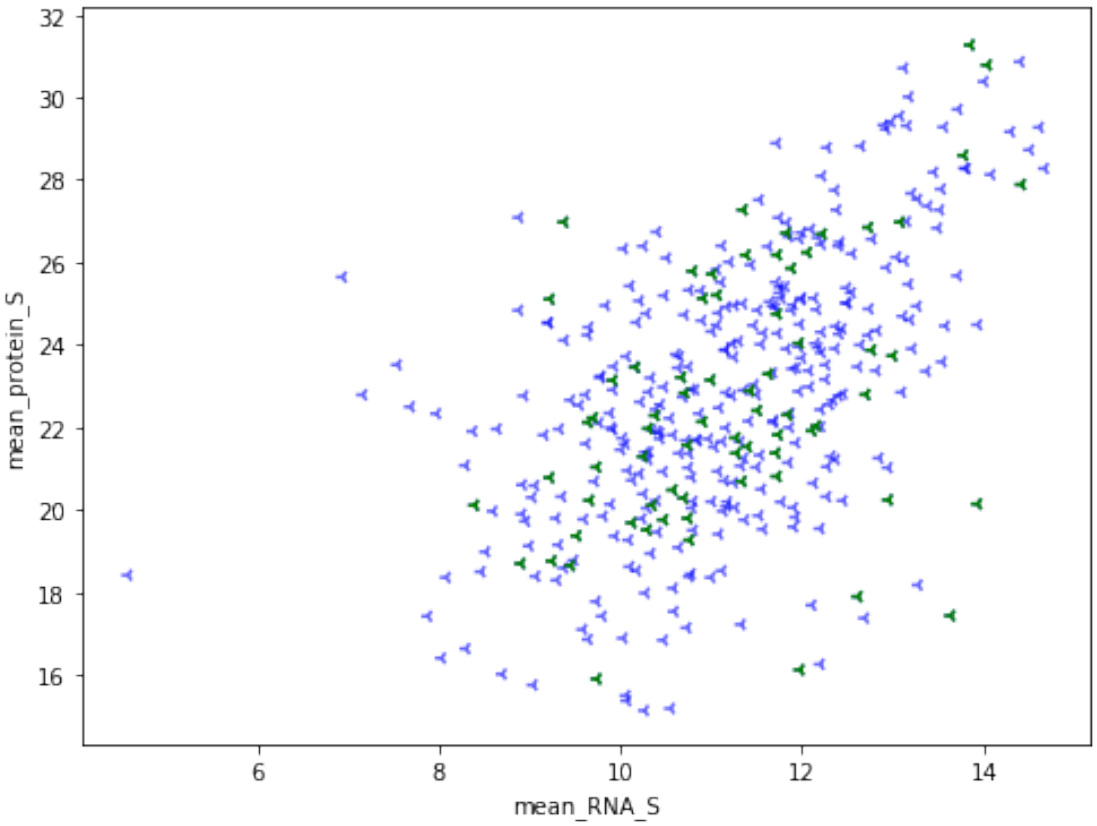
In [20]:
```
# Superimposing mean_RNA_G1 vs mean_protein_G1 with the elements of mean_RNA_G1 and m
# in their GOBP term
plt.figure(figsize=(8,6))
plt.scatter(df['mean_RNA_G1'], df['mean_protein_G1'], color='b', linestyle='dashed', m
plt.scatter(cellcgobp.mean_RNA_G1, cellcgobp.mean_protein_G1, color='g', linestyle='da
plt.xlabel('mean_RNA_G1')
plt.ylabel('mean_protein_G1')
```

Out[20]:
```
Text(0,0.5,'mean_protein_G1')
```



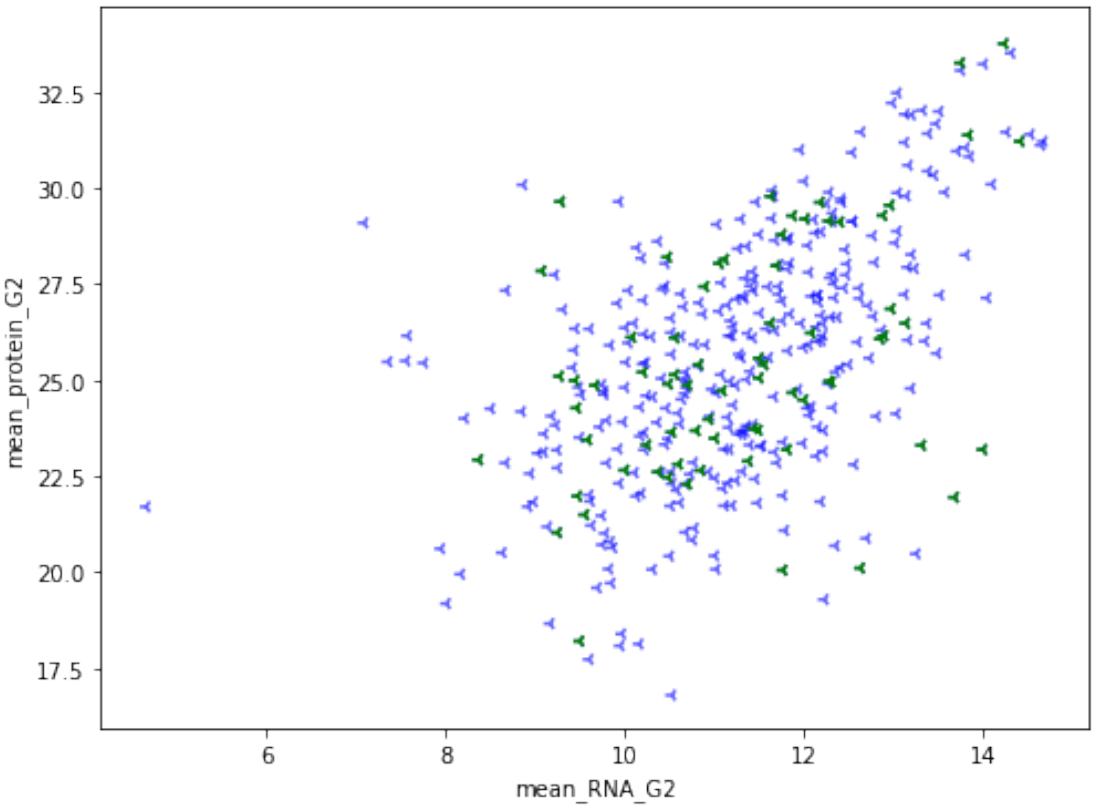
In [21]:
```
# Superimposing mean_RNA_S vs mean_protein_S with the elements of mean_RNA_S and mean_
# in their GOBP term
plt.figure(figsize=(8,6))
plt.scatter(df['mean_RNA_S'], df['mean_protein_S'], color='b', linestyle='dashed', ma
plt.scatter(cellcgobp.mean_RNA_S, cellcgobp.mean_protein_S, color='g', linestyle='das
plt.xlabel('mean_RNA_S')
plt.ylabel('mean_protein_S')
```

