```
#   Program to query Census API for ACS data for Social Vulnerability Index
#   Update SVI measures with uncertainty estimate
#   and compute SVI
#   By: Dennis Holt and Michael Laviolette, March 2015
#   michael.laviolette@dhhs.state.nh.us
#   dennis.holt@dhhs.state.nh.us

#   Reference: Flanagan BE et al., “A Social Vulnerability Index for Disaster
#    Management,” Journal of Homeland Security and Emergency Management s8(1),
#    ISSN (Online) 1547-7355, DOI: 10.2202/1547-7355.1792, January 2011.

library(acs)


################################################################################
#  Note: To access ACS through the Census API you must obtain and install an
#       API key.  This only needs to be done once.  For more information see
#       api.key.install in the help section of the acs R package.
################################################################################
ptm <- proc.time()
# suppress warnings "As of the date of this version of the acs package..."
options(warn = -1)
# get total state population (table B01003) for checking
# default span is 5 year aggregate estimate
nh.pop <- acs.fetch(endyear = 2013, geo = geo.make(state = 33),
               table.number = "B01003")
nh.pop  # for 2013
# 1-Year est. 1,323,459
# 3-Year est. 1,321,050
# 5-Year est. 1,319,171  This is the value returned

ctys <- seq(1,19,2)
nh.tract <- geo.make(state = "NH", county = "*", tract = "*")
nh.tract.pop <- acs.fetch(endyear = 2013, geography = nh.tract,
                     table.number = "B01003")
sum(estimate(nh.tract.pop))

##  Three census tracts in NH have zero population.  These causes devide by
##  zero errors and must be excluded from the set of geographic areas.
##  There seems to be a limit on the number of geography objects that can be
##  passed in an acs.fetch. NH has 292 populated census tracts which appears to
##  exceed this limit.  All census tracts within a county can be requested with
##  one geography object. The following determins which counties have tracts
##  with zero population.  For these counties, a list of populated tracts is
##  compiled. The geo set is formed using all tracts within counties that don't
##  have unpopulated tracts plus populated tracts in the other counties plus
##  the whole state.
```

```
zctys <- geography(nh.tract.pop[estimate(nh.tract.pop) == 0])[3]
ztracts <- geography(nh.tract.pop[estimate(nh.tract.pop) == 0])[4]

nh.tract <- geo.make(state = "NH", county = setdiff(ctys,zctys$county),
          tract = "*")

for (i in unique(zctys$county))
 {
 # get all county tracts then subtract those with zero pop
 geo.step2 <- geo.make(state = "NH", county = i, tract = "*")
 temp <- geography(acs.fetch(endyear = 2013, geography = geo.step2,
  table.number = "B01003"))[4]
 nztracts <- setdiff(temp$tract,ztracts$tract)
 geo.step3 <- geo.make(state = "NH", county = i, tract = nztracts)
 # append to geo.set that will not contain tracts with zero pop
 nh.tract <- nh.tract + geo.step3
 }
nh.tract <- nh.tract + geo.make(state = "NH")

## Now we compute the SVI indicators for each populated census tract and the
## state as a whole.
###############   POVERTY   ######################
#########################################################
poverty.data <- acs.fetch(endyear = 2013, geography = nh.tract,
                variable = c("B17001_001", "B17001_002"))
# We should have 292 populated census tracts plus the state value = 293
length(geography(poverty.data)$tract)

# Calculate proportion (% below federal poverty limit)
POVERTY_PCT <- divide.acs(poverty.data[,2], poverty.data[,1],
                method = "proportion")
## adding each indicator to an output dataset (svi.pak) and assign names
svi.pak <- cbind(poverty.data,POVERTY_PCT)
acs.colnames(svi.pak) <- c("POVERTY_DEN", "POVERTY_NUM", "POVERTY_PCT")

#############   UNEMPLOYMENT   ####################
#########################################################
# acs.lookup(endyear = 2013, table.number = "B23001")
den.var <- c("B23001_006", "B23001_013", "B23001_020", "B23001_027",
  "B23001_034", "B23001_041", "B23001_048", "B23001_055", "B23001_062",
  "B23001_069", "B23001_074", "B23001_079", "B23001_084", "B23001_092",
  "B23001_099", "B23001_106", "B23001_113", "B23001_120", "B23001_127",
  "B23001_134", "B23001_141", "B23001_148", "B23001_155", "B23001_160",
  "B23001_165", "B23001_170")
num.var <- c("B23001_008", "B23001_015", "B23001_022", "B23001_029",
