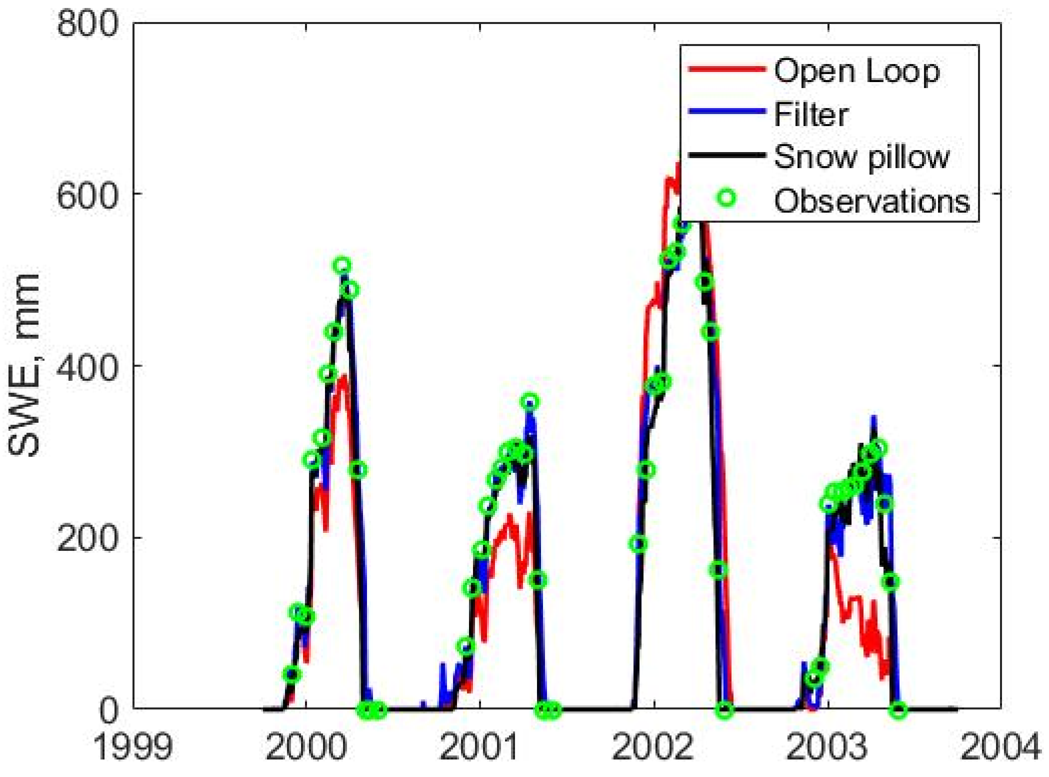
Homework 7

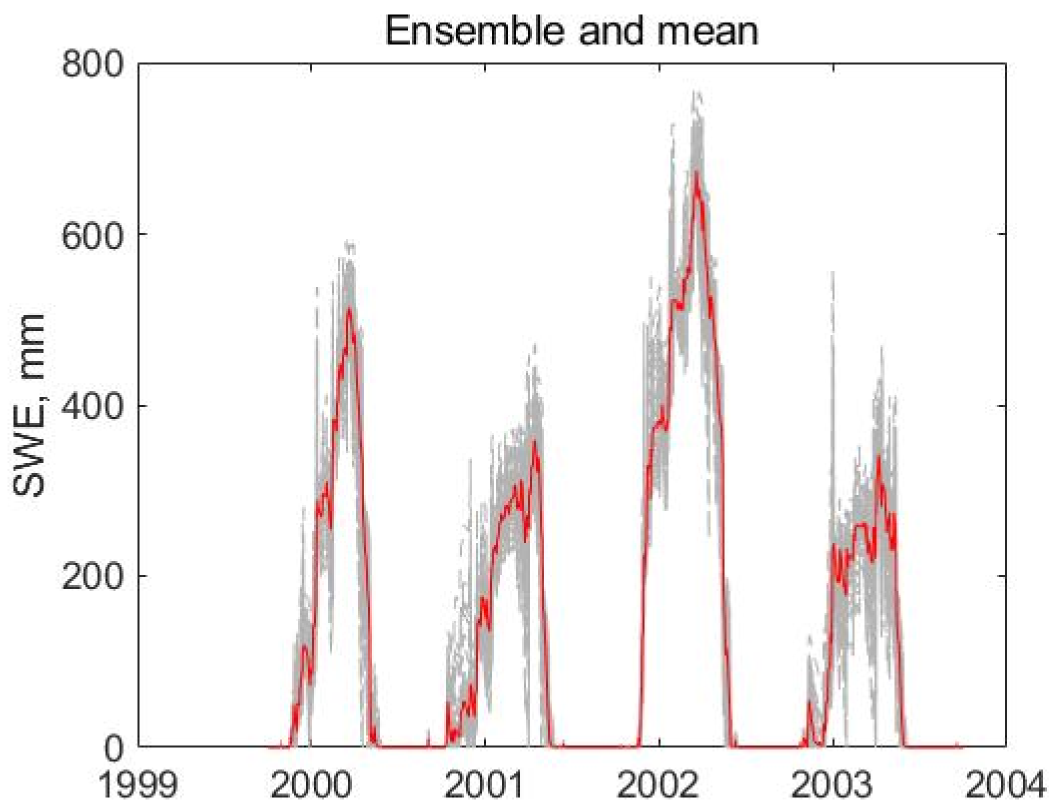
Problem 1 (60%) (based on Data Assimilation Lab, which is in Matlab):

Use Matlab, the ‘Assimilation Workshop 2014.zip’ folder of matlab files and your experience in the assimilation computer lab to answer the following questions:

A.

Use ‘ExtractObservationsCourse.m’ to change the analysis years to 2000 to 2003, and run ‘EnKFCourse.m’ with the default parameters, the same we used in lab. Provide a plot and describe the major problems with the open loop model. Are they the same across all years? In which year did the open loop model perform the worst? Looking at only the spread in the ensemble of model predictions (Figure 2), would you have picked the same year to perform the worst? Why? Report the RMSE statistics for the open loop model, the filter model (which includes data assimilation), and for the snow course data vs. the snow pillow data (obs). Using the RMSE code (lines 97-99) as an example, also calculate the bias for each of these. (Note: Bias is the same as mean error – just modify the RMSE code to not take the square or the square root. If there are equal positive and negative errors, the bias should be 0.)





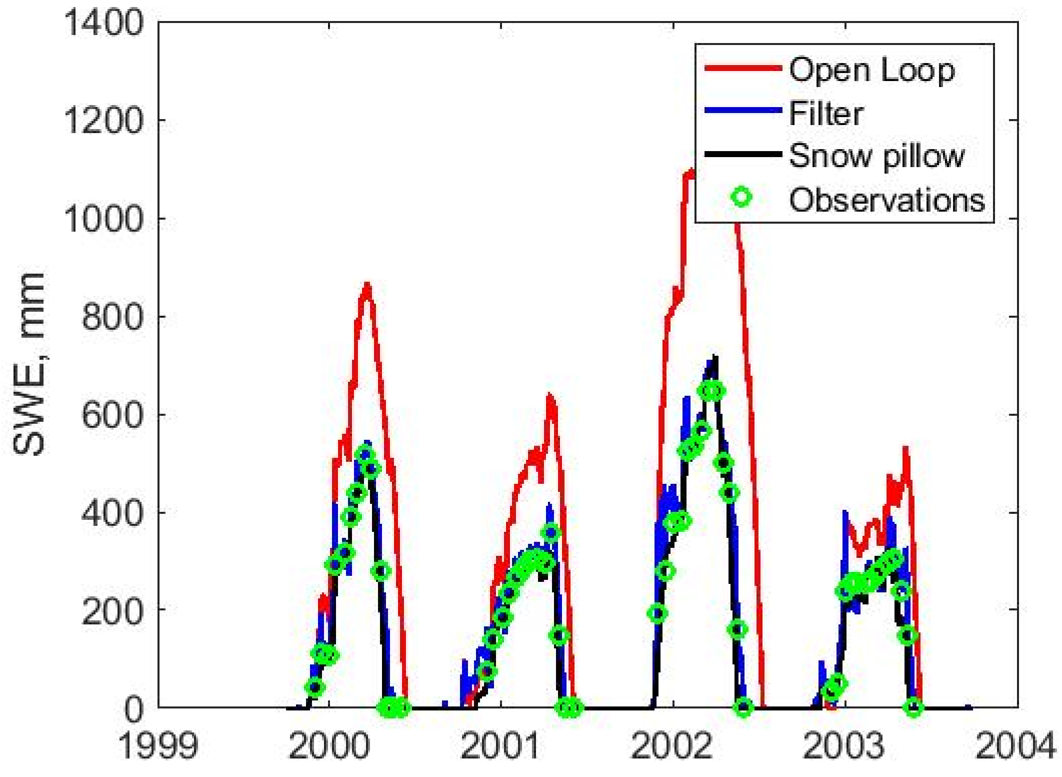
According to the figure we can tell that 2002 seems different from the others, and the year 2003 the open loop model performed the worst, when checking the figure 1 we can see the model fittings for 2003 are worse comparing to the other years. The 2003 year is also considered to be the worst one when only check figure 2 since we can clearly see there is a lot of residuals from the model fitting on figure 2 year 2003.

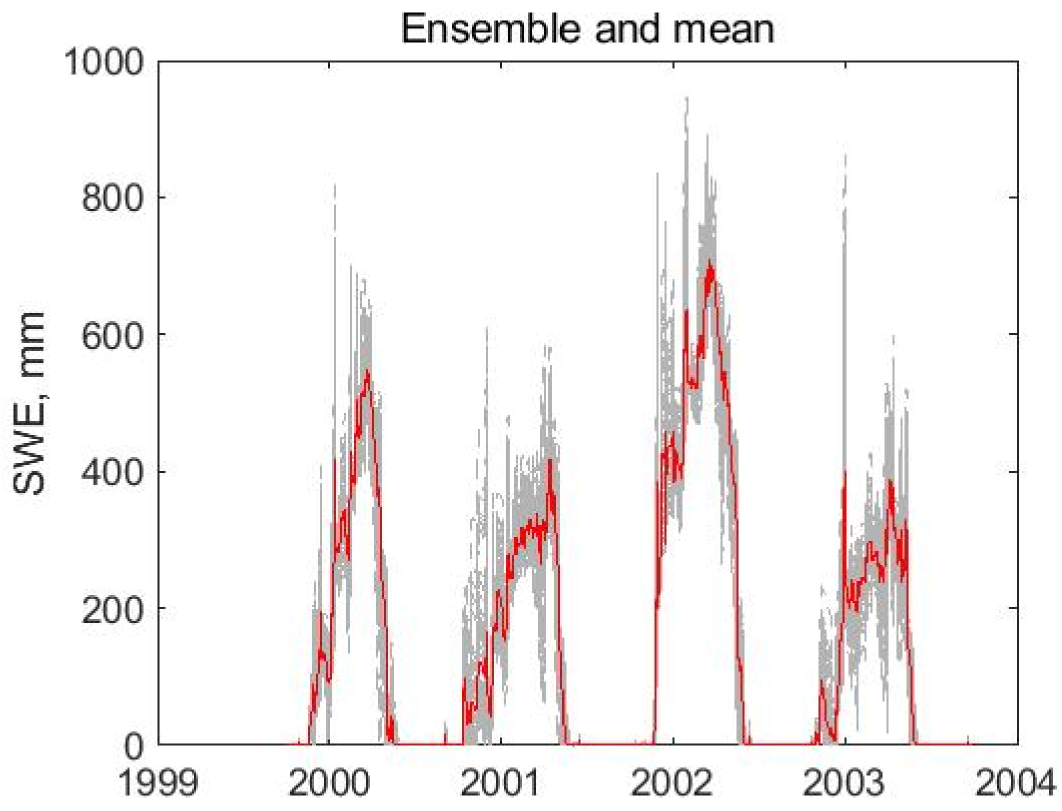
For the RMSEs: RMSE(open loop model)=72.3175, RMSE(filter model)=34.4743, RMSE(obs)=30.0596

The biases (mean error) for each case: Bias(open loop)=-10.6906, Bias(filter)=8.5330, Bias(obs)=18.3462

B.

The default parameters were for a fairly well-calibrated and unbiased model. What if the model had more problems than this? Precipitation is very hard to measure. Assign the gauge undercatch (line 17 in ‘EnKFCourse.m’, NominalParams.GUC) a factor of 2 and re-run the simulation. Compare the plot of the results with the one you created in part a; what differences do you see? Report the RMSE and bias for the open loop and filter loop (the EnKF model) and compare to those you calculated in part a. What do these results tell you about data assimilation when using a biased model?





From the new parameter adjusted model we can see the open loop model is upward altered comparing to the previous model.

