Validating Equations

Some open questions:

1. Clarify: is q__n__t either the short term or long term moving average?
2. how are N__s and N__l determined? L__l and L__s ? Q1 and Q2 ?
3. Are equation 4 inequalities mixed up?
4. Should equation 2 have q__s instead of q__l?
5. How are the 10 unknowns estimated via the two part sum squared error?
6. Authors say "q__j = sum of Trenton, Schuylkill, and PST model inflow on day j" … is our inflow data already in terms of daily total flow?

10 unknowns below – we would just plug into R to find ad-hocly with two part sum square error?

a. d__s / d__l = decay coefficient for short-term / long-term moving average flow;

b. a__s / a__l = constant for short-term / long-term moving average flow relationship;

c. P__s / P__l = power coefficient for short-term / long-term moving average flow relationship;

d. C__s / C__l = offset for short-term / long-term-term mov-ing average flow relationship;

e. tao__s / tao__l = tidal coefficient for short-term / long-term term moving average flow relationship;

10 unknowns below – we need to ask the authors how they were derived?

k__st / k__lt = short-term / long term moving average flow SC at time t; -> calculated via equation (2) / (3)

q__st / q__lt = short-term / long term moving average flow at time t;

N__s / N__l = number of days for short term / long term

Q__1 / Q__2 = maximum flow at which k__st / k__lt contributes to K__t;

L__s / L__l = short-term / long-term maximum 1-day change in SC;

Reading in Ben Franklin Bridge SC data with Delaware River Inflow data to use as a first pass cross check with Figure 3 in Meyer paper

```r
library(data.table)
library(tidyr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:data.table':
##
##     between, first, last

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
dat = fread('../data/processed/training_data.csv')

# use this to pull in actual meyer inputs
meyer_inputs <- fread('../data/raw/meyer_raw_data.csv')
meyer_inputs$Date = as.Date(meyer_inputs$Date)
meyer_water_level = meyer_inputs[, c('Date', 'water_level')]

# use this for SC values
bfb = dat[location == 'del_bfb']
bfb <- bfb %>% drop_na(mean_sc)

# convert inflow data back to cubic feet per second and convert sea level (meters) to feet...
#will revisit this to confirm raw tide data was provided in meters
input_data <- dat %>% group_by(date) %>%
  summarise(inflow = sum(inflow, na.rm = TRUE)/0.0283168, #convert inflow from cmps to cfps
                        reedy_sl = ((reedy_sl- 49.229)/12)) # convert reedy_sl from inches to feet
```

```
## `summarise()` regrouping output by 'date' (override with `.groups` argument)
```

```r
input_data$date = as.Date(input_data$date)
input_data = merge(input_data, meyer_water_level, by.x = 'date', by.y = 'Date')
```

Initalize parameter values based on table of values in Meyer paper

```r
#n_s = 10
#n_l = 30
#Q1 = 2800  # they provide Q1 / Q2 in cfps; our raw inflows are in cubic meters per second
#Q2 = 4800

#as = 355000
#Ps = -0.8998
#Cs = 132.08
#ds = .19
#Ls = 3
#tao_s = 35312.54

#al = 2600000
#Pl = -1.06
#Cl = 118.2
#dl = .0426
#Ll = 5.5
#tao_l = -2544.23


n_s = 10
n_l = 30
thresh_10 = 3700  # they provide Q1 / Q2 in cfps; our raw inflows are in cubic meters per second
thresh_30 = 3000

as = 52352.7976
Ps = -0.654734
Cs = 105.45053
ds = .80988481
Ls = 3
tao_s = 35312.54
```

```
al = 37912467
Pl = -1.394098647
Cl = -.434095416
dl = 1.912292411
Ll = 5.5
tao_l = -2544.23
```

