Linear Regression Assignment

In a watershed monitoring project, automated samplers are often used to collect water samples. However, these automated samplers may fail from time to time due to various instrumentation issues, such as power failure and damaged instruments. When the automated sampler failed to collect water samples, it leaves a data gap from a certain period of time.

Base flow nutrient concentration data are often linearly interpolated (or sometimes extrapolated) using nutrient data of the last and next sample collected. This approach cannot be applied to event flow nutrient and sediment (hereinafter refer as nutrient) data, especially phosphorus (P) and sediment (TSS), because of the large variations in nutrient concentration that can be caused by flow.

The objectived of this task are to:
1) develop a suitable model using existing dissolved reactive P (DRP), total P (TP), and TSS
2) predict the missing DRP, TP, TSS data using the model developed in (1)

In [99]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import statsmodels.formula.api as smf
import matplotlib.pyplot as plt
from scipy import stats
```

In [21]:
```
# import data from excel
data = pd.read_excel('JY_linear_reg_data.xlsx', sheet_name = 'Data')
data.head()
```

Out[21]:

	Sample date	Site	Flow (cms)	DRP (mg P/L)	TP (mg P/L)	TSS (mg/L)	VSS (mg/L)
0	2015-05-19	S11	0.028210	0.018	0.0015	22.333667	19.264296
1	2015-06-16	S11	0.189993	0.027	0.1770	1006.666667	892.000000
2	2015-06-30	S11	1.296460	0.203	1.2590	1338.666667	1194.666667
3	2015-08-18	S11	0.131722	0.050	1.3280	2026.000000	1812.000000
4	2015-08-25	S11	0.120204	0.010	3.1180	1722.000000	1560.000000

In [22]:
```
# drop 'VSS' column, rename the columns into shorter names

data = data.drop(columns=['VSS (mg/L)'])
data = data.rename(columns={'Sample date': 'Date', 'Flow (cms)':'Flow', 'DRP (mg P/L)':'DR
P', 'TP (mg P/L)':'TP', 'TSS (mg/L)':'TSS'})
data.head()
```

Out[22]:

	Date	Site	Flow	DRP	TP	TSS
0	2015-05-19	S11	0.028210	0.018	0.0015	22.333667
1	2015-06-16	S11	0.189993	0.027	0.1770	1006.666667
2	2015-06-30	S11	1.296460	0.203	1.2590	1338.666667
3	2015-08-18	S11	0.131722	0.050	1.3280	2026.000000
4	2015-08-25	S11	0.120204	0.010	3.1180	1722.000000

In [117]:
```
data_11 = data[data['Site'] == 'S11']
data_12 = data[data['Site'] == 'S12']
print(data_11.tail())
print(data_12.tail())
```

```
         Date Site      Flow    DRP     TP         TSS
24 2018-09-11  S11  0.120500  0.052  0.338  363.333333
25 2018-09-11  S11  0.115396  0.014  0.498  162.000000
26 2018-09-26  S11  0.085015  0.010  0.293  474.500000
27 2018-10-11  S11  0.113744  0.047  0.190  148.000000
28 2018-10-11  S11  0.156634    NaN    NaN         NaN
         Date Site      Flow    DRP     TP        TSS
60 2018-08-31  S12  0.028923    NaN    NaN        NaN
61 2018-09-06  S12  0.024730    NaN    NaN        NaN
62 2018-09-26  S12  0.022594    NaN    NaN        NaN
63 2018-10-11  S12  0.024522  0.001  0.042  11.666667
64 2018-10-11  S12  0.044039  0.003  0.040  20.333333
```

In [43]:
```
# check the number of null values
print('S11')
print(data_11.isnull().sum())
print('S12')
print(data_12.isnull().sum())
```