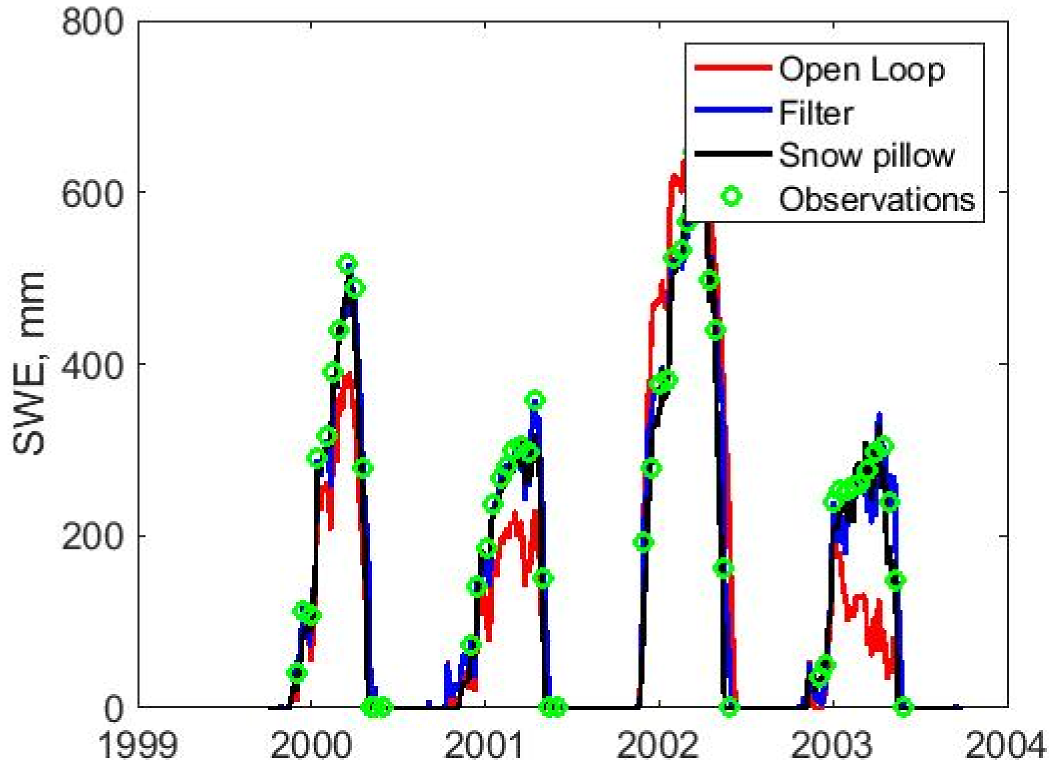
For the RMSEs: RMSE(open loop model)=72.3175(previous) 284.0299(now), RMSE(filter model)=34.4743(previous) 48.7182(now)

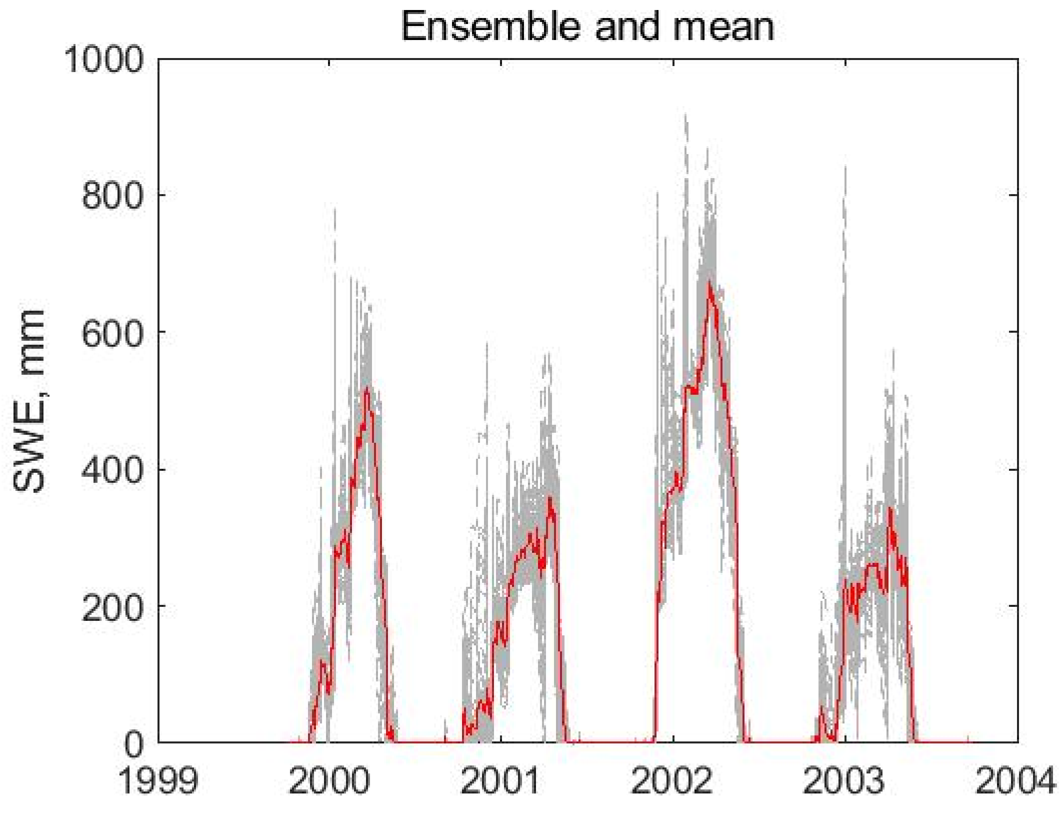
The biases (mean error) for each case: Bias(open loop)=-10.6906(previous) 173.8612(now), Bias(filter)=8.5330(previous) 24.7231(now)

We can get the conclusion that if using a biased model then all the errors will be amplified.

C.

Return the gauge undercatch parameter to its original value (1.2). Now, instead of looking at a biased basic model, look at a model with greater precipitation uncertainty. Change the coefficient of variation on the precipitation from 40% to 80%. (Change line 30 to 'covPrecip=0.8;'). Run the EnKFCourse.m code again, and compare the plots, RMSE, and bias with those in parts a and b.





The plot is similar to those in part a.

For the RMSEs: RMSE(open loop model)=72.3175, RMSE(filter model)=34.0864

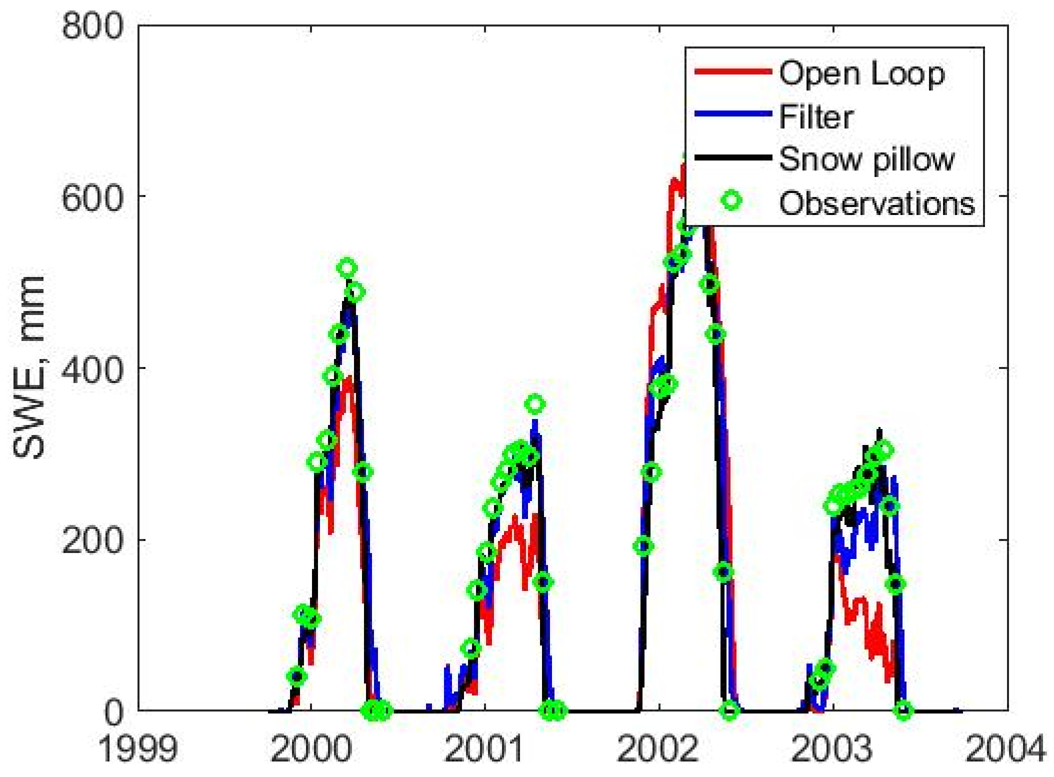
The biases (mean error) for each case: Bias(open loop)=-10.6906, Bias(filter)=9.0055

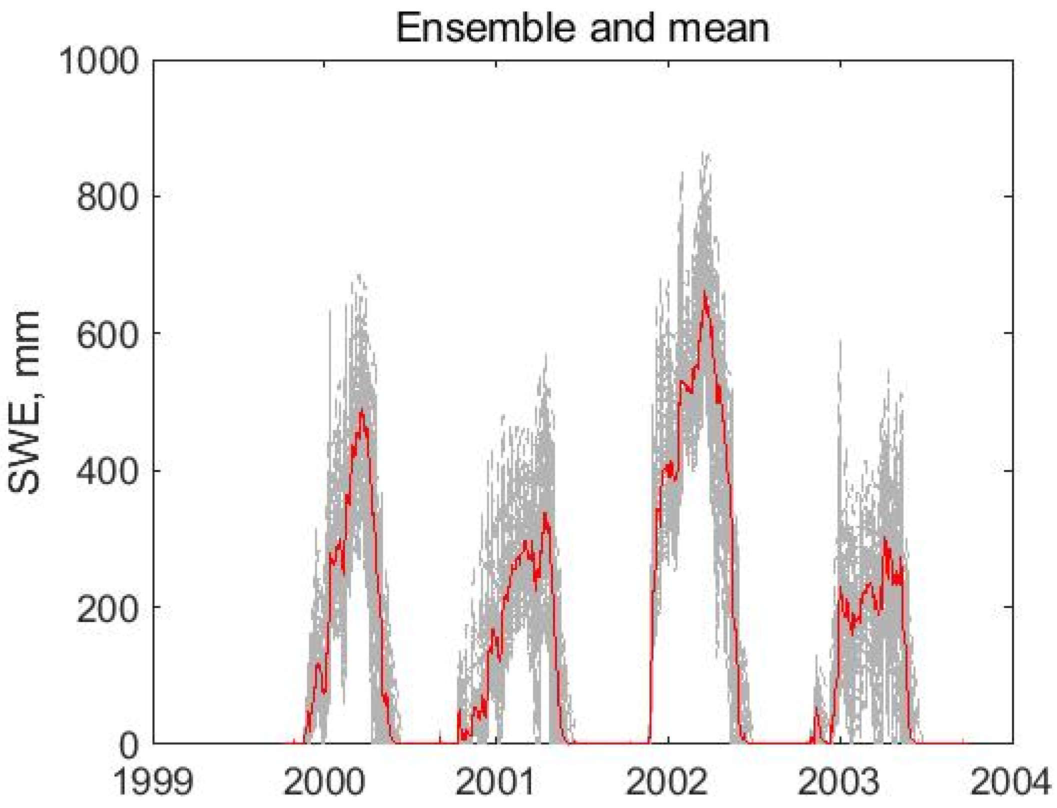
Comparing to part a result we can conclude that the variation on the precipitation won't affect that much on the model performance.

D.

Return covPrecip to 0.4. Now, test the effect of changing the uncertainty associated with the observations being assimilated. stdZ represents the standard deviation in the snow course measurements (30 mm), and r is the associated variance (stdZ^2). Increase the standard deviation. (Type ‘stdZ=100;’ and ‘r=stdZ.^2;’ in the code at line 11 (after the ObservationsCourse.mat file is loaded).) Run the code again, and compare the plots, RMSE, and bias with those in parts a, b, and c. Repeat the above with ‘stdZ=10;’ and compare both. Discuss your results.

When stdZ=100:



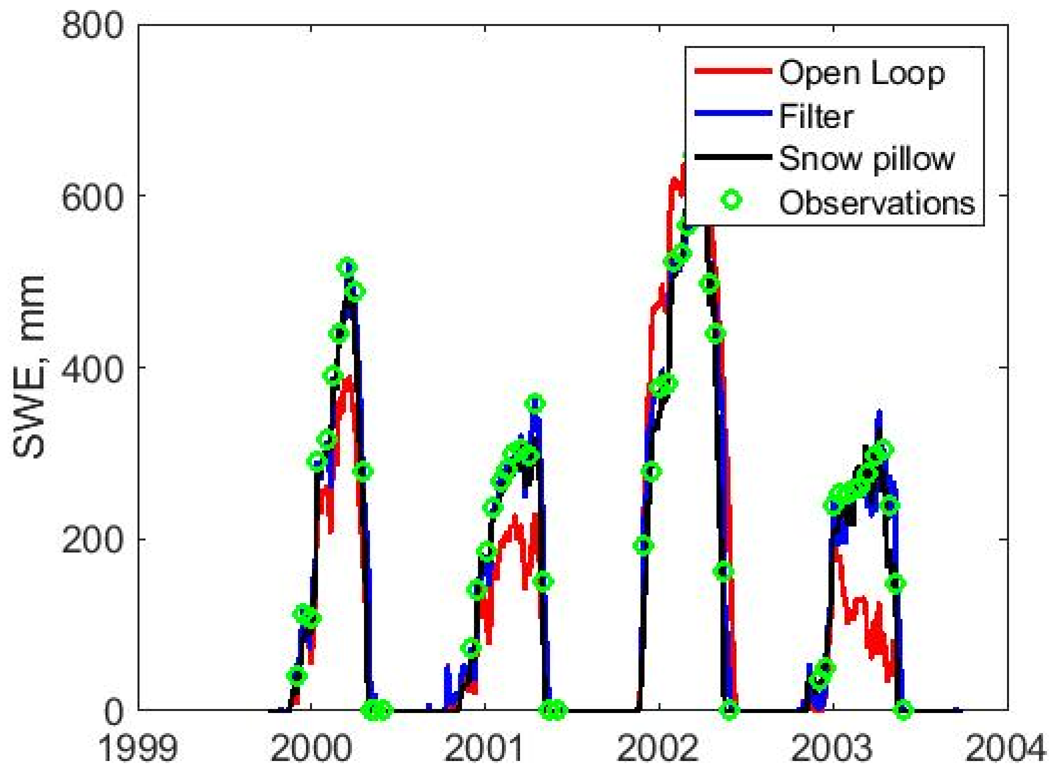


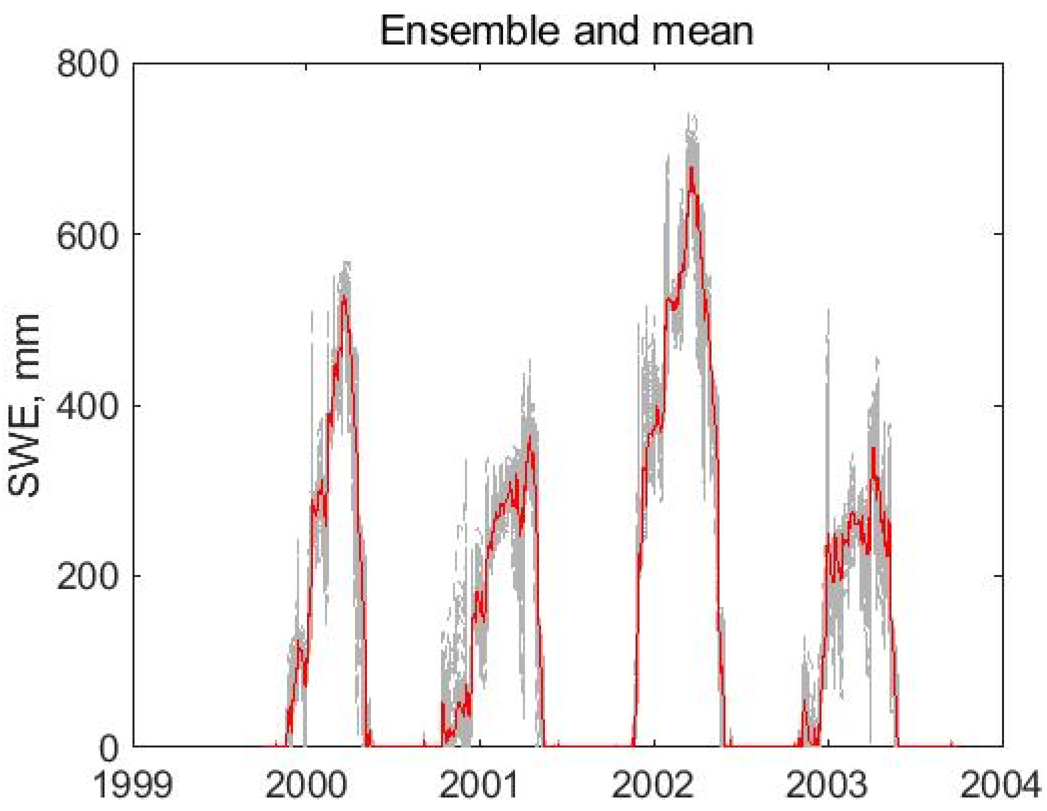
For the RMSEs: RMSE(open loop model)=72.3175, RMSE(filter model)=41.4563

The biases (mean error) for each case: Bias(open loop)=-10.6906, Bias(filter)=6.5516

Thus comparing to previous models, when stdZ=100 the filtered model actually performed worse while the open loop model keeps the same.

