week1

February 19, 2021

[2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

[3]:
```
original_data = pd.read_excel("Cell-Cycle-Set.xlsx")
df = original_data.dropna()

#view the head so we can determine the column names and get a feel for the data
df.head()

#each of the columns detail the concentration of RNA/protein during one of the␣
 ↪3 phases

#G1 phase is first growth phase/post mitotic gap phase
#S phase is DNA replication phase
#G2 phase is growth phase

#GOBP indicates the biological process
#GOMF indicates the molecular function
#GOCC indicates the location in the cell
```

[3]:
```
  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…
```

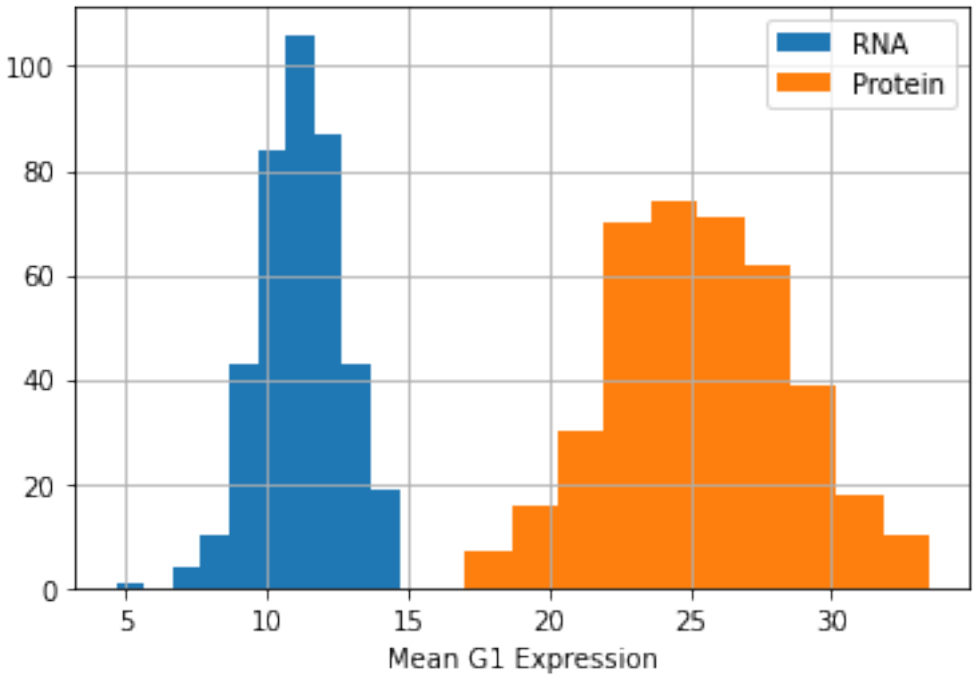
```
2  activation of MAPKK activity;activation of pro…
3  catabolic process;cellular catabolic process;c…
5  axon guidance;chemotaxis;locomotion;response t…
6  cellular macromolecule metabolic process;cellu…

                                                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…
```

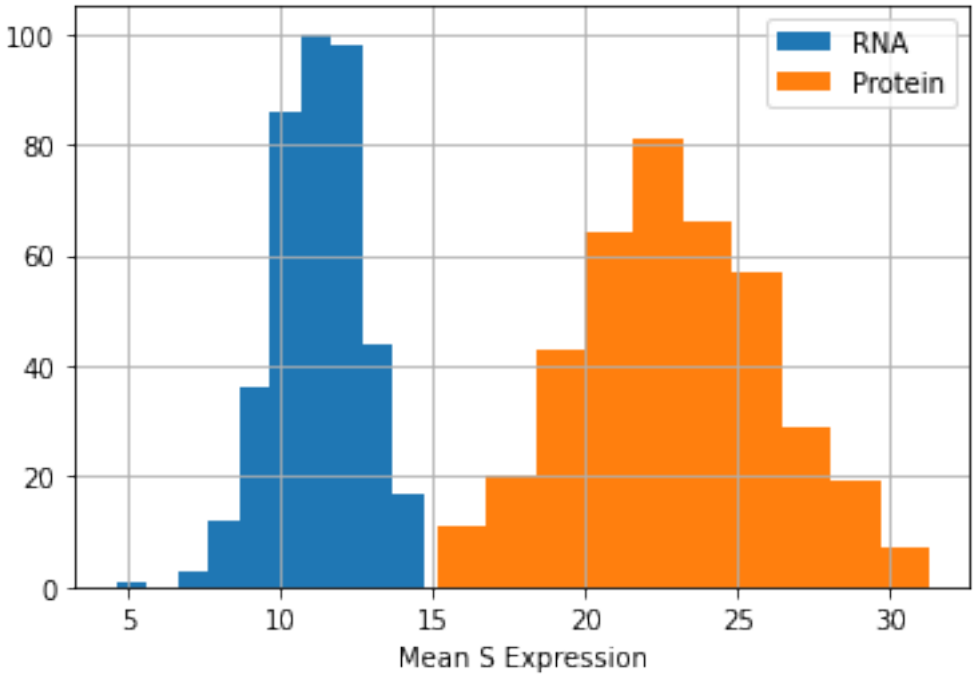
0.1 Task 1

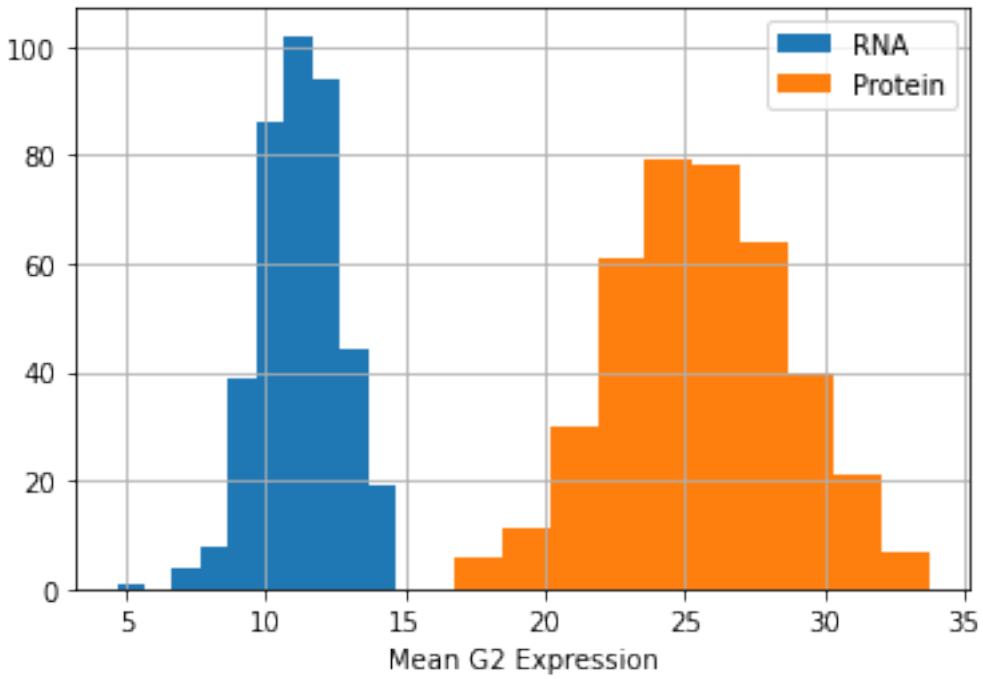
- Let's render histograms for the RNA and protein data in the G1, S and G2 phases:

[4]:
```
fig1, ax1 = plt.subplots()

#.hist draws histogram on a pandas series using matplotlib
#default bin size is 10
df.mean_RNA_G1.hist(ax=ax1, label='RNA', bins = 10)
df.mean_protein_G1.hist(ax=ax1, label='Protein', bins = 10)

#puts a colour coded legend of what each data set is
ax1.legend()

#set the label of the x axis
ax1.set_xlabel('Mean G1 Expression')

fig2, ax2 = plt.subplots()
df.mean_RNA_S.hist(ax=ax2, label='RNA')
df.mean_protein_S.hist(ax=ax2, label='Protein')
ax2.legend()
ax2.set_xlabel('Mean S Expression')

fig3, ax3 = plt.subplots()
df.mean_RNA_G2.hist(ax=ax3, label='RNA')
df.mean_protein_G2.hist(ax=ax3, label='Protein')
ax3.legend()
ax3.set_xlabel('Mean G2 Expression')
```

[4]: Text(0.5, 0, 'Mean G2 Expression')





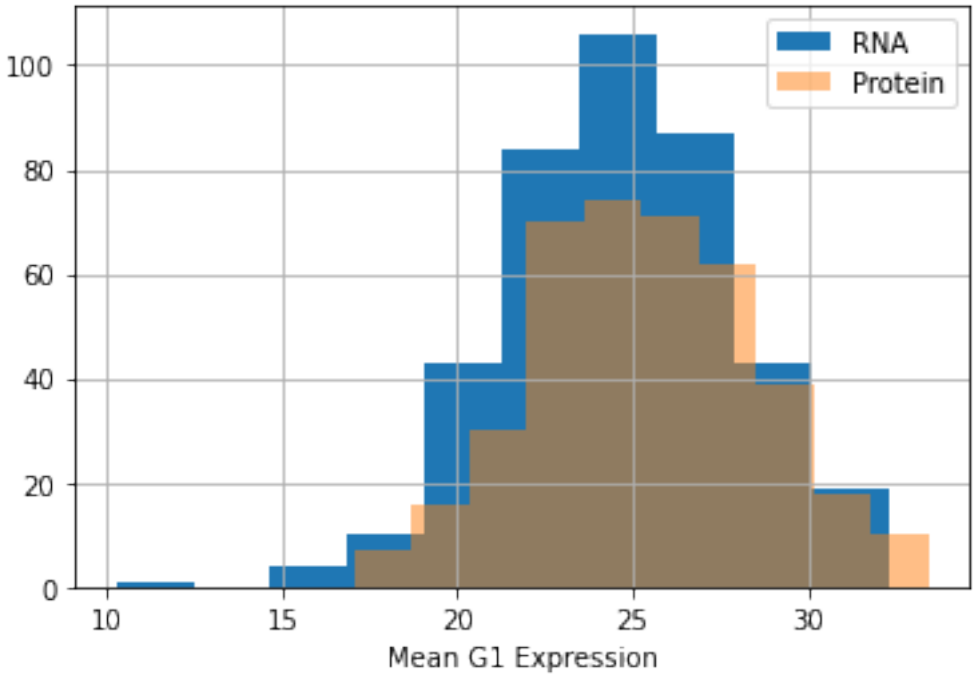


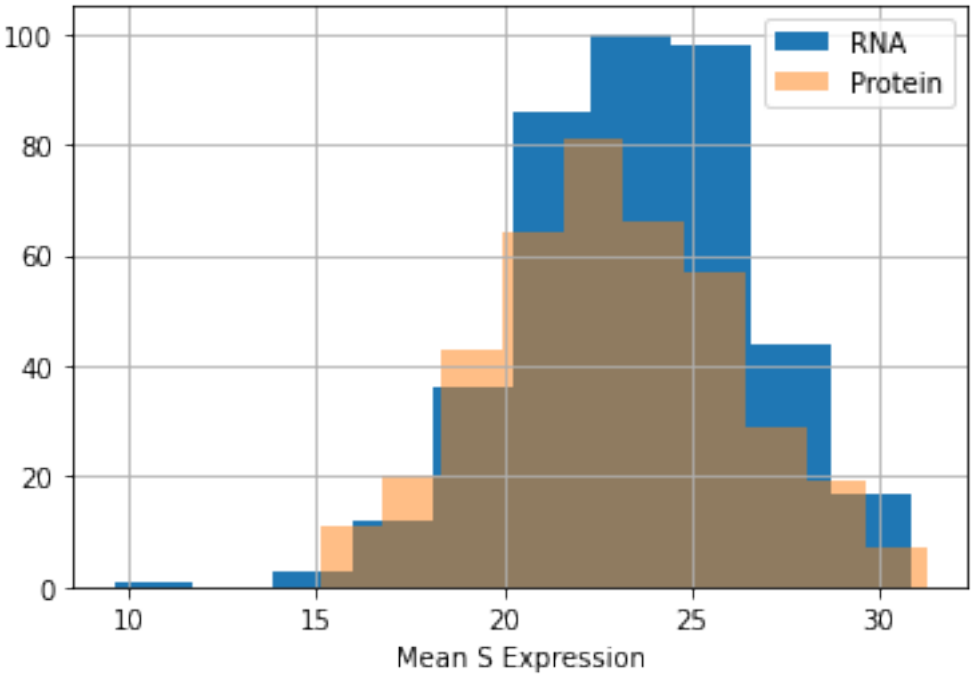
- From the above, straight away we can notice that in every phase there is a higher protein concentration than RNA. This could be due to the fact that a single mRNA strand used by ribosomes to create a protein could be used to create the same protein multiple times
- Now, if we apply a linear scalar to the RNA data from each phase and make the histogram reperesenting protein data slightly transparent in order to visualize how the distributions of the data compare, we can observe that the distributions of the RNA and protein data from each phase are similar apart from the S phase, where the RNA data is slightly more negatively skewed than in other phases.
- The negative skew of data in the S phase suggests there is a higher than normal concentration of RNA during this phase than in the other 2 phases
- See below for the scaled data:

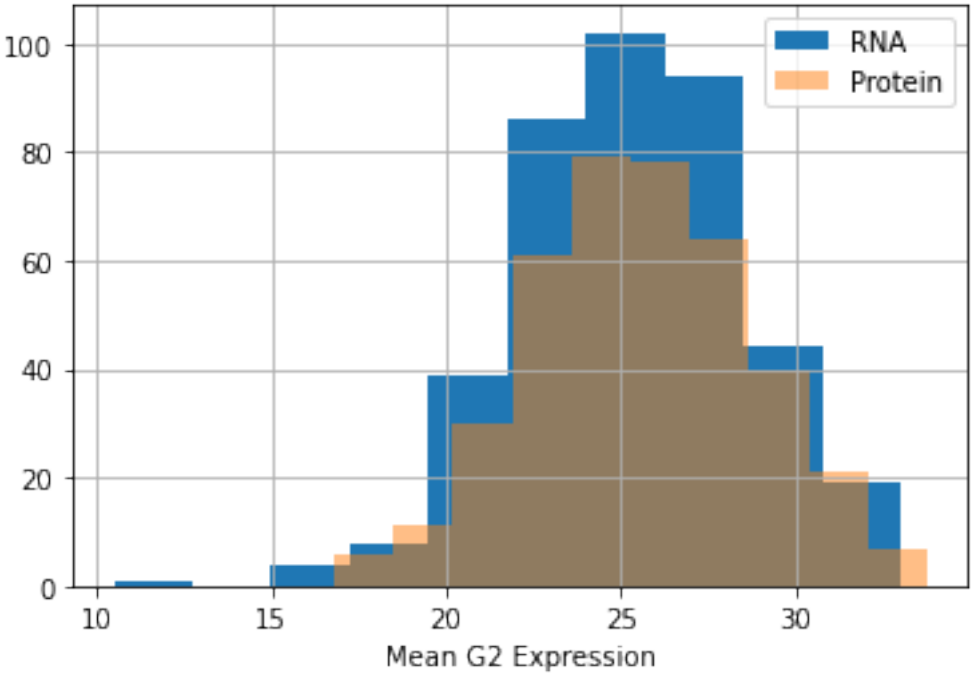
[5]:
```
fig1, ax1 = plt.subplots()
df.mean_RNA_G1.map(lambda x: x * 2.2).hist(ax=ax1, label='RNA', bins = 10)
df.mean_protein_G1.hist(ax=ax1, label='Protein', bins = 10, alpha = 0.5)
ax1.legend()
ax1.set_xlabel('Mean G1 Expression')

fig2, ax2 = plt.subplots()
df.mean_RNA_S.map(lambda x: x * 2.1).hist(ax=ax2, label='RNA')
df.mean_protein_S.hist(ax=ax2, label='Protein', alpha = 0.5)
ax2.legend()
ax2.set_xlabel('Mean S Expression')

fig3, ax3 = plt.subplots()
```

```
df.mean_RNA_G2.map(lambda x: x * 2.25).hist(ax=ax3, label='RNA')
df.mean_protein_G2.hist(ax=ax3, label='Protein', alpha = 0.5)
ax3.legend()
ax3.set_xlabel('Mean G2 Expression')
```

[5]: Text(0.5, 0, 'Mean G2 Expression')




RNA
Protein
100
80
60
40
20
0
10
15
20
25
30
Mean S Expression


RNA
Protein
100
80
60
40
20
0
10
15
20
25
30
Mean G2 Expression

0.2 Task 2

- We can see that when we look at the pairwise correlations accurate to 3 significant figures, there is only at most a difference of 0.01 in the spearmans rank score between any two phases in the pairwise correlation between RNA and protein concentration, hence the change in timestep has little effect on the relationship between the concentration amounts of RNA and protein

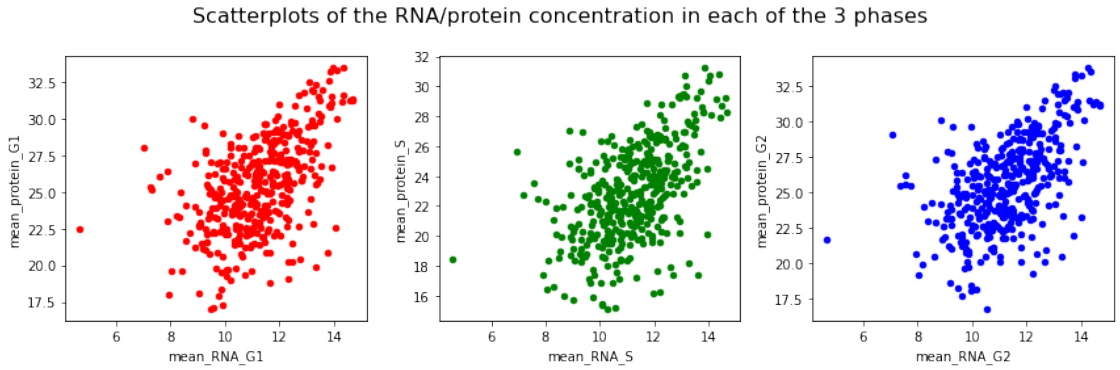
[6]:
```
#pairwise correlation

print("G1 correlation:", "%0.3f" % df.mean_RNA_G1.corr(df.mean_protein_G1,␣
 ↪method = "spearman"))
print("S correlation:", "%0.3f" % df.mean_RNA_S.corr(df.mean_protein_S, method␣
 ↪= "spearman"))
print("G2 correlation:", "%0.3f" % df.mean_RNA_G2.corr(df.mean_protein_G2,␣
 ↪method = "spearman"))
```

```
G1 correlation: 0.513
S correlation: 0.523
G2 correlation: 0.526
```

0.3 Task 3

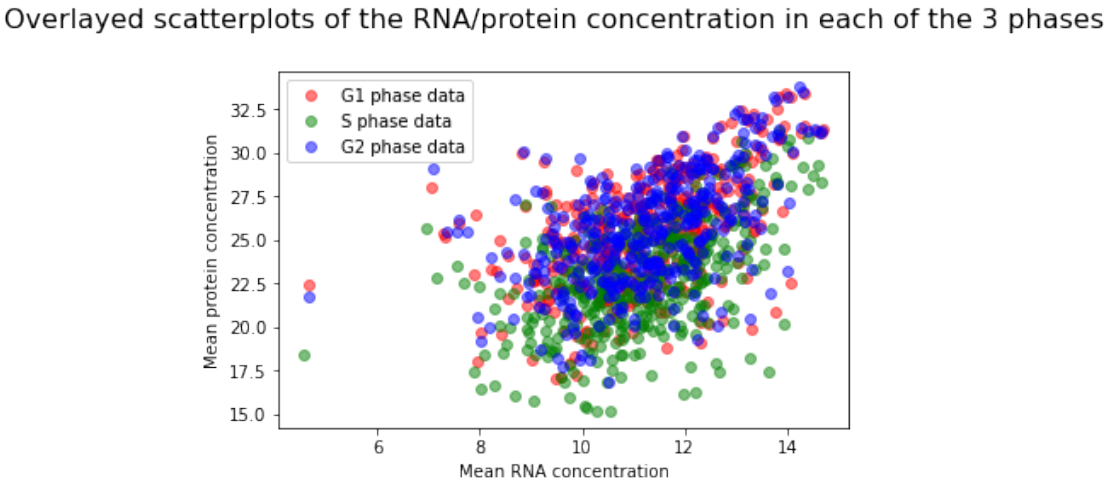
- Let's generate scatterplots of the RNA vs protein concentration for each stage:

[7]:
```
multiFig, multiAx = plt.subplots (ncols = 3, figsize= (12, 4))

df.plot.scatter("mean_RNA_G1", "mean_protein_G1", color = "r", ax = multiAx[0])
df.plot.scatter("mean_RNA_S", "mean_protein_S", color = "g", ax = multiAx[1])
df.plot.scatter("mean_RNA_G2", "mean_protein_G2", color = "b", ax = multiAx[2])

#make the plots spaced apart
multiFig.tight_layout()

multiFig.suptitle("Scatterplots of the RNA/protein concentration in each of the␣
 ↪3 phases", fontsize = 16)

#Make the title not overlapping
multiFig.subplots_adjust(top=0.85)
```