```
S11
Date    0
Site    0
Flow    0
DRP     3
TP      3
TSS     3
dtype: int64
S12
Date    0
Site    0
Flow    0
DRP     4
TP      4
TSS     4
dtype: int64
```

In [47]:
```
# create separate dataframe that only contain null values
data_11_null = data_11[data_11.isnull().any(axis=1)]
data_12_null = data_12[data_12.isnull().any(axis=1)]
print('S11')
print(data_11_null)
print('S12')
print(data_12_null)
```

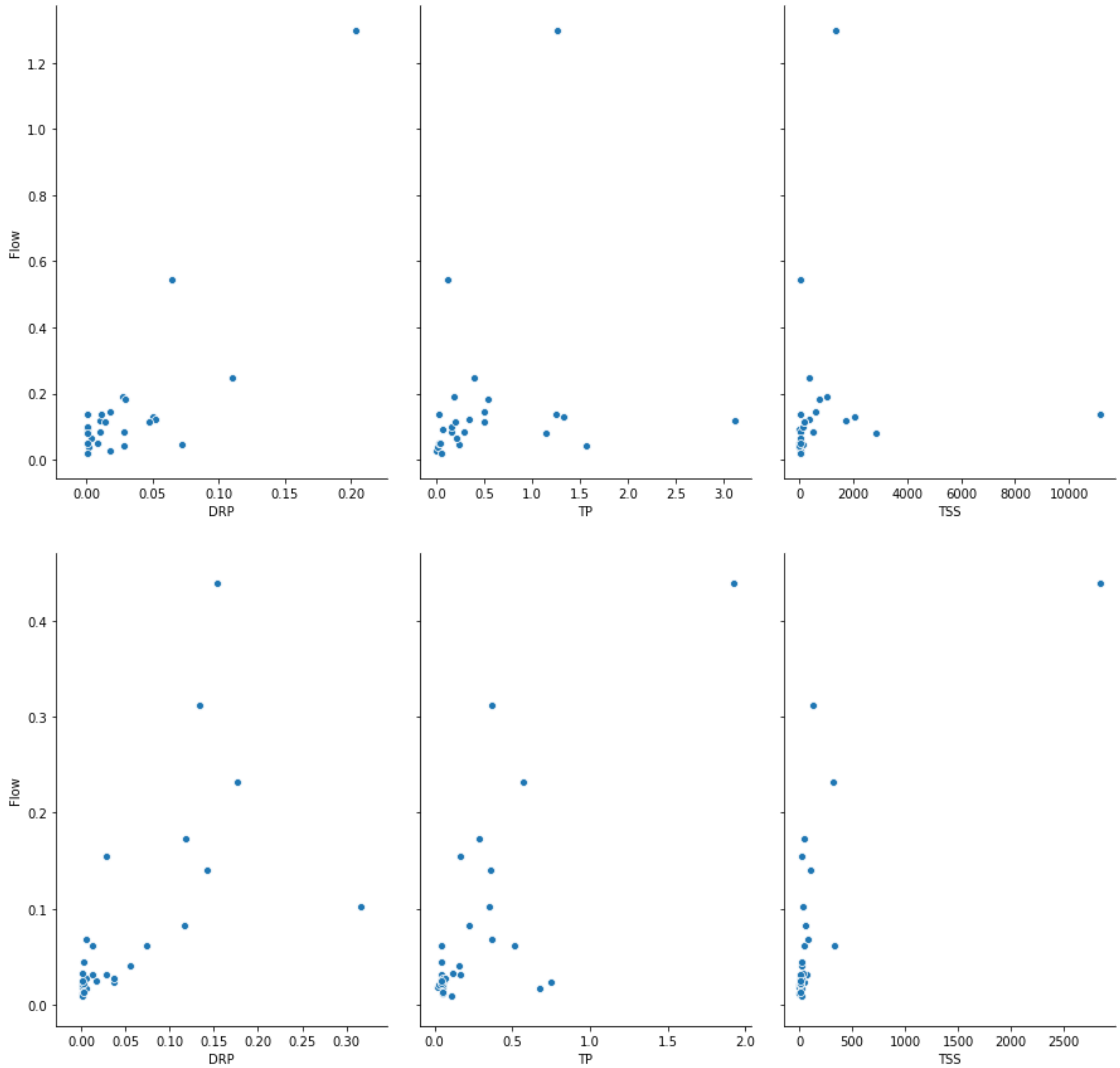
```
S11
         Date Site      Flow  DRP  TP  TSS
21 2018-06-22  S11  0.103414  NaN NaN  NaN
22 2018-06-27  S11  0.189099  NaN NaN  NaN
28 2018-10-11  S11  0.156634  NaN NaN  NaN
S12
         Date Site      Flow  DRP  TP  TSS
59 2018-08-21  S12  0.128687  NaN NaN  NaN
60 2018-08-31  S12  0.028923  NaN NaN  NaN
61 2018-09-06  S12  0.024730  NaN NaN  NaN
62 2018-09-26  S12  0.022594  NaN NaN  NaN
```

In [195]:
```
# drop the null values
data_11 = data_11.dropna()
data_12 = data_12.dropna()
print('S11')
print(data_11.isnull().sum())
print('S12')
print(data_12.isnull().sum())
```