When stdZ=10:





For the RMSEs: RMSE(open loop model)=72.3175, RMSE(filter model)=32.5748

The biases (mean error) for each case: Bias(open loop)=-10.6906, Bias(filter)=9.8289

We can tell that the stdZ won't affact the open loop model, but for the filtered model lower stdZ will yield lower

Problem 2: The best graphics (20%)

One of our greatest challenges in data analysis is to be able to visualize the information in the data and convey that information to others. Consider various scientific papers you have read (on any subject related to water or air) and pick out your favorite graphical representation of data (e.g., the best figure). Paste your top two choices here with a brief statement of why you chose these.

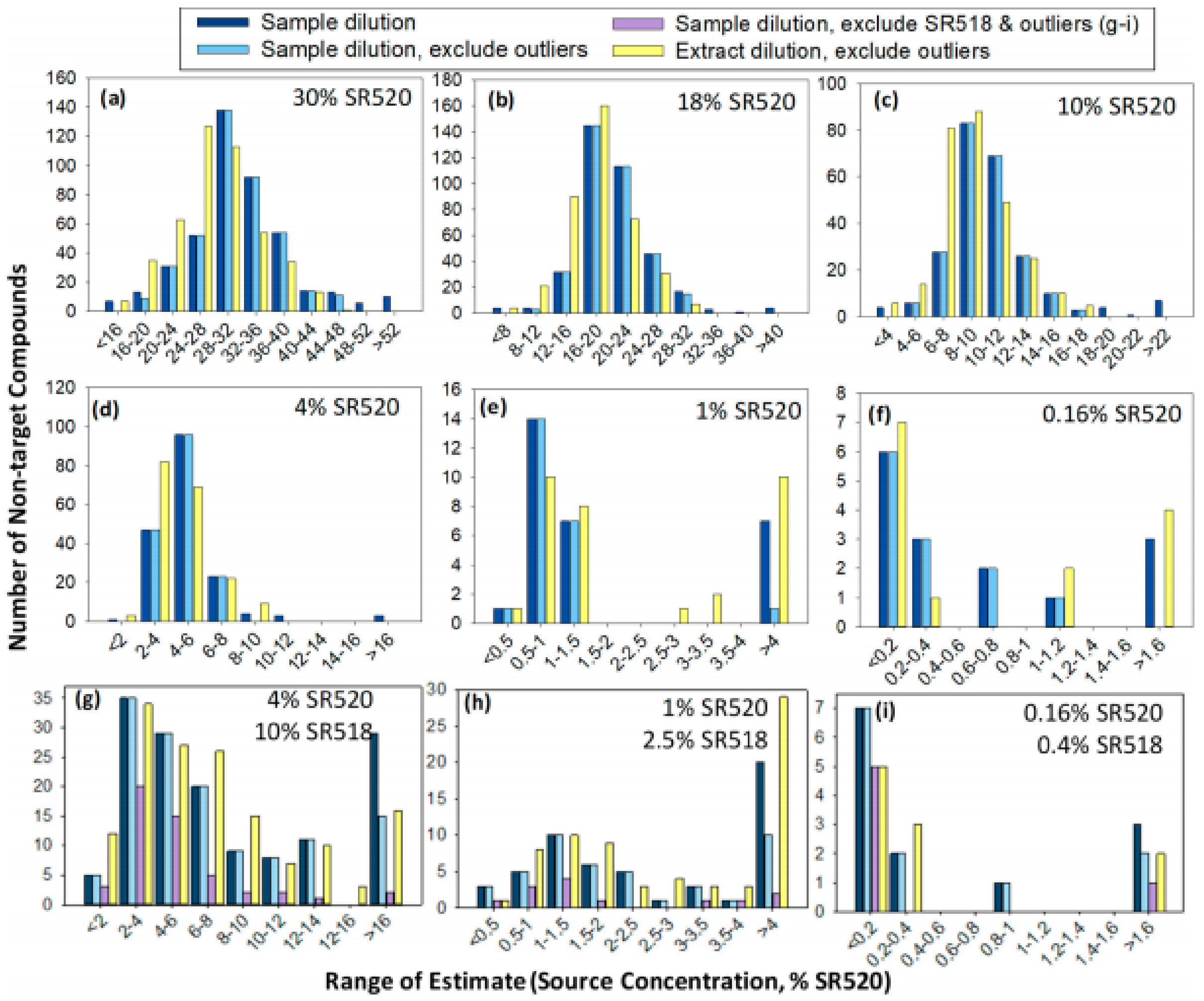


Figure 3. Histograms showing the distribution of SR520 concentration estimates and number of contributing nontarget compounds for the watershed mixtures: (a) Mix 1 (30% SR520), (b) Mix 2 (18% SR520), (c) Mix 3 (10% SR520), (d) Mix 4A (4% SR520), (e) Mix 5A (1% SR520), (f) Mix 6A (0.16% SR520), (g) Mix 4B (4% SR520 + 10% SR518), (h) Mix 5B (1% SR520 + 2.5% SR518), and (i) Mix 6B (0.16% SR520 + 0.4% SR518). Estimates from the SR520 sample dilution (including and excluding outliers) and extract dilution (excluding outliers) curves are shown for all nine mixtures; estimates using the SR520 sample dilution curve made only with SR520-unique compounds (i.e., excluding outliers and excluding compounds also detected in SR518 runoff) are only shown for the multisource mixtures (g−i).

Application of Nontarget High Resolution Mass Spectrometry Data to Quantitative Source Apportionment, Katherine T. Peter, Christopher Wu, Zhenyu Tian, and Edward P. Kolodziej, Environmental Science & Technology 2019 53 (21), 12257-12268, DOI: 10.1021/acs.est.9b04481

This graph did a good visualization on MS features PDF based on different dilution rate from a highway runoff samples, which gave us a good idea of how the features looks like during dilution and further help the researchers to do the apportionment efforts.

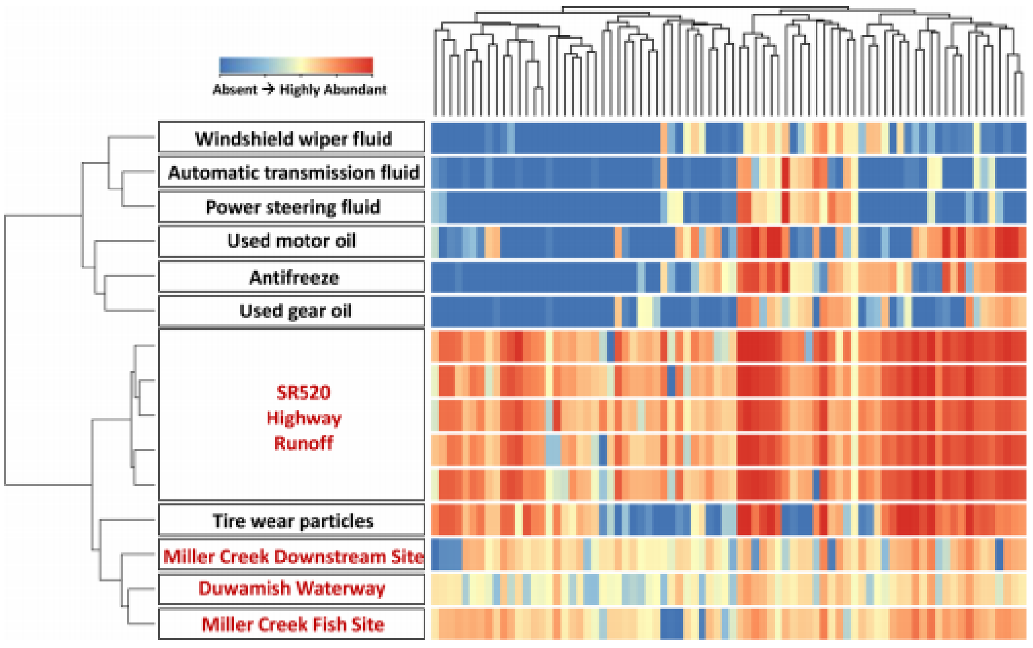


Figure 2. Hierarchical cluster analysis of waters associated with adult coho spawner mortality (red text) and vehicle-related sources (black text), for the features that comprise the coho mortality signature (n = 78 here, due to iterative software processing of different sample sets). Each row in the diagram represents an average of 3 or 4 replicate samples, each vertical line represents an individual feature, and the color of each bar represents the peak area (absent = dark blue, increasing to light blue, light yellow, peach, and dark red with increasing peak area, up to the maximum observed peak area for all data shown).

Using High-Resolution Mass Spectrometry to Identify Organic Contaminants Linked to Urban Stormwater Mortality Syndrome in Coho Salmon, Katherine T. Peter, Zhenyu Tian, Christopher Wu, Peter Lin, Sarah White, Bowen Du, Jenifer K. McIntyre, Nathaniel L. Scholz, and Edward P. Kolodziej, Environmental Science & Technology 2018 52 (18), 10317-10327, DOI: 10.1021/acs.est.8b03287

This graph deal with the visualization of clustering from a LCMS data, which give a great vision for identifying the potentially toxic fractions of the compounds that can be toxic to coho salmons. Thus give a direction to go for future researches.

Graduate Students: Problem 3: Project Update (20%)

Write out three clear (specific) questions that you are trying to answer in your project. For each question, provide 1-2 graphs that illustrate the answer to that question, as well as a couple sentences of explanation to accompany each graph (explain what you did to generate the graph, and what the graph shows). Do you think your answers are clear, or is further work needed? If you feel additional tests/analyses are needed, describe what you plan to do. If there are areas where you feel you need additional guidance, please describe the issues here.

In [68]:

```
import numpy as np
import scipy.stats as st
import statistics as stats
import matplotlib.pyplot as plt
import pandas as pd
import math
import seaborn as sns


%matplotlib inline
```

For the illustration of our problem we'll start from the Q01 site as example

In [69]:

```
df_Q1=pd.read_csv("Lyell_blw_Maclure_timeseries_stage_Q_T_2005_2015.csv",low_memory=False)
df_Q01=df_Q1[df_Q1[' estimated_discharge(cms)'].isna()==False]
df_Q01['date_time(inGMT_PDTplus7)']=pd.to_datetime(df_Q01['date_time(inGMT_PDTplus7)'])
```

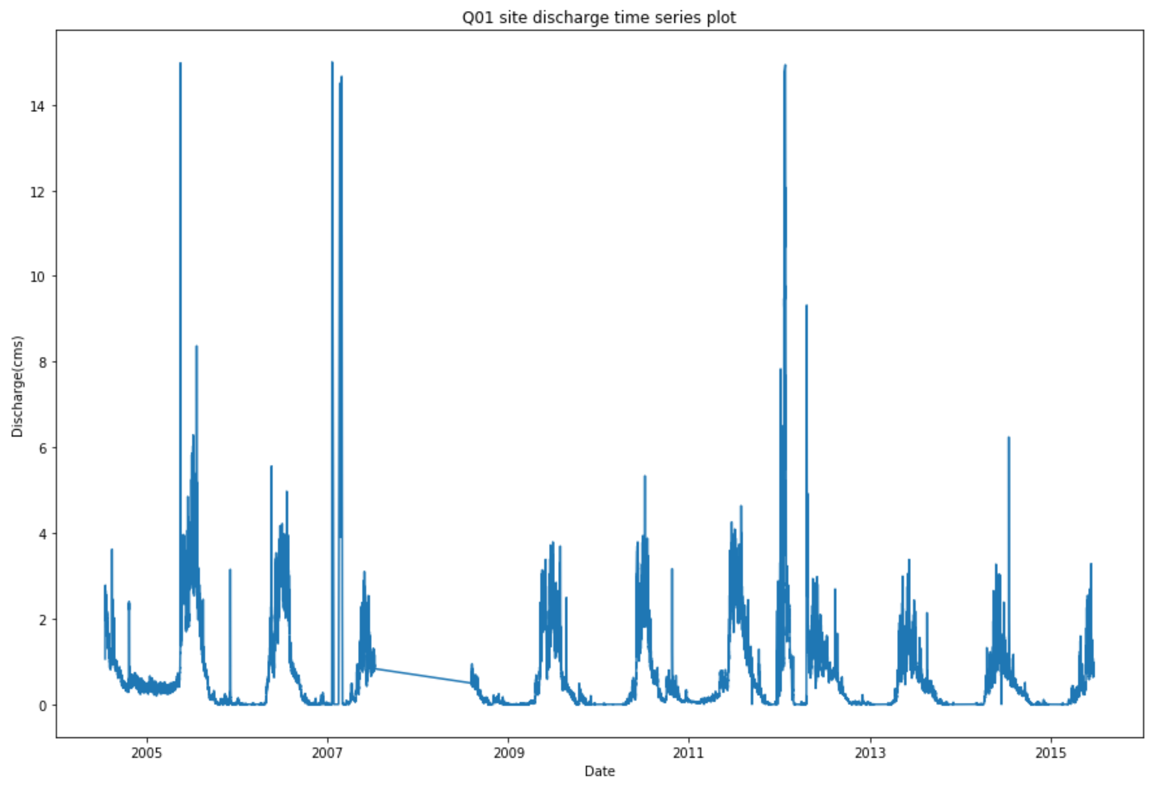
```
d:\python\anaconda\lib\site-packages\ipykernel_launcher.py:3: SettingWithCopyWarni
ng:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/
indexing.html#indexing-view-versus-copy
  This is separate from the ipykernel package so we can avoid doing imports until
```

