Supplementary material 2: Sensitivity of results to inclusion of additional auxiliaries

Rob Boyd

13 March 2023

## Purpose of this document

The purpose of this document is to demonstrate that including additional auxiliary variables does not improve estimates of mean occupancy or the time trend between them. In fact, it barely changes the results at all. The three additional auxiliaries are the first and second principal components of climate space in Britain and soil pH.

I do not provide any text beyond this introductory spiel in this document. The reason is that the code is explained in plain English in supplementary material one. The code here is roughly identical; the only major difference is that I do not drop the three additional auxiliary variables. Looking at the figures at the bottom of the document, it is clear that the results are very similar to those obtained when the three auxiliaries are omitted.

library(raster)

## Warning: package 'raster' was built under R version 4.1.2

## Loading required package: sp

## Warning: package 'sp' was built under R version 4.1.2

library(ggplot2)

## Warning: package 'ggplot2' was built under R version 4.1.2

library(survey)

## Warning: package 'survey' was built under R version 4.1.3

## Loading required package: grid

## Loading required package: Matrix

## Loading required package: survival

##   
## Attaching package: 'survey'

## The following object is masked from 'package:raster':  
##   
## cv

## The following object is masked from 'package:graphics':  
##   
## dotchart

library(rstanarm)

## Warning: package 'rstanarm' was built under R version 4.1.3

## Loading required package: Rcpp

## Warning: package 'Rcpp' was built under R version 4.1.2

## This is rstanarm version 2.21.3

## - See https://mc-stan.org/rstanarm/articles/priors for changes to default priors!

## - Default priors may change, so it's safest to specify priors, even if equivalent to the defaults.

## - For execution on a local, multicore CPU with excess RAM we recommend calling

## options(mc.cores = parallel::detectCores())

library(PracTools)

## Warning: package 'PracTools' was built under R version 4.1.3

##   
## Attaching package: 'PracTools'

## The following object is masked from 'package:survey':  
##   
## deff

library(reshape2)

## Warning: package 'reshape2' was built under R version 4.1.3

library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:raster':  
##   
## intersect, select, union

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

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

library(caret)

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following objects are masked from 'package:rstanarm':  
##   
## compare\_models, R2

## The following object is masked from 'package:survival':  
##   
## cluster

library(boot)

##   
## Attaching package: 'boot'

## The following object is masked from 'package:lattice':  
##   
## melanoma

## The following object is masked from 'package:rstanarm':  
##   
## logit

## The following object is masked from 'package:survival':  
##   
## aml

## load population data  
pop <- read.csv("W:/PYWELL\_SHARED/Pywell Projects/BRC/Rob Boyd/NERC\_exploring\_frontiers/Data/all\_data.csv")  
  
pop <- pop[complete.cases(pop), ]  
  
pop\_disc <- pop  
  
names(pop\_disc)

## [1] "heather\_true\_dist\_1987.1999" "heather\_true\_dist\_2010.2019"   
## [3] "sampled\_units\_1987.1999" "sampled\_units\_2010.2019"   
## [5] "postcode\_density\_299\_neighbours" "openAccessGB"   
## [7] "allPACoverage" "road\_length\_299\_neighbours"   
## [9] "UKelv" "X\_1"   
## [11] "X\_3" "inclusionProbs\_1987.1999"   
## [13] "inclusionProbs\_2010.2019" "layer"

## pull out the auxiliary data for the whole population  
pop\_aux\_cont <- pop\_disc[,c(5:11,14)]  
  
## for reasons that will become clear later, we need to discretize the auxiliary data in pop  
for (i in c(5,8,9,10,11,14)) {  
  
 q <- as.numeric(quantile(pop\_disc[,i], probs = c(0,0.33,0.66,1)))  
  
 print(q)  
   
 pop\_disc[,i] <- cut(pop\_disc[,i],   
 breaks = q,  
 labels = FALSE,  
 include.lowest = TRUE,  
 right = TRUE)  
   
 pop\_disc[,i] <- as.numeric(pop\_disc[,i])  
   