  "B23001_036", "B23001_043", "B23001_050", "B23001_057", "B23001_064",
  "B23001_071", "B23001_076", "B23001_081", "B23001_086", "B23001_094",
  "B23001_101", "B23001_108", "B23001_115", "B23001_122", "B23001_129",
```

```
 "B23001_136", "B23001_143", "B23001_150", "B23001_157", "B23001_162",
 "B23001_167", "B23001_172")

unemp.data <- acs.fetch(endyear = 2013, geography = nh.tract,
  variable = append(den.var,num.var))

dim(unemp.data)  #  293  52

denom <- apply(unemp.data[,den.var], MARGIN = 2, FUN = sum,
          agg.term = c("y", "denom"), one.zero = TRUE)
numer <- apply(unemp.data[,num.var], MARGIN = 2, FUN = sum,
          agg.term = c("y", "numer"), one.zero = TRUE)
UNEMPLOYED_PCT <-  divide.acs(numer, denom, method = "proportion")

svi.pak <- cbind(svi.pak, denom)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- "UNEMPLOYED_DEN"
svi.pak <- cbind(svi.pak, numer)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- "UNEMPLOYED_NUM"
svi.pak <- cbind(svi.pak, UNEMPLOYED_PCT)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- "UNEMPLOYED_PCT"

############  Per capita income  ###################
#########################################################
# acs.lookup(endyear = 2013, table.number = "B19301")
# B19301 PER CAPITA INCOME IN THE PAST 12 MONTHS (IN 2013 INFLATION-
#   ADJUSTED DOLLARS)
# The Census Bureau uses the Bureau of Labor Statistics (BLS) Consumer Price
#   Index (CPI-U) to adjust for changes in the cost of living

pci <- acs.fetch(endyear = 2013, geography = nh.tract, variable = "B19301_001")
currency.year(pci)
dim(pci)
svi.pak <- cbind(svi.pak, pci)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- "INCOME_PER_CAP"

##########   Educational Attainment   ###############
#########################################################
# acs.lookup(endyear = 2013, table.number = "B15001")
den.var <- c("B15001_011", "B15001_019", "B15001_027", "B15001_035",
         "B15001_052", "B15001_060", "B15001_068", "B15001_076")
num.var <- c("B15001_012", "B15001_013", "B15001_020", "B15001_021",
         "B15001_028", "B15001_029", "B15001_036", "B15001_037",
         "B15001_053", "B15001_054", "B15001_061", "B15001_062",
         "B15001_069", "B15001_070", "B15001_077", "B15001_078")
edu.data <- acs.fetch(endyear = 2013, geography = nh.tract,
                  variable = append(den.var, num.var))

denom <- apply(edu.data[,den.var], MARGIN = 2, FUN = sum,
```

```
        agg.term = c("y", "denom"), one.zero = TRUE)
numer <- apply(edu.data[,num.var], MARGIN = 2, FUN = sum,
        agg.term = c("y", "numer"), one.zero = TRUE)
EDU_NO_HS_DIP_PCT <-  divide.acs(numer, denom, method = "proportion")
svi.pak <- cbind(svi.pak, denom)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- "EDU_NO_HS_DIP_DEN"
svi.pak <- cbind(svi.pak, numer)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- "EDU_NO_HS_DIP_NUM"
svi.pak <- cbind(svi.pak, EDU_NO_HS_DIP_PCT)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- "EDU_NO_HS_DIP_PCT"

#############   NO HEALTHINSURANCE   ####################
#####################################################################
##  insurance table apparently not accessible through acs package
##  pulled in from FactFinder2 as csv (zipped)
##  rearrange the rows to match the output from acs.fetch
#####################################################################
# change the following path to wherever you saved the FactFinder file
setwd("R:/OCPH/EPI/BHSDM/Group/Data Requests/Customer Folders/2015/H/Holt, Dennis/5548 SVI
up-date 2013 ACS data")
ins.data <- read.acs(filename = "ACS_13_5YR_B27010.zip", endyear = 2013)
dim(ins.data)  #   295  66
acs.colnames(ins.data)
den.var <- c(2, 18, 34)
num.var <- c(17, 33, 50)
tract.lst <- substr(geography(ins.data)$Id2,6,13)
length(tract.lst)  # 295  all here including zero pop tracts
denom <- ins.data[,2] + ins.data[,18] + ins.data[,34]
# couldn't get sum() to work on this imported acs object
numer <- ins.data[,17] + ins.data[,33] + ins.data[,50]
denom[estimate(denom) == 0]
# remove the zero pop tracts.  they cause error in divide.acs()
denom2 <- denom[!tract.lst %in% ztracts$tract]
numer2 <- numer[!tract.lst %in% ztracts$tract]
NO_INSURANCE_PCT <-  divide.acs(numer2, denom2, method = "proportion")
acs.colnames(denom2) <- "NO_INSURANCE_DEN"
acs.colnames(numer2) <- "NO_INSURANCE_NUM"
acs.colnames(NO_INSURANCE_PCT) <- "NO_INSURANCE_PCT"
NO_INSURANCE_PCT[288:292]