```
S11
Date    0
Site    0
Flow    0
DRP     0
TP      0
TSS     0
dtype: int64
S12
Date    0
Site    0
Flow    0
DRP     0
TP      0
TSS     0
dtype: int64
```

In [50]:
```
# visualize the relationship between flow and responses
sns.pairplot(data_11, x_vars=['DRP', 'TP', 'TSS'], y_vars=['Flow'], height=6, aspect=0.7)
sns.pairplot(data_12, x_vars=['DRP', 'TP', 'TSS'], y_vars=['Flow'], height=6, aspect=0.7)
```

Out[50]: <seaborn.axisgrid.PairGrid at 0x175d4795240>



Objective 1

Develop a suitable model using existing dissolved reactive P (DRP), total P (TP), and TSS

In [80]:
```
list_of_comparisons = ['DRP ~ Flow', 'TP ~ Flow', 'TSS ~ Flow']

for x in list_of_comparisons:
    print(x)
    result = smf.ols(formula=x, data=data_11).fit()
    print(result.summary())
```

```
DRP ~ Flow
                            OLS Regression Results
==============================================================================
Dep. Variable:                    DRP   R-squared:                       0.719
Model:                            OLS   Adj. R-squared:                  0.708
Method:                 Least Squares   F-statistic:                     61.48
Date:                Fri, 04 Oct 2019   Prob (F-statistic):           4.50e-08
Time:                        11:01:56   Log-Likelihood:                 61.156
No. Observations:                  26   AIC:                            -118.3
Df Residuals:                      24   BIC:                            -115.8
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.0068      0.006      1.199      0.242      -0.005       0.018
Flow           0.1487      0.019      7.841      0.000       0.110       0.188
==============================================================================
Omnibus:                       13.007   Durbin-Watson:                   2.306
Prob(Omnibus):                  0.001   Jarque-Bera (JB):               11.946
Skew:                           1.467   Prob(JB):                      0.00255
Kurtosis:                       4.554   Cond. No.                         4.15
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifie
d.
TP ~ Flow
                            OLS Regression Results
==============================================================================
Dep. Variable:                     TP   R-squared:                       0.036
Model:                            OLS   Adj. R-squared:                 -0.005
Method:                 Least Squares   F-statistic:                    0.8870
Date:                Fri, 04 Oct 2019   Prob (F-statistic):              0.356
Time:                        11:01:56   Log-Likelihood:                -26.991
No. Observations:                  26   AIC:                             57.98
Df Residuals:                      24   BIC:                             60.50
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.4394      0.167      2.626      0.015       0.094       0.785
Flow           0.5301      0.563      0.942      0.356      -0.632       1.692
==============================================================================
Omnibus:                       29.806   Durbin-Watson:                   1.758
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               59.814
Skew:                           2.346   Prob(JB):                     1.03e-13
Kurtosis:                       8.762   Cond. No.                         4.15
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifie
d.
TSS ~ Flow
                            OLS Regression Results
==============================================================================
Dep. Variable:                    TSS   R-squared:                       0.003
Model:                            OLS   Adj. R-squared:                 -0.038
Method:                 Least Squares   F-statistic:                   0.08284
Date:                Fri, 04 Oct 2019   Prob (F-statistic):              0.776
Time:                        11:01:56   Log-Likelihood:                -236.65
No. Observations:                  26   AIC:                             477.3
Df Residuals:                      24   BIC:                             479.8
```