- Now, as before let's observe how the data look when overlayed onto one another:

[8]:
```
#scatterFig, scatterAxes = plt.subplots(ncols = 3, figsize = (12,4))

overlayFig, overlayAx = plt.subplots()

overlayAx.scatter(x = df.mean_RNA_G1, y = df.mean_protein_G1, color = "r",␣
↪alpha = 0.5, label = "G1 phase data")
overlayAx.scatter(x = df.mean_RNA_S, y = df.mean_protein_S, color = "g", alpha␣
↪= 0.5, label = "S phase data")
overlayAx.scatter(x = df.mean_RNA_G2, y = df.mean_protein_G2, color = "b",␣
↪alpha = 0.5, label = "G2 phase data")
overlayAx.legend()
overlayAx.set_xlabel("Mean RNA concentration")
overlayAx.set_ylabel("Mean protein concentration")

overlayFig.suptitle("Overlayed scatterplots of the RNA/protein concentration in␣
↪each of the 3 phases", fontsize = 16)
overlayFig.subplots_adjust(top=0.85)
```



- As before, we can see that the distribution of data from each phase is fairly similar
- Now, let's use scikit-learn to train a linear regression models for each of the data sets from each phase:

[9]:
```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score
```

- We can see that the data in question is a Pandas Series object of floats:

[10]:
```
df.mean_RNA_G1
```

[10]:
```
1      10.330107
2      12.321340
3      10.827333
5      10.845517
6       9.042438
         ...
494    11.115773
495    11.251870
496    13.013263
497     9.048456
498    13.525467
Name: mean_RNA_G1, Length: 397, dtype: float64
```

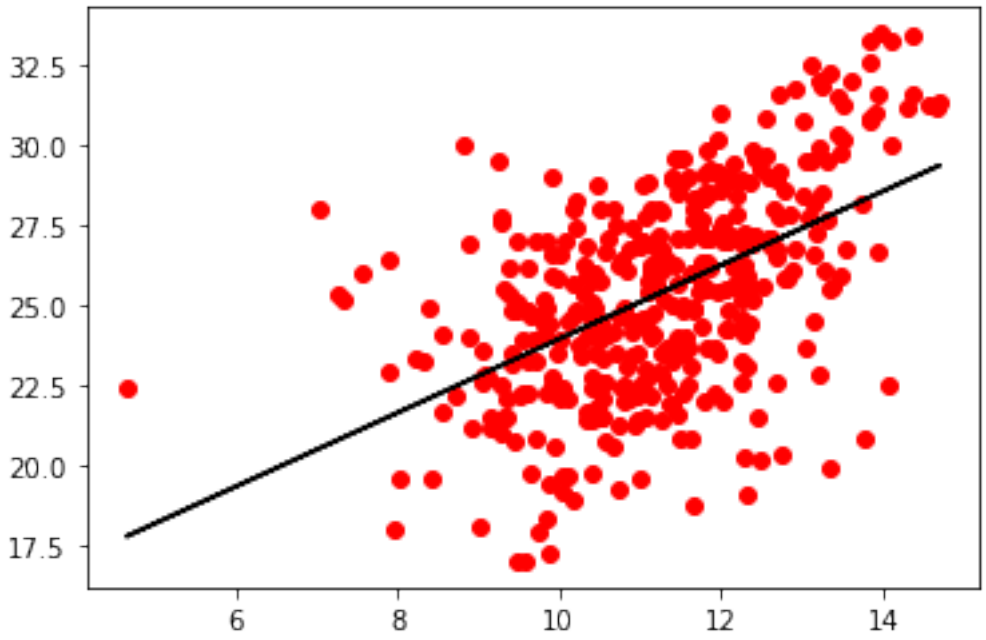
- We know that the data is actually just a 1D list, so let's flatten the pandas series into a numpy array which is suitable for feeding to a scikit-learn model.
- Let's train 3 linear regression models on the data for each cell stage which predicts protein concentration from RNA concentration, and plot the "line of best fit" that is found by the model, and also print the coefficient of determination score that is found between the predicted protein concentrations generated by the model and the actual protein concentrations:

[11]:
```
def process_column(col):
    return col.to_numpy().reshape(-1,1)
```

[12]:
```
G1_rna, G1_protein = process_column(df.mean_RNA_G1), process_column(df.
↪mean_protein_G1)
G1_model = LinearRegression().fit(G1_rna, G1_protein)

G1_protein_predictions = G1_model.predict(G1_rna)

plt.scatter(G1_rna, G1_protein, color = "r")
plt.plot(G1_rna, G1_protein_predictions, color = "k")

g1_score = r2_score(G1_protein, G1_protein_predictions)
```

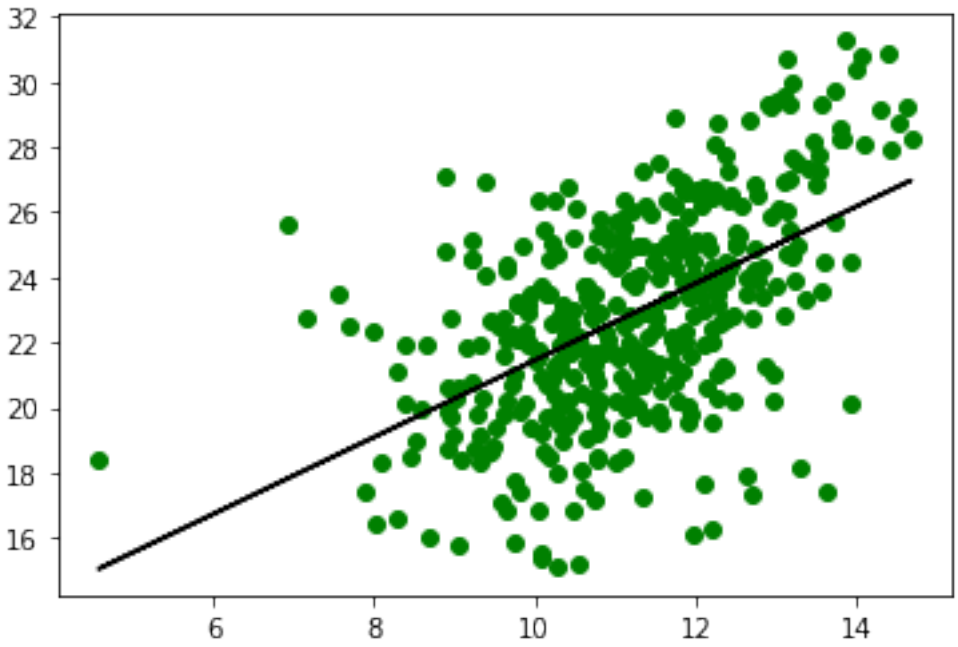
```
print(g1_score)
```

```
0.2731711059314553
```



