calibration_assignment2

AUTHOR
Tom Gibbens-Matsuyama

Calibration Assignment 2

```r
library(tidyverse)
library(here)

# read metric function
source(here("R/compute_highflowmetrics_all.R"))

# load model results from last class
msage = readRDS(here("Data/msage.RDS"))

# first rearrange so we can plot all results
msagel = msage %>% gather(key="sim",value="str", -date, -month, -day, -year, -wy,-obs)

# apply our accurracy measure to each parameter set
res <-  msage %>% select(-date, -month, -day, -year, -wy, -obs ) %>%
  apply(2, compute_highflowmetrics_all, o = msage$obs,
            month=msage$month, year=msage$year, day=msage$day, wy=msage$wy)
results = as.data.frame(matrix(unlist(res), byrow=T, ncol=5))
colnames(results)=c("annual_max_err","annual_max_cor", "high_month_err", "high_month_cor", "combi
results$sim <- msage %>% select(-date, -month, -day, -year, -wy, -obs ) %>% colnames()
```

Accuracy Measure

```r
summary(results$combined)
```

```
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
0.3668  0.5739  0.6772  0.6745  0.7878  0.8949 
```

```r
# 1) selecting behaviorial or acceptable parameters sets

threshold <- 0.3
results_acc <- subset(results, combined > threshold)
head(results_acc)
```

```
  annual_max_err annual_max_cor high_month_err high_month_cor  combined sim
1      6.9345354      0.9014004     42.8537530      0.9141555 0.5310607  V1
2      1.1514932      0.8985495      6.0981289      0.9502652 0.8586040  V2
3      0.8229786      0.8907377    -26.3129075      0.8983071 0.7846757  V3
4      2.2712197      0.9170244     -0.4800676      0.9369708 0.8057301  V4
```

```
5      2.4604070      0.8911067     -4.9528872      0.9369032 0.7682145   V5
6     -1.8622268      0.8817513    -50.3522647      0.8705280 0.6072294   V6
```

```
# as an alternative  what if you want the top N parameter sets
topN <- 50
tmp <- results[order(results$combined, decreasing = T), ]
results_acc <- tmp[1:topN, ]
head(results_acc)
```

```
   annual_max_err annual_max_cor high_month_err high_month_cor  combined sim
87      0.7259666      0.8752264     -0.3585295      0.9095890 0.8949480 V87
89      0.2769788      0.8770844    -12.6186279      0.9044074 0.8754783 V89
20      1.1147148      0.8905123     -2.3570769      0.9199460 0.8666259 V20
