```
#*************************************************************************
# Date: 03/09/2022
# Comment: R code for SRD analysis with polynomial progression bandwidth
selector
# Used for: Jiawei Yang's Honor Thesis - Section 5
# Data File: table_wo_final.dta
# Data Source: Lee (2008) U.S. house congressional election record from
Democrats
#*************************************************************************

# Clear all variables and prior sessions
rm(list=ls(all=TRUE))

# Load packages
library(datums)
library(data.table)
library(foreign)
library(ggplot2)

# Load dataset/initial insepection ----
dt_lee <- data.table(read.dta('Lee2008/table_two_final.dta'))
dt_lee
  #running var: difdemshare - democrat share margin of period (t)
  #outcome var: demsharenext - democrat raw share of period (t+1)

# EDA: visualizations (univariate running, bivariate running & outcome)
----
  #adjust output resolution
tiff("test.tiff", units="in", width=5, height=5, res=500)

  #histogram [difdemshare]: distribution of the running
ggplot(data = dt_lee, aes(x=difdemshare)) +
  geom_histogram(fill = 'white', color = 'black')+
  theme_classic() +
  xlab('democrat share margin for period t') +
  ylab('frequency') +
  scale_x_continuous(limits = c(-1.01,1.02), expand = c(0, 0)) +
  scale_y_continuous(limits = c(0,1200), expand = c(0, 0)) +
  theme(text = element_text(family = "Times New Roman")) +
  geom_vline(xintercept = 0, linetype ="dashed")

  #scatter [difdemshare ~ demsharenext]: running and outcome
  #note: raw plot
ggplot(data = dt_lee,
       aes(x = difdemshare, y = demsharenext)) +
  geom_point(size = 0.01, alpha = 0.7) +
  theme_classic() +
  geom_vline(xintercept = 0, linetype = "dashed", alpha = 0.5 )+
  xlab('democrat share margin for period t') +
  ylab('democrat raw share for period t+1') +
  theme(text = element_text(family = "Times New Roman"))

  #scatter [difdemshare ~ demsharenext]: running and outcome
  #note: omitted abnormal observations
ggplot(data = dt_lee[difdemshare != -1 & difdemshare != 1 &
                       demsharenext != 0 &demsharenext != 1],
       aes(x = difdemshare, y = demsharenext)) +
  geom_point(size = 0.01, alpha = 0.7) +
```

```
  theme_classic() +
  geom_vline(xintercept = 0, linetype = "dashed", alpha = 0.5 )+
  xlab('democrat share margin for period t') +
  ylab('democrat raw share for period t+1') +
  theme(text = element_text(family = "Times New Roman"))

# Data processing: reduce to bin-wise average ----
  #remove NA-containing columns
dt_lee_narm <- dt_lee[!is.na(difdemshare) & !is.na(demsharenext)]

  #helper functions: produce bin-wise averaged dataset
subset_by_bin <- function(b_s) {
    #param - b_s: bin size to average outcome
  bin_size = b_s
  bin_seq = seq(-1, 1, by = bin_size)
  bin_mean <-
    tapply(dt_lee_narm$demsharenext,
           cut(dt_lee_narm$difdemshare, bin_seq), mean)
  dt_bin_temp = data.table()
  dt_bin_temp$avg_demsharenext = bin_mean
  #assign step tick (start)
  dt_bin_temp$difdemshare_s = head(bin_seq,-1)
  #assign step tick (end)
  dt_bin_temp$difdemshare_e = bin_seq[-1]
  #compute mid points on X axis
  dt_bin_temp$avg_difdemshare =
    (dt_bin_temp$difdemshare_s + dt_bin_temp$difdemshare_e)/2
  #convert back to data.tale type
  dt_bin_temp = data.table(dt_bin_temp)
  return(dt_bin_temp)
    #return - dt_bin_temp: cols [avg_demsharenext, difdemsahre_s,
difdemsahre_e, avg_difdemsahre]
}

  #helper function: visualize bin-wise average step plot
vis_bin_size_avg <- function(bs) {
  ggplot(data = subset_by_bin(bs), aes(x = difdemshare_s, y =
avg_demsharenext)) +
    geom_step(size = 0.5) +
    xlab('democrat share margin, period t') +
    ylab('avg. dem. raw share, period t+1') +
    theme_classic()+
    theme(text = element_text(family = "Times New Roman"))+
    geom_vline(xintercept = 0, linetype = 'dashed', alpha = 0.5)
}

  #step plot [avg_demsharenext ~ avg_difdemsahre]: averaged outcome
against running
vis_bin_size_avg(0.05) #bin size used in Lee(2008)