```
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept    813.7883    531.769      1.530      0.139    -283.730    1911.306
Flow         514.7888   1788.619      0.288      0.776   -3176.739    4206.316
==============================================================================
Omnibus:                       55.816   Durbin-Watson:                   1.639
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              362.575
Skew:                           4.085   Prob(JB):                     1.85e-79
Kurtosis:                      19.369   Cond. No.                         4.15
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifie
d.
```

In [52]:
```
list_of_comparisons = ['DRP ~ Flow', 'TP ~ Flow', 'TSS ~ Flow']

for x in list_of_comparisons:
    print(x)
    result = smf.ols(formula=x, data=data_12).fit()
    print(result.summary())
```

```
DRP ~ Flow
                            OLS Regression Results
==============================================================================
Dep. Variable:                    DRP   R-squared:                       0.413
Model:                            OLS   Adj. R-squared:                  0.393
Method:                 Least Squares   F-statistic:                     21.10
Date:                Fri, 04 Oct 2019   Prob (F-statistic):           7.32e-05
Time:                        10:13:48   Log-Likelihood:                 47.461
No. Observations:                  32   AIC:                            -90.92
Df Residuals:                      30   BIC:                            -87.99
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.0116      0.013      0.918      0.366      -0.014       0.037
Flow           0.4836      0.105      4.593      0.000       0.269       0.699
==============================================================================
Omnibus:                       47.570   Durbin-Watson:                   1.320
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              236.046
Skew:                           3.113   Prob(JB):                     5.54e-52
Kurtosis:                      14.759   Cond. No.                         10.6
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifie
d.
TP ~ Flow
                            OLS Regression Results
==============================================================================
Dep. Variable:                     TP   R-squared:                       0.558
Model:                            OLS   Adj. R-squared:                  0.543
Method:                 Least Squares   F-statistic:                     37.81
Date:                Fri, 04 Oct 2019   Prob (F-statistic):           9.20e-07
Time:                        10:13:48   Log-Likelihood:               0.093905
No. Observations:                  32   AIC:                             3.812
Df Residuals:                      30   BIC:                             6.744
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.0357      0.055      0.644      0.524      -0.078       0.149
Flow           2.8447      0.463      6.149      0.000       1.900       3.790
==============================================================================
Omnibus:                       13.785   Durbin-Watson:                   2.480
Prob(Omnibus):                  0.001   Jarque-Bera (JB):               15.234
Skew:                           1.211   Prob(JB):                     0.000492
Kurtosis:                       5.359   Cond. No.                         10.6
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifie
d.
TSS ~ Flow
                            OLS Regression Results
==============================================================================
Dep. Variable:                    TSS   R-squared:                       0.556
Model:                            OLS   Adj. R-squared:                  0.541
Method:                 Least Squares   F-statistic:                     37.55
Date:                Fri, 04 Oct 2019   Prob (F-statistic):           9.76e-07
Time:                        10:13:48   Log-Likelihood:                -230.79
No. Observations:                  32   AIC:                             465.6
Df Residuals:                      30   BIC:                             468.5
```