2       1.1514932      0.8985495      6.0981289      0.9502652 0.8586040  V2
98      1.1892488      0.8758119     -3.8805810      0.9089188 0.8489358 V98
12      1.1889142      0.9166257      9.7859996      0.9525483 0.8462532 V12
```

Defining Weights

```
# create a weight for each parameter set based on its relative accuracy - we do this so all weigh
max_acc <- max(results_acc$combined)
min_acc <- min(results_acc$combined)

# normalize weights
results_acc$w_acc <- (results_acc$combined - min_acc) / (max_acc - min_acc)

# make sure weights sum to one
sum_acc <- sum(results_acc$combined)
results_acc$wt_acc <- results_acc$combined / sum_acc

# look at values
summary(results_acc$wt_acc)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
0.01724 0.01933 0.02010 0.02000 0.02076 0.02277 
```

```
# check to see that they sum to one
sum(results_acc$wt_acc)
```

```
[1] 1
```

```
Nacc <- nrow(results_acc)
Nacc
```

```
[1] 50
```

MLE

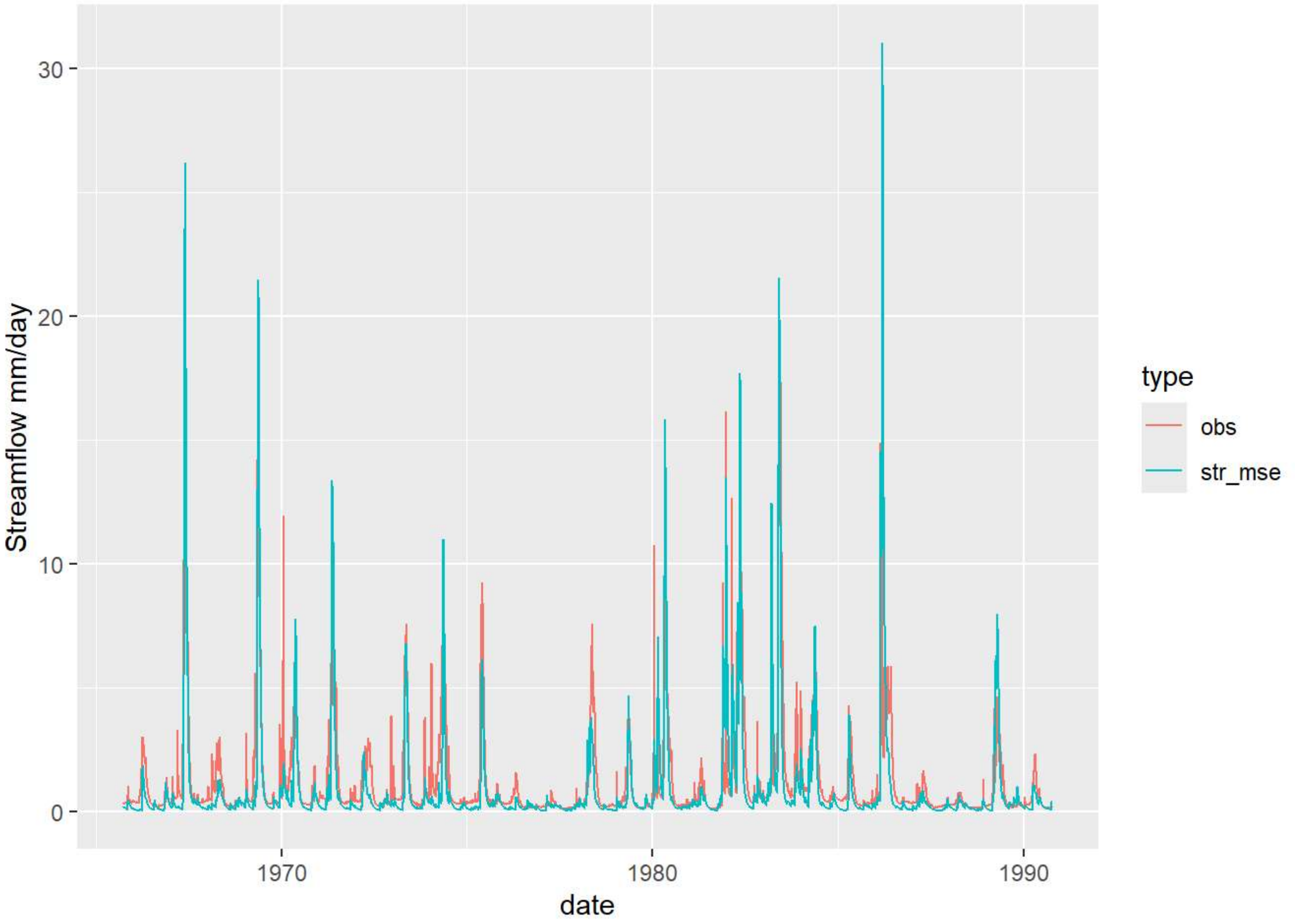
```r
# subset only acceptable runs
msagel_acc <- subset(msagel, sim %in% results_acc$sim)
# join with weights from res_acc, left_join will repeat weights for each day in streamflow trajec
msagel_acc <- left_join(msagel_acc, results_acc, by = "sim")