vis_bin_size_avg(0.025) #smaller bin sizes for comparison
vis_bin_size_avg(0.020)
vis_bin_size_avg(0.010)

# Analysis: Sharp Regression Discontinuity threshold at 0 ----
  #note: asymmetric bandwidths optimized with Polynomial Progression

  #helper function: bandwidth selection and SRD anlaysis
poly_opt <- function(dt_generic){
```

```
  dt_bin_generic = dt_generic
    #assign binary treatment: threshold at 0
  dt_bin_generic[,trtmt := (avg_difdemshare > 0)*1]

    #extract unique running var values
  x_tick_unique = unique(dt_bin_generic$avg_difdemshare)
  x_tick_pos = x_tick_unique[x_tick_unique > 0]
  x_tick_neg = x_tick_unique[x_tick_unique < 0]
    #inverse sort negative ticks for iteration
  x_tick_neg = sort(x_tick_neg, decreasing = TRUE)

  #treatment group
  fit_score_list = numeric()
  locality_score_list = numeric()
  h_pos_list = numeric()
    #loop through possible running var values
  for(h_pos in tail(x_tick_pos, length(x_tick_pos) - 4)){
    #local linear regression within a given bandwidth
    reg_sum <- summary(lm("avg_demsharenext ~ avg_difdemshare",
                          data = dt_bin_generic[trtmt == 1 &
avg_difdemshare < h_pos]))
      #calculate/append component objective scores
    fit_score = reg_sum$r.squared
    locality_score = 1- nrow(dt_bin_generic[trtmt == 1 & avg_difdemshare <
h_pos])/nrow(dt_bin_generic[trtmt == 1])
    fit_score_list = append(fit_score_list, fit_score)
    locality_score_list = append(locality_score_list, locality_score)
    h_pos_list = append(h_pos_list, h_pos)
  }
    #instantiate summary data table
  dt_pos_score = data.table(fit = fit_score_list,
                            locality = locality_score_list,
                            h_pos = h_pos_list)
    #calculate overall objective scores by product
  dt_pos_score[,O := fit*locality,]
    #locate optimal treatment bandwidth: as argmax(Objective)
  h_pos_opt = dt_pos_score[O == max(O)]$h_pos

  #control group
  fit_score_list = numeric()
  locality_score_list = numeric()
  h_neg_list = numeric()
    #loop through possible running var values
  for(h_neg in tail(x_tick_neg, length(x_tick_neg) - 4)){
    #local linear regression within a given bandwidth
    reg_sum <- summary(lm("avg_demsharenext ~ avg_difdemshare",
                          data = dt_bin_generic[trtmt == 0 &
avg_difdemshare > h_neg]))
    #calculate/append component objective scores
    fit_score = reg_sum$r.squared
    locality_score = 1- nrow(dt_bin_generic[trtmt == 0 & avg_difdemshare >
h_neg])/nrow(dt_bin_generic[trtmt == 0])
    fit_score_list = append(fit_score_list, fit_score)
    locality_score_list = append(locality_score_list, locality_score)
    h_neg_list = append(h_neg_list, h_neg )
  }

  #instantiate summary data table
```

```
  dt_neg_score = data.table(fit = fit_score_list,
                            locality = locality_score_list,
                            h_neg = h_neg_list)
  #calculate overall objective and locate optimal control bandwidth
  dt_neg_score[,O := fit*locality,]
  h_neg_opt = dt_neg_score[O == max(O)]$h_neg

    #itrct - interaction term[treatment ~ running]: allow different slopes
  dt_bin_generic[,itrct:= avg_difdemshare*trtmt]
    #SRD analysis with local linear approximation and optimzed asymmetric
bandwidths
  srd_reg_sum <-
    summary(lm("avg_demsharenext ~ avg_difdemshare + itrct + trtmt",
               data = dt_bin_generic[avg_difdemshare >= h_neg_opt &
avg_difdemshare <= h_pos_opt]))

  return(c(srd_reg_sum, h_neg_opt, h_pos_opt))
    #return values:
      #srd_reg_sum: summary of SRD analysis
      #h_neg_opt: control group bandwidth optimization summary
      #h_pos_opt: treatment group bandwithd optimzation summary
}

  #SRD analysis output with optimized asymmetric bandwidth
poly_opt(subset_by_bin(0.020)) #bin size used in Lee(2008)
poly_opt(subset_by_bin(0.020)) #smaller bin sizes for comparison
poly_opt(subset_by_bin(0.020))
poly_opt(subset_by_bin(0.020))
```