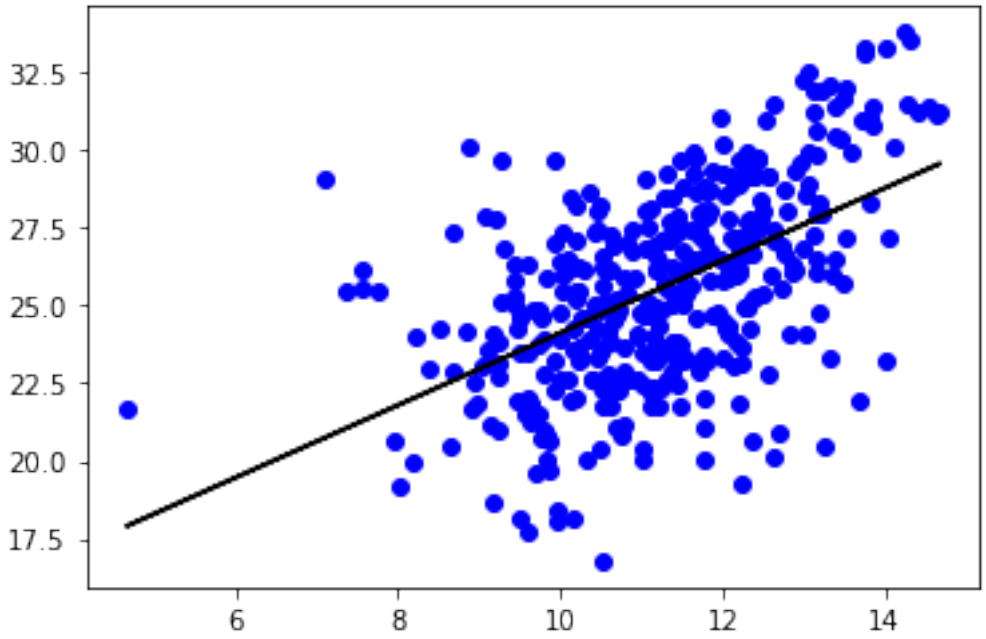
[13]:
```
S_rna, S_protein = process_column(df.mean_RNA_S), process_column(df.
 ↪mean_protein_S)
S_model = LinearRegression().fit(S_rna, S_protein)

S_protein_predictions = S_model.predict(S_rna)

plt.scatter(S_rna, S_protein, color = "g")
plt.plot(S_rna, S_protein_predictions, color = "k")

s_score = r2_score(S_protein, S_protein_predictions)

print(s_score)
```

```
0.28750000422102284
```



[14]:
```
G2_rna, G2_protein = process_column(df.mean_RNA_G2), process_column(df.
 ↪mean_protein_G2)
G2_model = LinearRegression().fit(G2_rna, G2_protein)

G2_protein_predictions = G2_model.predict(G2_rna)

plt.scatter(G2_rna, G2_protein, color = "b")
plt.plot(G2_rna, G2_protein_predictions, color = "k")

g2_score = r2_score(G2_protein, G2_protein_predictions)

print(g2_score)
```

```
0.28362549895654476
```



- The coefficient of determination scores aren't huge, but that is to be expected as a score near to 1 would indicate a high correlation and would mean the data are in a near perfect straight line shape which they are not, they are fairly spread out around the "line of best fit"
- Visually looking at the data above we can see that most of the data points with RNA concentration values of less than 8 or so are sort of outliers and are not representative of the general distribution of the data and are perhaps "skewing" our "line of best fit", so let's see if filtering out these data points can improve our coefficient of determination score at all:

[15]:
```
G1_filtered = df[df.mean_RNA_G1 > 8]
S_filtered = df[df.mean_RNA_S > 8]
G2_filtered = df[df.mean_RNA_G2 > 8]
```

[16]:
```
G1_rna, G1_protein = process_column(G1_filtered.mean_RNA_G1),␣
↪process_column(G1_filtered.mean_protein_G1)
G1_model = LinearRegression().fit(G1_rna, G1_protein)
G1_protein_predictions = G1_model.predict(G1_rna)
print("G1 filtered score:", r2_score(G1_protein, G1_protein_predictions))
print("G1 original score:", g1_score)
```

```
G1 filtered score: 0.3089380822670228
G1 original score: 0.2731711059314553
```

[17]:
```
S_rna, S_protein = process_column(S_filtered.mean_RNA_S),␣
↪process_column(S_filtered.mean_protein_S)
S_model = LinearRegression().fit(S_rna, S_protein)
```

```
S_protein_predictions = S_model.predict(S_rna)
print("S filtered score:",r2_score(S_protein, S_protein_predictions))
print("S original score:", s_score)
```

```
S filtered score: 0.318061690581281
S original score: 0.28750000422102284
```

[18]:
```
G2_rna, G2_protein = process_column(G2_filtered.mean_RNA_G2),␣
↪process_column(G2_filtered.mean_protein_G2)
G2_model = LinearRegression().fit(G2_rna, G2_protein)

G2_protein_predictions = G2_model.predict(G2_rna)
print("G2 filtered score:",r2_score(G2_protein, G2_protein_predictions))
print("G2 original score:", g2_score)
```

```
G2 filtered score: 0.32148617805391766
G2 original score: 0.28362549895654476
```

- By removing outliers, we can see that the predictions of protein concentration now have a slightly higher correlation with the actual reference values
- However, as mentioned before the correlation between the predictions and actual values are not strong enough for a linear model alone to be used for accurate prediction of protein concentration from RNA concentration. Perhaps a deep learning approach may produce more accurate predictions

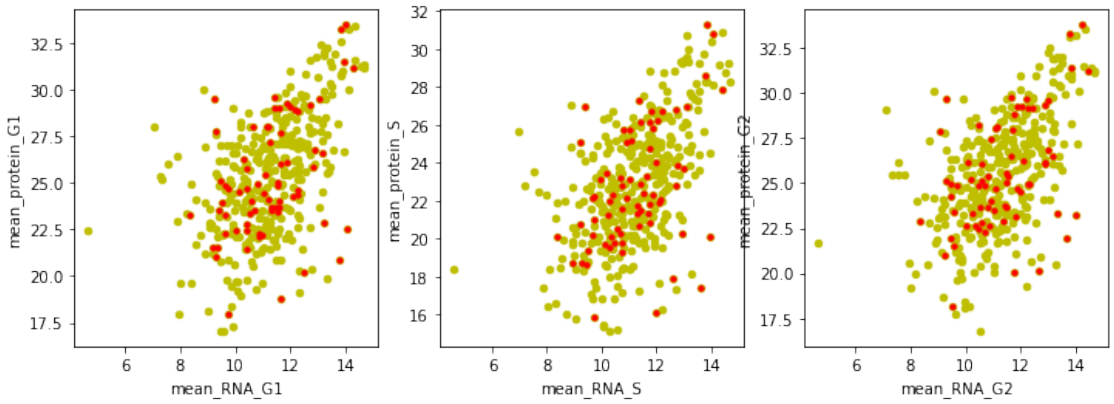
1 Week 2

1.1 Task 1

First plot the points where the string "cell cycle" is contained in the GOBP term for each phase G1, G2 and S and colour it red so we can clearly see how the data differs from the unfiltered data:

[19]:
```
gobp = df[df.GOBP.str.contains('cell cycle')]
fig4,ax4 = plt.subplots(ncols=3, figsize=(12,4))
df.plot.scatter('mean_RNA_G1', 'mean_protein_G1', ax=ax4[0], color='y')
ax4[0].scatter(gobp.mean_RNA_G1, gobp.mean_protein_G1, color='r', s=10.)
df.plot.scatter('mean_RNA_S', 'mean_protein_S', ax=ax4[1], color='y')
ax4[1].scatter(gobp.mean_RNA_S, gobp.mean_protein_S, color='r', s=10.)
df.plot.scatter('mean_RNA_G2', 'mean_protein_G2', ax=ax4[2], color='y')
ax4[2].scatter(gobp.mean_RNA_G2, gobp.mean_protein_G2, color='r', s=10.)
print(len(gobp))
```

```
71
```



Now lets see if the correlation is stronger or weaker for any of the 3 phases, lets remind ourselves of the pairwise correlation between RNA concentration and protein concentration for each of the 3 phases in the original unfiltered data:

[20]:
```
print("G1 unfiltered data correlation:", "%0.3f" % df.mean_RNA_G1.corr(df.
 ↪mean_protein_G1, method = "spearman"))
print("S unfiltered data correlation:", "%0.3f" % df.mean_RNA_S.corr(df.
 ↪mean_protein_S, method = "spearman"))
print("G2 unfiltered data correlation:", "%0.3f" % df.mean_RNA_G2.corr(df.
 ↪mean_protein_G2, method = "spearman"))
```

```
G1 unfiltered data correlation: 0.513
S unfiltered data correlation: 0.523
G2 unfiltered data correlation: 0.526
```

Now let's see how the pairwise correlation is in only the genes that contain the string "cell cycle" in their GOBP column

[21]:
```
print("cell cycle GOBP G1 data correlation:", "%0.3f" % gobp.mean_RNA_G1.
 ↪corr(gobp.mean_protein_G1, method = "spearman"))
print("cell cycle GOBP S data correlation:", "%0.3f" % gobp.mean_RNA_S.
 ↪corr(gobp.mean_protein_S, method = "spearman"))
print("cell cycle GOBP G2 data correlation:", "%0.3f" % gobp.mean_RNA_G2.
 ↪corr(gobp.mean_protein_G2, method = "spearman"))
```

```
cell cycle GOBP G1 data correlation: 0.393
cell cycle GOBP S data correlation: 0.420
cell cycle GOBP G2 data correlation: 0.403
```

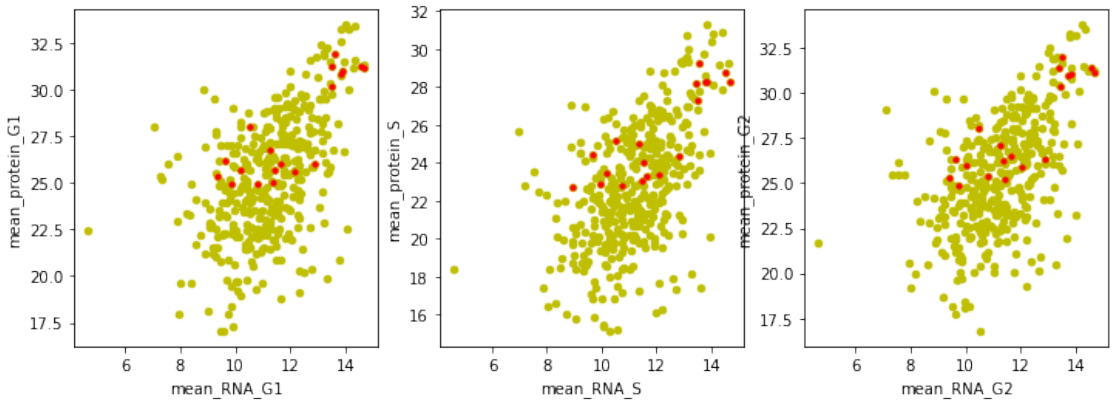
So, we can see that the correlation is weaker in the set of data that only includes genes that contain the string "cell cycle" in their GOBP field.

1.2 Task 2

Now let's do the same for only those data instances that contain the substring "ribosome" in their GOCC term:

[22]:
```
gocc = df[df.GOCC.str.contains('ribosome')]
fig5,ax5 = plt.subplots(ncols=3, figsize=(12,4))
df.plot.scatter('mean_RNA_G1', 'mean_protein_G1', ax=ax5[0], color='y')
ax5[0].scatter(gocc.mean_RNA_G1, gocc.mean_protein_G1, color='r', s=10.)
df.plot.scatter('mean_RNA_S', 'mean_protein_S', ax=ax5[1], color='y')
ax5[1].scatter(gocc.mean_RNA_S, gocc.mean_protein_S, color='r', s=10.)
df.plot.scatter('mean_RNA_G2', 'mean_protein_G2', ax=ax5[2], color='y')
ax5[2].scatter(gocc.mean_RNA_G2, gocc.mean_protein_G2, color='r', s=10.)
```

[22]:
```
<matplotlib.collections.PathCollection at 0x7fd81e1079a0>
```



And now let's compare the correllations again:

[23]:
```
print("G1 unfiltered data correlation:", "%0.3f" % df.mean_RNA_G1.corr(df.
 ↪mean_protein_G1, method = "spearman"))
print("S unfiltered data correlation:", "%0.3f" % df.mean_RNA_S.corr(df.
 ↪mean_protein_S, method = "spearman"))
print("G2 unfiltered data correlation:", "%0.3f" % df.mean_RNA_G2.corr(df.
 ↪mean_protein_G2, method = "spearman"))
print("specific GOCC G1 data correlation:", "%0.3f" % gocc.mean_RNA_G1.
 ↪corr(gocc.mean_protein_G1, method = "spearman"))
print("specific GOCC S data correlation:", "%0.3f" % gocc.mean_RNA_S.corr(gocc.
 ↪mean_protein_S, method = "spearman"))
print("specific GOCC G2 data correlation:", "%0.3f" % gocc.mean_RNA_G2.
 ↪corr(gocc.mean_protein_G2, method = "spearman"))
```

```
G1 unfiltered data correlation: 0.513
S unfiltered data correlation: 0.523
G2 unfiltered data correlation: 0.526
```

```
specific GOCC G1 data correlation: 0.732
specific GOCC S data correlation: 0.770
specific GOCC G2 data correlation: 0.744
```

- We can see that there is a greater amount of postive corelation between protein and RNA concentration in only those instances of data that contain the substring "ribosome" in their GOCC term compared to both the original unfiltered data and also the set consisting of only those instances with "cell cycle" in their gobp term
- This sort of makes sense as the GOCC value is "ribosome" meaning the location where the concentration data is being recorded is around the ribosome, logically a ribosome is bound to have a greater concentration of RNA around it as there is an abundance of both RNA in mRNA strands being delivered to ribosomes to synthesize proteins and also the RNA that makes up the actual ribosomes themselves. A greater correlation indicates that the RNA concentration levels are generally higher here than in other data previously in the notebook, due to the fact that the other data have smaller correlations whilst the RNA concentration was generally smaller than protein concentration, meaning there has to have been an increase in RNA concentration for the correlation to go up