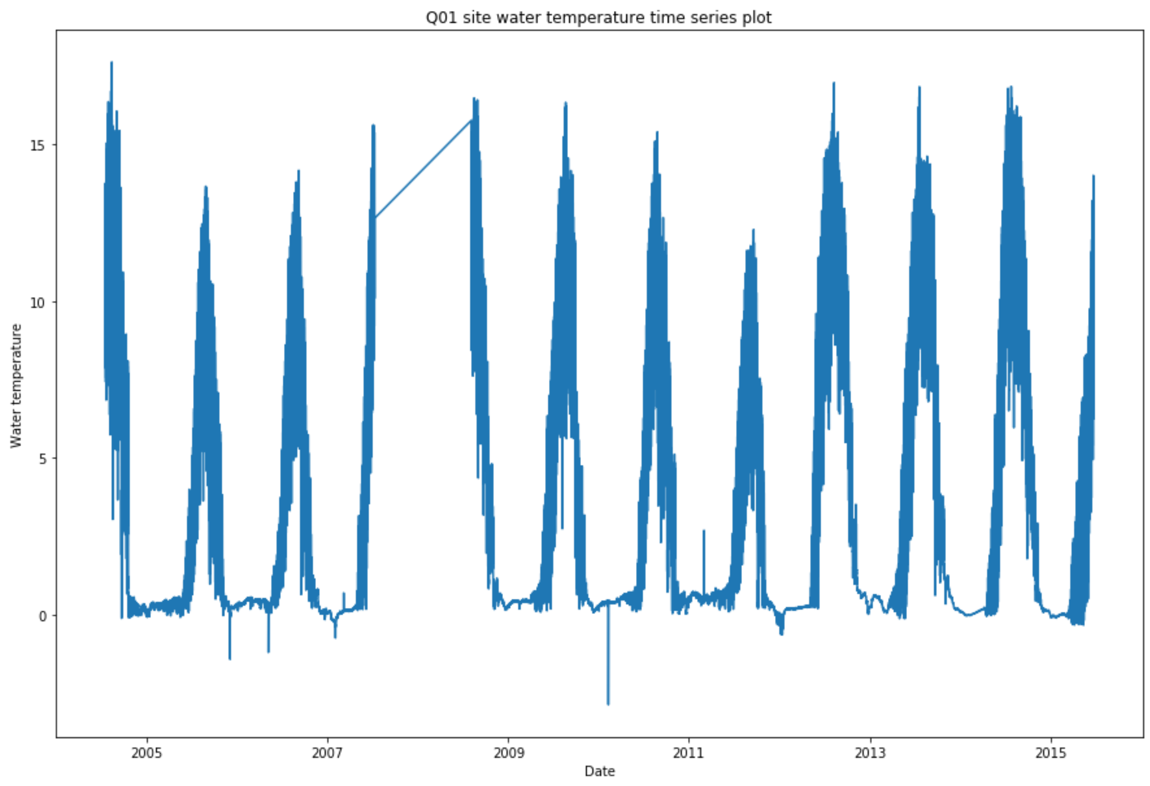
In [70]:

```
plt.figure(figsize=(15,10))
plt.plot(df_Q01['date_time(inGMT_PDTplus7)'],df_Q01[' estimated_discharge(cms)'])
plt.xlabel('Date')
plt.ylabel('Discharge(cms)')
plt.title('Q01 site discharge time series plot')

plt.figure(figsize=(15,10))
plt.plot(df_Q01['date_time(inGMT_PDTplus7)'],df_Q01[' water_temperature(deg_C)'])
plt.xlabel('Date')
plt.ylabel('Water temperature')
plt.title('Q01 site water temperature time series plot')
```

Out[70]:

```
Text(0.5,1,'Q01 site water temperature time series plot')
```





Note

1. do data cleaning first
2. recheck the process: i.e. if air temp is available, consider to set question as : is air temp controls discharge, is air temp controls water temp, or air&water temp controls discharge
3. parse the data to avoid autocorrelation, do the average of discharge based on weeks/month for the analysis, also try to get the min/max/avg temp from the parse period for the analysis

In [71]:

```
df_Q01.columns
```

Out[71]:

```
Index(['date_time(inGMT_PDTplus7)', ' raw_pressure(cm)',
       ' barocorrected_pressure(cm)',
       ' stage_in_feet(used_in_rating_curve_ft_referenced_to_bolt)',
       ' estimated_discharge(cms)',
       'lower_confidence_discharge_cms_bestestimate',
       'upper_confidence_discharge_cms_bestestimate',
       ' water_temperature(deg_C)', ' instrument_ID',
       ' offset_cm(cm_to_correct_solinst_to_bolt)',
       ' offset_ft(ft_to_correct_solinst_to_bolt)'],
      dtype='object')
```

In [72]:

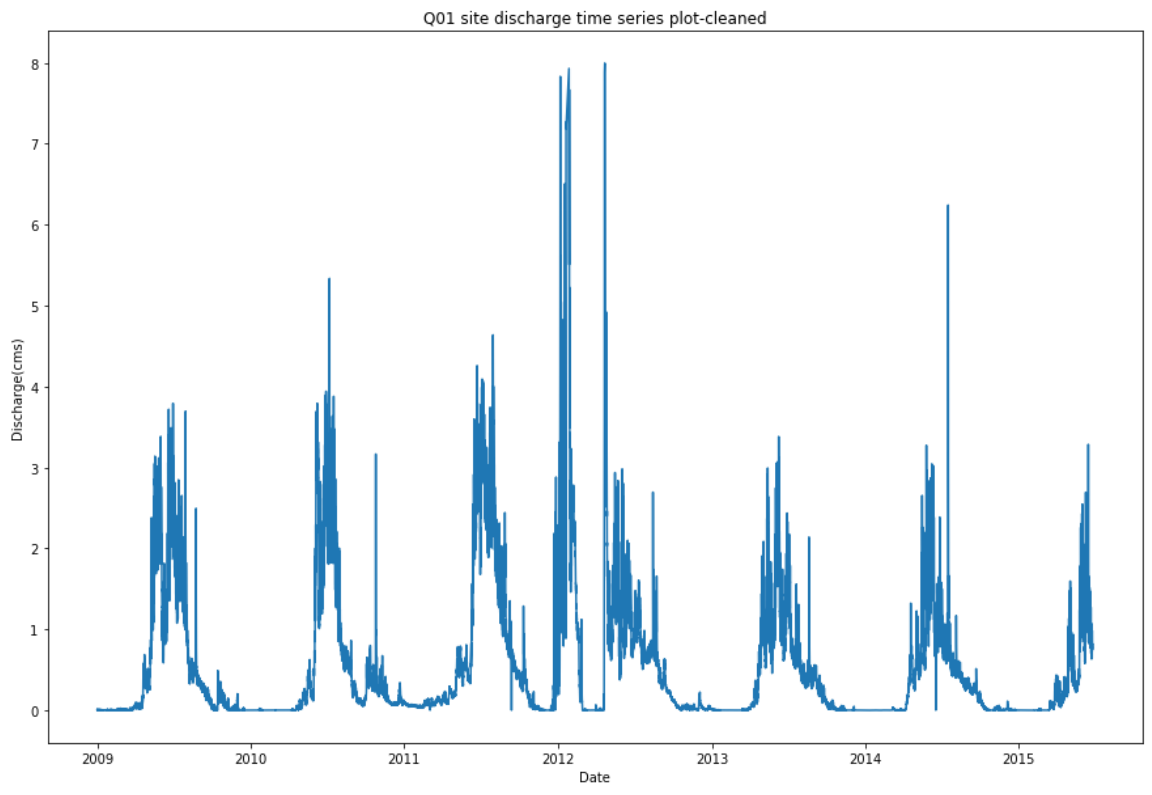
```
df_c=df_Q01[df_Q01['date_time(inGMT_PDTplus7)']>='2009-01-01 00:00:00']
#Clean the abnormal points from original data
df_c=df_c[df_c[' estimated_discharge(cms)']<=8]
```

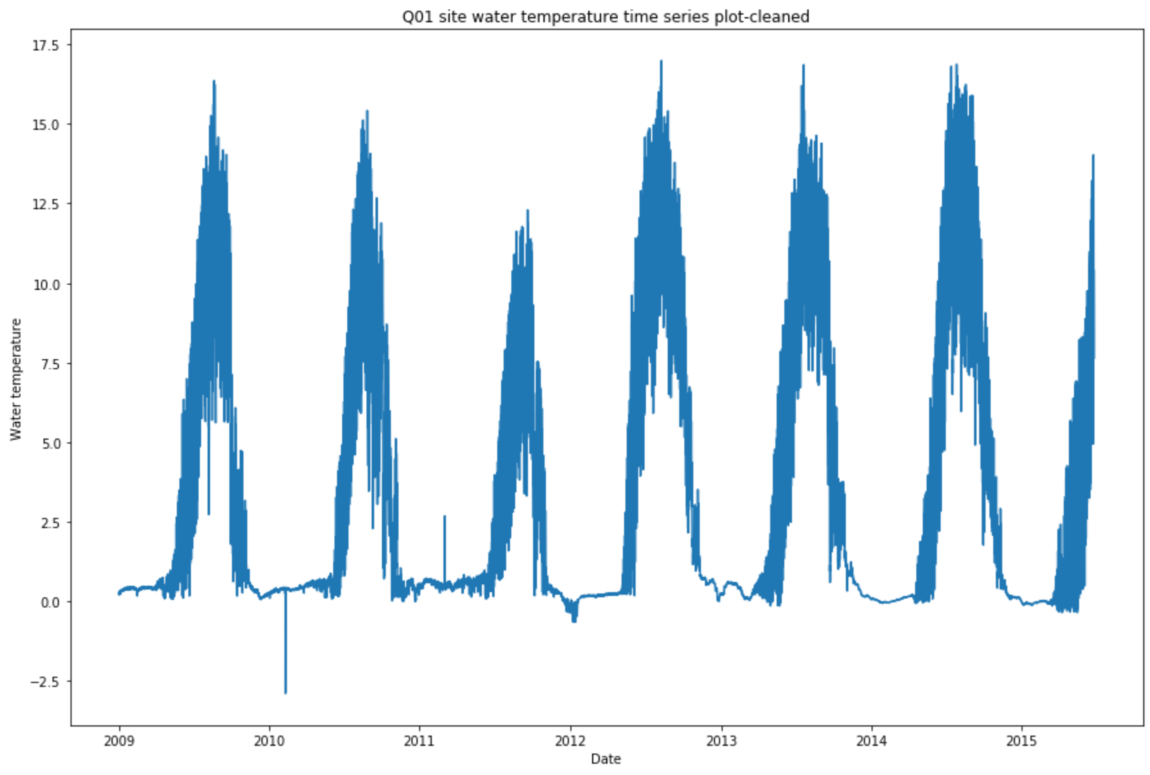
In [73]:

```
#Note that we only use the data from 2009 to 2015

plt.figure(figsize=(15,10))
plt.plot(df_c['date_time(inGMT_PDTplus7)'],df_c[' estimated_discharge(cms)'])
plt.xlabel('Date')
plt.ylabel('Discharge(cms)')
plt.title('Q01 site discharge time series plot-cleaned')

plt.figure(figsize=(15,10))
plt.plot(df_c['date_time(inGMT_PDTplus7)'],df_c[' water_temperature(deg_C)'])
plt.xlabel('Date')
plt.ylabel('Water temperature')
plt.title('Q01 site water temperature time series plot-cleaned')
```

Out[73]:

```
Text(0.5,1,'Q01 site water temperature time series plot-cleaned')
```





In [74]:

```
df_c=df_c.set_index('date_time(inGMT_PDTplus7)')
```

In [75]:

```
df_c.head()
```

Out[75]:

	raw_pressure(cm)	barocorrected_pressure(cm)	stage_in_feet(used
date_time(inGMT_PDTplus7)			
2009-01-01 00:00:00	96.2	30	
2009-01-01 00:30:00	96.7	30.3	
2009-01-01 01:00:00	96.3	29.8	
2009-01-01 01:30:00	96.1	29.3	
2009-01-01 02:00:00	96.6	29.7	

In [76]:

```
#Average for all years in different months
df_c['estimated_discharge(cms)'].groupby(lambda x : x.month).mean()
```

Out[76]:

```
1     0.308070
2     0.198115
3     0.024638
4     0.315827
5     0.931441
6     1.623214
7     1.577680
8     0.736647
9     0.294823
10    0.157225
11    0.049205
12    0.078547
Name: estimated_discharge(cms), dtype: float64
```

In [77]:

```
df_c['barocorrected_pressure(cm)']=df_c['barocorrected_pressure(cm)'].astype(float)
```

In [78]:

```
#downsampling
discharge=df_c['estimated_discharge(cms)'].resample('W').mean()
#pressure=df_c['barocorrected_pressure(cm)'].resample('M',kind='period').mean()
offset=df_c['offset_cm(cm_to_correct_solinst_to_bolt)'].resample('W').mean()
pressure=df_c['barocorrected_pressure(cm)'].resample('W').mean()
tmean=df_c['water_temperature(deg_C)'].resample('W').mean()
tmin=df_c['water_temperature(deg_C)'].resample('W').min()
tmax=df_c['water_temperature(deg_C)'].resample('W').max()
```

In [79]:

```
df_month=pd.DataFrame(discharge)
df_month['offset']=offset
df_month['pressure']=pressure
df_month['tmean']=tmean
df_month['tmin']=tmin
df_month['tmax']=tmax
df_month.columns=['discharge','offset','pressure','tmean','tmin','tmax']
```

In [80]:

```
#Resampled dataframe
df_month.head()
```

Out[80]:

	discharge	offset	pressure	tmean	tmin	tmax
date_time(inGMT_PDTplus7)						
2009-01-04	0.001777	164.8	28.613021	0.259740	0.22	0.31
2009-01-11	0.000694	164.8	28.300893	0.333839	0.28	0.38
2009-01-18	0.000077	164.8	27.811607	0.381458	0.35	0.43
2009-01-25	0.000809	164.8	27.896429	0.406696	0.37	0.43
2009-02-01	0.000770	164.8	28.546429	0.411607	0.38	0.44

