THE UNIVERSITY *of* EDINBURGH

Generating synthetic data with the synthpop package for R

Synthesising larger datasets

Gillian Raab
Administrative Data Research Centre – Scotland

Administrative Data Research Network

An ESRC Data Investment

Outline

- Problems with large data sets
- Tips and tricks to overcome them
  - Choose visit.sequence to preserve the relationships you want
  - Use stratified synthesis
  - Simplify predictors or use nesting for big categories
  - Some new untested methods

Problems with large data sets

- Number of variables and/or number of records
  - You may run out of memory
  - It may take a long time
  - Relationships between some variables may not be preserved

  (Variables with lots of categories are a problem)

What can you do about it?

- Get a more powerful machine
  - The computer scientist’s solution, but not practical when you have to work with what you have within a secure environment
- Synthesise only a sample of your big data
- Use only the variables the user really needs/wants
- Customise your synthesis
  - Choose a visit.sequence to preserve the relationships you want
  - Use stratified synthesis
  - Use different methods

Tips for customising syn()

Run syn() with m=0 so no syntheses are done

Then use the output for methods, predictor.matrix as starting points for your custom synthesis.

```
synbig0 <- syn(SD2011,m = 0, method = "ctree", cont.na = list(income = -8,
unempdur = -8, nofriend = -8, nociga = -8))
```

```
synbig0$method
```

```
 sex         age        agegr     placesize      region       edu     eduspec     socprof . . .
"sample"  "ctree"    "ctree"     "ctree"     "ctree"     "ctree"    "ctree"     "ctree“ . . .
. . . height     weight      bmi
. . . "ctree"   "ctree"    "ctree"
```

```
mymethod <- synbig0$method

mymethod[35] <- "~I(weight/height^2*10000)“

system.time(
synbig1 <- syn(SD2011, method = mymethod, cont.na = list(income = -8,
unempdur = -8, nofriend = -8, nociga = -8))
)
```

Synthesising all of SD2011

- Took 56 seconds on my machine (35 variables for 5000 cases – modest size)
- My machine would not handle it with parametric methods
- Your machine may take longer or fail
- If this happens get rid of some variables for today so you can get on
- Now to examine some results