Based on this data, 1964-12-15 is the first day we can make a prediction of K (31 days into the data). So, we need to pick an initial condition to test the PBEM model. Based on the rolling average values of mean sc leading up to those dates, we will choose 640 as the initial k__st value and 884 as the initial k__lt value.

```
library(zoo)
```

```
##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
# is it right to calculate the q_st and q_lt values as rolling averages
input_data <- input_data %>% mutate(q_st = rollmean(x = inflow, n_s, align = 'right', fill = NA), # n_s
                          q_lt = rollmean(x = inflow, n_l, align = 'right', fill = NA), # n_l = 30 days
                          power_10d = as*lag(q_st)^(Ps) + Cs,
                          power_30d = al*lag(q_lt)^(Pl) + Cl)
```

Some of the sea level days of data are missing, which breaks equations (2) and (3) when we multiply by the tide values. So as a first pass I just imputed the missing days of tide data with the previous non-NA value.

```
input_data <- input_data %>% drop_na(q_lt) %>%
          filter(date >= '1964-11-15' & date <= '2017-06-04') %>%
  fill(reedy_sl)
```

Initialize 1964-12-15 values of k__st and k__lt based on rolling average method and run the for loop to calculate the subsequent values of k__st and k__lt.

```
st_k_values = c(558)
lt_k_values = c(924)


input_data = as.data.table(input_data)
# loop through all rows of data after the first initial row
for (t in (2:nrow(input_data))){

  # store the previous time step slice of data
  t_minus1 = input_data[t-1]
  t_0 = input_data[t]

  # set k_st equal to equation 2 -- note this assumes that equation (2) is supposed to have q_lt rather
  k_st <- st_k_values[t-1] + min(Ls, ds*(t_0$power_10d - st_k_values[t-1]) + tao_s/t_0$q_st*t_0$reedy_s

  # set k_lt equal to equation 3
  k_lt <- lt_k_values[t-1] + min(Ll, dl*(t_0$power_30d - lt_k_values[t-1]) + tao_l/t_0$q_st*t_0$reedy_s

  st_k_values <- append(st_k_values, k_st)
  lt_k_values <- append(lt_k_values, k_lt)
}
```

Add the k_st and k_lt values to our existing data

```
input_data <- input_data %>% mutate(model_10d = st_k_values,
                                    model_30d = lt_k_values)
```