#  Need to build acs object with geo in same order as svi.pak
#  Need to add state level geo.  May require another import because MOE could be
#  larger if we sum over all the tracts.  We will check this.
str(geography(svi.pak))
str(geography(NO_INSURANCE_PCT))
tract.ins <- substr(geography(NO_INSURANCE_PCT)$Id2, 6, 13)
tract.svi <- geography(svi.pak)$tract
```

```
# combine measures into one acs object
no_ins <- cbind(denom2, numer2)
no_ins <- cbind(no_ins, NO_INSURANCE_PCT)

# rearrange the order of the geos (rows) to match the svi.pak
tract.sort <- data.frame(tract = tract.ins, indx = 1:292,
                         stringsAsFactors = FALSE)
tract.to <- data.frame(tract = tract.svi[1:292], org.seq = 1:292,
                       stringsAsFactors = FALSE)
temp <- merge(tract.to, tract.sort, by = "tract")
temp2 <- temp[order(temp$org.seq),]
# acs objects can not be rearranged in a vector assignment the elements must be
# unpacked and the new object built up in the desired geo order
rm(no_ins2)
no_ins2 <- no_ins[temp2$indx[1],]
for (i in 2:292) no_ins2 <- rbind(no_ins2, no_ins[temp2$indx[i],])

# Check to see that the order is really changed to what we want
# str(no_ins2)
# no_ins[253:257,]
# no_ins2[288:292,]
# attach proper formated tract number in first column
tract.ins2 <- substr(geography(no_ins2)$Id2, 6, 13)
temp3 <- cbind(tract.svi[1:292], tract.ins2)

# now we need to get the last geo which is the state level data
ins_nh.data <- read.acs(filename = "ACS_13_5YR_B27010_NH.zip", endyear= 2013,
                        span = 5)
denom <- ins_nh.data[,2] + ins_nh.data[,18] + ins_nh.data[,34]
numer <- ins_nh.data[,17] + ins_nh.data[,33] + ins_nh.data[,50]
no_ins_state <-  divide.acs(numer, denom, method = "proportion")
no_ins_nh <- cbind(denom, numer)
no_ins_nh <- cbind(no_ins_nh, no_ins_state)
acs.colnames(no_ins_nh) <- c("NO_INSURANCE_DEN", "NO_INSURANCE_NUM",
                             "NO_INSURANCE_PCT")
no_ins2 <- rbind(no_ins2, no_ins_nh)
dim(no_ins2)
no_ins2[288:293,]
dim(svi.pak)
svi.pak <- cbind(svi.pak, no_ins2)
dim(svi.pak)  # 13 variables

#  Explore difference in MOE when summing to state level
svi.pak[292, 11]
d <- sum(svi.pak[1:292, 11])
n <- sum(svi.pak[1:292, 12], one.zero = TRUE)
pct <- divide.acs(n, d, method = "proportion")  # 0.1212335 se=0.00158995
lims <- as.data.frame(confint(pct, level = 0.6826))
```

```
ag.pct.se <- (lims[2] - lims[1]) / (2 * estimate(pct))   #  0.01311469
svi.pak[293,13]    # 0.121233482525812 se=0.00163425   from state value in table
lims2 <- as.data.frame(confint(svi.pak[293,13], level = 0.6826))
ag.pct.se2 <- (lims2[2] - lims2[1]) / (2 * estimate(svi.pak[293,13]))
  # 0.01348029
# Aggregate percent standard error is very slightly smaller 0.036% difference
# So didn't need to import the state value; could have gotten there with sum()

##########   Population Measures    #################
#########################################################
# ACS_POP          B01001
# POP_0_17_PCT
# POP_18_24_PCT
# POP_65PLUS_PCT
acs.lookup(endyear = 2013, table.number = "B01001")
# 1     B01001_001                 Total:
# 2     B01001_002                  Male:
# 3     B01001_003       Male: Under 5 years
# 4     B01001_004        Male: 5 to 9 years
# 5     B01001_005      Male: 10 to 14 years
# 6     B01001_006      Male: 15 to 17 years
# 7     B01001_007     Male: 18 and 19 years
# 8     B01001_008           Male: 20 years
# 9     B01001_009           Male: 21 years
# 10    B01001_010      Male: 22 to 24 years
# 11    B01001_011      Male: 25 to 29 years
# 12    B01001_012      Male: 30 to 34 years
# 13    B01001_013      Male: 35 to 39 years
# 14    B01001_014      Male: 40 to 44 years
# 15    B01001_015      Male: 45 to 49 years
# 16    B01001_016      Male: 50 to 54 years
# 17    B01001_017      Male: 55 to 59 years
# 18    B01001_018     Male: 60 and 61 years
# 19    B01001_019      Male: 62 to 64 years
# 20    B01001_020     Male: 65 and 66 years
# 21    B01001_021      Male: 67 to 69 years
# 22    B01001_022      Male: 70 to 74 years
# 23    B01001_023      Male: 75 to 79 years
# 24    B01001_024      Male: 80 to 84 years
# 25    B01001_025    Male: 85 years and over
# 26    B01001_026                Female:
# 27    B01001_027     Female: Under 5 years
# 28    B01001_028      Female: 5 to 9 years
# 29    B01001_029     Female: 10 to 14 years
# 30    B01001_030     Female: 15 to 17 years
# 31    B01001_031    Female: 18 and 19 years
# 32    B01001_032          Female: 20 years
# 33    B01001_033          Female: 21 years
```

```
