Supplement for: bmjopen-2020-048211.R1

Development of a healthcare system COVID hotspotting score in California: an observational study with prospective validation

```
################################
######## CPAFuntions.R ########
################################
# https://sites.google.com/site/changepointanalysis/library-1

#' Title Change Point Analysis -  by Taha Kass-Hout, MD, MS and Zhiheng Xu, PhD
#'
#' @param DTA: dataset must have column 1 as date and column 2 as value
#'
#' @return:    a data frame with break point information
#' @export
#'
#' @examples: getCPA(df)
getCPA <- function(DTA){
  ds <- DTA
  x <- ds[,2]
  level<-1
  a<-find_break(x)
  a[3]<-ifelse(a[3]<1,1,floor(a[3]))
  a[4]<-ifelse(a[4]<1,1,floor(a[4]))
  breakpoint <- data.frame(level=level,
                           bp=ds[a[1],1],
                           bpindex=a[1],
                           conflevel=a[2],
                           value=x[a[1]],
                           begin=1,
                           end=length(x),
                           lower=ds[a[3],1],
                           upper=ds[a[4],1])

  for (level in 2:20) {
```

```
# print(sprintf('level: %s',level))
breakpoint<-breakpoint[order(breakpoint$bpindex),]
order<-c(1,breakpoint$bpindex,length(x)+1)
begin<-order[1]
jlevel<-min(2^(level-1),length(order)-1)
for (j in 1: jlevel){
  end<-order[j+1]-1
  # print(c(begin,end))
  if (end > (begin+2)){
    subx<-x[begin:end]
    if (!is.na(breakpoint$conflevel[j])) {
      a1<-find_break(subx)
      a1[3]<-ifelse(a1[3]<1,1,floor(a1[3]))
      a1[4]<-ifelse(a1[4]<1,1,floor(a1[4]))
      if (a1[1] < length(subx) & a1[1]>1) {
        a1[1]<-a1[1]+begin-1
        breakpoint1<-data.frame(level=level,
                                bp=ds[a1[1],1],
                                bpindex=a1[1],
                                conflevel=a1[2],
                                value=x[a1[1]],
                                begin=begin,
                                end=end,
                                lower=ds[a1[3],1],
                                upper=ds[a1[4],1])
        breakpoint<-rbind(breakpoint,breakpoint1)
      }
      begin<-end+2
    }
    else if (!is.na(breakpoint$conflevel[j])){
      j<-j+1
      begin<-order[j+1]+1
    }
  }
```

```
      else {
        j<-j+1
        begin<-end+2
      }
    }
  }
  breakpoint<-breakpoint[which(breakpoint$conflevel >= 95.0),]
  breakpoint<-na.omit(breakpoint[order(breakpoint$bpindex),])
  return(breakpoint)
}

find_break<-function(x) {
  ratio_jk_cp<-mat.or.vec(length(x),1)
  x2<-cumsum(x-mean(x))
  bp<-which.max(abs(x2))
  if (bp <length(x)) bp=bp+1
  diff<-max(x2)-min(x2)
  xlength<-length(x)
  n <-ifelse(xlength > 7, 1000, factorial(xlength))
  bpwo<-mat.or.vec(n,1)
  bpdiff<-mat.or.vec(n,1)
  bpjk<-jackknife_z(x,bp)
  ratio_jk_zscore<-mat.or.vec(n,1)
  set.seed(234)
  for (n1 in 1: n){
    ratio<-sample(x,replace=T)
    ratio2<-cumsum(ratio-mean(ratio))
    bpwo[n1]<-which.max(abs(ratio2))+1
    bpdiff[n1]<-max(ratio2)-min(ratio2)
    bpjk2<-jackknife_z(ratio,bpwo[n1])
    ratio_jk_zscore[n1]<-(bpjk2[1]-bpwo[n1])/bpjk2[2]
  }
  ratio_jk_zscore<-sort(ratio_jk_zscore)
  bplower<-bp+ratio_jk_zscore[25]*bpjk[2]
```

```
  bpupper<-bp+ratio_jk_zscore[975]*bpjk[2]
  conflevel<-sum(bpdiff <= diff)/n *100
  return(c(bp,conflevel,bplower,bpupper))
}

jackknife_z<-function(ratio,bp){
  ratio_jk_cp<-mat.or.vec(length(ratio),1)
  #Bootstrapping Jacknife T-interval
  for (n2 in 1: length(ratio)){
    ratio_jk<-ratio[-n2]
    ratio_jk2<-cumsum(ratio_jk-mean(ratio_jk))
    ratio_jk_cp[n2]<-which.max(abs(ratio_jk2))+1
  }
  ratio_b<-mean(ratio_jk_cp)
  ratio_sd<-sd(ratio_jk_cp)
  return(c(ratio_b, ratio_sd))
}
```

```
################################
######### functions.R #########
################################

#' Title: extract pvalue from Poisson test in CPTsum
#'
#' @param ptext
#'
#' @return
#' @export
#'
#' @examples
getPoissonP <- function(ptext){
  PTEXT <- ptext
  p <- NULL
  for (i in PTEXT){
    if (is.na(i)){
      pi <- NA
    } else{
      pi <- eval(parse(text=i))$p.value
    }
    p <- c(p,pi)
  }
  return(p)
}



#' Title
#'
#' @param m1 the sample means
#' @param m2
#' @param s1 the sample standard deviations
#' @param s2
#' @param n1 the same sizes
```

```
#' @param n2
#' @param m0 The null value for the difference in means to be tested for. Default is 0
#' @param equal.variance whether or not to assume equal variance. Default is FALSE.