In [81]:

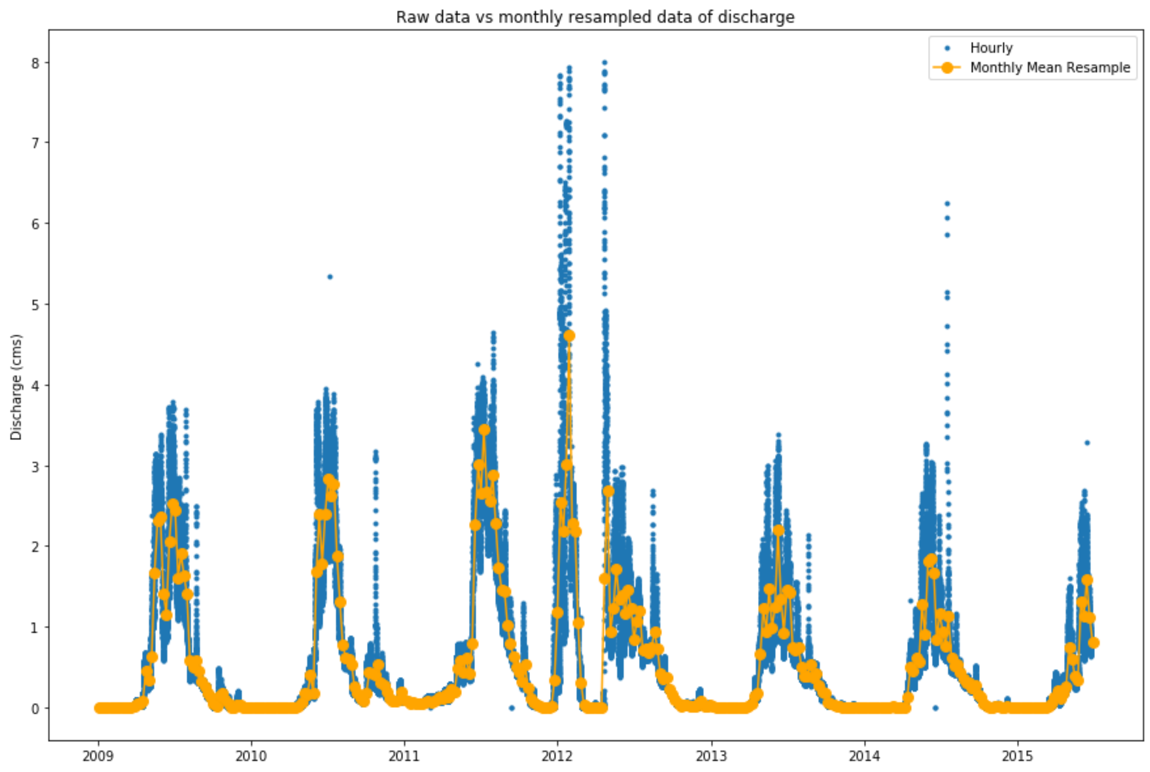
```
df_month[df_month['discharge']>=0.4].shape
```

Out[81]:

```
(128, 6)
```

In [82]:

```
fig, ax = plt.subplots(figsize=(15,10))
ax.plot(df_c.loc[:, ' estimated_discharge(cms)'],
marker='.', linestyle='None', linewidth=0.5, label='Hourly')
ax.plot(df_month.loc[:, 'discharge'],
marker='o', markersize=8, linestyle='-',color='orange', label='Monthly Mean Resample')
ax.set_ylabel('Discharge (cms)')
plt.title('Raw data vs monthly resampled data of discharge')
ax.legend();
```

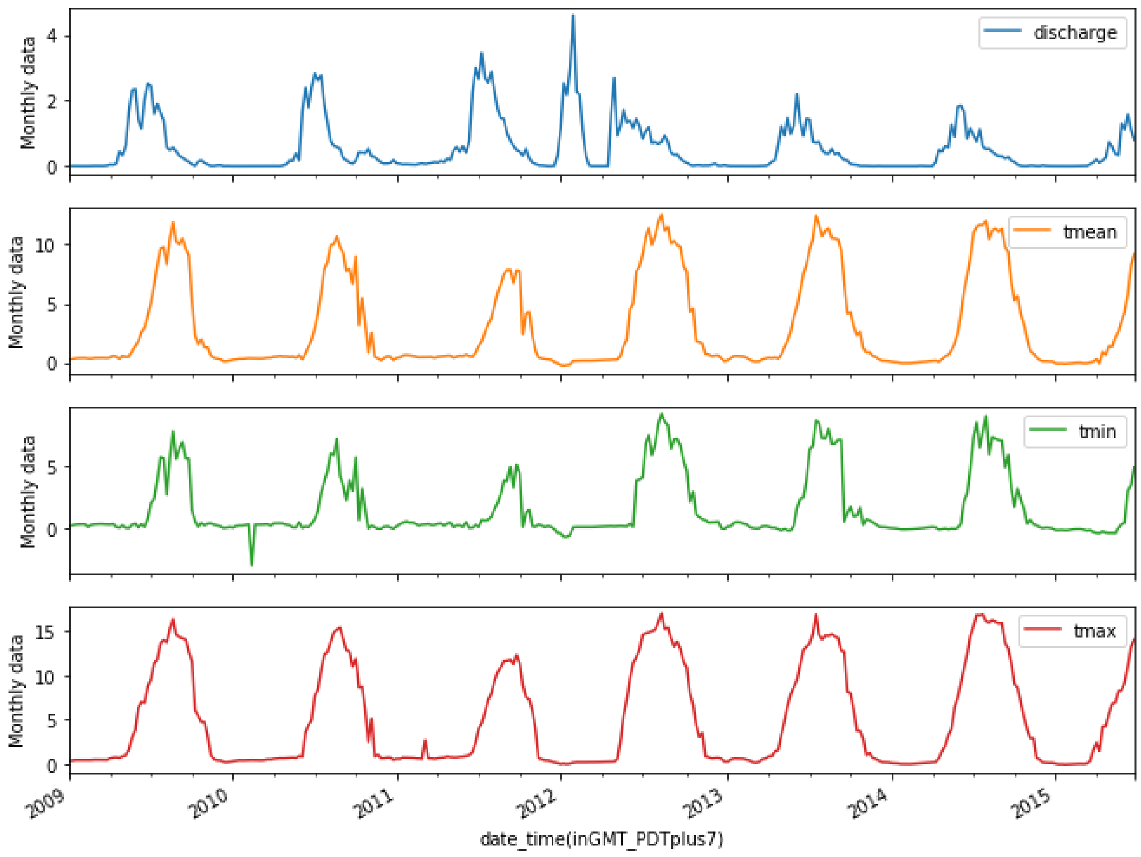


In [83]:

```
import seaborn as sns
# Use seaborn style defaults and set the default figure size
#sns.set(rc={'figure.figsize':(11, 4)})
```

In [84]:

```
cols_plot = ['discharge', 'tmean', 'tmin','tmax']
axes = df_month[cols_plot].plot( linestyle='-', figsize=(11, 9), subplots=True)
for ax in axes:
    ax.set_ylabel('Monthly data')
```



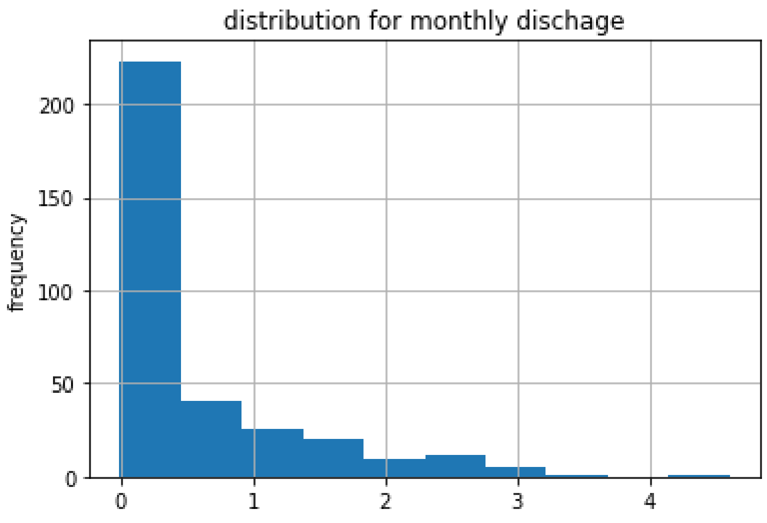
As the graph above shows there is a change happened around 2012 period. The discharge looks be smaller after 2012. By checking in detail we propose the change take place around 2012-7-29

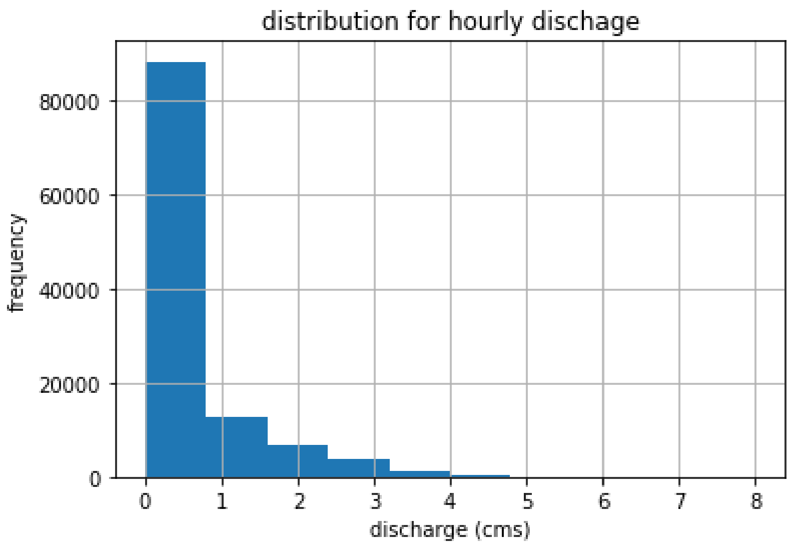
In [85]:

```
plt.figure(1)
df_month['discharge'].hist()
plt.title('distribution for monthly dischage')
plt.ylabel('frequency')

plt.figure(2)
df_c[' estimated_discharge(cms)'].hist()
plt.title('distribution for hourly dischage')
plt.xlabel('discharge (cms)')
plt.ylabel('frequency')
```

Out[85]:

```
Text(0,0.5,'frequency')
```



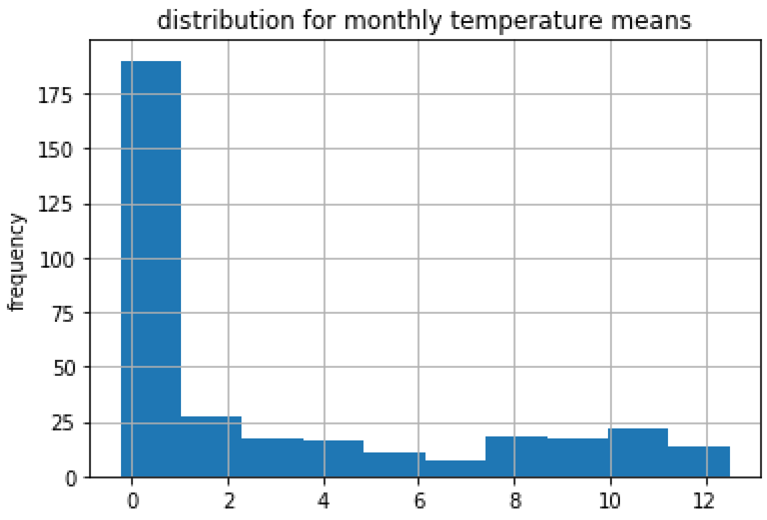


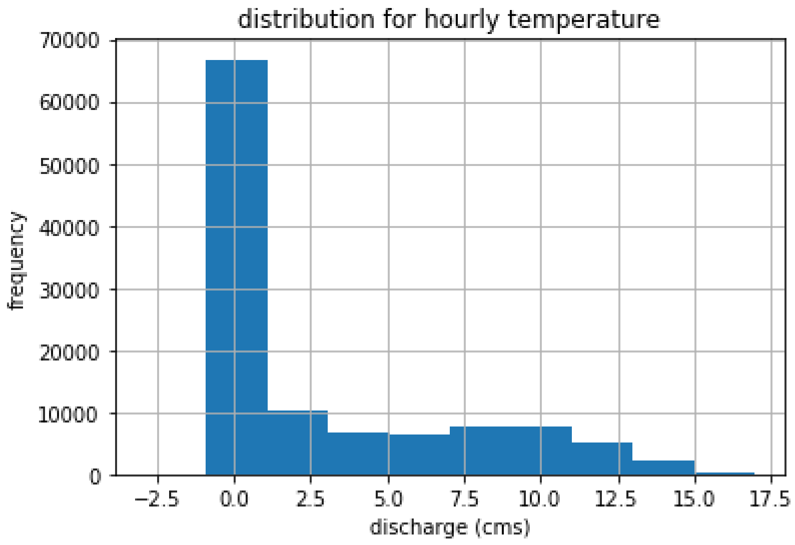
In [86]:

```
plt.figure(1)
df_month['tmean'].hist()
plt.title('distribution for monthly temperature means')
plt.ylabel('frequency')

plt.figure(2)
df_c[' water_temperature(deg_C)'].hist()
plt.title('distribution for hourly temperature')
plt.xlabel('discharge (cms)')
plt.ylabel('frequency')
```

Out[86]:

```
Text(0,0.5,'frequency')
```





Q1. In different period with different temperatures, is there any significant changes for the discharge means?

We can clearly see that the tmin before 2012 is much lower than after 2012, and also there is a change on the dischage happened around 2012, so we can propose that there is a change for the discharge means happened due to tmin change. Then we can validate it below.

In [87]:

```
df_pre=df_month[df_month.index<='2012-07-29 00:00:00']
df_aft=df_month[df_month.index>'2012-07-29 00:00:00']
```

In [88]:

```
print('before 2012 tmin mean=',df_pre['tmin'].mean())
print('before 2012 discharge mean=',df_pre['discharge'].mean())
print('after 2012 tmin mean=',df_aft['tmin'].mean())
print('after 2012 discharge mean=',df_aft['discharge'].mean())
```

```
before 2012 tmin mean= 1.258395721925133
before 2012 discharge mean= 0.6950501481104051
after 2012 tmin mean= 2.0209210526315804
after 2012 discharge mean= 0.3451053944228666
```

Then we validate if there is any significant change on the discharge by hypothesis testing, discharge after 2012 is smaller than before 2012

First, we need to create a null and an alternative hypothesis. We were told to use a two sample test, and to set α at 5%.