Out[21]:
```
Text(0,0.5,'mean_protein_S')
```



In [22]:
```
# Superimposing mean_RNA_G2 vs mean_protein_G2 with the elements of mean_RNA_G2 and m
# in their GOBP term
plt.figure(figsize=(8,6))
plt.scatter(df['mean_RNA_G2'], df['mean_protein_G2'], color='b', linestyle='dashed', m
plt.scatter(cellcgobp.mean_RNA_G2, cellcgobp.mean_protein_G2, color='g', linestyle='da
plt.xlabel('mean_RNA_G2')
plt.ylabel('mean_protein_G2')
```

Out[22]:
```
Text(0,0.5,'mean_protein_G2')
```




```
In [23]: print("Correlation between mean_RNA_G1 and mean_protein_G1 =", df["mean_RNA_G1"].corr
         print("Correlation between mean_RNA_S and mean_protein_S =", df["mean_RNA_S"].corr(df
         print("Correlation between mean_RNA_G2 and mean_protein_G2 =", df["mean_RNA_G2"].corr

         print("Correlation between mean_RNA_G1 and mean_protein_G1 containing 'cell cycle' in
               cellcgobp.mean_RNA_G1.corr(cellcgobp.mean_protein_G1))
         print("Correlation between mean_RNA_S and mean_protein_S containing 'cell cycle' in t
               cellcgobp.mean_RNA_S.corr(cellcgobp.mean_protein_S))
         print("Correlation between mean_RNA_G2 and mean_protein_G2 containing 'cell cycle' in
               cellcgobp.mean_RNA_G2.corr(cellcgobp.mean_protein_G2))
```

```
Correlation between mean_RNA_G1 and mean_protein_G1 = 0.522657733063862
Correlation between mean_RNA_S and mean_protein_S = 0.5361902686743043
Correlation between mean_RNA_G2 and mean_protein_G2 = 0.5325650185250103
Correlation between mean_RNA_G1 and mean_protein_G1 containing 'cell cycle' in their GOBP term
Correlation between mean_RNA_S and mean_protein_S containing 'cell cycle' in their GOBP term =
Correlation between mean_RNA_G2 and mean_protein_G2 containing 'cell cycle' in their GOBP term
```

5.2 2) Repeat task 1 by finding genes that contain 'ribosome' in their GOCC term.

Answer =

When superimposing mean_RNA_G1 vs mean_protein_G1 with the elements of mean_RNA_G1 and mean_protein_G1 containing 'ribosome' in their GOCC term it can be noticed that there are just 19 corrisponding cells. Because of this, there are not enough features to fully understand if there is a strong corraltion. The same can be obesrved when considering mean_RNA_S vs mean_protein_S and mean_RNA_G2 vs mean_protein_G2. When comparing the different cell stages (G1,S,G2), it can be noticed that the distance between each of the point is almost the same in all the cases, there is just a scaling factor that varies when comparing the different cell phases. That doesn't enable us to describe the elements of mean_RNA_G1 and mean_protein_G1 containing 'ribosome' in their GOCC using a straight line (the data is too sparse, and there are not enough data-points) (Same for mean_RNA_S vs mean_protein_S and mean_RNA_G2 vs mean_protein_G2). Correlation between mean_RNA_G1 and mean_protein_G1 = 0.522657733063862 Correlation between mean_RNA_S and mean_protein_S = 0.5361902686743043 Correlation between mean_RNA_G2 and mean_protein_G2 = 0.5325650185250103 Correlation between mean_RNA_G1 and mean_protein_G1 containing 'ribosome' in their GOCC term = 0.8408005925694084 Correlation between mean_RNA_S and mean_protein_S containing 'ribosome' in their GOCC term = 0.8448011378787456 Correlation between mean_RNA_G2 and mean_protein_G2 containing 'ribosome' in their GOCC term = 0.8477056210062089

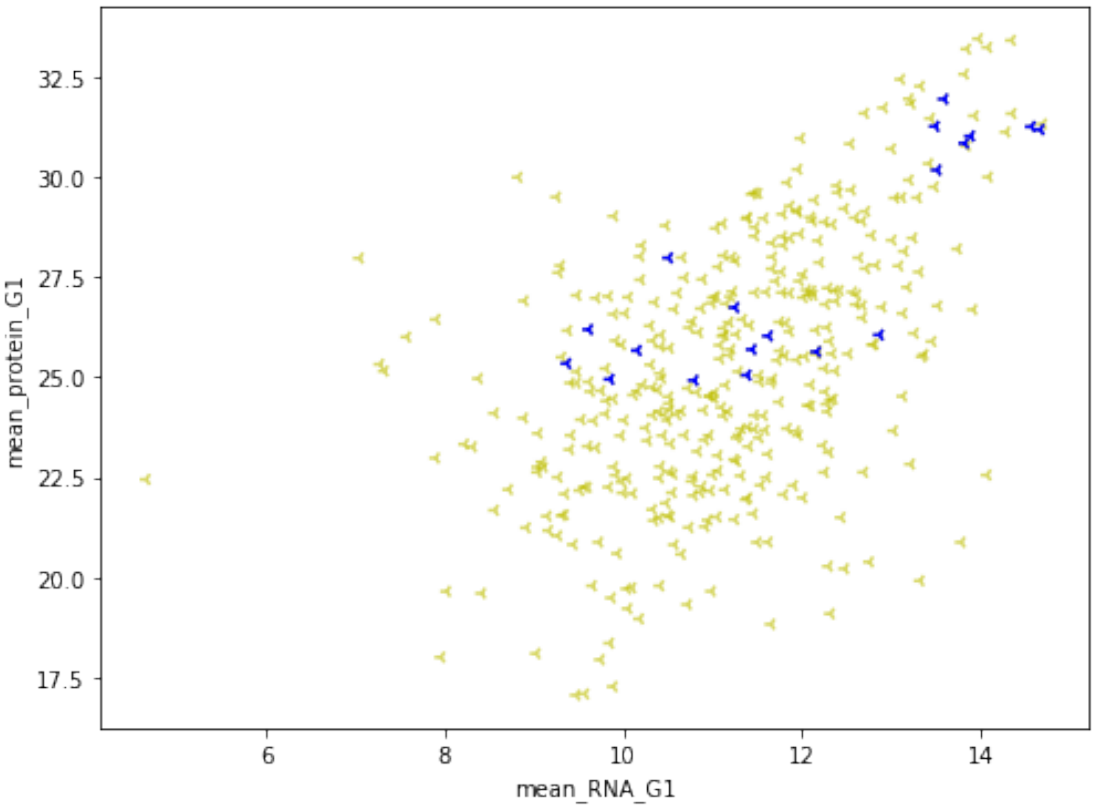
In [24]:
```
# Finding all genes that contain 'ribosome' in their GOBP term
ribgocc = df[df.GOCC.str.contains('ribosome')]
print("Number of genes containging 'ribosome' in their GOCC term =", len(ribgocc))
ribgocc.GOCC.head()
```

```
Number of genes containging 'ribosome' in their GOCC term = 19
```

Out[24]:
```
22    cell part;cytoplasmic part;endoplasmic reticul...
27    cell part;cytoplasmic part;cytosolic small rib...
45    cell part;cytoplasmic part;intracellular membr...
66    cell part;cytoplasmic part;intracellular non-m...
72    90S preribosome;cell part;cytosolic small ribo...
Name: GOCC, dtype: object
```