#'
#' @return
#' @export
#'
#' @examples
getTTestP <- function(m1,m2,s1,s2,n1,n2,m0=0,equal.variance=FALSE)
{
  if( equal.variance==FALSE )
  {
    se <- sqrt( (s1^2/n1) + (s2^2/n2) )
    df <- ( (s1^2/n1 + s2^2/n2)^2 )/( (s1^2/n1)^2/(n1-1) + (s2^2/n2)^2/(n2-1) )
  } else
  {
    se <- sqrt( (1/n1 + 1/n2) * ((n1-1)*s1^2 + (n2-1)*s2^2)/(n1+n2-2) )
    df <- n1+n2-2
  }
  t <- (m1-m2-m0)/se
  p <-2*pt(-abs(t),df)
  return(p)
}
```

```
#######################################
######### getHotspotReport.R #########
#######################################

### Load some libraries.
suppressMessages(library(tidyquant))
suppressMessages(library(zoo))
suppressMessages(library(tidyverse))
suppressMessages(library(lubridate))
suppressMessages(library(hrbrthemes))
suppressMessages(library(fastDummies))
suppressMessages(library(forecast))
suppressMessages(library(fpp2))
suppressMessages(library(rateratio.test))

source("./CPAFunctions.R")
source("./functions.R")

### Read in the dataset
dta <- read_csv("./example_data.csv")

### The dataframe should have 2 columns (at least) CalendarDate and value
head(dta)
# A tibble: 6 x 2
# CalendarDate value
# <date>      <dbl>
# 1 2015-01-01      5
# 2 2015-01-02      8
# 3 2015-01-03      2
# 4 2015-01-04      5
# 5 2015-01-05     10
# 6 2015-01-06     12

### Generate some time serie features
```

```
sub <- dta %>%
      mutate(D=as_date(as.Date(CalendarDate,format='%Y-%m-%d')),
          M=as.character(month(D,label = TRUE, abbr = TRUE)),
          DOW=as.character(wday(D,label = TRUE, abbr = TRUE))) %>%
      mutate(M=factor(M,levels=c('Aug','Sep','Oct','Nov','Dec','Jan','Feb','Mar','Apr','May','Jun','Jul')),
          DOW=factor(DOW,levels=c('Mon','Tue','Wed','Thu','Fri','Sun','Sat'))) %>%
      arrange(CalendarDate)
sub <- fastDummies::dummy_cols(sub, select_columns = c('DOW','M'),remove_first_dummy = TRUE)

### (1) create train/validation sets
full <- sub %>%
  arrange(D)
train <- sub %>%
  filter(D <= as_date('2019-12-31','%Y-%m-%d')) %>%
  arrange(D)
test <- sub %>%
  filter(D > as_date('2019-12-31','%Y-%m-%d')) %>%
  arrange(D)

covars <- names(select(sub,DOW,starts_with('M_')))
covars_string <- paste0(covars,collapse = '+')