head(msagel_acc)
```

```
        date month year day   wy       obs sim       str annual_max_err
1 1965-10-01    10 1965   1 1966 0.3358678  V2 0.3316747       1.151493
2 1965-10-02    10 1965   2 1966 0.3208737  V2 0.3179167       1.151493
3 1965-10-03    10 1965   3 1966 0.3058796  V2 0.3047440       1.151493
4 1965-10-04    10 1965   4 1966 0.2968832  V2 0.2921237       1.151493
5 1965-10-05    10 1965   5 1966 0.2968832  V2 0.2800427       1.151493
6 1965-10-06    10 1965   6 1966 0.2968832  V2 0.2684613       1.151493
  annual_max_cor high_month_err high_month_cor combined     w_acc     wt_acc
1      0.8985495       6.098129      0.9502652 0.858604 0.8326761 0.02184062
2      0.8985495       6.098129      0.9502652 0.858604 0.8326761 0.02184062
3      0.8985495       6.098129      0.9502652 0.858604 0.8326761 0.02184062
4      0.8985495       6.098129      0.9502652 0.858604 0.8326761 0.02184062
5      0.8985495       6.098129      0.9502652 0.858604 0.8326761 0.02184062
6      0.8985495       6.098129      0.9502652 0.858604 0.8326761 0.02184062
```

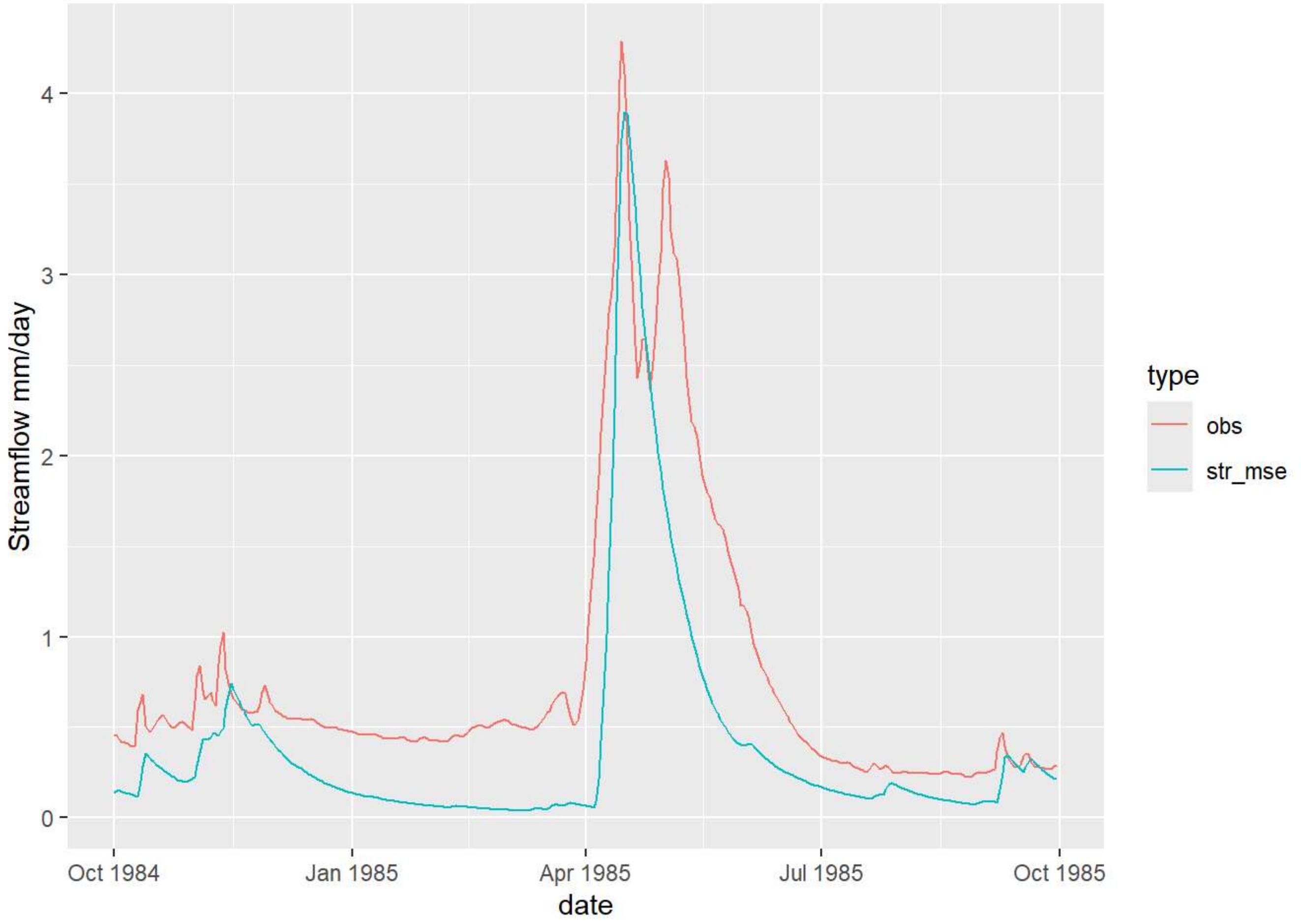