H0 :

$$\mu_p = \mu_l$$

H1 :

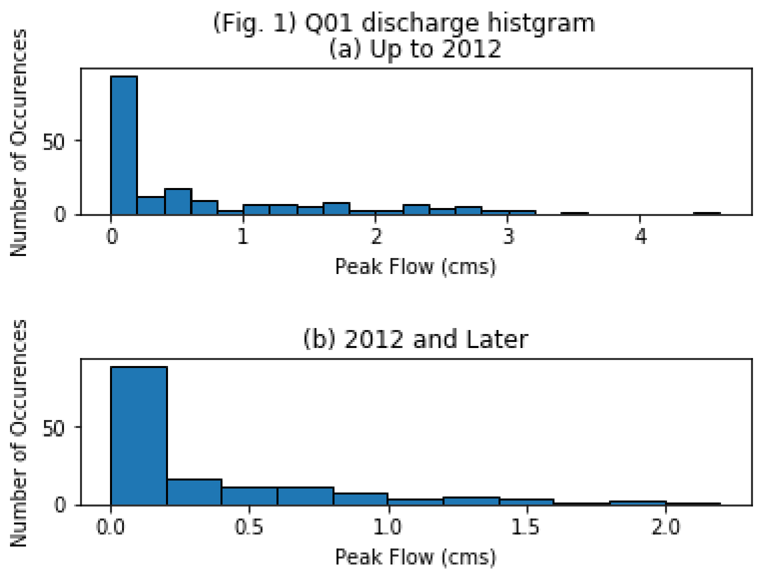
$$\mu_p < \mu_l$$

In [89]:

```
#Make a histogram for each period

bin_width = 0.2 # bin width in cfs

# Create a figure with subplots
fig1, axs = plt.subplots(2, 1)
fig1.subplots_adjust(hspace=1)
fig1.suptitle('(Fig. 1) Q01 discharge histgram')

# set bin number so that intervals are equal in these two plots
nbins = int(( np.max(df_pre['discharge']) - np.min(df_pre['discharge']) ) / bin_width )
axs[0].hist(df_pre['discharge'], nbins, ec="black")
axs[0].set_title('(a) Up to 2012')
axs[0].set_xlabel('Peak Flow (cms)')
axs[0].set_ylabel('Number of Occurences')


# set bin number so that intervals are equal in these two plots
nbins = int(( np.max(df_aft['discharge']) - np.min(df_aft['discharge']) ) / bin_width )
axs[1].hist(df_aft['discharge'], nbins, ec="black")
axs[1].set_title('(b) 2012 and Later')
axs[1].set_xlabel('Peak Flow (cms)')
axs[1].set_ylabel('Number of Occurences')
```

Out[89]:

```
Text(0,0.5,'Number of Occurences')
```



In [90]:

```
n = len(df_pre['discharge'])
m = len(df_aft['discharge'])
print(n)
print(m)
```

```
187
152
```

In [91]:

```
conf = 0.95
delta_0 = 0
z_alpha = st.norm.ppf(conf)
print("z_alpha =")
print(z_alpha)
mean2 = stats.mean(df_pre['discharge'])
mean1 = stats.mean(df_aft['discharge'])
sd2 = stats.stdev(df_pre['discharge'])
sd1 = stats.stdev(df_aft['discharge'])
pooled_sd = math.sqrt(sd1**2/m + sd2**2/n)
ztest = (mean2 - mean1 - delta_0)/pooled_sd
print("z_test =")
print(ztest)
p = 1 - st.norm.cdf(ztest)
print("p = ")
print(p)
```

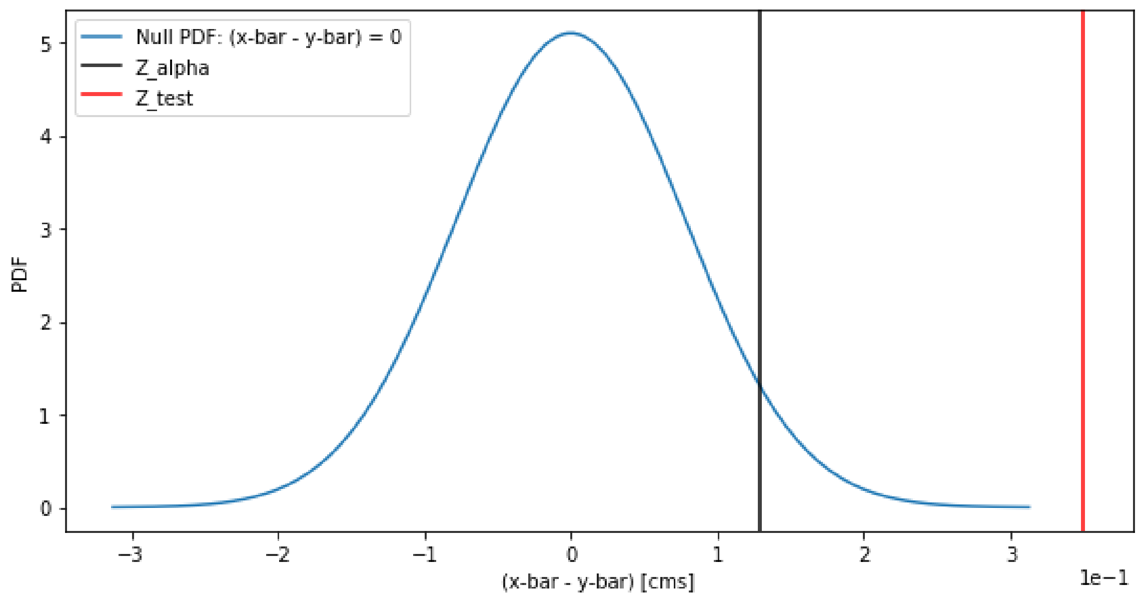
```
z_alpha =
1.6448536269514722
z_test =
4.476411763083108
p =
3.7954009569629577e-06
```

In [92]:

```
plt.figure(figsize=(10,5))
x = np.linspace(-4, 4, num=160)
x = [i * pooled_sd for i in x]
plt.plot(x, st.norm.pdf(x, 0, pooled_sd), label='Null PDF: (x-bar - y-bar) = 0')
plt.axvline(z_alpha*pooled_sd, color='black', label='Z_alpha')
plt.axvline(ztest*pooled_sd, color='red', label='Z_test')
plt.xlabel('(x-bar - y-bar) [cms]')
plt.ylabel('PDF')
plt.ticklabel_format(axis='x', style='sci', scilimits=(0,0))
plt.ticklabel_format(axis='y', style='sci', scilimits=(0,0))
plt.legend(loc='best')
```

Out[92]:

```
<matplotlib.legend.Legend at 0x1b917d39e48>
```



As the results shows, since P=3e-6<0.05, we can rejected the null hypothesis within the 95% confidence.

Which means the discharge means changed significantly between prior 2012 and after 2012 periods.

In [94]:

```
conf = 0.95
delta_0 = 0
z_alpha = st.norm.ppf(conf)
print("z_alpha =")
print(z_alpha)
mean1 = stats.mean(df_pre['tmin'])
mean2 = stats.mean(df_aft['tmin'])
sd1 = stats.stdev(df_pre['tmin'])
sd2 = stats.stdev(df_aft['tmin'])
pooled_sd = math.sqrt(sd1**2/m + sd2**2/n)
ztest = (mean2 - mean1 - delta_0)/pooled_sd
print("z_test =")
print(ztest)
p = 1 - st.norm.cdf(ztest)
print("p = ")
print(p)
```

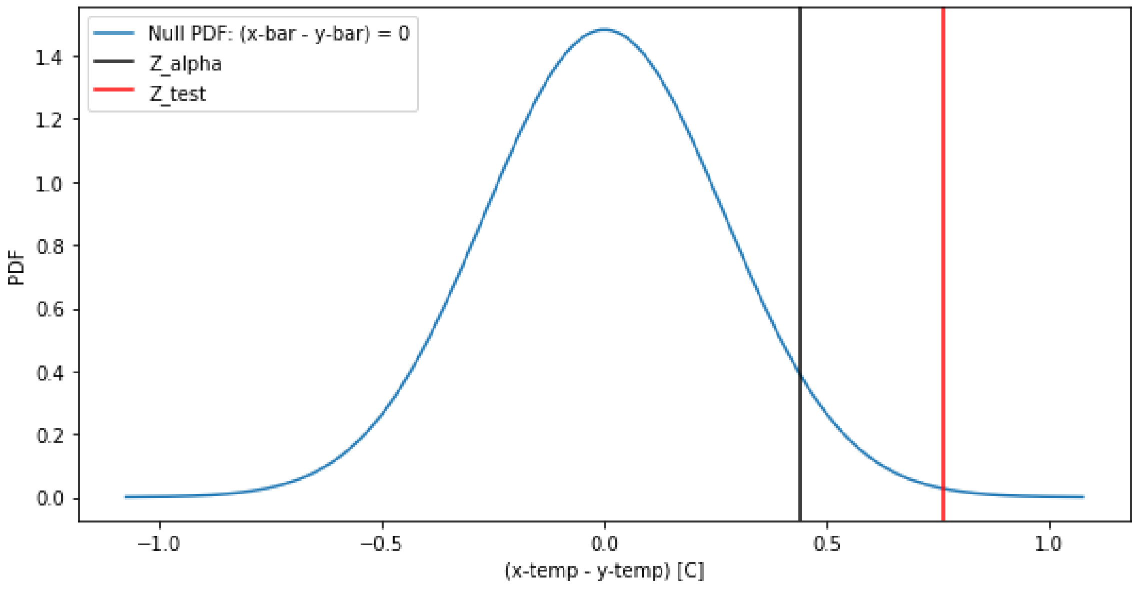
```
z_alpha =
1.6448536269514722
z_test =
2.8350409177588958
p =
0.00229099083279094
```

In [95]:

```
plt.figure(figsize=(10,5))
x = np.linspace(-4, 4, num=160)
x = [i * pooled_sd for i in x]
plt.plot(x, st.norm.pdf(x, 0, pooled_sd), label='Null PDF: (x-bar - y-bar) = 0')
plt.axvline(z_alpha*pooled_sd, color='black', label='Z_alpha')
plt.axvline(ztest*pooled_sd, color='red', label='Z_test')
plt.xlabel('(x-temp - y-temp) [C]')
plt.ylabel('PDF')
plt.ticklabel_format(axis='x', style='sci', scilimits=(0,0))
plt.ticklabel_format(axis='y', style='sci', scilimits=(0,0))
plt.legend(loc='best')
```

Out[95]:

```
<matplotlib.legend.Legend at 0x1b91989fe10>
```



As the results shows, since P=0.002<0.05, we can rejected the null hypothesis within the 95% confidence.

Which means the Tmin changed significantly between prior 2012 and after 2012 periods.Tmin after 2012 is significantly higher than before 2012.

Q2. is there any relationship between temperatures and discharge during certain periods?

Here we try to fit a model for discharge use all the variables provided.

In [96]:

```
from scipy.linalg import lstsq
```

In [97]:

```
df_month.columns
```

Out[97]:

```
Index(['discharge', 'offset', 'pressure', 'tmean', 'tmin', 'tmax'], dtype='objec
t')
```

In [98]:

```
Xmulti = np.array([df_month['tmin'],
                   df_month['tmax'],
                    df_month['tmean'],
                   ]).T
print(Xmulti.shape)
print(df_month['discharge'].shape)
```

```
(339, 3)
(339,)
```

In [99]:

```
B, res, rnk, s = lstsq(Xmulti, df_month['discharge'])
print('The trend for each variable(first one is B0)=', B)
```

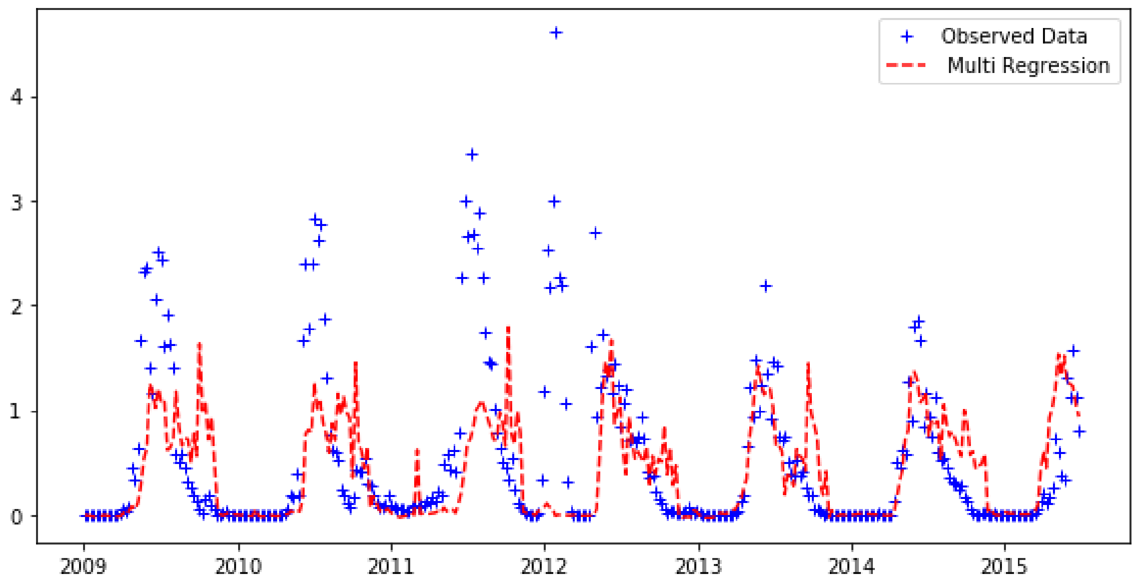
```
The trend for each variable(first one is B0)= [-0.02032358  0.29338467 -0.3351295
4]
```

In [100]:

```
plt.figure(figsize=(10,5))
plt.plot(df_month['discharge'],'b+', label='Observed Data')
plt.plot(df_month.index,Xmulti.dot(B),'r--', label=' Multi Regression')

#n = df_month['discharge'].size
#Bv2 = np.polyfit(df_month['discharge'],df_month['discharge'],1)
#print('The trend for precipitation= ', Bv2)
#x = np.linspace(np.min(df_cas['year']), np.max(df_cas['year']),n)
#y = Bv2[1] + Bv2[0]*x
#plt.plot(x,y,'k--',label='Linear Regression');

plt.legend()
```

Out[100]:

```
<matplotlib.legend.Legend at 0x1b915f1ac88>
```



The model performance enhanced significantly after taking pressure into account. Note that offset and pressure may plays more important role to the discharge change.Also according the trends, offset and pressure determines the most trend for the model (if not considering the constant viriable).