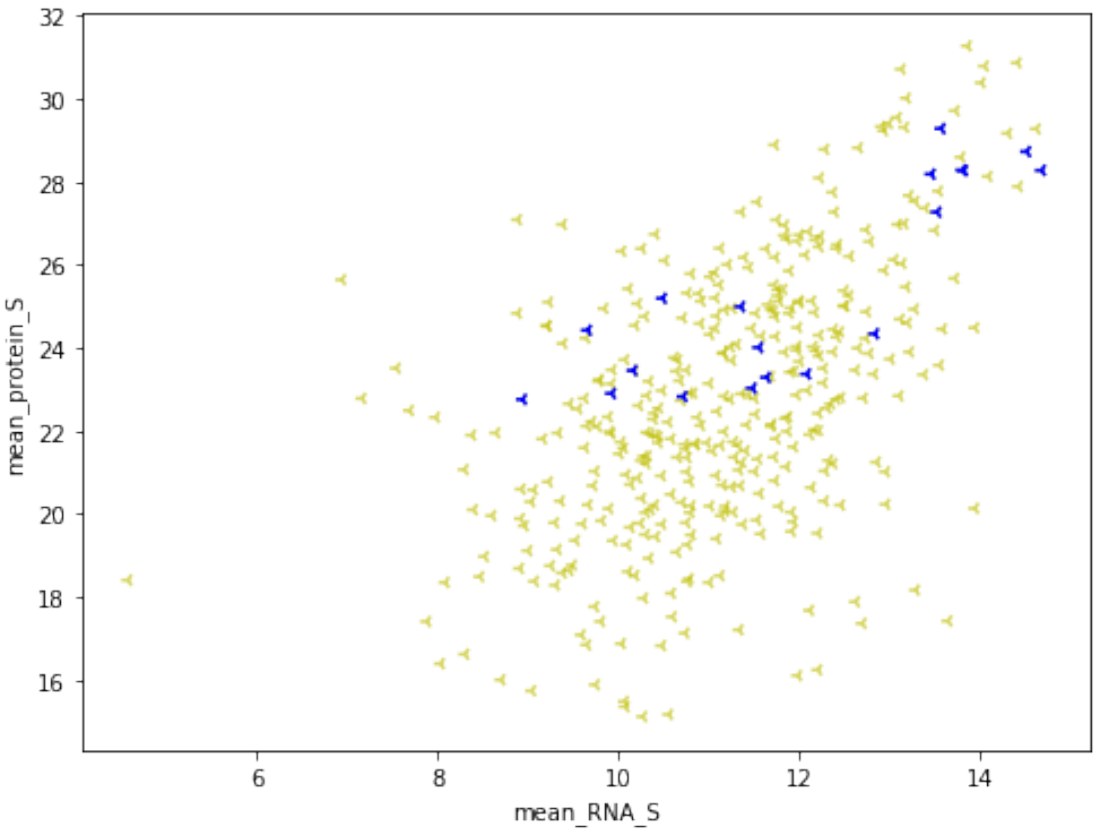
In [25]:
```
# Superimposing mean_RNA_G1 vs mean_protein_G1 with the elements of mean_RNA_G1 and m
# in their GOCC term
plt.figure(figsize=(8,6))
plt.scatter(df['mean_RNA_G1'], df['mean_protein_G1'], color='y', linestyle='dashed', r
plt.scatter(ribgocc.mean_RNA_G1, ribgocc.mean_protein_G1, color='b', linestyle='dashe
plt.xlabel('mean_RNA_G1')
plt.ylabel('mean_protein_G1')
```

Out[25]:
```
Text(0,0.5,'mean_protein_G1')
```



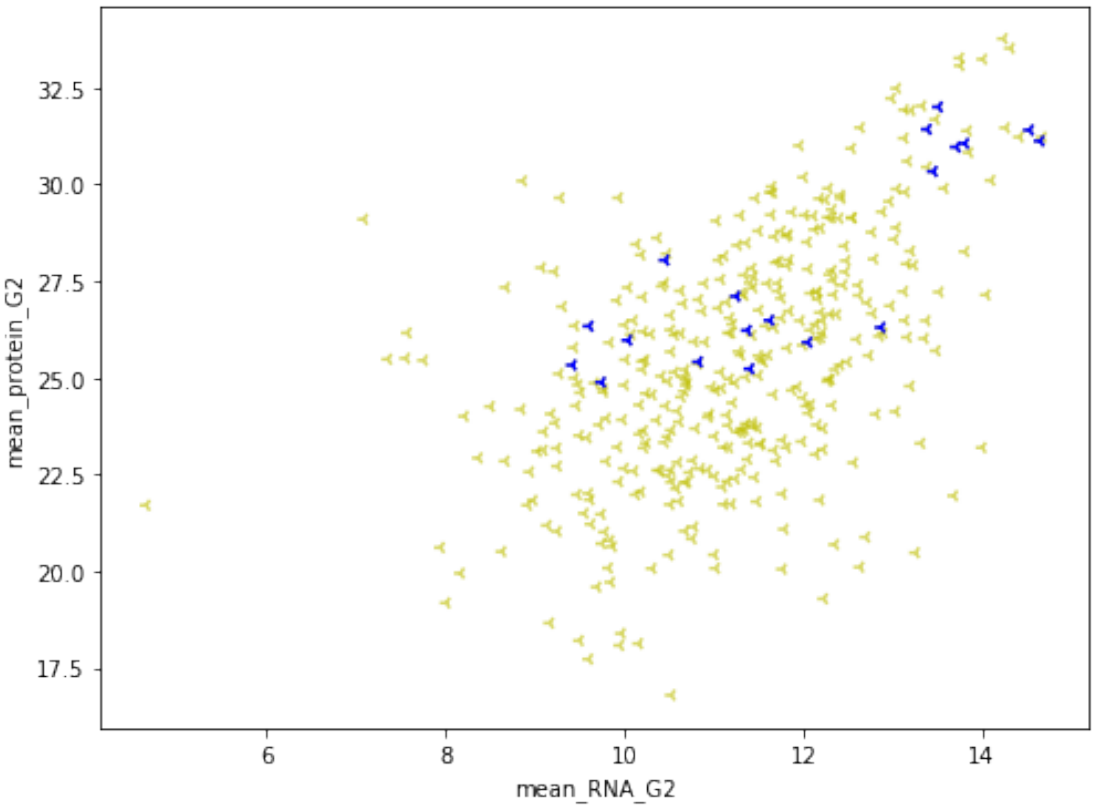
In [26]:
```
# Superimposing mean_RNA_S vs mean_protein_S with the elements of mean_RNA_S and mean_
# in their GOCC term
plt.figure(figsize=(8,6))
plt.scatter(df['mean_RNA_S'], df['mean_protein_S'], color='y', linestyle='dashed', ma
plt.scatter(ribgocc.mean_RNA_S, ribgocc.mean_protein_S, color='b', linestyle='dashed'
plt.xlabel('mean_RNA_S')
plt.ylabel('mean_protein_S')
```