# 34    B01001_034     Female: 22 to 24 years
# 35    B01001_035     Female: 25 to 29 years
# 36    B01001_036     Female: 30 to 34 years
# 37    B01001_037     Female: 35 to 39 years
# 38    B01001_038     Female: 40 to 44 years
# 39    B01001_039     Female: 45 to 49 years
# 40    B01001_040     Female: 50 to 54 years
# 41    B01001_041     Female: 55 to 59 years
# 42    B01001_042    Female: 60 and 61 years
# 43    B01001_043     Female: 62 to 64 years
# 44    B01001_044    Female: 65 and 66 years
# 45    B01001_045     Female: 67 to 69 years
# 46    B01001_046     Female: 70 to 74 years
# 47    B01001_047     Female: 75 to 79 years
# 48    B01001_048     Female: 80 to 84 years
# 49    B01001_049  Female: 85 years and over
pop.data <- acs.fetch(endyear = 2013, geography = nh.tract,
                table.number = "B01001")
# dim(pop.data)
svi.pak <- cbind(svi.pak, pop.data[,1])
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- "ACS_POP"
# dim(svi.pak)
# svi.pak[293,14]   #  New Hampshire 1319171 +/- 0
POP_0_17_NUM <- apply(pop.data[,c(3,4,5,6,27,28,29,30)], MARGIN = 2,
                FUN = sum, agg.term = c("y", "POP_0_17_NUM"),
                one.zero = TRUE)
POP_0_17_NUM[292:293,]
POP_18_24_NUM <- apply(pop.data[,c(7,8,9,10,31,32,33,34)], MARGIN = 2,
                 FUN = sum, agg.term = c("y", "POP_18_24_NUM"),
                 one.zero = TRUE)
POP_18_24_NUM[292:293,]
POP_25_64_NUM <- apply(pop.data[,c(11,12,13,14,15,16,17,18,19,35,36,37,38,39,
                         40,41,42,43)], MARGIN = 2, FUN = sum,
                 agg.term = c("y", "POP_25_64_NUM"), one.zero = TRUE)
POP_25_64_NUM[292:293,]
POP_65PLUS_NUM <- apply(pop.data[,c(20,21,22,23,24,25,44,45,46,47,48,49)],
                  MARGIN = 2, FUN = sum,
                  agg.term = c("y", "POP_65PLUS_NUM"), one.zero = TRUE)
POP_65PLUS_NUM[292:293,]
acs_pop2 <- POP_0_17_NUM + POP_18_24_NUM + POP_25_64_NUM + POP_65PLUS_NUM
# acs_pop2[292:293,] # Tract 1075, 4384 +/- 328.5
# svi.pak[292:293, 14] # Tract 1075, 4384 +/- 111
# adding the groups increases the standard error by about 3 times
svi.pak <- cbind(svi.pak, POP_0_17_NUM)
# svi.pak[292:293, 15]
POP_0_17_PCT <-  divide.acs(POP_0_17_NUM, svi.pak[,14], method = "proportion")
svi.pak <- cbind(svi.pak, POP_0_17_PCT)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- "POP_0_17_PCT"
```

```
# svi.pak[292:293, 14:16]
svi.pak <- cbind(svi.pak, POP_18_24_NUM)
POP_18_24_PCT <- divide.acs(POP_18_24_NUM, svi.pak[,14], method = "proportion")
svi.pak <- cbind(svi.pak, POP_18_24_PCT)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- "POP_18_24_PCT"
# svi.pak[292:293, 14:18]
svi.pak <- cbind(svi.pak, POP_65PLUS_NUM)
POP_65PLUS_PCT <-  divide.acs(POP_65PLUS_NUM, svi.pak[,14],
                      method="proportion")
svi.pak <- cbind(svi.pak, POP_65PLUS_PCT)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- "POP_65PLUS_PCT"
# svi.pak[292:293, 14:20]
# dim(svi.pak)

################  DISABLED  ########################
#########################################################
# acs.lookup(endyear = 2013, table.number = "B18101")
# disability table apparently not accessible through acs package
# pulled in from FactFinder2 as csv (ziped)

disabl.data <- read.acs(filename="ACS_13_5YR_B18101.zip", endyear= 2013,
                  span = 5)
dim(disabl.data)   #   295  39
acs.colnames(disabl.data)
# [1] 01Total:"
# [3] Male: - Under 5 years:"
# [22] Female: - Under 5 years:"
# [7] 07Male: - 5 to 17 years: - With a disability"
# [10] 10Male: - 18 to 34 years: - With a disability"
# [13] 13Male: - 35 to 64 years: - With a disability"
# [16] 16Male: - 65 to 74 years: - With a disability"
# [19] 19Male: - 75 years and over: - With a disability"
# [26] 26Female: - 5 to 17 years: - With a disability"
# [29] 29Female: - 18 to 34 years: - With a disability"
# [32] 32Female: - 35 to 64 years: - With a disability"
# [35] 35Female: - 65 to 74 years: - With a disability"
# [38] 38Female: - 75 years and over: - With a disability"

tract.lst <- substr(geography(disabl.data)$Id2, 6, 13)
length(tract.lst)  # 295  all here including zero pop tracts
denom <- disabl.data[,1] - disabl.data[,3] - disabl.data[,22]
acs.colnames(denom) <- "DISABLED_DEN"
numer <- disabl.data[,7] + disabl.data[,10] + disabl.data[,13] +
  disabl.data[,16] + disabl.data[,19] + disabl.data[,26] + disabl.data[,29] +
  disabl.data[,32] + disabl.data[,35] + disabl.data[,38]
acs.colnames(numer) <- "DISABLED_NUM"
numer[292:295]
denom[estimate(denom) == 0]
```

```
# remove the zero pop tracts.  they cause error in divide.acs()
denom2 <- denom[!tract.lst %in% ztracts$tract]
numer2 <- numer[!tract.lst %in% ztracts$tract]
DISABLED_PCT <-  divide.acs(numer2, denom2, method = "proportion")
acs.colnames(DISABLED_PCT) <- "DISABLED_PCT"
DISABLED_PCT[288:292]
svi.pak[288:293, 3]