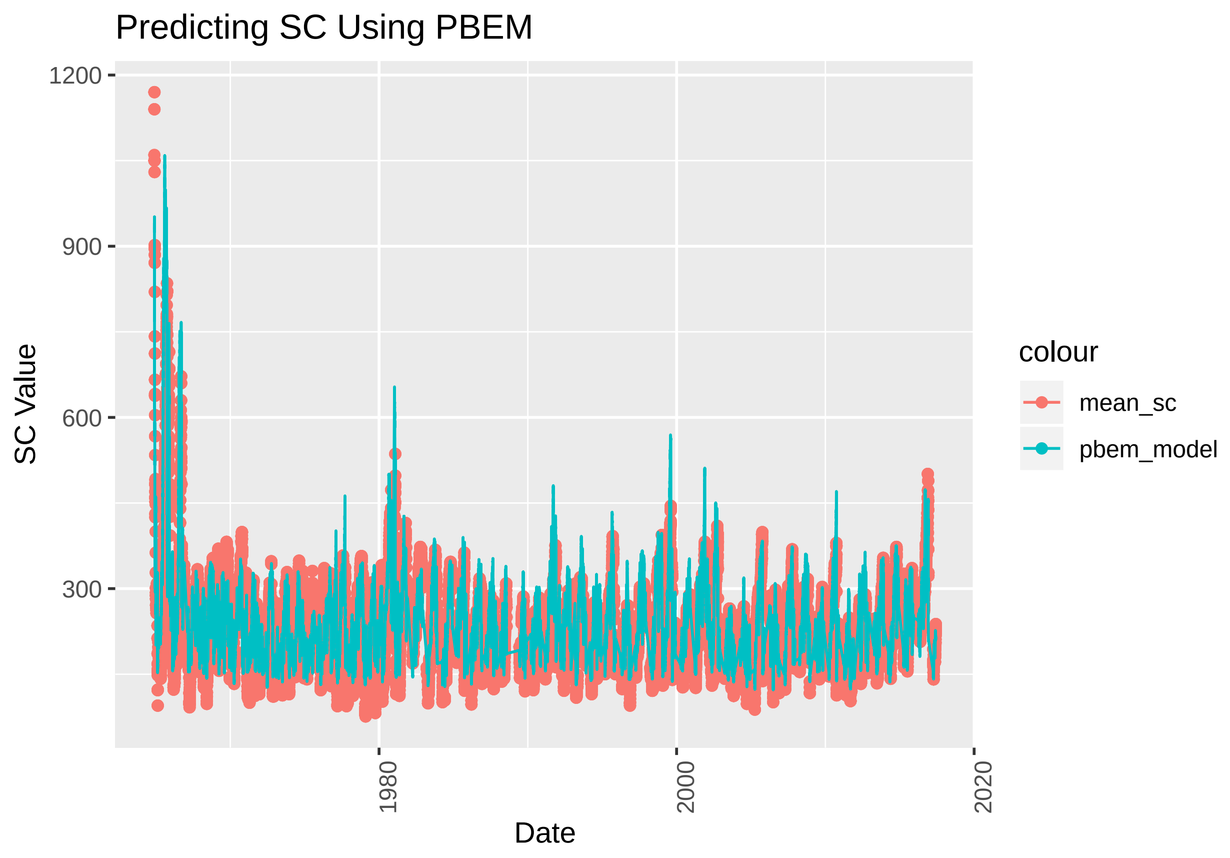
Finally, add the logic to predict capital K depending on the relationship between q and Q

```
# ql_t <= Q1       -> k_st (maybe authors have this swapped?)
# ql_t >= Q2       -> k_lt (maybe authors have this swapped?)
# Ql < qlt <= Q2 ->
#bf <- bf %>% mutate(K_hat = ifelse(q_lt <= Q1, k_lt,
 #                          ifelse(q_lt >= Q2, k_st, k_st*(q_lt - Q1)/(Q2 - Q1) + k_lt*(Q2 - q_lt)/(Q2
  #                   inequality = ifelse(q_lt <= Q1, 'first',
   #                         ifelse(q_lt >= Q2, 'third', 'second')))




#preds = c()
#for (t in (1:nrow(input_data))){
#  time_step = input_data[t]
#  p = time_step$model_10d*max(0, min(1, (time_step$q_lt - thresh_30)/ 700)) + time_step$model_30d*min(
#  preds = append(preds, p)
#}

final_output <- input_data %>%
                select(date, q_lt, model_10d, model_30d) %>%
                group_by(date) %>%
                mutate(model_output = model_10d*max(0, min(1, (q_lt - thresh_30)/ 700)) + model_30d*min
  ungroup()

#final_output$model_output = preds
```

Cross validate K_hat (predicted SC value) with mean_sc (actual SC values)