Out[26]:
```
Text(0,0.5,'mean_protein_S')
```



In [27]:
```
# Superimposing mean_RNA_G2 vs mean_protein_G2 with the elements of mean_RNA_G2 and m
# in their GOCC term
plt.figure(figsize=(8,6))
plt.scatter(df['mean_RNA_G2'], df['mean_protein_G2'], color='y', linestyle='dashed', r
plt.scatter(ribgocc.mean_RNA_G2, ribgocc.mean_protein_G2, color='b', linestyle='dashe
plt.xlabel('mean_RNA_G2')
plt.ylabel('mean_protein_G2')
```

Out[27]:
```
Text(0,0.5,'mean_protein_G2')
```




```
In [28]: print("Correlation between mean_RNA_G1 and mean_protein_G1 =", df["mean_RNA_G1"].corr
         print("Correlation between mean_RNA_S and mean_protein_S =", df["mean_RNA_S"].corr(df
         print("Correlation between mean_RNA_G2 and mean_protein_G2 =", df["mean_RNA_G2"].corr

         print("Correlation between mean_RNA_G1 and mean_protein_G1 containing 'ribosome' in th
               ribgocc.mean_RNA_G1.corr(ribgocc.mean_protein_G1))
         print("Correlation between mean_RNA_S and mean_protein_S containing 'ribosome' in thei
               ribgocc.mean_RNA_S.corr(ribgocc.mean_protein_S))
         print("Correlation between mean_RNA_G2 and mean_protein_G2 containing 'ribosome' in th
               ribgocc.mean_RNA_G2.corr(ribgocc.mean_protein_G2))
```

```
Correlation between mean_RNA_G1 and mean_protein_G1 = 0.522657733063862
Correlation between mean_RNA_S and mean_protein_S = 0.5361902686743043
Correlation between mean_RNA_G2 and mean_protein_G2 = 0.5325650185250103
Correlation between mean_RNA_G1 and mean_protein_G1 containing 'ribosome' in their GOCC term =
Correlation between mean_RNA_S and mean_protein_S containing 'ribosome' in their GOCC term = 0
Correlation between mean_RNA_G2 and mean_protein_G2 containing 'ribosome' in their GOCC term =
```

5.3 3) Count the number of occurrences of every GOBP term across all genes, what are some of the difficulties that arise when using these terms?

Answer =

It might have been better to have used a readme file or a legend attached to this dataset (Cell-Cycle-Set.xlsx) in order to make clear the meaning and the implications of all the terms used. This would in fact make easier for the public to make use of this data-set and would avoid the use of repetitive words across all the description terms. There are too many terms that are present just once, that makes analysis more difficult.

```
In [29]: print(df.GOBP.str.split(';',expand=True).stack().value_counts())
```

```
cellular process
metabolic process
cellular metabolic process
primary metabolic process
biological regulation
regulation of biological process
macromolecule metabolic process
regulation of cellular process
cellular macromolecule metabolic process
nitrogen compound metabolic process
cellular nitrogen compound metabolic process
nucleobase-containing compound metabolic process
response to stimulus
cellular component organization or biogenesis
nucleic acid metabolic process
cellular component organization
regulation of metabolic process
regulation of cellular metabolic process
regulation of primary metabolic process
regulation of macromolecule metabolic process
cellular component organization or biogenesis at cellular level
cellular component organization at cellular level
RNA metabolic process
biosynthetic process
cellular biosynthetic process
cellular response to stimulus
protein metabolic process
regulation of nitrogen compound metabolic process
establishment of localization
regulation of nucleobase-containing compound metabolic process

negative regulation of reactive oxygen species metabolic process
mesenchyme development
positive regulation of protein dephosphorylation
nuclear RNA surveillance
sterol transmembrane transport
```

```
melanin biosynthetic process
CUT catabolic process
regulation of DNA damage response, signal transduction by p53 class mediator
nuclear polyadenylation-dependent ncRNA catabolic process
antigen processing and presentation of endogenous peptide antigen via MHC class I
peroxisome fission
isoprenoid catabolic process
posterior midgut development
T cell differentiation in thymus
ethanol oxidation
negative regulation of cell fate commitment
lacrimal gland development
lymphocyte migration into lymphoid organs
regulation of telomere maintenance
regulation of lipid transport by positive regulation of transcription from RNA polymerase II p
kinetochore assembly
foam cell differentiation
NAD metabolic process
regulation of S phase
histone H4-K16 acetylation
nucleoside salvage
cellular response to indole-3-methanol
positive regulation of cAMP biosynthetic process
activation of Rac GTPase activity
regulation of glycolysis by negative regulation of transcription from RNA polymerase II promot
Length: 2854, dtype: int64
```

```
In [30]: print(df.GOMF.str.split(';',expand=True).stack().value_counts())
```

```
binding                                                          366
protein binding                                                  276
catalytic activity                                               180
nucleotide binding                                               116
nucleic acid binding                                             111
ribonucleotide binding                                            86
purine ribonucleotide binding                                     86
purine ribonucleoside triphosphate binding                        86
purine nucleotide binding                                         86
hydrolase activity                                                84
ion binding                                                       84
metal ion binding                                                 84
cation binding                                                    84
adenyl ribonucleotide binding                                     69
ATP binding                                                       69
adenyl nucleotide binding                                         69
RNA binding                                                       61
transferase activity                                              60
```

```
enzyme binding                                                                 59
DNA binding                                                                    56
hydrolase activity, acting on acid anhydrides, in phosphorus-containing anhydrides    45
hydrolase activity, acting on acid anhydrides                                  45
pyrophosphatase activity                                                       45
transferase activity, transferring phosphorus-containing groups                43
nucleoside-triphosphatase activity                                             43
transition metal ion binding                                                   40
kinase activity                                                                40
structural molecule activity                                                   35
zinc ion binding                                                               34
cytoskeletal protein binding                                                   32
                                                                              ...
lipoprotein particle receptor activity                                          1
androsterone dehydrogenase activity                                             1
RNA polymerase II transcription factor binding                                  1
selenium binding                                                                1
transmembrane receptor protein tyrosine phosphatase activity                    1
RNA polymerase II core promoter proximal region sequence-specific DNA binding   1
prostaglandin receptor activity                                                 1
SH2 domain binding                                                              1
phosphatidylinositol 3-kinase binding                                           1
fibroblast growth factor binding                                                1
carboxylesterase activity                                                       1
non-membrane spanning protein tyrosine kinase activity                          1
gated channel activity                                                          1
amine transmembrane transporter activity                                        1
SMAD binding                                                                    1
glycerol kinase activity                                                        1
17-alpha,20-alpha-dihydroxypregn-4-en-3-one dehydrogenase activity              1
citrate hydro-lyase (cis-aconitate-forming) activity                            1
receptor signaling protein activity                                             1
dihydrotestosterone 17-beta-dehydrogenase activity                              1
spermidine synthase activity                                                    1
extracellular matrix binding                                                    1
protein tyrosine kinase activator activity                                      1
CD4 receptor binding                                                            1
loop DNA binding                                                                1
protein-arginine N-methyltransferase activity                                   1
lysine N-acetyltransferase activity                                             1
ARF GTPase activator activity                                                   1
phosphatidylinositol-4,5-bisphosphate binding                                   1
acyl binding                                                                    1
Length: 842, dtype: int64
```

5.4 4) Calculate the change in mRNA/protein level across the cell cycle by taking the difference at each stage (G1-S, S-G2, G2-G1), and standardize the differences by mean-centering and variance scaling. Repeat tasks 1 and 2 by plotting the changes in levels with GOBP/GOCC labelling. What do we notice about changes in the cell cycle? Is there any apparent clustering of GO terms?