Q3. Using the Anova test to see if there is any change happens in different periods

In [101]:

```
#df_c=df_c.set_index(' date_time(inGMT_PDTplus7)')
discharge=df_c[ ' estimated_discharge(cms)'].resample('W').mean()
tmean=df_c[' water_temperature(deg_C)'].resample('W').mean()
df_week=pd.DataFrame(discharge)
df_week['tmean']=tmean
df_week.columns=['discharge', 'tmean']
df_week[df_week['discharge']<0]=0
df_week.head()
```

Out[101]:

	discharge	tmean
date_time(inGMT_PDTplus7)		
2009-01-04	0.001777	0.259740
2009-01-11	0.000694	0.333839
2009-01-18	0.000077	0.381458
2009-01-25	0.000809	0.406696
2009-02-01	0.000770	0.411607

In [102]:

```
df_week['datetime']=df_week.index
cols = df_week.columns.tolist()
cols1 = cols[-1:] + cols[:-1]
df_week1=df_week[cols1]
df_week1=df_week1.reset_index(drop = True)
df_week1.head()
```

Out[102]:

	datetime	discharge	tmean
0	2009-01-04	0.001777	0.259740
1	2009-01-11	0.000694	0.333839
2	2009-01-18	0.000077	0.381458
3	2009-01-25	0.000809	0.406696
4	2009-02-01	0.000770	0.411607

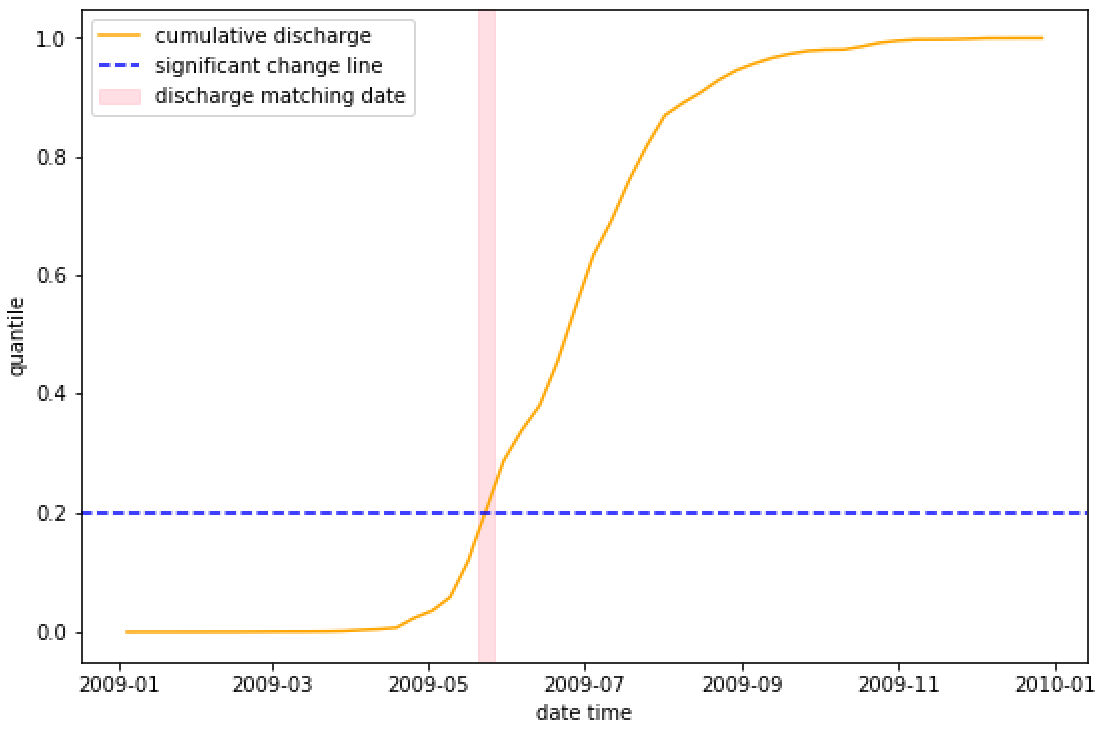
Note that we interpreted cumulative discharge to determine the periods of discharge rising. We use 20% of total cumulative discharge as the starting point. By this approach we assume that the periods we selected is according to the air temperatrue rising periods thus we can got a general idea of when was the snow began to melt which eventually led to discharge increase.

In [111]:

```
df_2009=df_week1[0:52]
from scipy.interpolate import interp1d

dsum2009=np.sum(df_2009['discharge'])
dcsum2009=np.cumsum(df_2009['discharge'])
dcp2009=dcsum2009/dsum2009

tsum2009=np.sum(df_2009['tmean'])
tcsum2009=np.cumsum(df_2009['tmean'])
tcp2009=tcsum2009/tsum2009


plt.figure(figsize=(9,6))
plt.plot(df_2009['datetime'], dcp2009, label='cumulative discharge', color='orange')
#plt.plot(df_2009['datetime'], tcp2009, label='cumulative tmean', color='cyan')
plt.axhline(0.2, color='blue', linestyle='--', label='significant change line')
plt.axvspan('2009-05-21','2009-05-27', alpha=0.5, color='pink', label='discharge matching date')
#plt.axvspan('2009-07-05','2009-07-12', alpha=0.2, color='blue', label='temperature matching dat
e')
plt.xlabel('date time')
plt.ylabel('quantile')
plt.legend(loc="best")
```

Out[111]:

```
<matplotlib.legend.Legend at 0x1b90c435cc0>
```

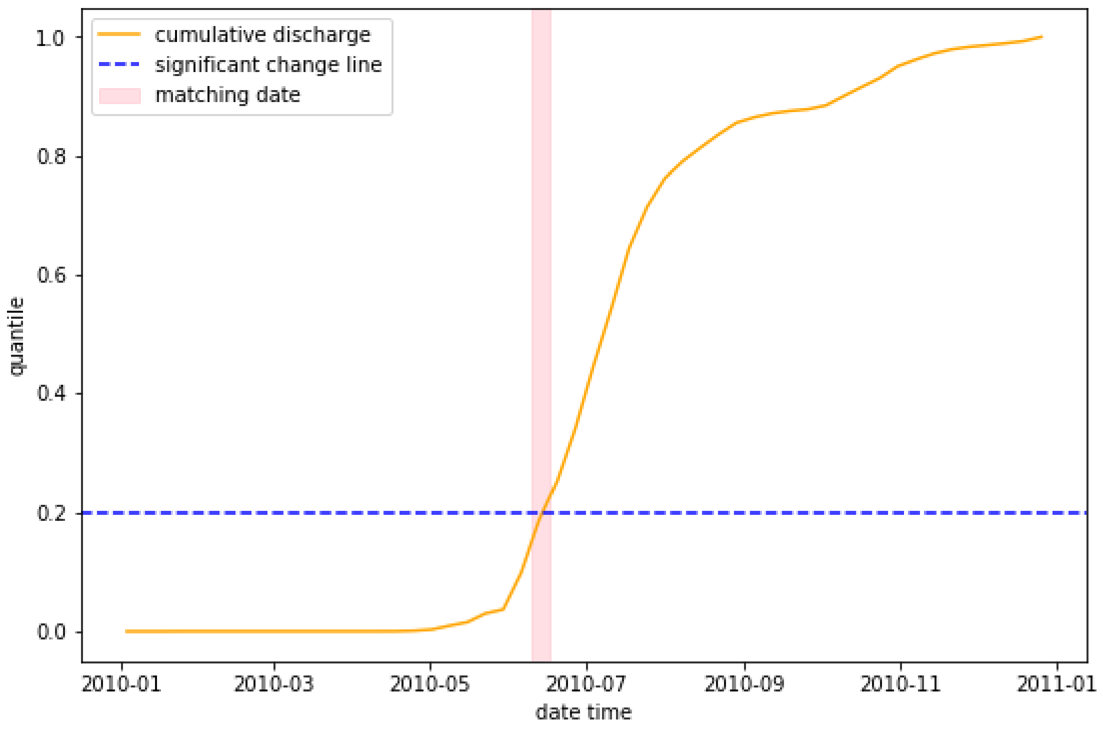


In [112]:

```
df_2010=df_week1[52:104]
dsum2010=np.sum(df_2010['discharge'])
dcsum2010=np.cumsum(df_2010['discharge'])
dcp2010=dcsum2010/dsum2010

tsum2010=np.sum(df_2010['tmean'])
tcsum2010=np.cumsum(df_2010['tmean'])
tcp2010=tcsum2010/tsum2010

plt.figure(figsize=(9,6))
plt.plot(df_2010['datetime'], dcp2010, label='cumulative discharge', color='orange')
#plt.plot(df_2010['datetime'], tcp2010, label='cumulative tmean', color='cyan')
plt.axhline(0.2, color='blue', linestyle='--', label='significant change line')
plt.axvspan('2010-06-10','2010-06-17', alpha=0.5, color='pink', label='matching date')
#plt.axvspan('2010-07-18','2010-07-25', alpha=0.2, color='blue', label='temperature matching dat
e')
plt.xlabel('date time')
plt.ylabel('quantile')
plt.legend(loc="best")
```

Out[112]:

```
<matplotlib.legend.Legend at 0x1b90c20e898>
```

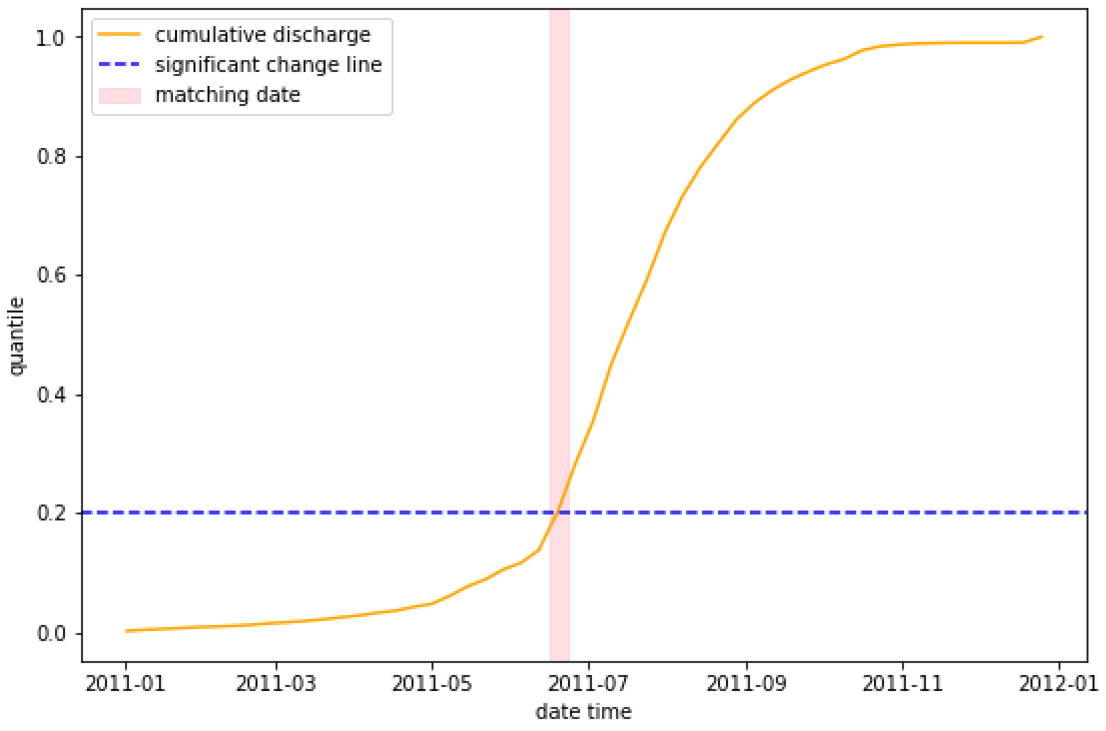


In [113]:

```
df_2011=df_week1[104:156]

dsum2011=np.sum(df_2011['discharge'])
dcsum2011=np.cumsum(df_2011['discharge'])
dcp2011=dcsum2011/dsum2011

tsum2011=np.sum(df_2011['tmean'])
tcsum2011=np.cumsum(df_2011['tmean'])
tcp2011=tcsum2011/tsum2011



plt.figure(figsize=(9,6))
plt.plot(df_2011['datetime'], dcp2011, label='cumulative discharge', color='orange')

plt.axhline(0.2, color='blue', linestyle='--', label='significant change line')
plt.axvspan('2011-06-16','2011-06-23', alpha=0.5, color='pink', label='matching date')

plt.xlabel('date time')
plt.ylabel('quantile')
plt.legend(loc="best")
```