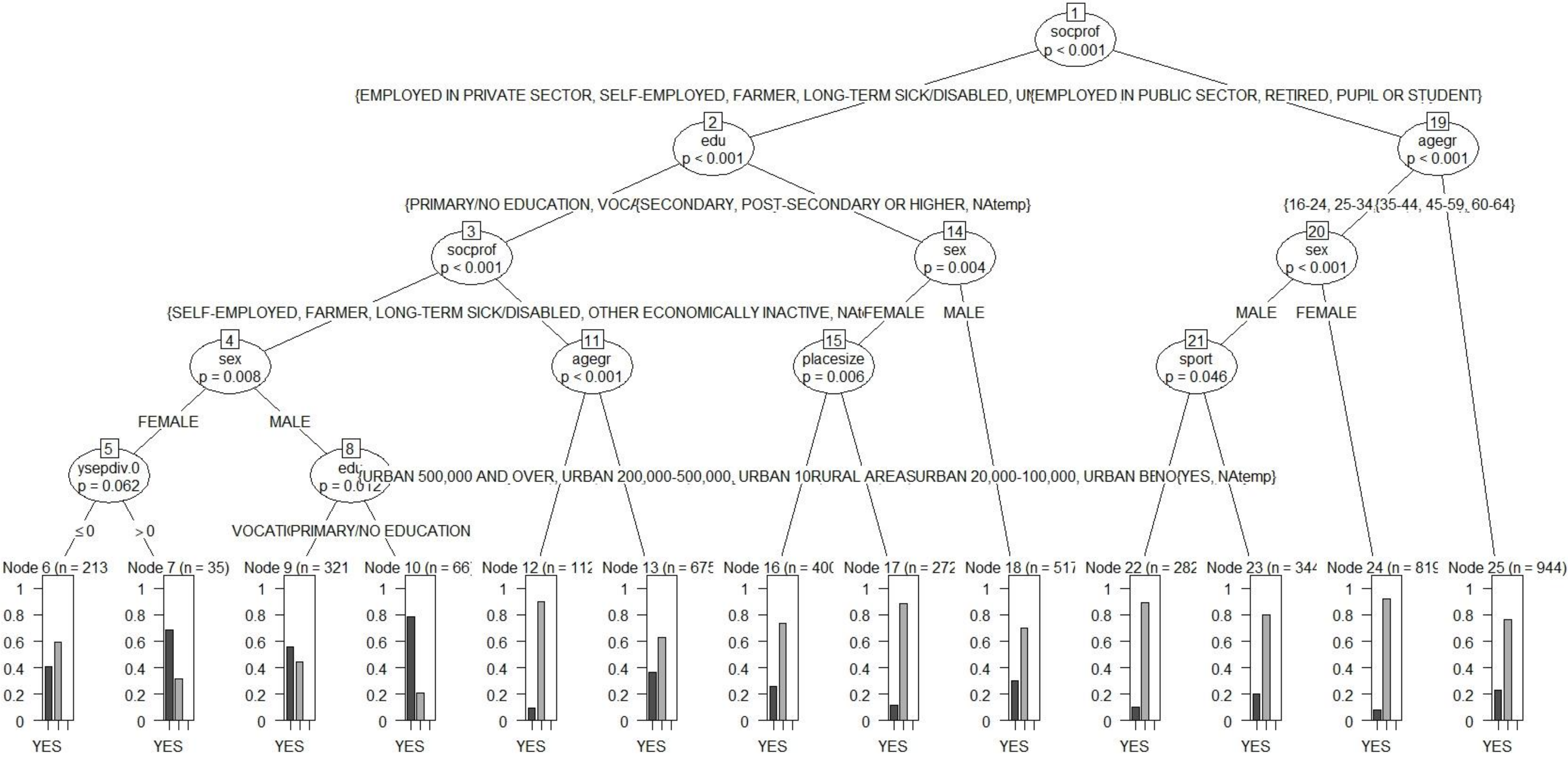
Relationships between variables

- Often we wish 2 way tables to be preserved
- If pairs of variables are together at the start of the visit sequence – usually OK
- But if further apart
  - Relationships between some variables may not be preserved
  - But they may be
- We will look at 2 examples