Answer =

Now all the graphs are centred around 0. When superimposing mean_RNA_g1s vs mean_protein_g1s with the elements of mean_RNA_g1s and mean_protein_g1s containing 'cell cycle' in their GOBP term and 'ribosome' in their GOCC term we can see: 1) mean_RNA_g1s and mean_protein_g1s form a clear clouster centered around 0 (in blue) 2) the elements of mean_RNA_g1s and mean_protein_g1s containing 'cell cycle' in their GOBP term form another clouster (in green) centered at X=0 and Y=3. 3) the elements of mean_RNA_g1s and mean_protein_g1s containing 'ribosome' in their GOCC term form another clouseter (in yellow) centred at X=0 and Y=2.6/2.7. When superimposing mean_RNA_sg2 vs mean_protein_sg2 with the elements of mean_RNA_sg2 and mean_protein_sg2 containing 'cell cycle' in their GOBP term and 'ribosome' in their GOCC term we can see: 1) mean_RNA_sg2 and mean_protein_sg2 form a clear clouster centered around 0 (in blue) (the blue clouster has some big outliers values) 2) the elements of mean_RNA_sg2 and mean_protein_sg2 containing 'cell cycle' in their GOBP term form another clouster (in green) centered at X=0 and Y=-3. 3) the elements of mean_RNA_sg2 and mean_protein_sg2 containing 'ribosome' in their GOCC term form another clouseter (in yellow) centred at X=0 and Y=-2.6/2.7. (the yellow clouser is more compact than the green one) When superimposing mean_RNA_g2g1 vs mean_protein_g2g1 with the elements of mean_RNA_g2g1 and mean_protein_g2g1 containing 'cell cycle' in their GOBP term and 'ribosome' in their GOCC term we can see: 1) mean_RNA_g2g1 and mean_protein_g2g1 form a clear clouster centered around 0 (in blue) 2) the elements of mean_RNA_g2g1 and mean_protein_g2g1 containing 'cell cycle' in their GOBP term form another clouster (in green) centered at X=0 and Y=0 (smaller dimensions than the blue clouster). 3) the elements of mean_RNA_g2g1 and mean_protein_g2g1 containing 'ribosome' in their GOCC term form another clouseter (in yellow) centred at X=0 and Y=0 (smaller dimensions than the blue clouster and more compact than the green clouster). During each different cell cycle the blue clouster position remains unaltered but the green and yellow clousters change alsways their position

In [31]:
```
# Calculating the change in mRNA/protein level across the cell cycle by taking the di
# and standardizing the differences by mean-centering and veriance scaling

from sklearn import preprocessing

df['mean_RNA_g1s'] = (df.mean_RNA_G1 - df.mean_RNA_S)
df['mean_RNA_sg2'] = (df.mean_RNA_S - df.mean_RNA_G2)
df['mean_RNA_g2g1'] = (df.mean_RNA_G2 - df.mean_RNA_G1)
df['mean_protein_g1s'] = (df.mean_protein_G1 - df.mean_protein_S)
df['mean_protein_sg2'] = (df.mean_protein_S - df.mean_protein_G1)
df['mean_protein_g2g1'] = (df.mean_protein_G2 - df.mean_protein_G1)

print("Dataset mean before standardizing and normalizing the data")
print(df.mean())
print("\nDataset variance before standardizing and normalizing the data")
```

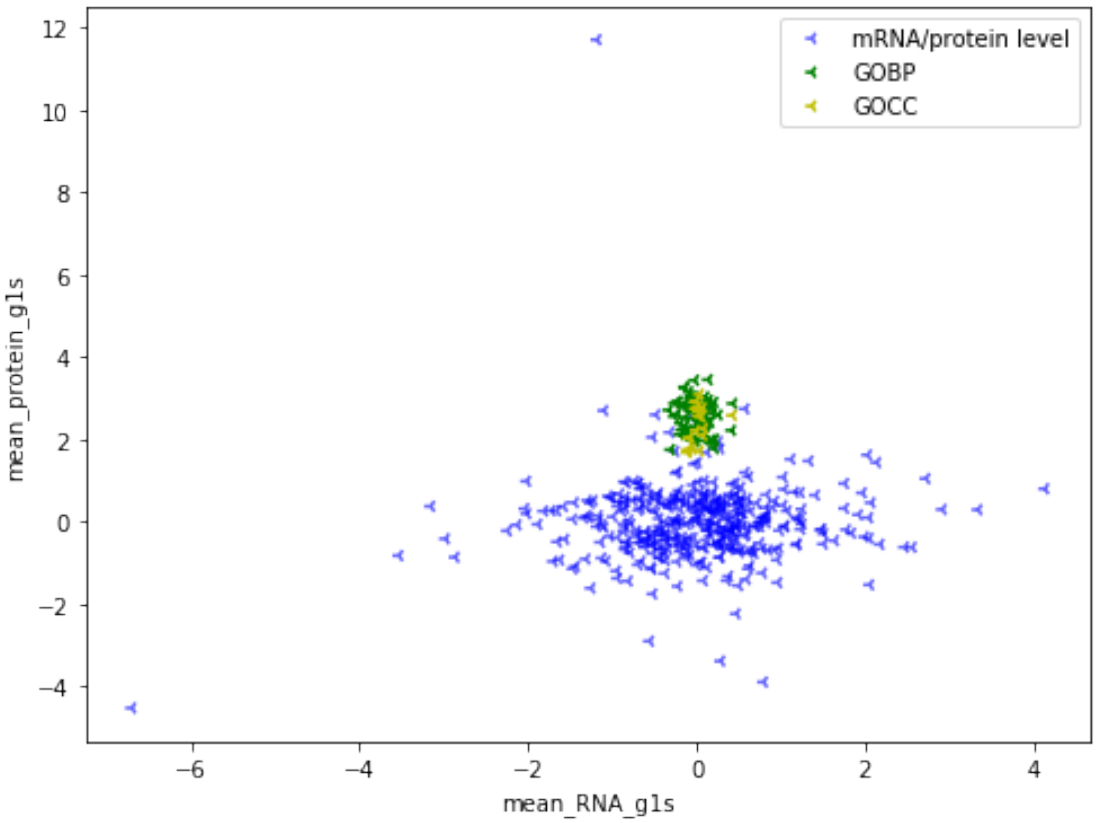
```
print(df.std())

mean_RNA_G1 = preprocessing.scale(df["mean_RNA_G1"])
mean_RNA_S = preprocessing.scale(df["mean_RNA_S"])
mean_RNA_G2 = preprocessing.scale(df["mean_RNA_G2"])
mean_protein_G1 = preprocessing.scale(df["mean_protein_G1"])
mean_protein_S = preprocessing.scale(df["mean_protein_S"])
mean_protein_G2 = preprocessing.scale(df["mean_protein_G2"])
mean_RNA_g1s = preprocessing.scale(df["mean_RNA_g1s"])
mean_RNA_sg2 = preprocessing.scale(df["mean_RNA_sg2"])
mean_RNA_g2g1 = preprocessing.scale(df["mean_RNA_g2g1"])
mean_protein_g1s = preprocessing.scale(df["mean_protein_g1s"])
mean_protein_sg2 = preprocessing.scale(df["mean_protein_sg2"])
mean_protein_g2g1 = preprocessing.scale(df["mean_protein_g2g1"])

print("\nSome exaples of Dataset after having been standardized and normalized:")
print("1) mean_protein_G2\n",mean_protein_G2.mean())
print("",mean_protein_G2.std())
print("2) mean_RNA_G1\n",mean_RNA_G1.mean())
print("",mean_RNA_G1.std())
print("3) mean_protein_g1s\n",mean_protein_g1s.mean())
print("",mean_protein_g1s.std())

# Alternative way to standardise and normilize data
# from sklearn.preprocessing import StandardScaler
# ss = StandardScaler()
# scaled = ss.fit_transform(df[["mean_RNA_G1","mean_RNA_S","mean_RNA_G2","mean_protei
#                                 "mean_protein_G2","mean_RNA_g1s","mean_RNA_sg2"
#                               "mean_protein_sg2", "mean_protein_g2g1"]])
# print(scaled.mean(),scaled.std())
```

```
Dataset mean before standardizing and normalizing the data
mean_RNA_G1          11.215627
mean_RNA_S           11.186962
mean_RNA_G2          11.257939
mean_protein_G1      25.351672
mean_protein_S       22.847658
mean_protein_G2      25.573553
mean_RNA_g1s          0.028665
mean_RNA_sg2         -0.070977
mean_RNA_g2g1         0.042312
mean_protein_g1s      2.504014
mean_protein_sg2     -2.504014
mean_protein_g2g1     0.221881
dtype: float64

Dataset variance before standardizing and normalizing the data
mean_RNA_G1          1.469866
```

```
mean_RNA_S           1.464784
mean_RNA_G2          1.449706
mean_protein_G1      3.233199
mean_protein_S       3.225371
mean_protein_G2      3.165157
mean_RNA_g1s         0.196234
mean_RNA_sg2         0.236924
mean_RNA_g2g1        0.185482
mean_protein_g1s     0.787225
mean_protein_sg2     0.787225
mean_protein_g2g1    0.689230
dtype: float64

Some exaples of Dataset after having been standardized and normalized:
1) mean_protein_G2
 5.145618048640524e-16
 0.9999999999999999
2) mean_RNA_G1
 9.93328005911475e-16
 1.0
3) mean_protein_g1s
 8.501455906449561e-17
 1.0
```