#  Need to build acs object with geo in same order as svi.pak
#  Need to add state level geo.  Will require another import because MOE will be
#  way too large if we sum over all the tracts.
tract.ins <- substr(geography(DISABLED_PCT)$Id2, 6, 13)
tract.svi <- geography(svi.pak)$tract
# combine measures into one acs object
disabled <- cbind(denom2, numer2)
disabled <- cbind(disabled, DISABLED_PCT)
dim(disabled)
disabled[292]

# rearrange the order of the geos (rows) to match the svi.pak
tract.sort <- data.frame(tract = tract.ins, indx = 1:292,
                         stringsAsFactors = FALSE)
tract.to <- data.frame(tract = tract.svi[1:292], org.seq = 1:292,
                       stringsAsFactors = FALSE)
temp <- merge(tract.to,tract.sort, by = "tract")
temp2 <- temp[order(temp$org.seq),]
# acs objects can not be rearranged in a vector assignment the elements must be
# unpacked and the new object built up in the desired geo order
rm(disabled2)
disabled2 <- disabled[temp2$indx[1],]
for (i in 2:292) disabled2 <- rbind(disabled2,disabled[temp2$indx[i],])

# Check to see that the order is really changed to what we want
# str(disabled2)
# disabled[253:257,]
# disabled2[288:292,]
tract.ins2 <- substr(geography(disabled2)$Id2, 6, 13)
temp3 <- cbind(tract.svi[1:292], tract.ins2)
temp3[!temp3[,1]==temp3[,2],]
# dim(temp3)
# now we need to get the last geo which is the state level data
dis_nh.data <- read.acs(filename = "ACS_13_5YR_B18101_NH.zip", endyear= 2013,
                        span = 5)
acs.colnames(dis_nh.data)
denom <- dis_nh.data[,1] - dis_nh.data[,3] - dis_nh.data[,22]
acs.colnames(denom) <- "DISABLED_DEN"
numer <- dis_nh.data[,7] + dis_nh.data[,10] + dis_nh.data[,13] +
  dis_nh.data[,16] + dis_nh.data[,19] + dis_nh.data[,26] + dis_nh.data[,29] +
```

```
  dis_nh.data[,32] + dis_nh.data[,35] + dis_nh.data[,38]
acs.colnames(numer) <- "DISABLED_NUM"
dis_state <-  divide.acs(numer, denom, method = "proportion")
acs.colnames(dis_state) <- "DISABLED_PCT"
dis_nh <- cbind(denom, numer)
dis_nh <- cbind(dis_nh, dis_state)
dis_nh
disabled2 <- rbind(disabled2, dis_nh)
# dim(disabled2)
# disabled2[288:293,]
# dim(svi.pak)
svi.pak <- cbind(svi.pak, disabled2)
# dim(svi.pak)
# acs.colnames(svi.pak)

###############   SINGLE_PARENT   ######################
############################################################
# SINGLE_PARENT_PCT  B11005 Universe:  Households
# Married-couple family: family in which the householder and in the household.
# Other family:
#  Male householder, no wife present.
#  Female householder, no husband present.
# Nonfamily household. householder living alone or with nonrelatives only.
#
acs.lookup(endyear = 2013, table.number = "B11005")
# 2 B11005_002 Households with one or more people under 18 years (denominator)
# 6 B11005_006 Households with child: Male householder, no wife present
# 7 B11005_007 Households with child: Female householder, no husband present
# 9 B11005_009 Households with child: Nonfamily households: Male householder
# 10 B11005_010 Households with child: Nonfamily households: Female householder
# (numerator = 6 + 7 + 9 + 10)

den.var <- "B11005_002"
num.var <- c("B11005_006", "B11005_007", "B11005_009", "B11005_010")
dat <- acs.fetch(endyear = 2013, geography = nh.tract,
                 variable = append(den.var, num.var))
#pop.data <- acs.fetch(endyear = 2013, geography = nh.tract,
#                  table.number = "B01001")
# dim(dat)
# acs.colnames(dat)
den <- dat[,1]
svi.pak <- cbind(svi.pak, den)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- "SINGLE_PARENT_DEN"
# dim(svi.pak)
# acs.colnames(svi.pak)
num <- apply(dat[,c(2,3,4,5)], MARGIN = 2, FUN = sum, agg.term = c("y", "NUM"),
          one.zero = TRUE)
svi.pak <- cbind(svi.pak, num)
```

```
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- "SINGLE_PARENT_NUM"
pct <-  divide.acs(num, den, method = "proportion")
svi.pak <- cbind(svi.pak,pct)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- "SINGLE_PARENT_PCT"
# dim(svi.pak)
# acs.colnames(svi.pak)
# svi.pak[288:293, 24:26]
# dim(svi.pak)

#################   MINORITY   ##########################
############################################################
#  B02001 & B03002 Universe:  Total population
measure <- "MINORITY"
acs.lookup(endyear = 2013, table.number = "B02001")
acs.lookup(endyear = 2013, table.number = "B03002")
# B02001_001  Total:
# B02001_002  White alone
# B03002_013  Hispanic or Latino: White alone
dat <- acs.fetch(endyear = 2013, geography = nh.tract,
            variable = c("B02001_001", "B02001_002", "B03002_013"))
# dim(dat)
# acs.colnames(dat)
den <- dat[,1]
svi.pak <- cbind(svi.pak, den)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- paste0(measure, "_DEN")
# dim(svi.pak)
# acs.colnames(svi.pak)
num <- dat[,1] - dat[,2] + dat[,3]
svi.pak <- cbind(svi.pak, num)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- paste0(measure, "_NUM")
pct <- divide.acs(num, den, method = "proportion")
svi.pak <- cbind(svi.pak, pct)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- paste0(measure, "_PCT")
size <- dim(svi.pak)
# acs.colnames(svi.pak)
# svi.pak[seq(size[1] - 5, size[1]), seq(size[2] - 3, size[2])]
# dim(svi.pak)

#################   LIMITED ENGLISH   ######################