}

## [1] 0 506 2220 154697  
## [1] 0.0 195369.5 339891.7 649403.7  
## [1] -5 71 173 1199  
## [1] -17.384830 -1.294301 2.252261 5.551067  
## [1] -8.4327745 -0.2794245 0.6739962 5.7398550  
## [1] 3.809253 5.583358 6.713431 9.172379

## we'll make PA and open access land coverage binary   
pop\_disc$openAccessGB <- ifelse(pop$openAccessGB > 0, 1, 0)  
  
pop\_disc$allPACoverage <- ifelse(pop$allPACoverage > 0, 1, 0)  
  
## pull out columns relevant to period 1  
names(pop)

## [1] "heather\_true\_dist\_1987.1999" "heather\_true\_dist\_2010.2019"   
## [3] "sampled\_units\_1987.1999" "sampled\_units\_2010.2019"   
## [5] "postcode\_density\_299\_neighbours" "openAccessGB"   
## [7] "allPACoverage" "road\_length\_299\_neighbours"   
## [9] "UKelv" "X\_1"   
## [11] "X\_3" "inclusionProbs\_1987.1999"   
## [13] "inclusionProbs\_2010.2019" "layer"

pop\_p1 <- pop[,c(1,3,5,6,7,8,9,10,11,12,14)]  
  
pop\_disc\_p1 <- pop\_disc[,c(1,3,5,6,7,8,9,10,11,12,14)]  
  
## and period 2  
pop\_p2 <- pop[,c(2,4,5,6,7,8,9,10,11,13,14)]  
  
pop\_disc\_p2 <- pop\_disc[,c(2,4,5,6,7,8,9,10,11,13,14)]  
  
## pull out sampled rows for periods 1 and 2  
samp\_disc\_p1 <- pop\_disc\_p1[pop\_disc\_p1$sampled\_units\_1987.1999 == 1, ]  
  
samp\_disc\_p2 <- pop\_disc\_p2[pop\_disc\_p2$sampled\_units\_2010.2019 == 1, ]  
  
## pull out the auxiliary data for the whole population  
pop\_aux <- pop\_disc[,c(5:11,14)]  
  
## calculate the population means  
pop\_mean\_p1 <- mean(pop\_p1$heather\_true\_dist\_1987.1999);pop\_mean\_p1

## [1] 0.3173697

pop\_mean\_p2 <- mean(pop\_p2$heather\_true\_dist\_2010.2019);pop\_mean\_p2

## [1] 0.2695963

## now set up survey designs using the survey package to construct estimators  
design\_p1 <- svydesign(ids=~0,  
 data = samp\_disc\_p1)

## Warning in svydesign.default(ids = ~0, data = samp\_disc\_p1): No weights or  
## probabilities supplied, assuming equal probability

design\_p2 <- svydesign(ids=~0,  
 data = samp\_disc\_p2)

## Warning in svydesign.default(ids = ~0, data = samp\_disc\_p2): No weights or  
## probabilities supplied, assuming equal probability

## calculate sample means  
samp\_mean\_p1 <- svymean(design = design\_p1,  
 x=~heather\_true\_dist\_1987.1999)  
  
samp\_mean\_p2 <- svymean(design = design\_p2,  
 x=~heather\_true\_dist\_2010.2019)  
  
## and their confidence intervals  
samp\_mean\_p1\_conf <- confint(object = samp\_mean\_p1,  
 level = 0.95)  
  
samp\_mean\_p2\_conf <- confint(object = samp\_mean\_p2,  
 level = 0.95)  
  
## now calculate a weighted mean  
## first create a new survey design with the estimated inclusion probabilities  
weighted\_design\_p1 <- svydesign(ids=~0,  
 data = samp\_disc\_p1,  
 probs=~inclusionProbs\_1987.1999)  
  
weighted\_design\_p2 <- svydesign(ids=~0,  
 data = samp\_disc\_p2,  
 probs=~inclusionProbs\_2010.2019)  
  