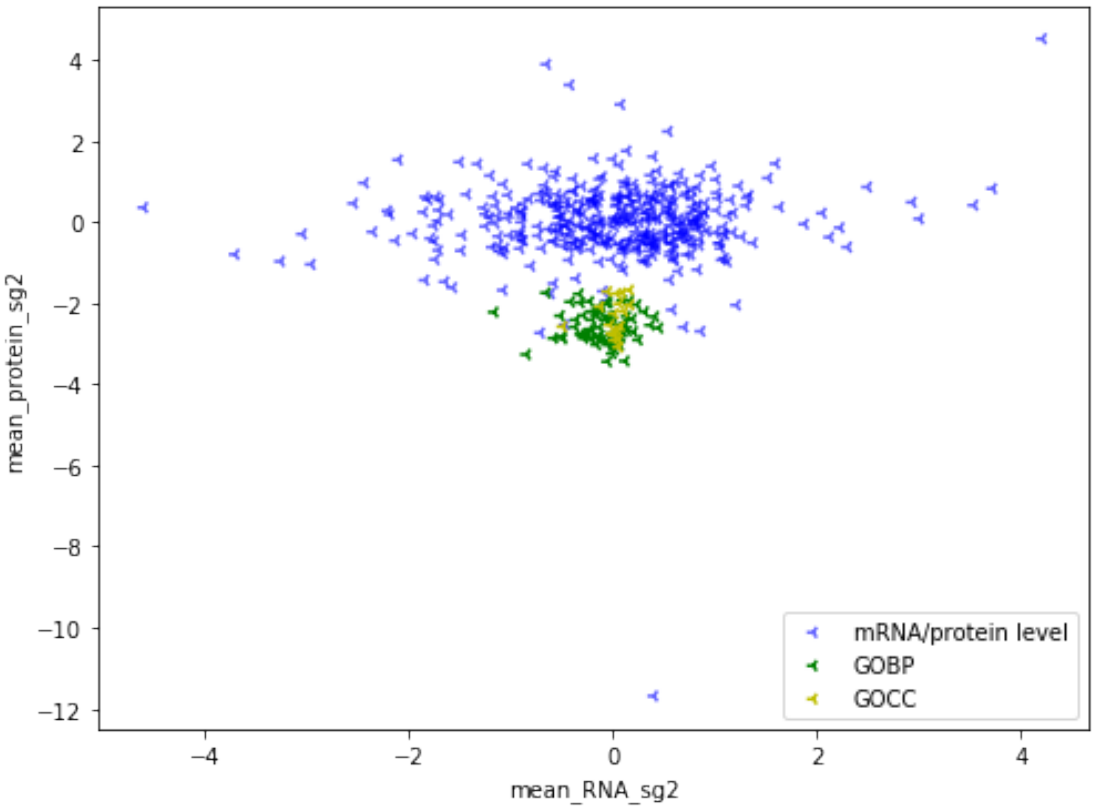
In [32]:
```
# Superimposing mean_RNA_g1s vs mean_protein_g1s with the elements of mean_RNA_g1s an
# in their GOBP term and 'ribosome' in their GOCC term
plt.figure(figsize=(8,6))

cellcgobp = df[df.GOBP.str.contains('cell cycle')]
ribgocc = df[df.GOCC.str.contains('ribosome')]

plt.scatter(mean_RNA_g1s, mean_protein_g1s, color='b', linestyle='dashed', marker='3'
plt.scatter(cellcgobp.mean_RNA_g1s, cellcgobp.mean_protein_g1s, color='g', linestyle=
            label='GOBP')
plt.scatter(ribgocc.mean_RNA_g1s, ribgocc.mean_protein_g1s, color='y', linestyle='dash
            label='GOCC')
plt.xlabel('mean_RNA_g1s')
plt.ylabel('mean_protein_g1s')
plt.legend(loc="best")
```