#############################################################
#  B16005  Universe:  Population 5 years and over
measure <- "POOR_ENGLISH"
acs.lookup(endyear = 2013, table.number = "B16005")
# B16005_001  Total:
# B16005_007  Native:  Spanish:  English 'not well'
# B16005_008  Native:  Spanish:  English 'not at all'
# B16005_012  Native:  other Indo-European:  English 'not well'
# B16005_013  Native:  other Indo-European:  English 'not at all'
```

```
# B16005_017  Native:  Asian and Pacific Island:  English 'not well'
# B16005_018  Native:  Asian and Pacific Island:  English 'not at all'
# B16005_022  Native:  other:  English 'not well'
# B16005_023  Native:  other:  English 'not at all'
# B16005_029  Foreign born:  Spanish:  English 'not well'
# B16005_030  Foreign born:  Spanish:  English 'not at all'
# B16005_034  Foreign born:  other Indo-European:  English 'not well'
# B16005_035  Foreign born:  other Indo-European:  English 'not at all'
# B16005_039  Foreign born:  Asian and Pacific Island:  English 'not well'
# B16005_040  Foreign born:  Asian and Pacific Island:  English 'not at all'
# B16005_044  Foreign born:  other:  English 'not well'
# B16005_045  Foreign born:  other:  English 'not at all'

dat <- acs.fetch(endyear = 2013, geography = nh.tract,
            variable= c("B16005_001", "B16005_007", "B16005_008",
                  "B16005_012", "B16005_013", "B16005_017",
                  "B16005_018", "B16005_022", "B16005_023",
                  "B16005_029", "B16005_030", "B16005_034",
                  "B16005_035", "B16005_039", "B16005_040",
                  "B16005_044", "B16005_045"))
# dim(dat)
# acs.colnames(dat)
den <- dat[,1]
svi.pak <- cbind(svi.pak, den)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- paste0(measure, "_DEN")
# dim(svi.pak)
# acs.colnames(svi.pak)
num <- apply(dat[,2:17], MARGIN = 2, FUN = sum, agg.term = c("y", "NUM"),
         one.zero = TRUE)
# dim(num)
# num[293]
svi.pak <- cbind(svi.pak,num)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- paste0(measure, "_NUM")
pct <- divide.acs(num, den, method = "proportion")
svi.pak <- cbind(svi.pak, pct)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- paste0(measure, "_PCT")
size <- dim(svi.pak)
# acs.colnames(svi.pak)
# svi.pak[seq(size[1] - 5, size[1]), seq(size[2] - 3, size[2])]
# dim(svi.pak)

##################   Large Apartments   ##########################
#######################################################################
#  B25024  Universe:  Housing units
measure <- "H_UNITS_10PLUS"
acs.lookup(endyear = 2013, table.number = "B25024")
#  B25024_001 Total:
#  B25024_007 10 to 19
```

```
#  B25024_008 20 to 49
#  B25024_009 50 or more
#  B25024_010 Mobile home

dat <- acs.fetch(endyear = 2013, geography = nh.tract,
              variable = c("B25024_001", "B25024_007", "B25024_008",
                       "B25024_009", "B25024_010"))
# dim(dat)
# acs.colnames(dat)
den <- dat[,1]
svi.pak <- cbind(svi.pak, den)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- paste0(measure,"_DEN")
# dim(svi.pak)
# acs.colnames(svi.pak)
num <- apply(dat[,2:4], MARGIN = 2, FUN = sum, agg.term = c("y", "NUM"),
             one.zero=TRUE)
# dim(num)
# num[293]
svi.pak <- cbind(svi.pak, num)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- paste0(measure,"_NUM")
pct <-  divide.acs(num, den, method = "proportion")
svi.pak <- cbind(svi.pak, pct)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- paste0(measure,"_PCT")
size <- dim(svi.pak)
# acs.colnames(svi.pak)
# svi.pak[seq(size[1] - 5, size[1]), seq(size[2] - 3, size[2])]

##################     MOBILE HOME     ##########################
#########################################################################
#  B25024  Universe:  Housing units
measure <- "MOBILE_HOME"
num <- dat[,5]
svi.pak <- cbind(svi.pak, num)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- paste0(measure, "_NUM")
pct <-  divide.acs(num, den, method = "proportion")
svi.pak <- cbind(svi.pak, pct)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- paste0(measure, "_PCT")
size <- dim(svi.pak)
# acs.colnames(svi.pak)
# svi.pak[seq(size[1] - 5, size[1]), seq(size[2] - 3, size[2])]

##################      CROWDING      ##########################
#########################################################################
#  B25050 & B25044  Universe:  Occupied housing units
measure <- "CROWDING"
acs.lookup(endyear = 2013, table.number = "B25050")
acs.lookup(endyear = 2013, table.number = "B25044")
#  B25050_001 Total:
```

```
#  B25050_007 Complete plumbing facilities: 1.01 or more occupants per room:
#  B25050_016 Lacking complete plumbing: 1.01 or more occupants per room:
#  B25044_003 Owner occupied: No vehicle available
#  B25044_010 Renter occupied: No vehicle available

dat <- acs.fetch(endyear = 2013, geography = nh.tract,
            variable = c("B25050_001", "B25050_007", "B25050_016",
                    "B25044_003", "B25044_010"))
# dim(dat)
# acs.colnames(dat)
den <- dat[,1]
svi.pak <- cbind(svi.pak, den)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- paste0(measure, "_DEN")
# dim(svi.pak)
# acs.colnames(svi.pak)
num <- apply(dat[,2:3], MARGIN = 2, FUN = sum, agg.term = c("y", "NUM"),