```r
# finally multiply flow by weight
msagel_acc <- msagel_acc %>% mutate(str_wt = str * wt_acc)

# now we can average streamflow for each day from all the runs # using the weights
aver_flow <- msagel_acc %>%
  group_by(date) %>%
  dplyr::summarize(str_mse = sum(str_wt))

# add some date information
aver_flow <- left_join(aver_flow, msage[, c("date", "month", "year", "day", "wy", "obs")],
  by = c("date")
)

# plot
aver_flowl = aver_flow %>% pivot_longer(
  cols = c(str_mse, obs),
  values_to = "str", names_to = "type"
)
```

Plotting MLE

```r
ggplot(aver_flowl, aes(x = date, y = str, col=type)) +
  geom_line()+
  labs(y = "Streamflow mm/day")
```



```
# add some of the other date info and plot a subset
wycheck <- 1985
ggplot(subset(aver_flowl, wy == wycheck), aes(x = date, y = str, col=type)) +  labs(y = "Streamfl
```



Monthly High MLE Flow

```
# we could also calculate mse for august flow
# get monthly flow

# summarize monthly flow
msagel_month <- msagel %>%
  group_by(month, wy, sim) %>%
  dplyr::summarize(str = sum(str), obs = sum(obs))

# extract only acceptable runs

msagel_acc_month <- subset(msagel_month, sim %in% results_acc$sim)