```
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept   -143.3081     75.413     -1.900      0.067    -297.323      10.706
Flow        3854.9033    629.085      6.128      0.000    2570.141    5139.665
==============================================================================
Omnibus:                       20.257   Durbin-Watson:                   2.516
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               68.576
Skew:                           0.986   Prob(JB):                     1.29e-15
Kurtosis:                       9.895   Cond. No.                         10.6
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specifie
d.
```

Discussion for objective 1

The quality of the OLS models was evaluated based on p-value and R2:
1) S11-DRP - will work well
2) S11-TP - will not work well
3) S11-TSS - will not work well based on R2, p-value, and high Y-intercept value
4) S12- DRP - will work well
5) S12 - TP - will work well
6) S12 - TSS - appear that it will work well based on R2 and p-value, but the Y-intercept is largely negative

Objective 2

Predict the missing DRP, TP, TSS data using the model developed in (1)

In [176]:
```python
# predict values and export the values into a dataframe for S11
list_of_comparisons = ['DRP ~ Flow', 'TP ~ Flow', 'TSS ~ Flow']
predictions_11 = pd.DataFrame(columns=list(list_of_comparisons))

for x in list_of_comparisons:
    result = smf.ols(formula=x, data=data_11).fit()
    prednull_11 = result.predict(data_11_null['Flow'])
    predictions_11.loc[:, x] = prednull_11
predictions_11
```

Out[176]:

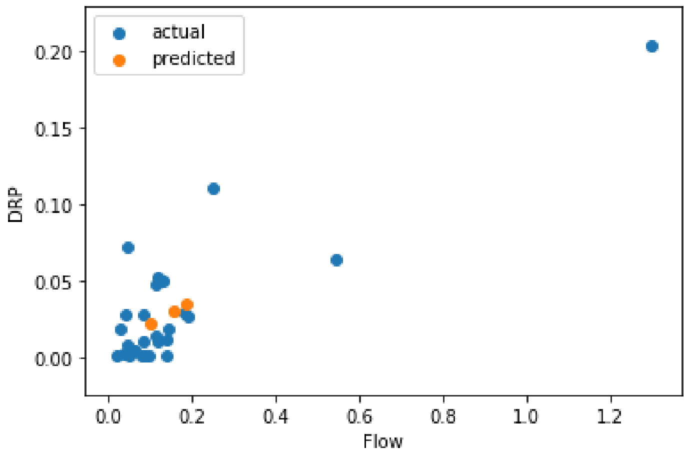
	DRP ~ Flow	TP ~ Flow	TSS ~ Flow
21	0.022142	0.494259	867.024807
22	0.034885	0.539682	911.134534
28	0.030057	0.522472	894.421904

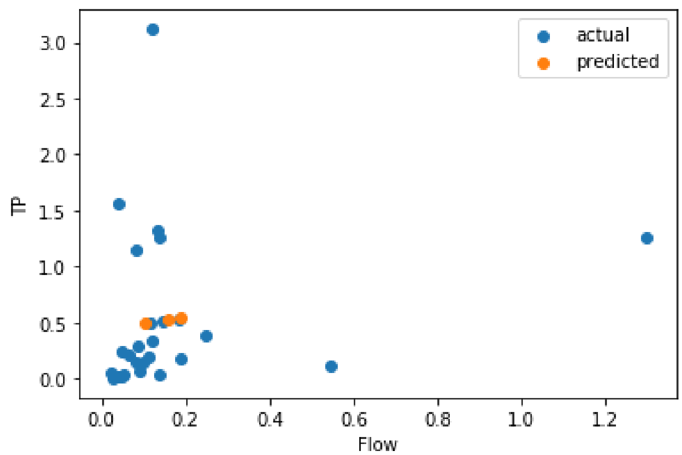
In [199]:
```
# add the values back into this new dataframe - getting the dataframe ready to merge with
 the main dataframe
predictions_11 ['Flow'] = data_11_null ['Flow']
predictions_11 ['Date'] = data_11_null ['Date']
predictions_11 ['Site'] = 'S11'
predictions_11 = predictions_11.rename(columns={'DRP ~ Flow': 'DRP', 'TP ~ Flow': 'TP', 'T
SS ~ Flow': 'TSS'})
predictions_11
```

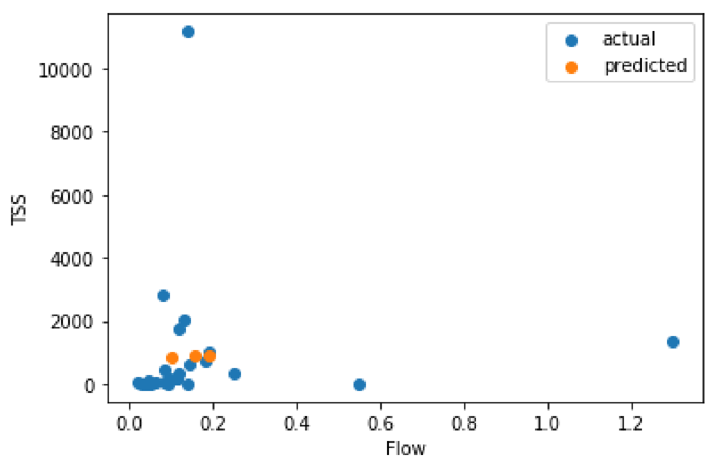
Out[199]:

	DRP	TP	TSS	Flow	Date	Site
21	0.022142	0.494259	867.024807	0.103414	2018-06-22	S11
22	0.034885	0.539682	911.134534	0.189099	2018-06-27	S11
28	0.030057	0.522472	894.421904	0.156634	2018-10-11	S11

In [219]:
```
analytes = ['DRP', 'TP', 'TSS']