Out[32]:
```
<matplotlib.legend.Legend at 0x1ba35043080>
```



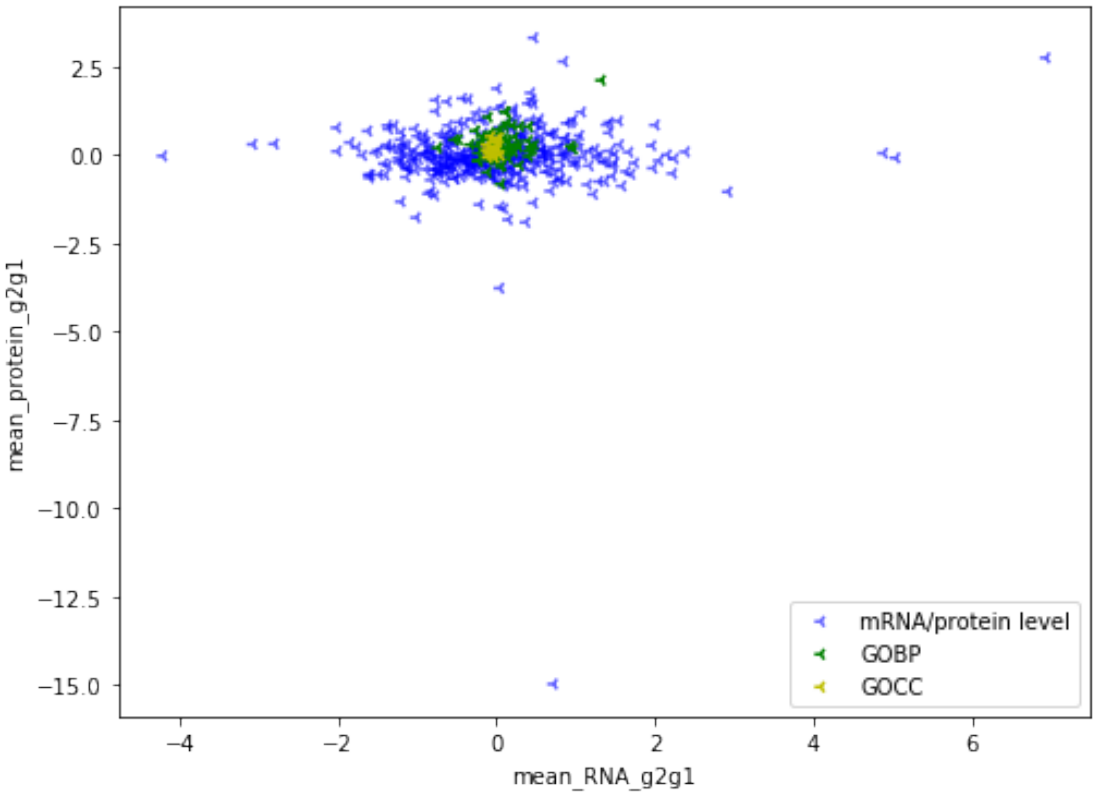
In [33]:
```
# Superimposing mean_RNA_sg2 vs mean_protein_sg2 with the elements of mean_RNA_sg2 an
# in their GOBP term and 'ribosome' in their GOCC term
plt.figure(figsize=(8,6))

cellcgobp = df[df.GOBP.str.contains('cell cycle')]
ribgocc = df[df.GOCC.str.contains('ribosome')]

plt.scatter(mean_RNA_sg2, mean_protein_sg2, color='b', linestyle='dashed', marker='3'
plt.scatter(cellcgobp.mean_RNA_sg2, cellcgobp.mean_protein_sg2, color='g', linestyle=
            label='GOBP')
plt.scatter(ribgocc.mean_RNA_sg2, ribgocc.mean_protein_sg2, color='y', linestyle='dash
            label='GOCC')
plt.xlabel('mean_RNA_sg2')
plt.ylabel('mean_protein_sg2')
plt.legend(loc="lower right")
```

Out[33]: <matplotlib.legend.Legend at 0x1ba36087710>



In [34]:
```
# Superimposing mean_RNA_g2g1 vs mean_protein_g2g1 with the elements of mean_RNA_g2g1
# 'cell cycle' in their GOBP term and 'ribosome' in their GOCC term
plt.figure(figsize=(8,6))

cellcgobp = df[df.GOBP.str.contains('cell cycle')]
ribgocc = df[df.GOCC.str.contains('ribosome')]

plt.scatter(mean_RNA_g2g1, mean_protein_g2g1, color='b', linestyle='dashed', marker='
plt.scatter(cellcgobp.mean_RNA_g2g1, cellcgobp.mean_protein_g2g1, color='g', linestyle
            label='GOBP')
plt.scatter(ribgocc.mean_RNA_g2g1, ribgocc.mean_protein_g2g1, color='y', linestyle='da
            label='GOCC')
plt.xlabel('mean_RNA_g2g1')
plt.ylabel('mean_protein_g2g1')
plt.legend(loc="lower right")
```

Out[34]: <matplotlib.legend.Legend at 0x1ba360ede48>



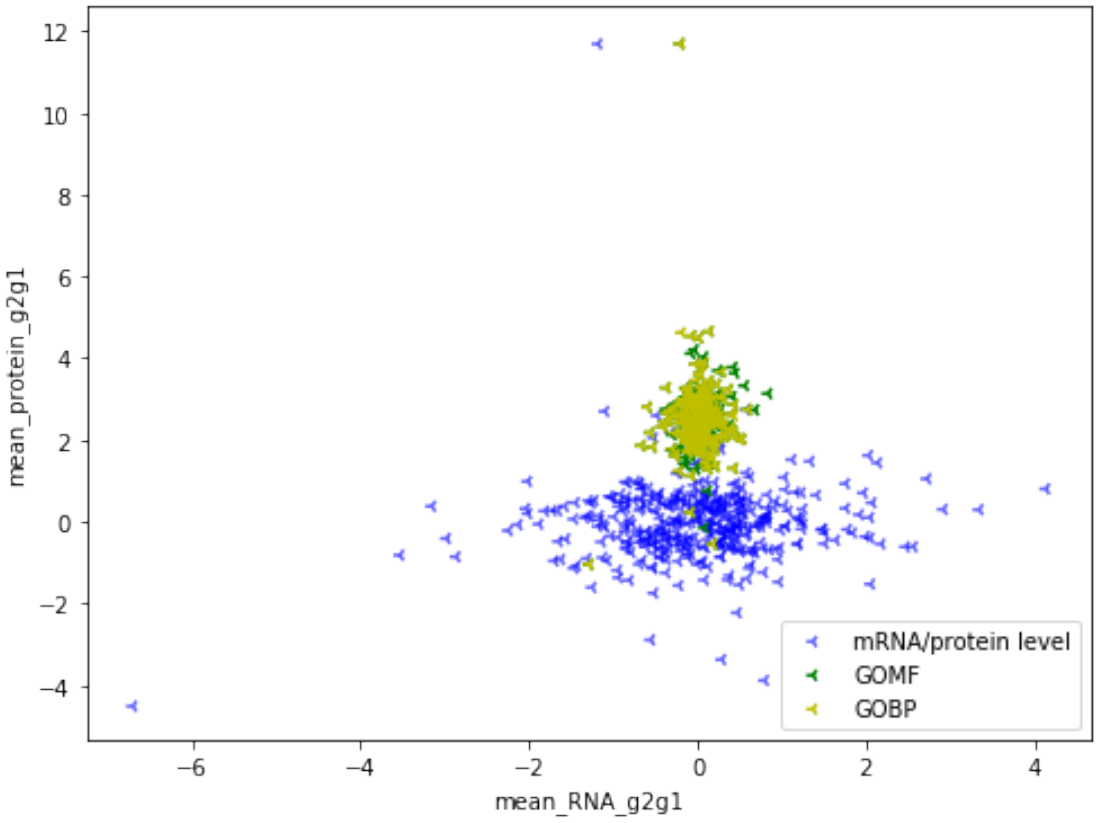
5.5 EXTRA: Finding clustering/correlations by using other terms in GOBP, GOMF or GOCC