# join with weights from res_acc, left_join will repeat weights for each month in streamflow traj
msagel_acc_month <- left_join(msagel_acc_month, results_acc, by = "sim")


# now MSE for monthly flow
mse_flow <- msagel_acc_month %>%
  group_by(wy, month) %>%
  dplyr::summarize(str_mse = sum(str*wt_acc), obs = mean(obs))

# For comparison, compute MSE for all simulations
```

```
mse_flow_all <- msagel_month %>%
  group_by(wy, month) %>%
  dplyr::summarize(str_mse = mean(str))

mse_flow_all = left_join(mse_flow_all, mse_flow, by = c("wy", "month"), suffix = c("_all", "_mse"

# see the impact of calibration
tmp = mse_flow_all %>% subset(month==8)
cor(tmp$str_mse_all, tmp$obs)
```

```
[1] 0.737085
```

```
cor(tmp$str_mse_mse, tmp$obs)
```

```
[1] 0.7097067
```

```
# but what about across all months
cor(mse_flow_all$str_mse_all, mse_flow_all$obs)
```

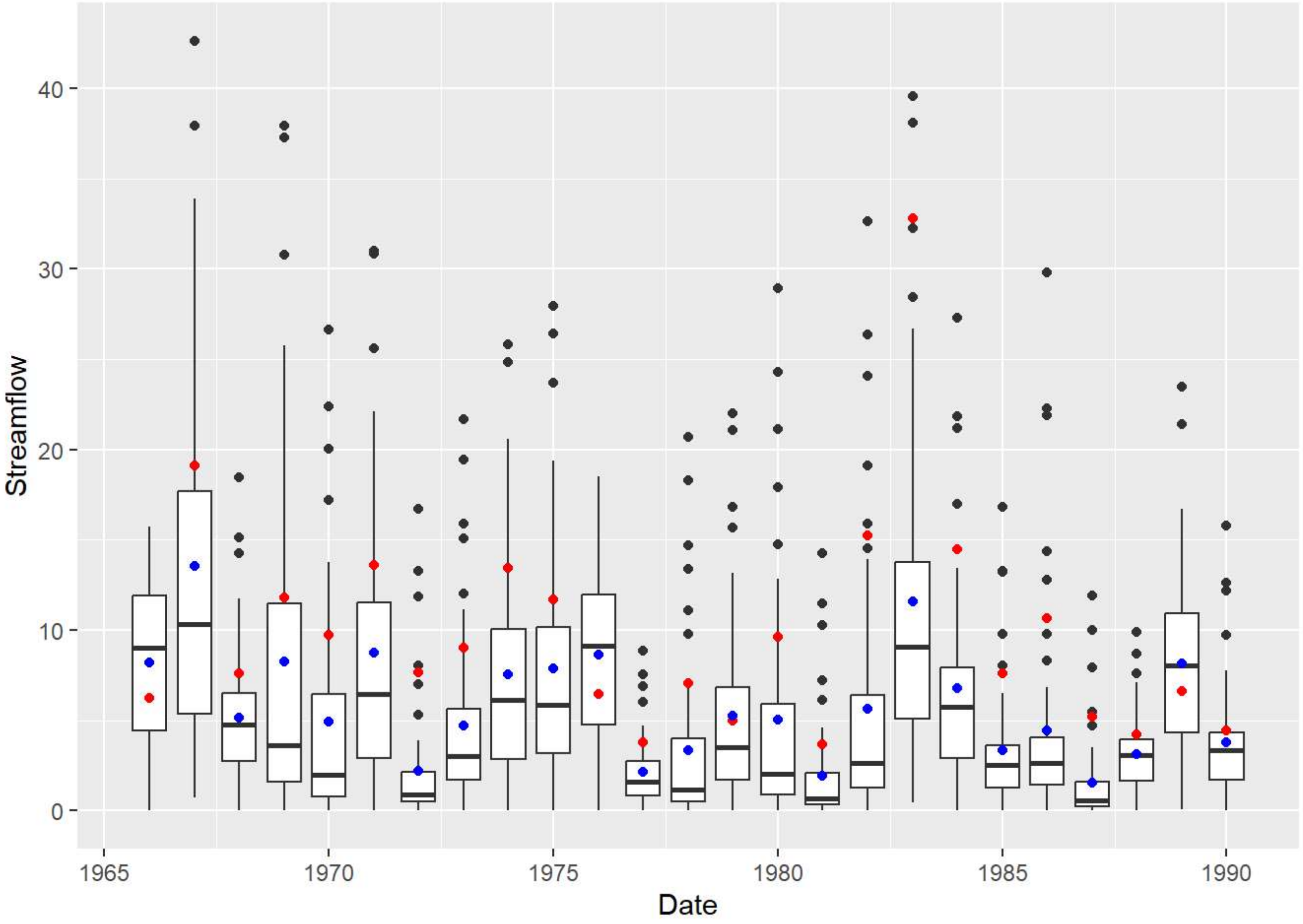
```
[1] 0.8660356
```

```
cor(mse_flow_all$str_mse_mse, mse_flow_all$obs)
```

```
[1] 0.8595476
```

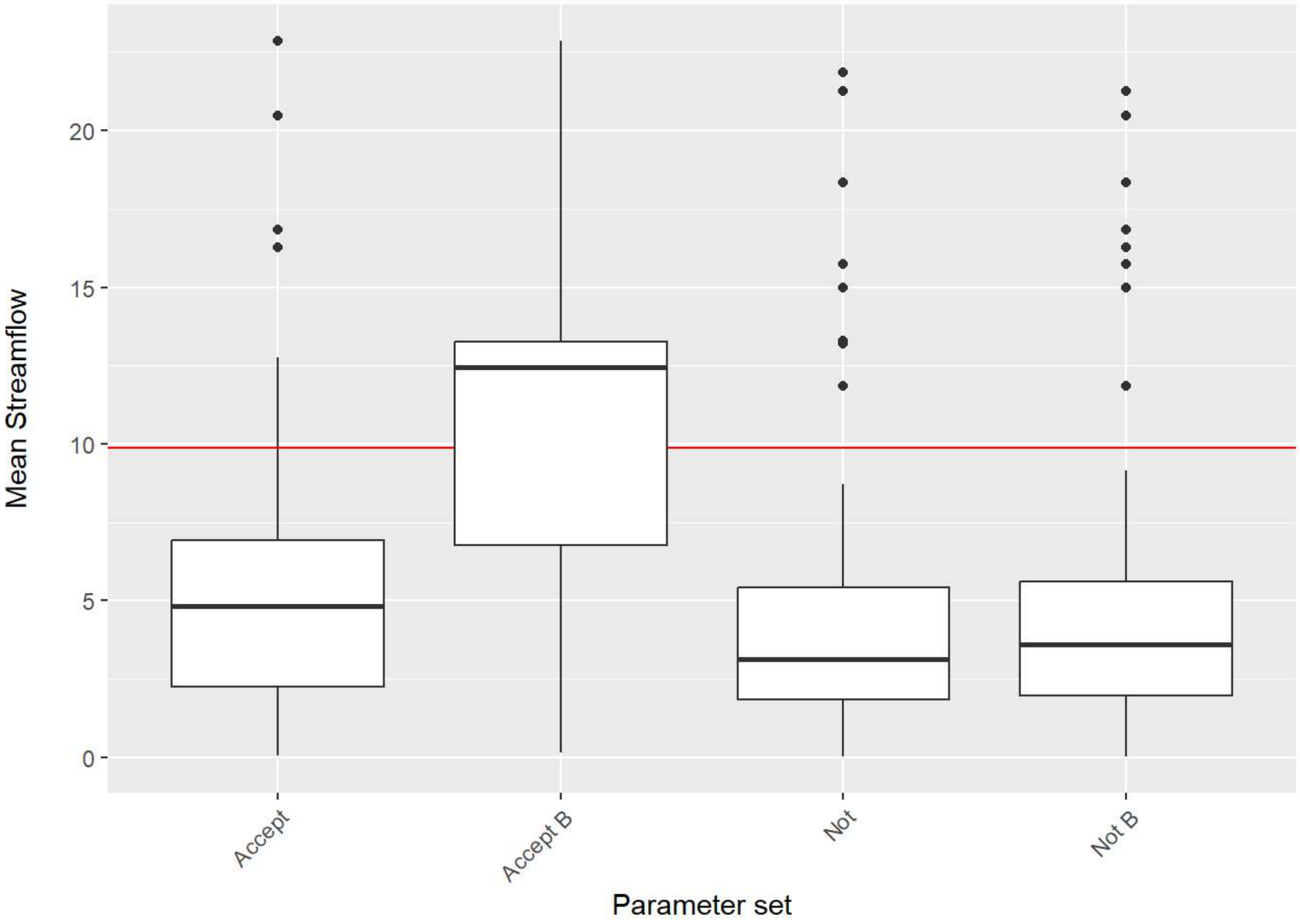
Plotting Monthly

```
p1=ggplot(subset(msagel_acc_month, month==8), aes(wy, str, group=wy)) +