## then get the weighted sample means  
weighted\_samp\_mean\_p1 <- svymean(design = weighted\_design\_p1,  
 x=~heather\_true\_dist\_1987.1999);weighted\_samp\_mean\_p1

## mean SE  
## heather\_true\_dist\_1987.1999 0.28211 0.0016

weighted\_samp\_mean\_p2 <- svymean(design = weighted\_design\_p2,  
 x=~heather\_true\_dist\_2010.2019);weighted\_samp\_mean\_p2

## mean SE  
## heather\_true\_dist\_2010.2019 0.30369 0.0014

## and their confidence intervals   
weighted\_samp\_mean\_p1\_conf <- confint(object = weighted\_samp\_mean\_p1,  
 level = 0.95)  
  
weighted\_samp\_mean\_p2\_conf <- confint(object = weighted\_samp\_mean\_p2,  
 level = 0.95)  
  
## next we want to postratify. We use the auxiliary data from earlier  
## first, cross the covariates to get the poststrata  
cells <- data.frame(table(pop\_aux))  
  
cells$Freq[cells$Freq==0] <- 1  
  
cellsSamp <- data.frame(table(samp\_disc\_p1[,c(3:7,9)]))  
  
  
## now poststratify   
ps\_design\_p1 <- survey::postStratify(design = design\_p1,  
 strata = samp\_disc\_p1[,c(3:9,11)],  
 population = cells,  
 partial = T)

## Warning in postStratify.survey.design(design = design\_p1, strata =  
## samp\_disc\_p1[, : Some strata absent from sample: ignored

ps\_design\_p2 <- survey::postStratify(design = design\_p2,  
 strata = samp\_disc\_p2[,c(3:9,11)],  
 population = cells,  
 partial = T)

## Warning in postStratify.survey.design(design = design\_p2, strata =  
## samp\_disc\_p2[, : Some strata absent from sample: ignored

## and get the weighted mean across poststrata  
ps\_samp\_mean\_p1 <- svymean(design = ps\_design\_p1,  
 x=~heather\_true\_dist\_1987.1999,  
 na.rm = T);ps\_samp\_mean\_p1

## mean SE  
## heather\_true\_dist\_1987.1999 0.34021 0.0013

ps\_samp\_mean\_p2 <- svymean(design = ps\_design\_p2,  
 x=~heather\_true\_dist\_2010.2019);ps\_samp\_mean\_p2

## mean SE  
## heather\_true\_dist\_2010.2019 0.3174 0.001

## and their confidence intervals  
ps\_samp\_mean\_p1\_conf <- confint(object = ps\_samp\_mean\_p1,  
 level = 0.95)  
  
ps\_samp\_mean\_p2\_conf <- confint(object = ps\_samp\_mean\_p2,  
 level = 0.95)  
  
## another estimator is regression-based  
## first, calculate sums of the auxiliary variables  
aux\_tots <- c(nrow(pop\_aux\_cont), colSums(pop\_aux\_cont))  
  
names(aux\_tots)[1] <- "(Intercept)"  
  
## create new designs with the continuous rather than discretized auxiliary variables  
samp\_p1 <- pop\_p1[pop\_p1$sampled\_units\_1987.1999 == 1, ]  
  
samp\_p2 <- pop\_p2[pop\_p2$sampled\_units\_2010.2019 == 1, ]  
  
pre\_calib\_design\_p1 <- svydesign(ids=~0,  
 data = samp\_p1)

## Warning in svydesign.default(ids = ~0, data = samp\_p1): No weights or  
## probabilities supplied, assuming equal probability

pre\_calib\_design\_p2 <- svydesign(ids=~0,  
 data = samp\_p2)

## Warning in svydesign.default(ids = ~0, data = samp\_p2): No weights or  
## probabilities supplied, assuming equal probability

## now calibrate   
calib\_design\_p1 <- calibrate(design = pre\_calib\_design\_p1,  
 formula = ~   
 road\_length\_299\_neighbours + postcode\_density\_299\_neighbours + openAccessGB + allPACoverage +   
 UKelv + X\_3 + X\_1 + layer,  
 population = aux\_tots,  
 calfun="linear")