          one.zero=TRUE)
# dim(num)
# num[293]
svi.pak <- cbind(svi.pak, num)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- paste0(measure, "_NUM")
pct <- divide.acs(num, den, method = "proportion")
svi.pak <- cbind(svi.pak, pct)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- paste0(measure, "_PCT")
# size <- dim(svi.pak)
# acs.colnames(svi.pak)
# svi.pak[seq(size[1] - 5, size[1]), seq(size[2] - 3, size[2])]

#################     NO_CAR     ##########################
#########################################################################
#  B25024  Universe:  Housing units
measure <- "NO_CAR"
num <- apply(dat[,4:5], MARGIN = 2, FUN = sum, agg.term = c("y", "NUM"),
          one.zero = TRUE)
svi.pak <- cbind(svi.pak, num)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- paste0(measure, "_NUM")
pct <- divide.acs(num, den, method = "proportion")
svi.pak <- cbind(svi.pak, pct)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- paste0(measure, "_PCT")
# size <- dim(svi.pak)
# acs.colnames(svi.pak)
# svi.pak[seq(size[1] - 5, size[1]), seq(size[2] -3 ,size[2])]


#################      GROUP_QTR      ##########################
#########################################################################
#  B26001  Universe: Population in Group Quarters
measure <- "GROUP_QTR"
```

```
acs.lookup(endyear = 2013, table.number = "B26001")
#  B26001_001   Group Quarters Population       Total:

dat <- acs.fetch(endyear = 2013, geography = nh.tract, variable = "B26001_001")
# denominator is total ACS population
acs.colnames(svi.pak)
den <- svi.pak[,14]
num <- dat[,1]
# num[293]
svi.pak <- cbind(svi.pak, num)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- paste0(measure, "_NUM")
pct <-  divide.acs(num, den, method = "proportion")
svi.pak <- cbind(svi.pak,pct)
acs.colnames(svi.pak)[length(acs.colnames(svi.pak))] <- paste0(measure,"_PCT")
# size <- dim(svi.pak)
# acs.colnames(svi.pak)
# svi.pak[seq(size[1] - 5, size[1]), seq(size[2] - 3, size[2])]

# dim(svi.pak)
# acs.colnames(svi.pak)
#############################################################################
##  We have completed getting the indicators from ACS
##  Save and build the SVI
#############################################################################
save(nh.tract, ztracts, svi.pak, file = "svi_pak.Rdata")

##############  build SVI  ##############################################
##  We need to identify the most vulnerable 10% of census tracts for each
##  indicator.
#############################################################################
pct.rank <- function(x) rank(x,na.last = FALSE,
                     ties.method = "average") / length(x)


TRACT_10 <- geography(svi.pak)$tract
COUNTY <- sprintf("%02d", geography(svi.pak)$county)
## get area of each tract for computing popultion density
tArea <- read.csv(file = "2010tract_Area.csv")
tArea$TRACT_10 <- sprintf("%06d", tArea$TRACTCE)
est1 <- as.data.frame(cbind(TRACT_10, COUNTY), stringsAsFactors = FALSE)
est2 <- merge(est1, tArea[5:6], by = "TRACT_10", all.x = TRUE)
est.svi <- as.data.frame(estimate(svi.pak), stringsAsFactors = FALSE)
est.svi <- cbind(TRACT_10, est.svi)
est <- merge(est2, est.svi, by = "TRACT_10", all.y = TRUE)

est$POP_DENSITY <- est$ACS_POP/est$LAND_SQ_MI
colnames(est)
# [1] "TRACT_10"         "COUNTY"           "LAND_SQ_MI"       "POVERTY_DEN"
```

```
# [5] "POVERTY_NUM"       "POVERTY_PCT"       "UNEMPLOYED_DEN"    "UNEMPLOYED_NUM"
# [9] "UNEMPLOYED_PCT"    "INCOME_PER_CAP"    "EDU_NO_HS_DIP_DEN"  "EDU_NO_HS_DIP_NUM"
# [13] "EDU_NO_HS_DIP_PCT"  "NO_INSURANCE_DEN"   "NO_INSURANCE_NUM"
"NO_INSURANCE_PCT"
# [17] "ACS_POP"          "POP_0_17_NUM"      "POP_0_17_PCT"      "POP_18_24_NUM"
# [21] "POP_18_24_PCT"     "POP_65PLUS_NUM"    "POP_65PLUS_PCT"    "DISABLED_DEN"
# [25] "DISABLED_NUM"      "DISABLED_PCT"      "SINGLE_PARENT_DEN"  "SINGLE_PARENT_NUM"
# [29] "SINGLE_PARENT_PCT"  "MINORITY_DEN"      "MINORITY_NUM"      "MINORITY_PCT"
# [33] "POOR_ENGLISH_DEN"   "POOR_ENGLISH_NUM"   "POOR_ENGLISH_PCT"
"H_UNITS_10PLUS_DEN"
# [37] "H_UNITS_10PLUS_NUM" "H_UNITS_10PLUS_PCT" "MOBILE_HOME_NUM"
"MOBILE_HOME_PCT"
# [41] "CROUDED_DEN"       "CROUDED_NUM"       "CROUDED_PCT"       "NO_CAR_NUM"
# [45] "NO_CAR_PCT"        "GROUP_QTR_NUM"     "GROUP_QTR_PCT"     "POP_DENSITY"

# extract columns for SVI Spreadsheet
est1 <- est[c(1,2,17,3,48,21,10,6,9,16,13,19,23,29,26,32,35,38,40,43,45,47)]
names(est1)
# [1] "TRACT_10"          "COUNTY"            "ACS_POP"           "LAND_SQ_MI"
# [5] "POP_DENSITY"       "POP_18_24_PCT"     "INCOME_PER_CAP"    "POVERTY_PCT"
# [9] "UNEMPLOYED_PCT"    "NO_INSURANCE_PCT"   "EDU_NO_HS_DIP_PCT"  "POP_0_17_PCT"
# [13] "POP_65PLUS_PCT"    "SINGLE_PARENT_PCT"  "DISABLED_PCT"      "MINORITY_PCT"
# [17] "POOR_ENGLISH_PCT"   "H_UNITS_10PLUS_PCT" "MOBILE_HOME_PCT"   "CROUDED_PCT"
# [21] "NO_CAR_PCT"        "GROUP_QTR_PCT"

# Make list of measures by colum number
mes <- c(6:22)

# Calculate percentile ranking for each measure
for (i in mes) {
  nm <- colnames(est1[i])
  est1[,paste0(substr(nm,1,nchar(nm)-4), "_PCTL")] <-
    100 * pct.rank(as.numeric(est1[,i]))
  est1[i] <- est1[i]*100   # Convert to percent to match previous year output.