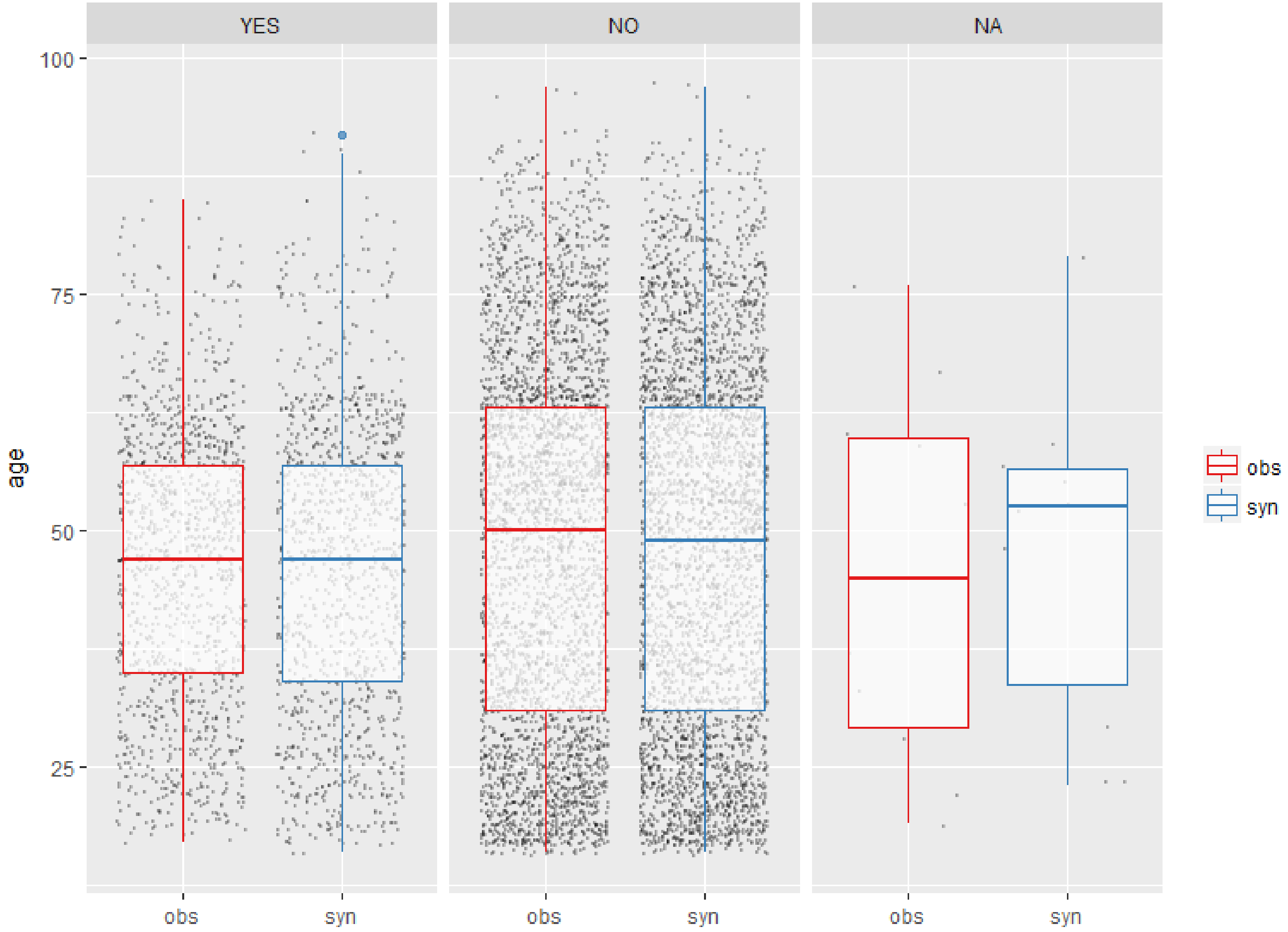
Example 1 Smoking and age

- Smoke is at position 23 in visit sequence and age at 2
- Looking at the model for age we can see that age only appears at a few nodes – much less often than socprof (social class)
- But despite this their relationship is well maintained