## Warning in regcalibrate.survey.design2(design, formula, population,  
## aggregate.stage = aggregate.stage, : Sample and population totals reordered to  
## make names agree: check results.

sum(weights(calib\_design\_p1))

## [1] 229584

summary(weights(calib\_design\_p1))

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -0.3914 1.7025 2.0034 2.3470 2.8528 7.4903

sp\_samp\_mean\_p1 <- svymean(~heather\_true\_dist\_1987.1999, design=calib\_design\_p1)  
  
calib\_design\_p2 <- calibrate(design = pre\_calib\_design\_p2,  
 formula = ~   
 road\_length\_299\_neighbours + postcode\_density\_299\_neighbours + openAccessGB + allPACoverage +   
 UKelv + X\_3 + X\_1 + layer,  
 population = aux\_tots,  
 calfun="linear")

## Warning in regcalibrate.survey.design2(design, formula, population,  
## aggregate.stage = aggregate.stage, : Sample and population totals reordered to  
## make names agree: check results.

sum(weights(calib\_design\_p2))

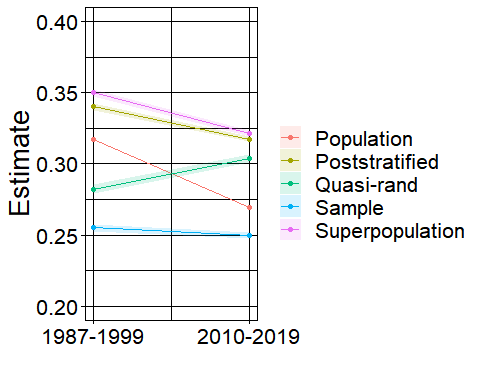
## [1] 229584

summary(weights(calib\_design\_p2))

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.4966 1.2862 1.4491 1.6235 1.9643 3.7275

sp\_samp\_mean\_p2 <- svymean(~heather\_true\_dist\_2010.2019, design=calib\_design\_p2)  
  
## and the confidence intervals   
sp\_samp\_mean\_p1\_conf <- confint(object = sp\_samp\_mean\_p1,  
 level = 0.95)  
  
sp\_samp\_mean\_p2\_conf <- confint(object = sp\_samp\_mean\_p2,  
 level = 0.95)  
  
  
## let's plot some of the estimates so far  
plotDat <- data.frame(p = c(1,2,1,2,1,2,1,2,1,2),  
 est = c(pop\_mean\_p1[1], pop\_mean\_p2[1], samp\_mean\_p1[1], samp\_mean\_p2[1], ps\_samp\_mean\_p1[1], ps\_samp\_mean\_p2[1], weighted\_samp\_mean\_p1[1], weighted\_samp\_mean\_p2[1], sp\_samp\_mean\_p1[1], sp\_samp\_mean\_p2[1]),  
 type = c("Population", "Population", "Sample", "Sample", "Poststratified", "Poststratified", "Quasi-rand", "Quasi-rand", "Superpopulation", "Superpopulation"),  
 lower = c(pop\_mean\_p1[1], pop\_mean\_p2[1], samp\_mean\_p1\_conf[1], samp\_mean\_p2\_conf[1], ps\_samp\_mean\_p1\_conf[1], ps\_samp\_mean\_p2\_conf[1], weighted\_samp\_mean\_p1\_conf[1], weighted\_samp\_mean\_p2\_conf[1], sp\_samp\_mean\_p1\_conf[1], sp\_samp\_mean\_p2\_conf[1]),  
 upper = c(pop\_mean\_p1[2], pop\_mean\_p2[2], samp\_mean\_p1\_conf[2], samp\_mean\_p2\_conf[2], ps\_samp\_mean\_p1\_conf[2], ps\_samp\_mean\_p2\_conf[2], weighted\_samp\_mean\_p1\_conf[2], weighted\_samp\_mean\_p2\_conf[2], sp\_samp\_mean\_p1\_conf[2], sp\_samp\_mean\_p2\_conf[2]))  
  