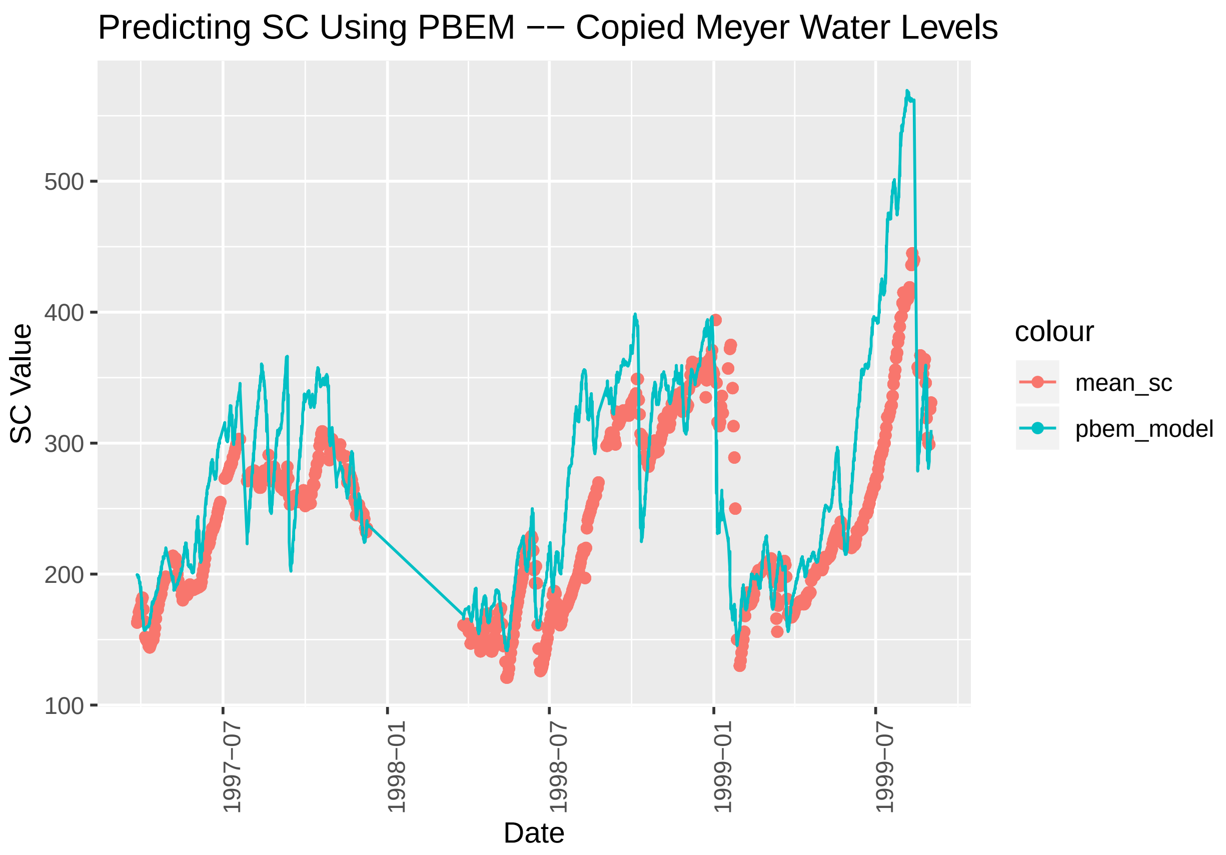
```
bfb$date = as.Date(bfb$date)
#copycat = merge(final_output, bfb[, c('date', 'mean_sc')])
risq_attempt = merge(final_output, bfb[, c('date', 'mean_sc')])
```

```
library(ggplot2)
risq_attempt$date = as.Date(risq_attempt$date)


ggplot()+
  geom_point(data = risq_attempt, aes(x = date, y = mean_sc, color = 'mean_sc')) +
  geom_line(data = risq_attempt, aes(x = date, y = model_output, color = 'pbem_model')) +
  xlab('Date') +
  ylab('SC Value')+
  ggtitle('Predicting SC Using PBEM')+
  theme(axis.text.x = element_text(angle = 90))
```