Out[113]:

```
<matplotlib.legend.Legend at 0x1b90c2455c0>
```

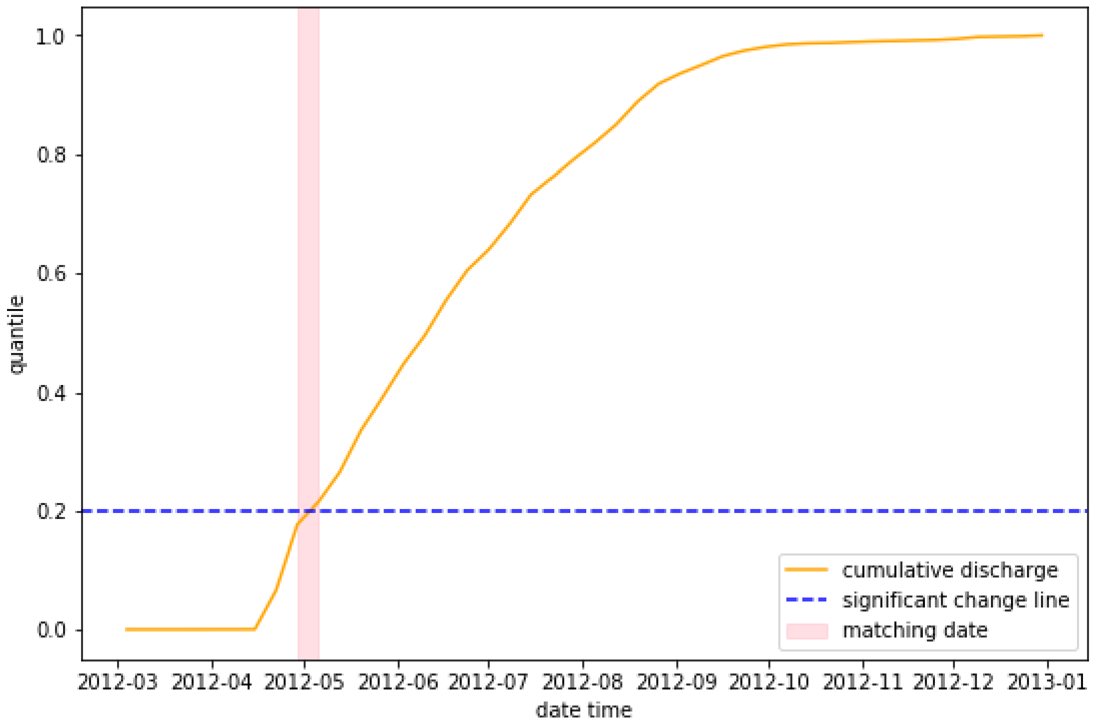


In [117]:

```
df_2012=df_week1[156:209]
df_2012=df_2012[df_2012['datetime']>='2012-03-01']
dsum2012=np.sum(df_2012['discharge'])
dcsum2012=np.cumsum(df_2012['discharge'])
dcp2012=dcsum2012/dsum2012

tsum2012=np.sum(df_2012['tmean'])
tcsum2012=np.cumsum(df_2012['tmean'])
tcp2012=tcsum2012/tsum2012


plt.figure(figsize=(9,6))
plt.plot(df_2012['datetime'], dcp2012, label='cumulative discharge', color='orange')
plt.axhline(0.2, color='blue', linestyle='--', label='significant change line')
plt.axvspan('2012-04-29','2012-05-06', alpha=0.5, color='pink', label='matching date')
plt.xlabel('date time')
plt.ylabel('quantile')
plt.legend(loc="best")
```

Out[117]:

```
<matplotlib.legend.Legend at 0x1b90d1937f0>
```

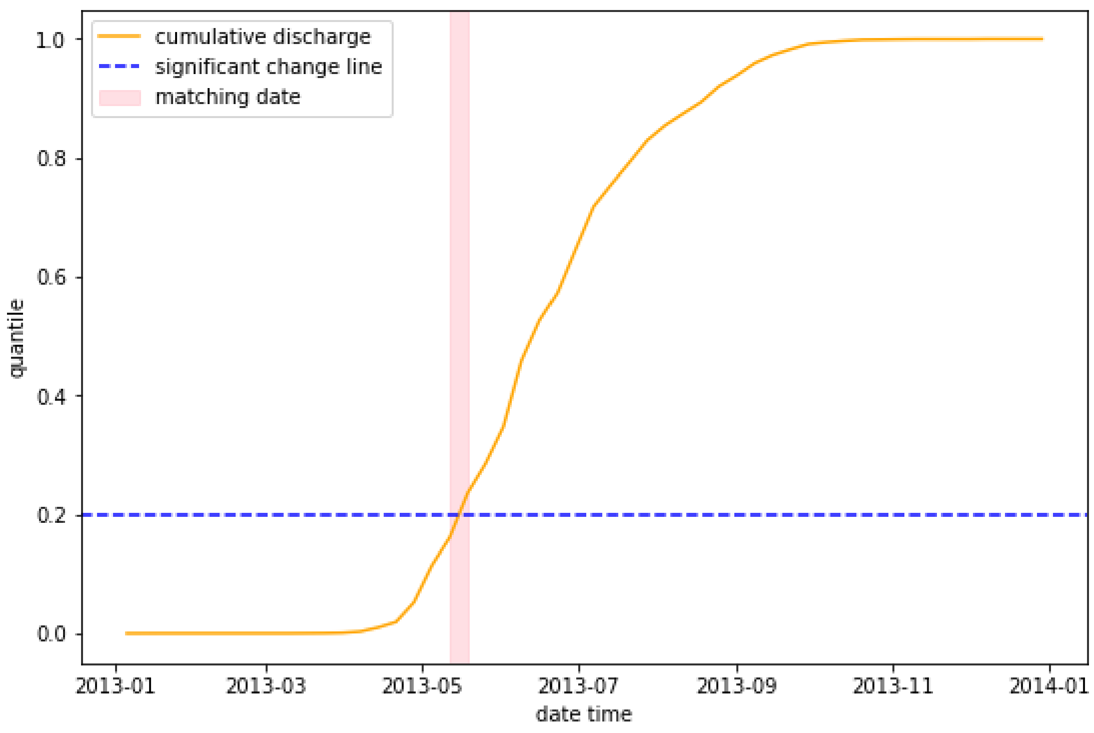


In [119]:

```
df_2013=df_week1[209:261]
dsum2013=np.sum(df_2013['discharge'])
dcsum2013=np.cumsum(df_2013['discharge'])
dcp2013=dcsum2013/dsum2013

tsum2013=np.sum(df_2013['tmean'])
tcsum2013=np.cumsum(df_2013['tmean'])
tcp2013=tcsum2013/tsum2013

plt.figure(figsize=(9,6))
plt.plot(df_2013['datetime'], dcp2013, label='cumulative discharge', color='orange')
plt.axhline(0.2, color='blue', linestyle='--', label='significant change line')
plt.axvspan('2013-05-12','2013-05-19', alpha=0.5, color='pink', label='matching date')
plt.xlabel('date time')
plt.ylabel('quantile')
plt.legend(loc="best")
```

Out[119]:

```
<matplotlib.legend.Legend at 0x1b90d20ebe0>
```

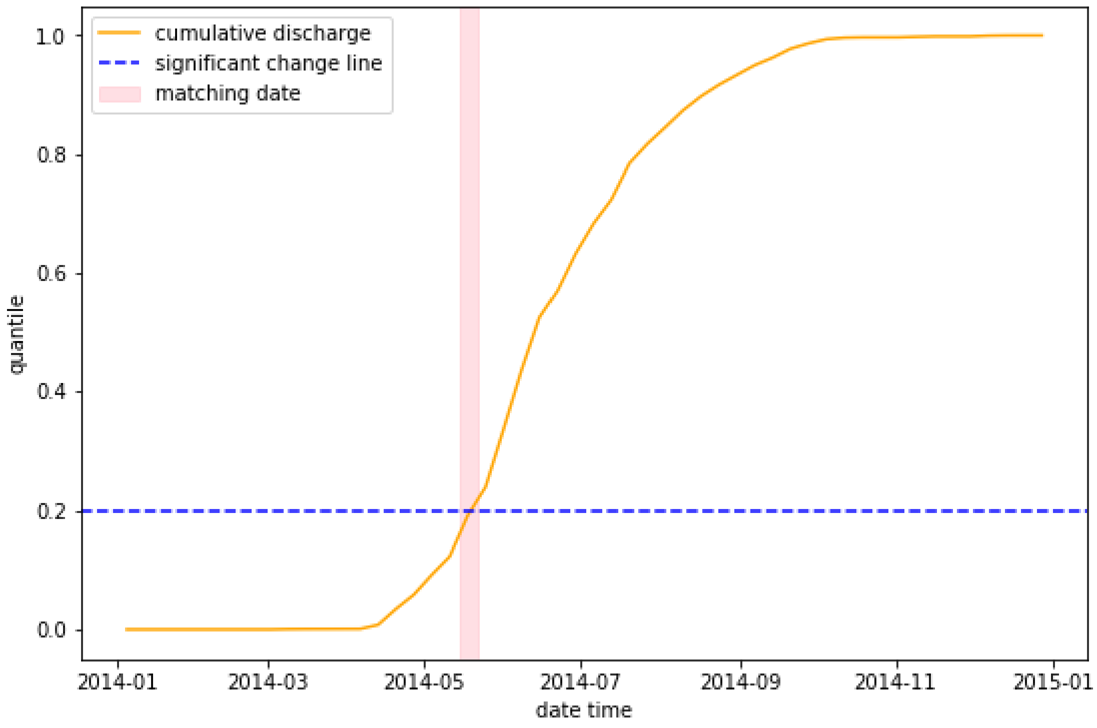


In [120]:

```
df_2014=df_week1[261:313]
dsum2014=np.sum(df_2014['discharge'])
dcsum2014=np.cumsum(df_2014['discharge'])
dcp2014=dcsum2014/dsum2014

tsum2014=np.sum(df_2014['tmean'])
tcsum2014=np.cumsum(df_2014['tmean'])
tcp2014=tcsum2014/tsum2014

plt.figure(figsize=(9,6))
plt.plot(df_2014['datetime'], dcp2014, label='cumulative discharge', color='orange')

plt.axhline(0.2, color='blue', linestyle='--', label='significant change line')
plt.axvspan('2014-05-15','2014-05-22', alpha=0.5, color='pink', label='matching date')

plt.xlabel('date time')
plt.ylabel('quantile')
plt.legend(loc="best")
```

Out[120]:

```
<matplotlib.legend.Legend at 0x1b914793780>
```



Discharge initiation date:

2009:5/24/2009

2010:6/13/2010

2011:6/20/2011

2012:5/2/2012

2013:5/15/2013

2014:5/20/2014

Here we can propose that the temperature after 2012 is slightly higher than before 2012. Then we use ANOVA test to see if there is any difference in the tmin and discharge in different periods.

In [122]:

```
df_discharge_anova=pd.read_excel('df_danova.xlsx')
df_discharge_anova.head()
```

Out[122]:

	index	years	discharge
0	0	2009	0.001777
1	1	2009	0.000694
2	2	2009	0.000077
3	3	2009	0.000809
4	4	2009	0.000770

In [123]:

```
import statsmodels.api as sm
from statsmodels.formula.api import ols
model = ols('discharge ~ C(years)', data=df_discharge_anova).fit()
anova_table = sm.stats.anova_lm(model, typ=1)
anova_table
```

Out[123]:

	df	sum_sq	mean_sq	F	PR(>F)
C(years)	5.0	8.681238	1.736248	2.815443	0.01672
Residual	307.0	189.322978	0.616687	NaN	NaN

In [124]:

```
from statsmodels.stats.multicomp import pairwise_tukeyhsd

m_comp = pairwise_tukeyhsd(endog=df_discharge_anova['discharge'], groups=df_discharge_anova['yea
rs'], alpha=0.05)
print(m_comp)
```

```
Multiple Comparison of Means - Tukey HSD, FWER=0.05
====================================================
group1 group2 meandiff p-adj  lower   upper  reject
----------------------------------------------------
  2009   2010  -0.0117    0.9 -0.4533    0.43  False
  2009   2011     0.17 0.8707 -0.2717  0.6117  False
  2009   2012   0.2932  0.398 -0.1464  0.7327  False
  2009   2013  -0.1506    0.9 -0.5923  0.2911  False
  2009   2014  -0.1747 0.8531 -0.6164  0.2669  False
  2010   2011   0.1816 0.8272   -0.26  0.6233  False
  2010   2012   0.3048 0.3516 -0.1348  0.7444  False
  2010   2013  -0.1389    0.9 -0.5806  0.3027  False
  2010   2014  -0.1631 0.8966 -0.6047  0.2786  False
  2011   2012   0.1232    0.9 -0.3164  0.5628  False
  2011   2013  -0.3206 0.2997 -0.7623  0.1211  False
  2011   2014  -0.3447 0.2234 -0.7864   0.097  False
  2012   2013  -0.4438 0.0464 -0.8833 -0.0042   True
  2012   2014  -0.4679 0.0295 -0.9075 -0.0283   True
  2013   2014  -0.0241    0.9 -0.4658  0.4176  False
----------------------------------------------------
```

As result we can see at least 2012'discharge is different from others.

We also checked mean water temperature:

In [125]:

```
df_tmean_anova=pd.read_excel('df_tanova.xlsx')
df_tmean_anova.head()
```

Out[125]:

	index	years	tmean
0	0	2009	0.259740
1	1	2009	0.333839
2	2	2009	0.381458
3	3	2009	0.406696
4	4	2009	0.411607

In [126]:

```
model = ols('tmean ~ C(years)', data=df_tmean_anova).fit()
anova_table = sm.stats.anova_lm(model, typ=1)
anova_table
```

Out[126]:

	df	sum_sq	mean_sq	F	PR(>F)
C(years)	5.0	134.031818	26.806364	1.722216	0.129165
Residual	307.0	4778.467539	15.565041	NaN	NaN

In [127]:

```
m_comp = pairwise_tukeyhsd(endog=df_tmean_anova['tmean'], groups=df_tmean_anova['years'], alpha=
0.05)
print(m_comp)
```

```
Multiple Comparison of Means - Tukey HSD, FWER=0.05
====================================================
group1 group2 meandiff p-adj  lower  upper  reject
----------------------------------------------------
  2009   2010  -0.3627    0.9 -2.5816 1.8562  False
  2009   2011  -0.9882 0.7711 -3.2071 1.2307  False
  2009   2012   0.8318 0.8843 -1.3766 3.0402  False
  2009   2013   0.5513    0.9 -1.6676 2.7702  False
  2009   2014    0.753    0.9 -1.4659 2.9719  False
  2010   2011  -0.6255    0.9 -2.8444 1.5934  False
  2010   2012   1.1945 0.6136 -1.0139 3.4029  False
  2010   2013    0.914 0.8262 -1.3049 3.1329  False
  2010   2014   1.1157 0.6763 -1.1032 3.3346  False
  2011   2012     1.82 0.1728 -0.3884 4.0284  False
  2011   2013   1.5395  0.351 -0.6794 3.7584  False
  2011   2014   1.7412 0.2181 -0.4777 3.9601  False
  2012   2013  -0.2805    0.9 -2.4889 1.9279  False
  2012   2014  -0.0788    0.9 -2.2872 2.1296  False
  2013   2014   0.2017    0.9 -2.0172 2.4206  False
----------------------------------------------------
```

As result we can see there isn't a significant difference in the mean temperature.

Conclusion

1. As tested in part 1, we can see that there is a significant change happened around 2012 on both discharge and tmin
2. Part 2 verified that the discharge can be predicted by some extent by temperature and tmean can explain most of the trend for the discharge change.
3. In part 3 we used ANOVA test to dig the relationship between air temperature and discharge, however we didn't get a positive result.

This may because the dataset is not large enough for us to really figure out the behavior of this site. We are planning to use larger dataset to develop the same analysis and compare that with this one to illustrate how will both size of data and real size of the measured site effect the statistical analysis. Also, this time using this small dataset, we developed our pipeline towards the project.

The next step for us is to really double check the physical process and confirm the questions to solve. Also as mentioned above, using a larger dataset to assess our existing pipeline and update it.