In [35]:
```
# Superimposing mean_RNA_g1s vs mean_protein_g1s with the elements of mean_RNA_g1s an
# 'binding' in their GOMF term and 'cellular metabolic process' in their GOBP term
plt.figure(figsize=(8,6))

bindgomf = df[df.GOMF.str.contains('binding')]
cmpgobp = df[df.GOBP.str.contains('cellular metabolic process')]
print("Number of genes containging 'binding' in their GOMF term =", len(bindgomf))
print(bindgomf.GOCC.head())
print("\n")
print("Number of genes containging 'cellular metabolic process' in their GOMF term ="
print(cmpgobp.GOBP.head())

plt.scatter(mean_RNA_g1s, mean_protein_g1s, color='b', linestyle='dashed', marker='3'
plt.scatter(bindgomf.mean_RNA_g1s, bindgomf.mean_protein_g1s, color='g', linestyle='da
            label='GOMF')
plt.scatter(cmpgobp.mean_RNA_g1s, cmpgobp.mean_protein_g1s, color='y', linestyle='dash
            label='GOBP')
plt.xlabel('mean_RNA_g2g1')
```

```
        plt.ylabel('mean_protein_g2g1')
        plt.legend(loc="lower right")
```

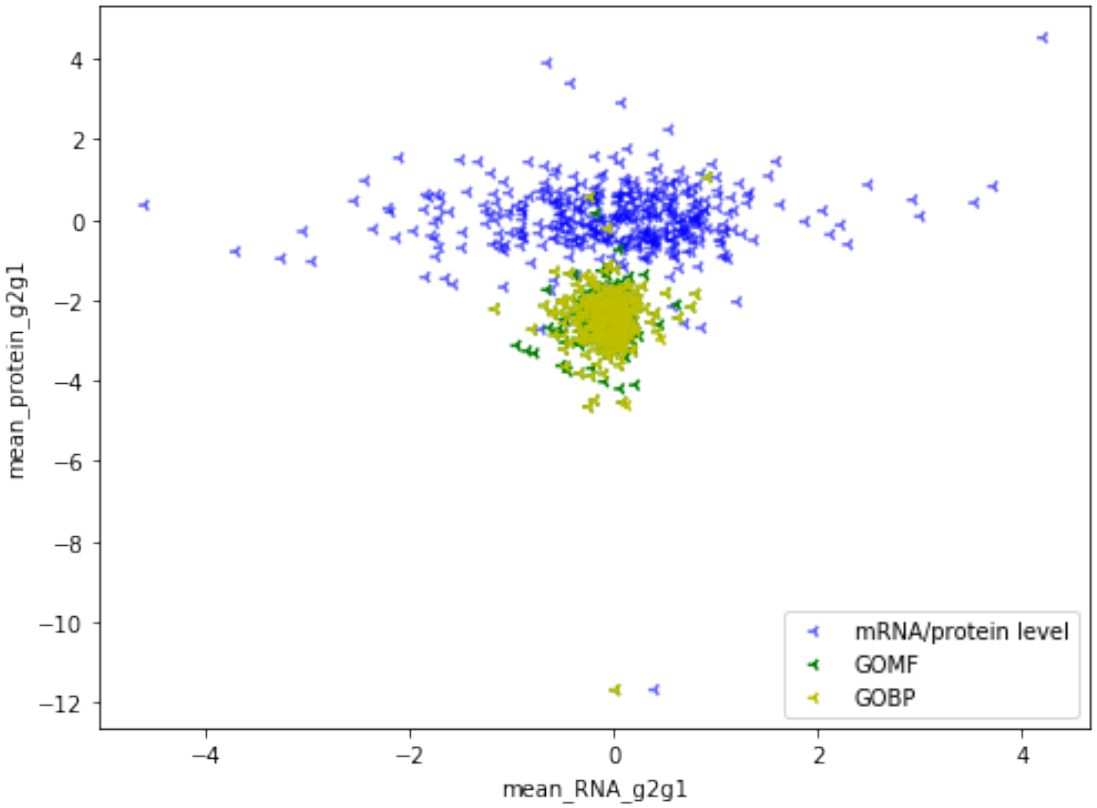
```
Number of genes containging 'binding' in their GOMF term = 366
1    apolipoprotein B mRNA editing enzyme complex;c...
2    cell part;extracellular region part;extracellu...
3                cell part;cytoplasm;intracellular part
5     axon;cell part;cell projection;neuron projection
6    cell part;cytoplasmic part;endoplasmic reticul...
Name: GOCC, dtype: object


Number of genes containging 'cellular metabolic process' in their GOMF term = 282
1     base conversion or substitution editing;biolog...
2     activation of MAPKK activity;activation of pro...
3     catabolic process;cellular catabolic process;c...
6     cellular macromolecule metabolic process;cellu...
10    biological regulation;cell cycle;cell cycle ch...
Name: GOBP, dtype: object
```

Out[35]: <matplotlib.legend.Legend at 0x1ba36162518>



In [36]:
```
# Superimposing mean_RNA_sg2 vs mean_protein_sg2 with the elements of mean_RNA_sg2 an
# 'binding' in their GOMF term and 'cellular metabolic process' in their GOBP term
plt.figure(figsize=(8,6))

plt.scatter(mean_RNA_sg2, mean_protein_sg2, color='b', linestyle='dashed', marker='3'
plt.scatter(bindgomf.mean_RNA_sg2, bindgomf.mean_protein_sg2, color='g', linestyle='da
            label='GOMF')
plt.scatter(cmpgobp.mean_RNA_sg2, cmpgobp.mean_protein_sg2, color='y', linestyle='dash
            label='GOBP')
plt.xlabel('mean_RNA_g2g1')
plt.ylabel('mean_protein_g2g1')
plt.legend(loc="lower right")
```

Out[36]: <matplotlib.legend.Legend at 0x1ba361d27b8>



In [37]:
```
# Superimposing mean_RNA_g2g1 vs mean_protein_g2g1 with the elements of mean_RNA_g2g1
# 'binding' in their GOMF term and 'cellular metabolic process' in their GOBP term
plt.figure(figsize=(8,6))

plt.scatter(mean_RNA_g2g1, mean_protein_g2g1, color='b', linestyle='dashed', marker='
plt.scatter(bindgomf.mean_RNA_g2g1, bindgomf.mean_protein_g2g1, color='g', linestyle=
```

```
                label='GOMF')
plt.scatter(cmpgobp.mean_RNA_g2g1, cmpgobp.mean_protein_g2g1, color='y', linestyle='da
                label='GOBP')
plt.xlabel('mean_RNA_g2g1')
plt.ylabel('mean_protein_g2g1')
plt.legend(loc="lower right")
```

Out[37]: <matplotlib.legend.Legend at 0x1ba36235ba8>