for x in analytes:
    plt.scatter(data_11['Flow'], data_11[x], label='actual')
    plt.scatter(predictions_11['Flow'], predictions_11[x], label='predicted')
    plt.legend()
    plt.xlabel('Flow')
    plt.ylabel(x)
    plt.show()
```







In [220]:
```
data_11.shape
```

Out[220]: (26, 6)

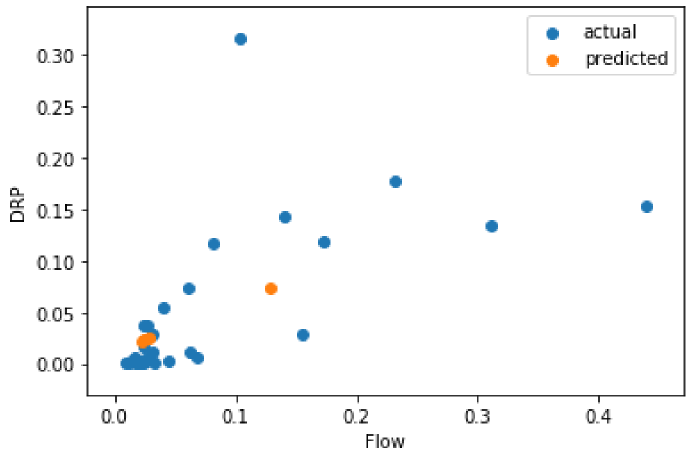
In [221]:
```
merged_11 = pd.concat([data_11, predictions_11], sort=True)
merged_11 = merged_11.sort_values(by=['Date'])
merged_11.shape
```

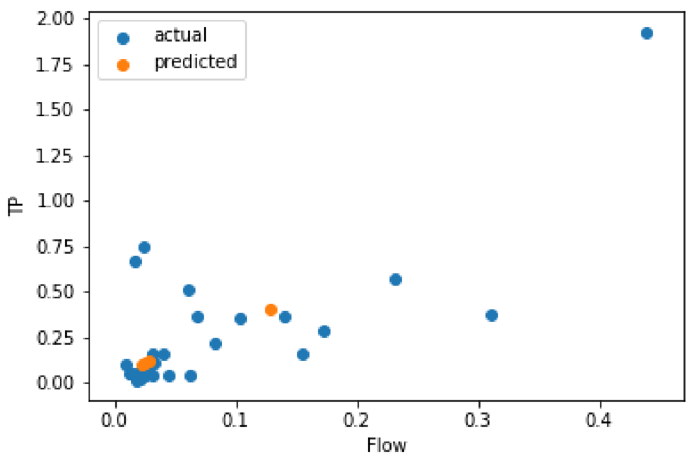
Out[221]: (29, 6)

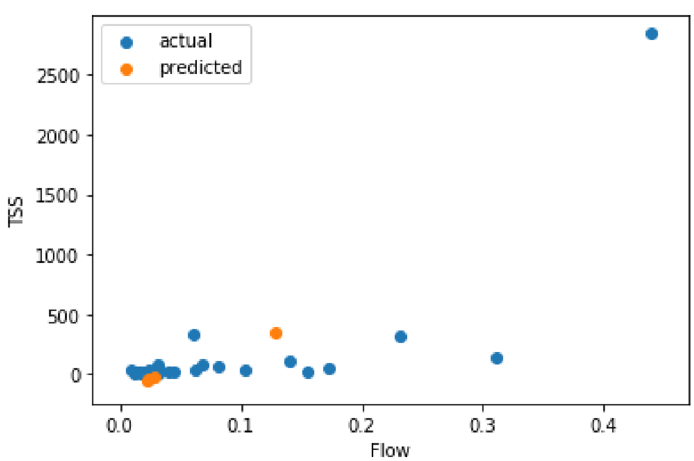
In [223]:
```
# predict values and export the values into a dataframe for S12
list_of_comparisons = ['DRP ~ Flow', 'TP ~ Flow', 'TSS ~ Flow']
predictions_12 = pd.DataFrame(columns=list(list_of_comparisons))