  
ggplot(data = plotDat, aes(x = p, y = est, colour = type, fill = type)) +  
 geom\_point() +  
 geom\_line() +  
 theme\_linedraw() +  
 geom\_ribbon(aes(ymin = lower, ymax = upper), alpha = 0.15, colour = NA) +  
 labs(x = "",  
 y = "Estimate",  
 fill = "",  
 colour = "") +  
 scale\_x\_continuous(breaks = c(1,2), labels = c("1987-1999", "2010-2019")) +  
 theme(text=element\_text(size = 20)) +  
 ylim(c(0.2,0.4))

## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning  
## -Inf



## now let's look at the trends   
trends <- lapply(unique(plotDat$type),  
 function(x) {  
 data.frame(difference = plotDat$est[plotDat$p == 2 & plotDat$type == x] - plotDat$est[plotDat$p == 1 & plotDat$type == x],  
 estimator = x)  
 })  
  
trends <- do.call("rbind", trends)  
  
## and the confidence intervals for the trends   
  
## sample mean   
sample\_mean\_se <- sqrt(SE(ps\_samp\_mean\_p1)^2 + SE(ps\_samp\_mean\_p2)^2)  
  
sample\_mean\_upper <- (samp\_mean\_p2 - samp\_mean\_p1) + 1.96 \* sample\_mean\_se  
  
sample\_mean\_lower <- (samp\_mean\_p2 - samp\_mean\_p1) - 1.96 \* sample\_mean\_se  
  
## quasi-randomisation   
weighted\_mean\_se <- sqrt(SE(weighted\_samp\_mean\_p1)^2 + SE(weighted\_samp\_mean\_p2)^2)  
  
weighted\_mean\_upper <- (weighted\_samp\_mean\_p2 - weighted\_samp\_mean\_p1) + 1.96 \* weighted\_mean\_se  
  
weighted\_mean\_lower <- (weighted\_samp\_mean\_p2 - weighted\_samp\_mean\_p1) - 1.96 \* weighted\_mean\_se  
  
## poststratification  
ps\_mean\_se <- sqrt(SE(ps\_samp\_mean\_p1)^2 + SE(ps\_samp\_mean\_p2)^2)  
  
ps\_mean\_upper <- (ps\_samp\_mean\_p2 - ps\_samp\_mean\_p1) + 1.96 \* ps\_mean\_se  
  
ps\_mean\_lower <- (ps\_samp\_mean\_p2 - ps\_samp\_mean\_p1) - 1.96 \* ps\_mean\_se  
  
## superpopulation  
sp\_mean\_se <- sqrt(SE(sp\_samp\_mean\_p1)^2 + SE(sp\_samp\_mean\_p2)^2)  
  
sp\_mean\_upper <- (sp\_samp\_mean\_p2 - sp\_samp\_mean\_p1) + 1.96 \* sp\_mean\_se  
  
sp\_mean\_lower <- (sp\_samp\_mean\_p2 - sp\_samp\_mean\_p1) - 1.96 \* sp\_mean\_se  
  
head(trends)

## difference estimator  
## 1 -0.047773364 Population  
## 2 -0.005404438 Sample  
## 3 -0.022802844 Poststratified  
## 4 0.021574504 Quasi-rand  
## 5 -0.028722478 Superpopulation

trends$lower <- c(NA, sample\_mean\_lower, ps\_mean\_lower, weighted\_mean\_lower, sp\_mean\_lower)  
  
trends$upper <- c(NA, sample\_mean\_upper, ps\_mean\_upper, weighted\_mean\_upper, sp\_mean\_upper)  
  
ggplot(data = trends, aes(x = difference, y = estimator)) +  
 geom\_point() +   
 theme\_linedraw() +  
 geom\_vline(xintercept = 0) +  
 labs(x = "Trend",  
 y = "") +  
 geom\_errorbar(aes(xmin = lower, xmax = upper, width = .2)) +  
 theme(text=element\_text(size = 20))