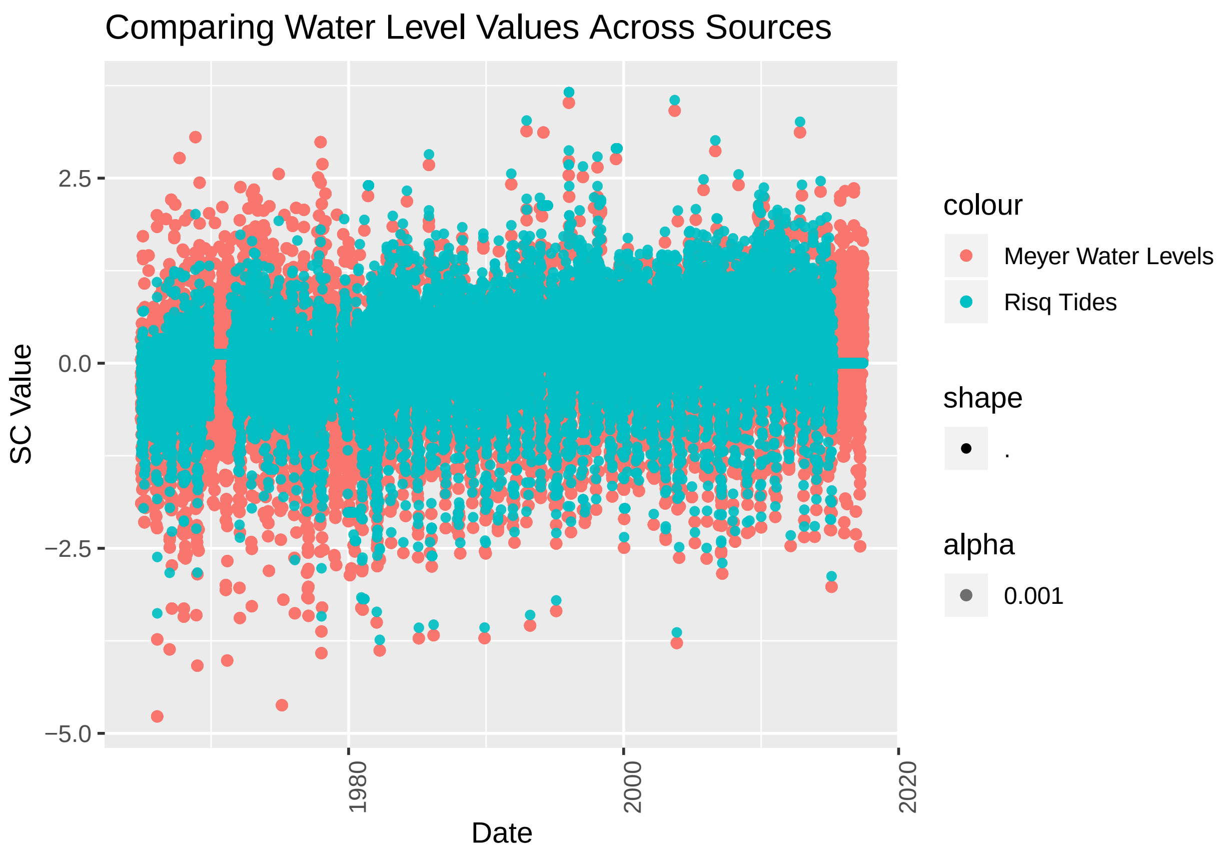
```r
  #ylim(-800,800)
  #scale_x_continuous(name ="Date", breaks=c('1964-11-01', '1966-11-01'))
```

```r
drill_down <- risq_attempt %>% filter(date >= '1997-01-01' & date <= '1999-09-01')

ggplot()+
  geom_point(data = drill_down, aes(x = date, y = mean_sc, color = 'mean_sc')) +
  geom_line(data = drill_down, aes(x = date, y = model_output, color = 'pbem_model')) +
  xlab('Date') +
  ylab('SC Value')+
  ggtitle('Predicting SC Using PBEM -- Copied Meyer Water Levels')+
  theme(axis.text.x = element_text(angle = 90))
```



```r
#meyer_output = copycat$model_output
#risq_output = risq_attempt$model_output

#compare = data.frame(meyer_output, risq_output)
```

```r
ggplot()+
  geom_point(data = input_data, aes(x = date, y = water_level, color = 'Meyer Water Levels', )) +
  geom_point(data = input_data, aes(x = date, y = reedy_sl, color = 'Risq Tides', alpha = .001, shape =
  xlab('Date') +
  ylab('SC Value')+
  ggtitle('Comparing Water Level Values Across Sources')+
  theme(axis.text.x = element_text(angle = 90))
```


Comparing Water Level Values Across Sources
SC Value
2.5
0.0
−2.5
−5.0
1980
2000
2020
Date
colour
Meyer Water Levels
Risq Tides
shape
.
alpha
0.001