1.3 Task 3

Now let's count the number of occurrences of every GOBP term across all genes:

[24]:
```
print(df.GOBP.str.split(';',expand=True).stack().value_counts()[:50])
```

```
cellular process                                                  377
metabolic process                                                 273
cellular metabolic process                                        260
primary metabolic process                                         255
biological regulation                                             236
regulation of biological process                                  225
regulation of cellular process                                    211
macromolecule metabolic process                                   211
cellular macromolecule metabolic process                          201
nitrogen compound metabolic process                               167
cellular nitrogen compound metabolic process                      166
nucleobase-containing compound metabolic process                  158
response to stimulus                                              148
cellular component organization or biogenesis                     145
nucleic acid metabolic process                                    144
cellular component organization                                  142
regulation of metabolic process                                   141
regulation of cellular metabolic process                          130
regulation of primary metabolic process                           125
regulation of macromolecule metabolic process                     123
cellular component organization or biogenesis at cellular level   122
cellular component organization at cellular level                 119
RNA metabolic process                                             119
biosynthetic process                                              118
cellular biosynthetic process                                     117
```

```
cellular response to stimulus                                    116
protein metabolic process                                        104
regulation of nitrogen compound metabolic process                100
establishment of localization                                     98
transport                                                         95
regulation of nucleobase-containing compound metabolic process    95
regulation of cellular biosynthetic process                       92
regulation of gene expression                                     92
regulation of biosynthetic process                                92
positive regulation of biological process                         90
organelle organization                                            87
regulation of macromolecule biosynthetic process                  86
positive regulation of cellular process                           85
regulation of cellular macromolecule biosynthetic process         85
signal transduction                                               84
catabolic process                                                 83
cellular protein metabolic process                                82
negative regulation of biological process                         81
regulation of RNA metabolic process                               79
negative regulation of cellular process                           75
macromolecule biosynthetic process                                73
cellular macromolecule biosynthetic process                       73
small molecule metabolic process                                  72
regulation of transcription, DNA-dependent                        72
cellular catabolic process                                        71
dtype: int64
```

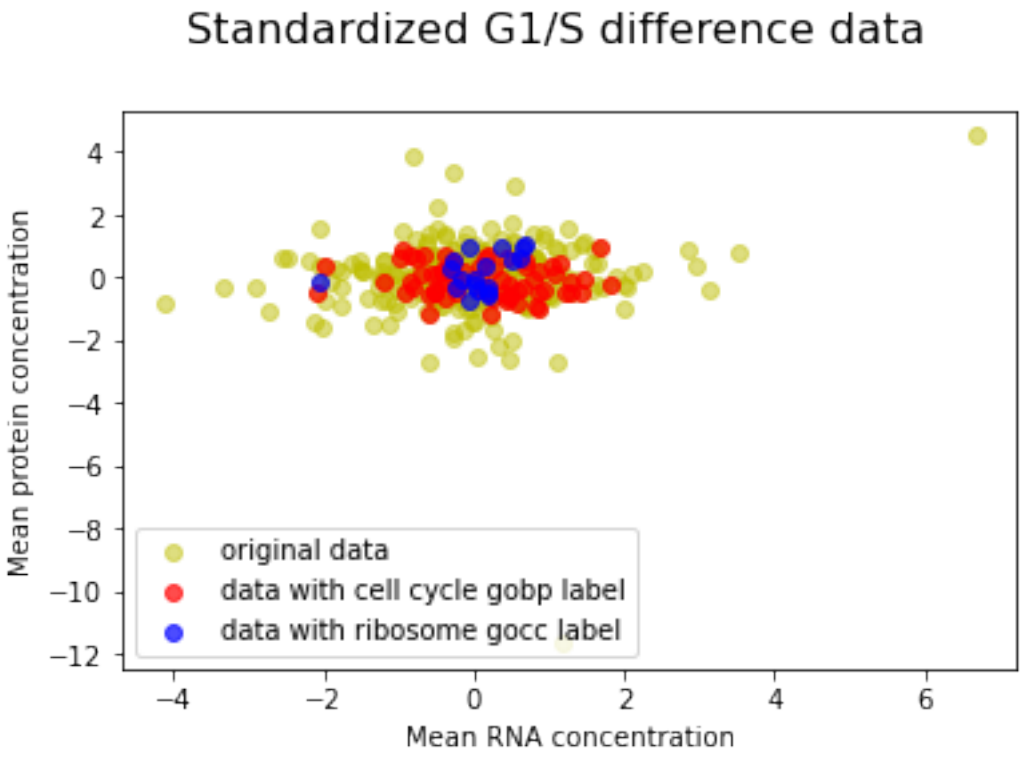
- One difficulty that arises when using these terms is that some are quite similar e.g. "cellular process", "metabolic process" and "cellular metabolic process" so perhaps they could be hard to distinguish