for x in list_of_comparisons:
    result = smf.ols(formula=x, data=data_12).fit()
    prednull_12 = result.predict(data_12_null['Flow'])
    predictions_12.loc[:, x] = prednull_12

# add the values back into this new dataframe - getting the dataframe ready to merge with
 the main dataframe
predictions_12 ['Flow'] = data_12_null ['Flow']
predictions_12 ['Date'] = data_12_null ['Date']
predictions_12 ['Site'] = 'S12'
predictions_12 = predictions_12.rename(columns={'DRP ~ Flow': 'DRP', 'TP ~ Flow': 'TP', 'T
SS ~ Flow': 'TSS'})
predictions_12.shape
```

Out[223]: (4, 6)

In [222]:
```python
analytes = ['DRP', 'TP', 'TSS']

for x in analytes:
    plt.scatter(data_12['Flow'], data_12[x], label='actual')
    plt.scatter(predictions_12['Flow'], predictions_12[x], label='predicted')
    plt.legend()
    plt.xlabel('Flow')
    plt.ylabel(x)
    plt.show()
```







In [225]:
```
merged_12 = pd.concat([data_12, predictions_12], sort=True)
merged_12 = merged_12.sort_values(by=['Date'])
merged_12.shape
```

Out[225]:
```
(36, 6)
```

In [229]:
```
# merge all the data back together
final_merged = pd.concat([merged_11, merged_12])
print(final_merged.shape)
print(final_merged.isnull().sum())
```

```
(65, 6)
DRP     0
Date    0
Flow    0
Site    0
TP      0
TSS     0
dtype: int64
```

Discussion for objective 2

From the figures above where predicted and actual values were plotted:
1) S11-DRP - worked well
2) S11-TP - worked okay (within somewhat expected range, but the model had weak R2 and p-value)
3) S11-TSS - worked okay (within somewhat expected range)
4) S12- DRP - worked well
5) S12 - TP - worked well
6) S12 - TSS - did not work at all (some of the predicted TSS concentrations were negative)
In summary, this OLS model worked well for DRP and TP, but not TSS. Another suitable model is needed to predict TSS concentration. A potential approach is to multiple-linear-regression to incorporate several parameters such as precipitation data (antecedent condition) and time of the year (seasonality - land cover density).

In [ ]:
```

```