### (2) fit OLS model to adjust for seasonality.
mytext <- paste0('lm(value ~ ',covars_string,',data=train)')
fit <- eval(parse(text=mytext))
fit_sum <- summary(fit)$coef

train$pred <- predict(fit,newdata = train)
train[,'pred'][train[,'pred'] < 0] <- 0
train$residual <- train$value - train$pred

test$pred <- predict(fit,newdata = test)
test[,'pred'][test[,'pred'] < 0] <- 0
test$residual <- test$value - test$pred
```

```
test <- as.data.frame(test)

### (4) apply getCPA() on the residual
dsCPA <- test[,c('D','residual')]
breakpoint <- getCPA(dsCPA)

breakpoint <- distinct(breakpoint,level,bp,bpindex,conflevel,value,begin,end, .keep_all= TRUE)
CPT <- c(breakpoint$bpindex,length(dsCPA$residual))
Date <- c(breakpoint$bp,dsCPA[length(dsCPA$residual),1])

CPTsum <-   bind_cols(D=Date,BPIndex=CPT) %>%
  arrange(D) %>%
  rownames_to_column("rn") %>%
  mutate(ChangeDetected = as.numeric(rn)-1) %>%
  select(-rn)

CPTsum <- left_join(test %>% select(D,residual),
                    CPTsum,
                    by='D') %>%
  fill(ChangeDetected,.direction='up') %>%
  group_by(ChangeDetected) %>%
  summarise(CPTMean = mean(residual),
            CPTVar  = var(residual),
            N=n()) %>%
  left_join(CPTsum,by='ChangeDetected') %>%
  select(D,CPTMean,CPTVar,ChangeDetected,N) %>%
  mutate(
    RelativeChangeLast = ifelse(ChangeDetected!=0,
                                ifelse(lag(CPTMean)!=0,
                                       (CPTMean-lag(CPTMean))/abs(lag(CPTMean)),
                                       (CPTMean-lag(CPTMean))/0.5),
                                0),
    PValLastPoisson = 1,
```

```
    PValLastTtest = ifelse(ChangeDetected!=0,
                    getTTestP(CPTMean,lag(CPTMean),sqrt(CPTVar),sqrt(lag(CPTVar)),N,lag(N)),
                    1),
    RelativeChangeFirst = ifelse(ChangeDetected!=0,
                        (CPTMean-CPTMean[which(CPTMean != 0)[1]])/abs(CPTMean[which(CPTMean != 0)[1]]),
                        0),
    PValFirstPoisson = 1,
    PValFirstTtest = ifelse(ChangeDetected!=0,
                     getTTestP(CPTMean,CPTMean[which(CPTMean != 0)[1]],sqrt(CPTVar),sqrt(CPTVar[which(CPTMean !=
0)[1]]),N,N[which(CPTMean != 0)[1]]),
                     1),
    CPTMeanMin = min(CPTMean),
    RelativeChangeMin =  (CPTMean-CPTMeanMin)/abs(CPTMeanMin),
    PValMinPoisson = 1,
    PValMinTtest =
getTTestP(CPTMean,CPTMeanMin,sqrt(CPTVar),sqrt(CPTVar[which.min(CPTMean[which(CPTMean!=0)])]),N,N[which.min(CPTM
ean[which(CPTMean!=0)])])
  ) %>%
  select(-CPTMeanMin,-ends_with('Text'))

CPTsumAll <- left_join(test %>% select(-M,-starts_with('DOW'),-starts_with('M')),
                   CPTsum,
                   by='D') %>%
            fill(CPTMean:PValMinTtest,.direction='up') %>%
            rename(ObservedValue = value,
                 ExpectedValue = pred)

ValueMean <-  CPTsumAll %>%
            group_by(ChangeDetected) %>%
            summarise(ObservedMean = mean(ObservedValue),
                   ExpectedMean = mean(ExpectedValue))

CPTsumAll <-   left_join(CPTsumAll,
                   ValueMean,
                   by='ChangeDetected')
```

```
### (5) generate slopes
### get slopes
CPTsumAll <- CPTsumAll %>%
  rownames_to_column(var="day_num") %>%
  mutate(day_num = as.numeric(day_num),
        value_ma03 = rollmean(residual, k = 3, fill = NA))

L <- dim(CPTsumAll)[1]
SlopeWeekTrend <- rep(NA,L)
SlopePValue <- rep(NA,L)
for(i in L:7){
  t <- CPTsumAll[(i-6):i,]
  fit <- lm(value_ma03 ~ day_num,t)
  SlopeWeekTrend[i] <- summary(fit)$coefficients[2,1]
  SlopePValue[i] <- summary(fit)$coefficients[2,4]
}

CPTsumAll$SlopeWeekTrend <- SlopeWeekTrend
CPTsumAll$SlopePValue <- SlopePValue

att1 <- CPTsumAll %>%
  select(CalendarDate,SlopeWeekTrend,SlopePValue) %>%
  filter(SlopeWeekTrend>0 & SlopePValue<=0.05)

att2 <- CPTsumAll %>%
  select(CalendarDate,SlopeWeekTrend,SlopePValue) %>%
  filter(SlopeWeekTrend>0 & SlopePValue<=0.01)

### View the report
View(CPTsumAll)

### (5) Visualize the result
CPTsumAll %>%
```

```
ggplot(aes(D,ObservedValue)) +
geom_line(color='dodgerblue2',size=1)+
geom_line(aes(D,ExpectedValue),color='forestgreen',size=1)+
geom_line(aes(D,ObservedMean),color='firebrick',size=1,linetype='dashed')+
geom_line(aes(CalendarDate,SlopeWeekTrend),linetype='solid',color='firebrick') +
geom_vline(xintercept = att1$CalendarDate,alpha=1,linetype='dashed',color='firebrick')+
geom_vline(xintercept = att2$CalendarDate,alpha=1,linetype='dashed',color='gold')+
labs(x='', y='Count',
    title='CPA and Slope week trend on an example data',
    subtitle=sprintf('%s - Slope off the residual',CPTsumAll$Variable[1]),
    caption='CPA by Taha Kass-Hout')+
scale_x_date(date_breaks = '1 month',date_labels = "%m-%Y")+
theme_ipsum(grid=FALSE)+
theme(axis.text.x=element_text(size = 9,angle=60, hjust=1),
     plot.title = element_text(size = 11))
```