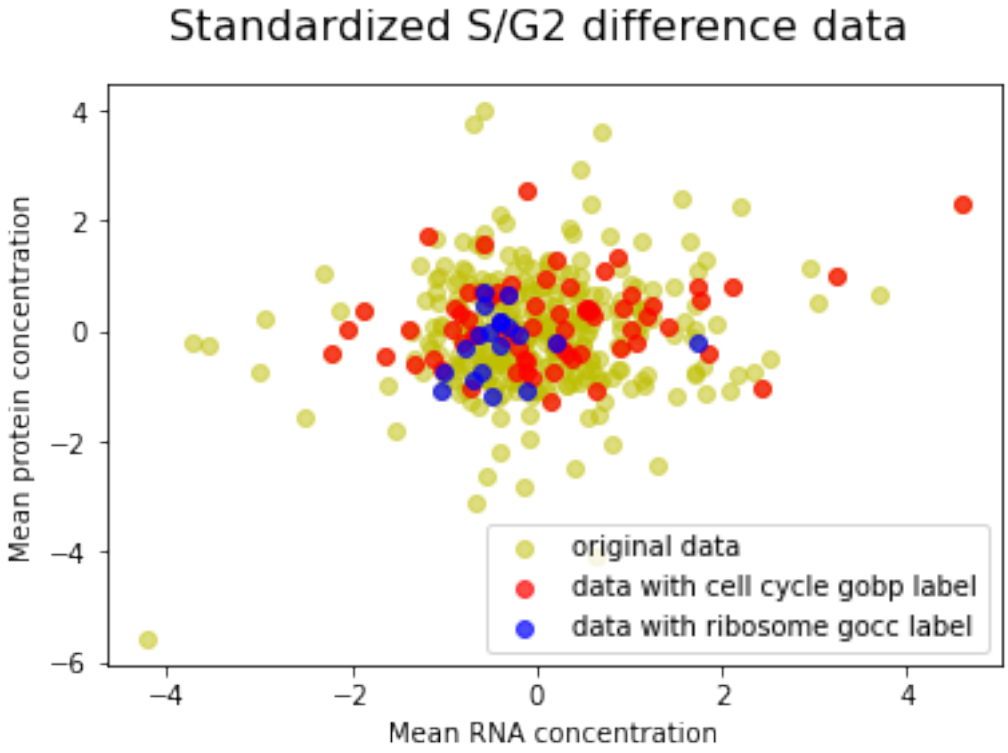
1.4 Task 4

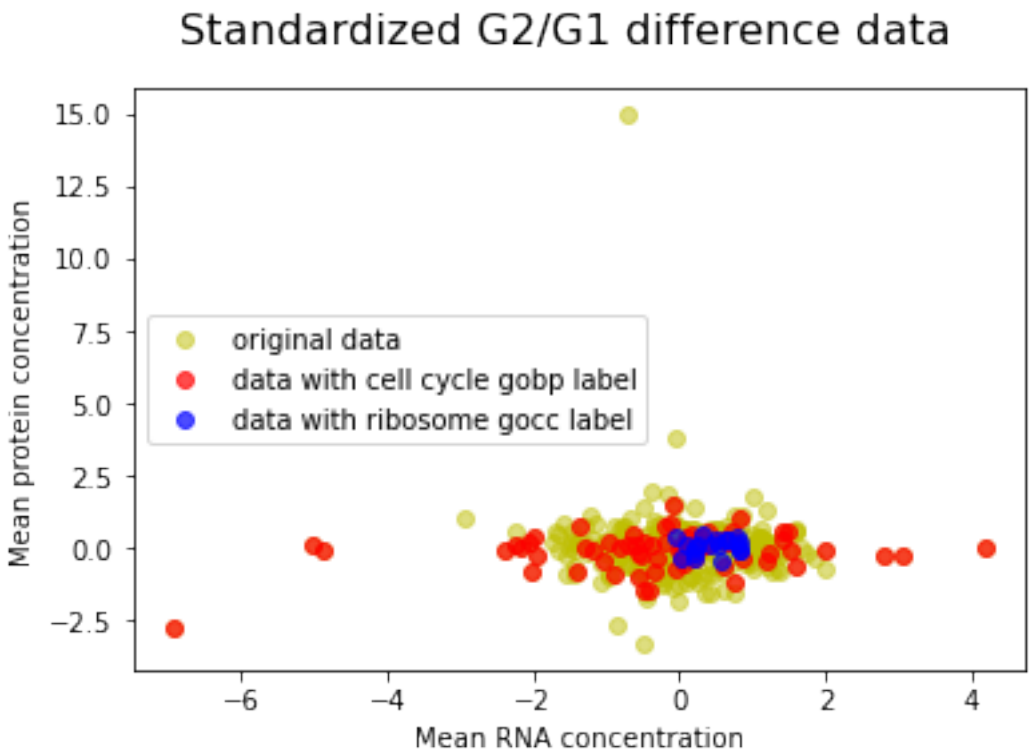
[27]:
```
#calculate the difference in mean RNA levels and mean protein levels
#between each stage by calculating the difference in RNA levels and the
#difference in protein levels for all possible pairs of stages
df['mean_RNA_g1s'] = (df.mean_RNA_S - df.mean_RNA_G1)
df['mean_RNA_sg2'] = (df.mean_RNA_G2 - df.mean_RNA_S)
df['mean_RNA_g2g1'] = (df.mean_RNA_G1 - df.mean_RNA_G2)
df['mean_protein_g1s'] = (df.mean_protein_S - df.mean_protein_G1)
df['mean_protein_sg2'] = (df.mean_protein_G2 - df.mean_protein_S)
df['mean_protein_g2g1'] = (df.mean_protein_G1 - df.mean_protein_G2)

# standardise
df.iloc[:,-6:] = (df.iloc[:,-6:] - df.iloc[:,-6:].mean(axis=0)) / df.iloc[:,-6:
 ↪].std(axis=0)

gobp = df[df.GOBP.str.contains('cell cycle')]
```

```
gocc = df[df.GOCC.str.contains('ribosome')]

g1sPlot, g1sAx = plt.subplots()
g1sAx.scatter(df.mean_RNA_g1s, df.mean_protein_g1s, color = 'y', alpha = 0.5,␣
↪label = "original data")
g1sAx.scatter(gobp.mean_RNA_g1s, gobp.mean_protein_g1s, color = 'r', alpha = 0.
↪7, label = "data with cell cycle gobp label")
g1sAx.scatter(gocc.mean_RNA_g1s, gocc.mean_protein_g1s, color = 'b', alpha = 0.
↪7, label = "data with ribosome gocc label")
g1sAx.legend()
g1sAx.set_xlabel("Mean RNA concentration")
g1sAx.set_ylabel("Mean protein concentration")

g1sPlot.suptitle("Standardized G1/S difference data", fontsize = 16)
g1sPlot.subplots_adjust(top=0.85)

sg2Plot, sg2Ax = plt.subplots()
sg2Ax.scatter(df.mean_RNA_sg2, df.mean_protein_sg2, color = 'y', alpha = 0.5,␣
↪label = "original data")
sg2Ax.scatter(gobp.mean_RNA_sg2, gobp.mean_protein_sg2, color = 'r', alpha = 0.
↪7, label = "data with cell cycle gobp label")
sg2Ax.scatter(gocc.mean_RNA_sg2, gocc.mean_protein_sg2, color = 'b', alpha = 0.
↪7, label = "data with ribosome gocc label")
sg2Ax.legend()
sg2Ax.set_xlabel("Mean RNA concentration")
sg2Ax.set_ylabel("Mean protein concentration")

sg2Plot.suptitle("Standardized S/G2 difference data", fontsize = 16)
#sg2Plot.subplot_adjust(top = 0.85)

g2g1Plot, g2g1Ax = plt.subplots()
g2g1Ax.scatter(df.mean_RNA_g2g1, df.mean_protein_g2g1, color = 'y', alpha = 0.
↪5, label = "original data")
g2g1Ax.scatter(gobp.mean_RNA_g2g1, gobp.mean_protein_g2g1, color = 'r', alpha =␣
↪0.7, label = "data with cell cycle gobp label")
g2g1Ax.scatter(gocc.mean_RNA_g2g1, gocc.mean_protein_g2g1, color = 'b', alpha =␣
↪0.7, label = "data with ribosome gocc label")
g2g1Ax.legend()
g2g1Ax.set_xlabel("Mean RNA concentration")
g2g1Ax.set_ylabel("Mean protein concentration")

g2g1Plot.suptitle("Standardized G2/G1 difference data", fontsize = 16)
#g2g1Plot.subplot_adjust(top = 0.85)
```

[27]: Text(0.5, 0.98, 'Standardized G2/G1 difference data')


Standardized G1/S difference data
Mean protein concentration
Mean RNA concentration
original data
data with cell cycle gobp label
data with ribosome gocc label


Standardized S/G2 difference data
Mean protein concentration
Mean RNA concentration
original data
data with cell cycle gobp label
data with ribosome gocc label



Interestingly from the graph we can see that the set of data points that contain "ribosome" in their gocc label have little change in protein or RNA concentration between any of the 3 phases, this could be explained by the fact that proteins are used to carry out a large majority of tasks in a cell, so the ribosomes need to continuously produce new proteins regardless of what cell cycle stage the cell is in, and obviously the ribosome constantly needs mRNA strands to be delivered to it in order to continute producing the proteins explaining for the little change in RNA concentration.