"ctree" model for smoke



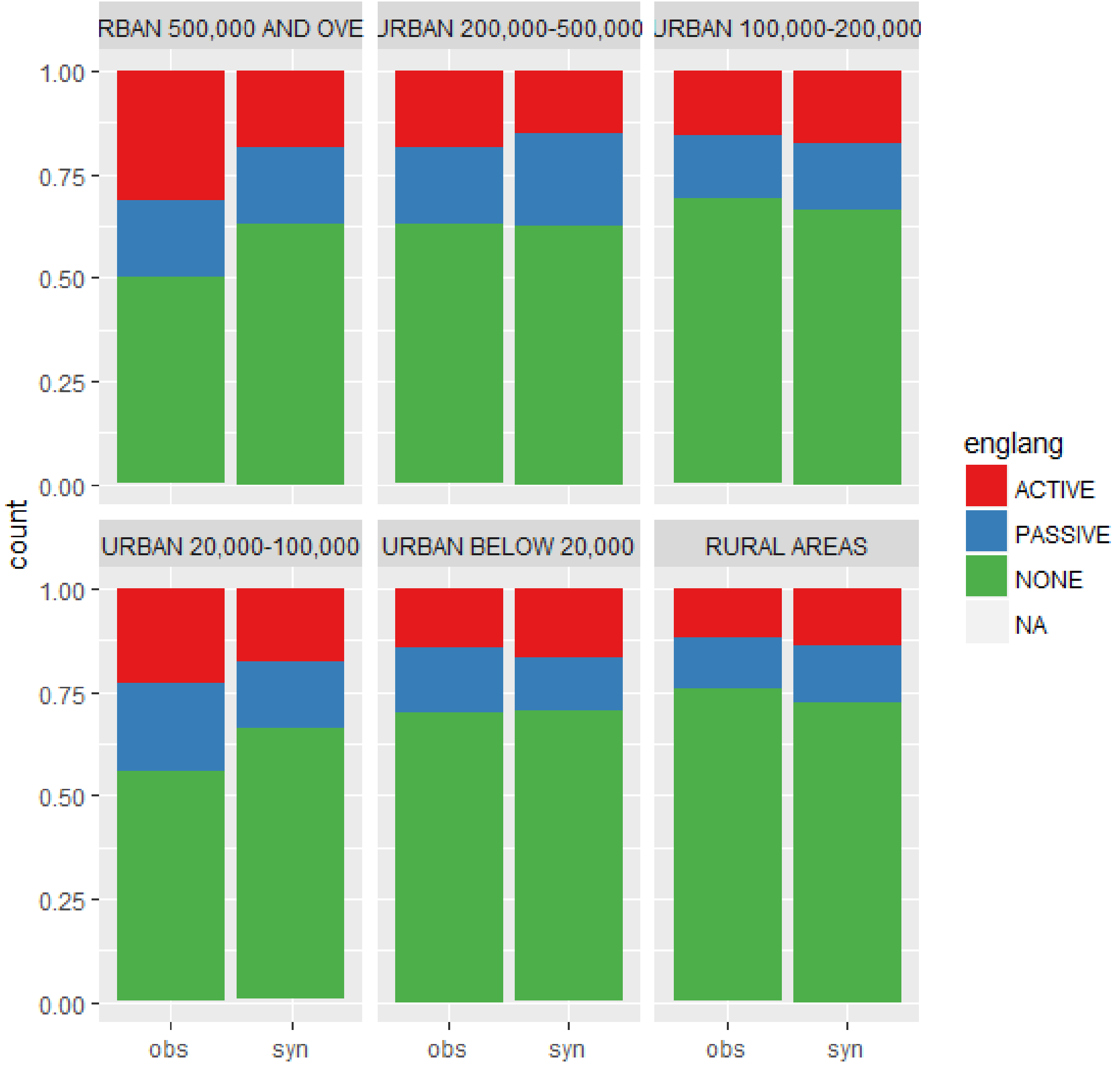
Age by smoking Tabular utility ratio 1.6



Example 2 English language and placesize

- englang is at position 32 in visit sequence and placesize at 2
- Model for englang dominated by education and educational specialty – too big a plot to show here
- Thus the relationship between englang and urban areas is underestimated