  geom_boxplot(position="dodge") +
  geom_point(aes(wy, obs), col="red") +
  labs(y = "Streamflow", x = "Date")

p1 + geom_point(data=subset(mse_flow, month==8), aes(wy, str_mse), col="blue")
```

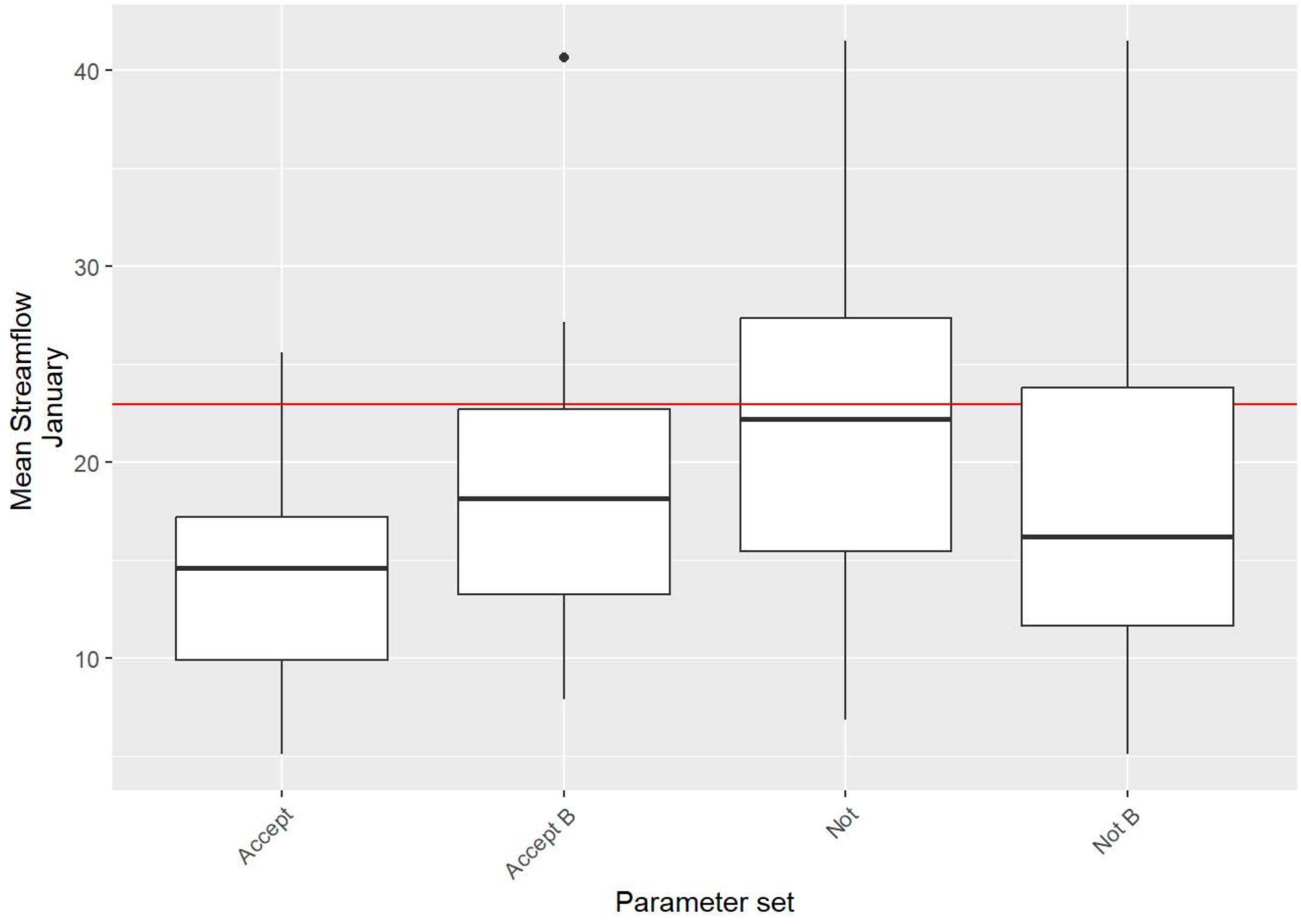


```
# plot mean August streamflow for all parameters and acceptable parameters
msagel_month_avg <- msagel_month %>%
  group_by(month, sim) %>%
  dplyr::summarize(str = mean(str), obs = mean(obs))
msagel_month_avg = msagel_month_avg %>% mutate(accept = ifelse(sim %in% results_acc$sim, "Accept"

# read in the results from last class
goodpar = c(40, 42,29, 68,9,20, 12,32, 10,77)
other_cal_parms = paste0("V",goodpar)
msagel_month_avg$othercal = ifelse(msagel_month_avg$sim %in% other_cal_parms, "Accept B", "Not B"

# compute average august flow in observed
mean_august_obs = mean(subset(msagel_month_avg, month==8)$obs)
```

```
p2 = ggplot(subset(msagel_month_avg, month==8), aes(x=accept, y=str))+geom_boxplot()+geom_hline(y
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

p2
```

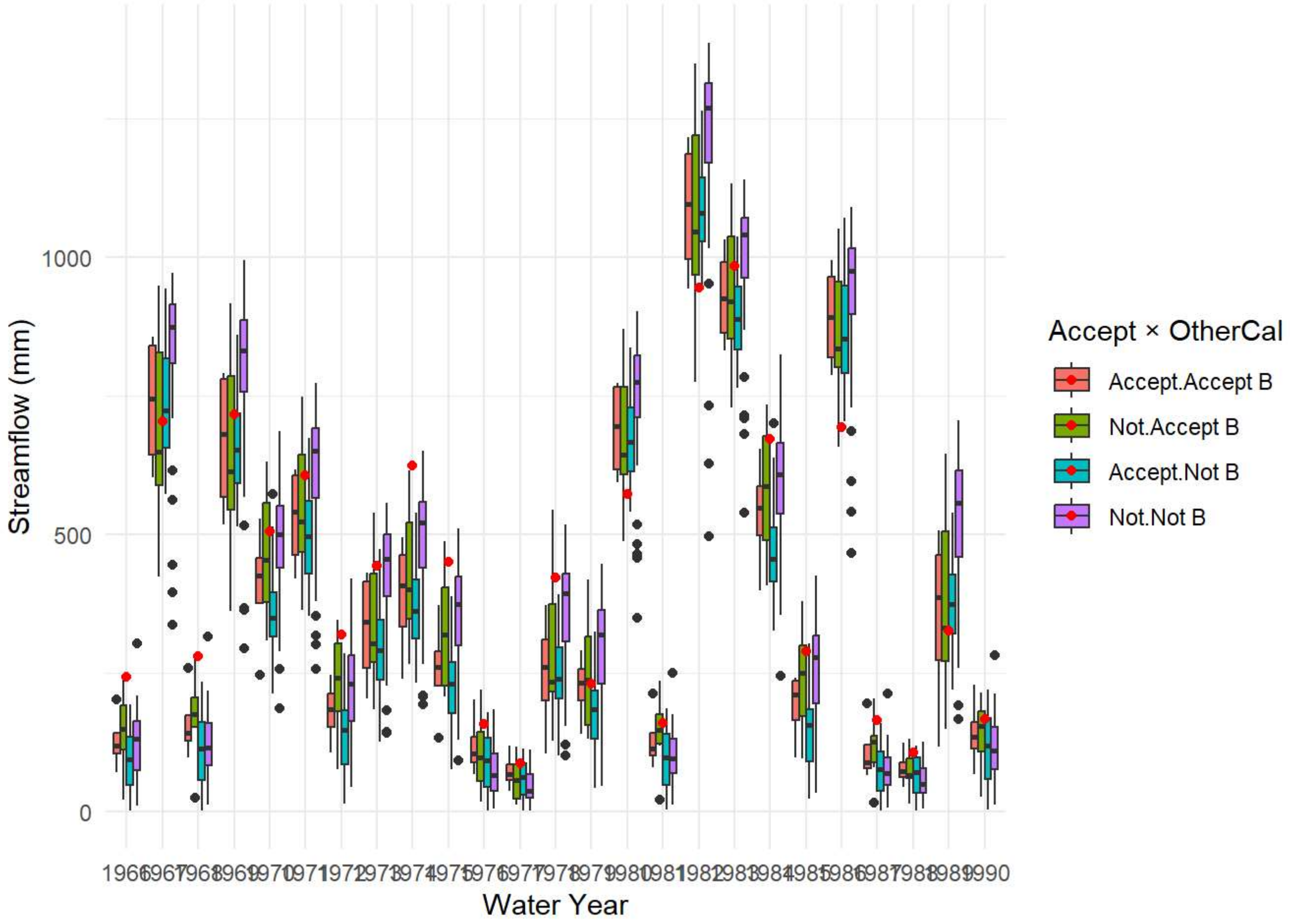


```
# compute average december flow in observed
mean_dec_obs = mean(subset(msagel_month_avg, month==1)$obs)

p3 = ggplot(subset(msagel_month_avg, month==1), aes(x=accept, y=str))+geom_boxplot()+geom_hline(y
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

p3
```



```
# calculate annual total streamflow
msagel_annual <- msagel %>%
  group_by(wy, sim) %>%
  dplyr::summarize(str = sum(str), obs = sum(obs), minstr=min(str), minobs=min(obs))

# added a column to indicate if the parameter set is acceptable
msagel_annual = msagel_annual %>% mutate(accept = ifelse(sim %in% results_acc$sim, "Accept", "Not

# also add a column for class good parameters
msagel_annual$othercal = ifelse(msagel_annual$sim %in% other_cal_parms, "Accept B", "Not B")
```

```
# plot annual total streamflow
  p4 = ggplot(msagel_annual, aes(x=factor(wy), y=minstr,
  fill=interaction(accept, othercal))) +
              geom_boxplot(position = position_dodge(width = 0.75)) + labs(x = "Water Year", y =
  theme_minimal() + geom_point(aes(x=factor(wy), y=minobs), col="red")
  p4
```


Streamflow (mm)
1000
500
0
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
Water Year
Accept × OtherCal
Accept.Accept B
Not.Accept B
Accept.Not B
Not.Not B