  }
# Undo multiplying income per capita by 100
est1$INCOME_PER_CAP <- est1$INCOME_PER_CAP / 100
# invert the percentile for Per Capata Income so lower income is higher pctl
est1$INCOME_PCTL <- 100 - est1$INCOME_PER_PCTL

# Add SPACER variable and placeholder for old SVI and College age flag
est1$SPACER <- 100
est1$SVI_15 <- 0
est1$AGE_18_24_FLAG <- 0
est1$AGE_18_24_FLAG[est1$POP_18_24_PCTL >= 90]  <- 1
summary(est1$POP_18_24_PCTL)
summary(est1$AGE_18_24_FLAG)
```

```
detach("package:acs")
library(dplyr)
library(tidyr)

# SVI is sum of the measures at or above 90th percentile
# for arranging and manipulating the data we find the use of the
# dplyr and tidyr package very useful
test2 <- est1 %>%
  tbl_df() %>%
  select(contains("_PCTL"), -INCOME_PER_PCTL, -POP_18_24_PCTL) %>%
  mutate_each(funs(flag= as.numeric(. >= 90.0)))
SVI_16 <- apply(as.data.frame(test2), 1, sum)
est1 <- cbind(est1, SVI_16)
names(est1)
# rearrange to match old spreadsheet
est1 <- est1[c(1:22,40,41,25:39,44,42,23,43)]
# when importing this file in Excel set first two columns to Text
write.csv(est1, file = "est1.csv", row.names = FALSE, quote = TRUE)
# Aft import add census tracts with no population to the end and remove NH avg
# Calculate disparity for each indicator
# est > select > sort > subset > calc

# perhaps acs is fouling up rbind() below


##############################################################################
##  For each indicator calculate the ratio of most vulnerable 20% of census
##  tracts to the least vulnerable 20% of tracts. This ratio is a gauge of
##  disparity.  We will build a summary table of the indicators
##############################################################################
disTable <- new('data.frame')
rcrd <- data.frame(ind = "NA", nhNum = 0, nhDen = 0, nhInd = 0, mNeedy = 0,
                   lNeedy = 0, disRato = 0)
state <- as.data.frame(est[293,])

# calc per capita income seperately because it has different structure
rcrd$ind <- "Per capita income"
rcrd$nhNum <- as.numeric(0)
rcrd$nhDen <- as.numeric(0)
rcrd$nhInd <- state$INCOME_PER_CAP
temp <- est[1:292,] %>%
  tbl_df() %>%
  select(ACS_POP, INCOME_PER_CAP) %>%
  mutate(popIncome = ACS_POP * INCOME_PER_CAP) %>%
  arrange(INCOME_PER_CAP)
mNdy <- as.data.frame(summarise_each(temp[1:58,], funs(sum)))
rcrd$mNeedy  <- mNdy$popIncome/mNdy$ACS_POP
lNdy <- as.data.frame(summarise_each(temp[(292-57):292,], funs(sum)))
```

```
rcrd$lNeedy  <- lNdy$popIncome / lNdy$ACS_POP
rcrd$disRato <- rcrd$lNeedy / rcrd$mNeedy
disTable <- rcrd

indName <- c("Poverty", "Unemployed", "Education", "Health Insurance",
          "Children", "Elderly", "Disability", "Single Parent",
          "Minority", "Limited English", "Large Apt. bldgs.",
          "Mobile homes", "Crowding", "No vehicle", "Group Quarters")

num <- c(5,8,12,15,18,22,25,28,31,34,37,39,42,44,46)
den <- c(4,7,11,14,17,17,24,27,30,33,36,36,41,41,17)
ind <- c(6,9,13,16,19,23,26,29,32,35,38,40,43,45,47)

for(i in 1:length(indName)){
  rcrd <- data.frame(lapply(rcrd, function(x){x <- NA}))
  rcrd$ind <- indName[i]; rcrd$nhNum <- state[num[i]];
  rcrd$nhDen <- state[den[i]]; rcrd$nhInd <- state[ind[i]];
  indNa <- names(est)[ind[i]]
  temp <- est[,c(num[i],den[i],ind[i])] %>% tbl_df() %>% arrange_(indNa)
  lNdy <- as.data.frame(summarise_each(temp[1:58,], funs(sum)))
  rcrd$lNeedy  <- lNdy[1] / lNdy[2]
  mNdy <- as.data.frame(summarise_each(temp[(292-57):292,], funs(sum)))
  rcrd$mNeedy  <- mNdy[1] / mNdy[2]
  rcrd$disRato <- rcrd$mNeedy / rcrd$lNeedy
  disTable <- rbind(disTable, rcrd)
  }

#  some ratio values are 'inf' (divide by zero)
#  I think this messes up the data frame (becomes a dataframe of lists)
#  This flattens the data frame so write.csv can handle it
temp2 <- data.frame(lapply(disTable, as.character), stringsAsFactors = FALSE)

#Write table of indicator summary and disparity measure
write.csv(temp2, file = "disTable.csv", row.names = FALSE, quote = TRUE)
proc.time() - ptm

### END
```