Placesize by englang utility.tab ratio 4.65



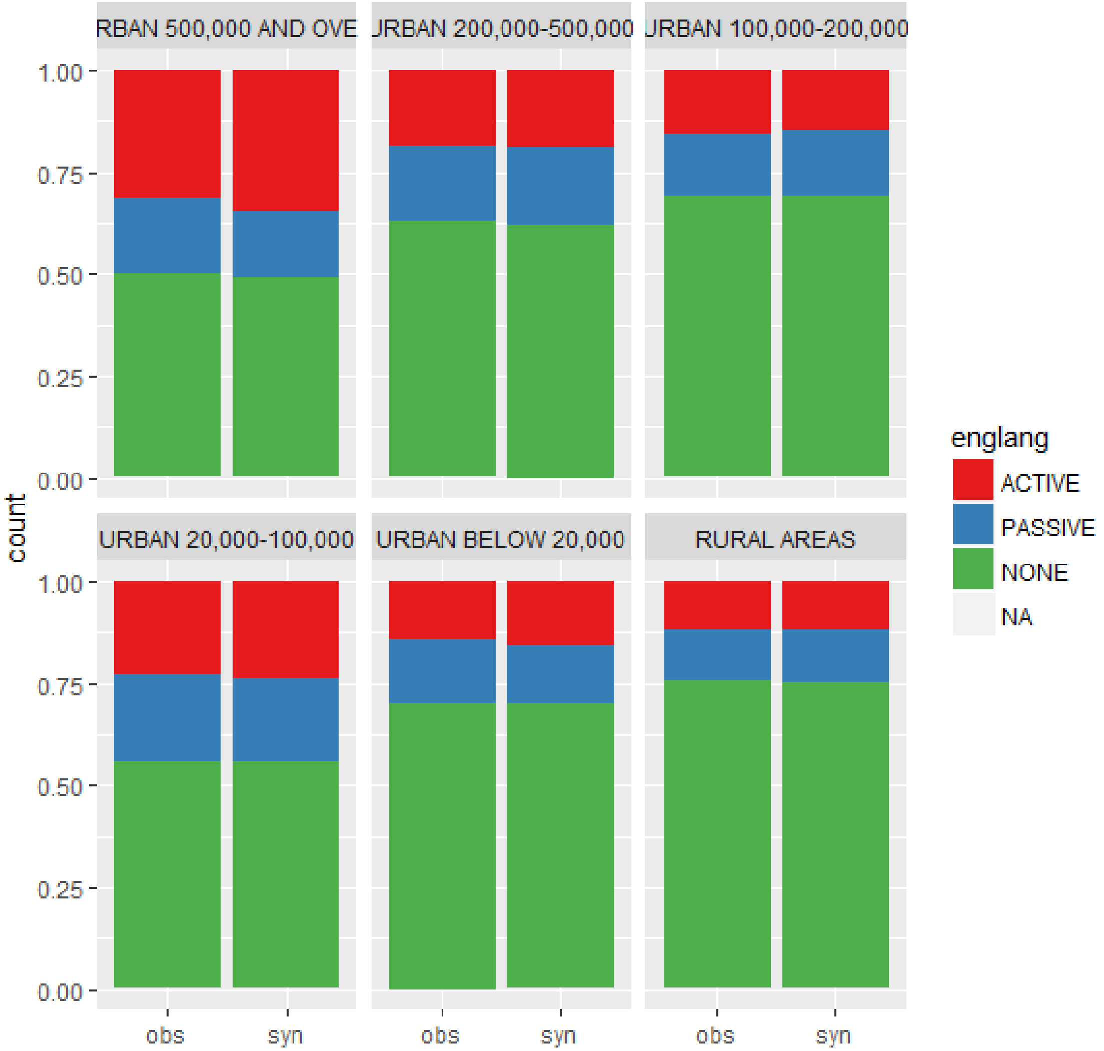
Moving englang and placesize to the start of the visit.sequence

```
newvs <- c(4,32,1:3,5:31,33:35)

system.time(
synbig2 <- syn(SD2011, visit.sequence = newvs, cont.na =
list(income = -8, unempdur = -8, nofriend = -8, nociga = -
8))
)
```

Results are better utility.tab ratio 1.04

placesize by englang



Data sets with many records and usually many variables

- Can lead to memory and computing time problems
- Stratify your synthesis
  - Choose strata of interest
  - Make sure there are no small groups
  - NA values are OK if not small
- All strata use the same methods and predictor matrix
- Stratification can improve utility too
  - Relationships between other variables and strata are maintained better

Example age-sex groups

This gives 10 strata, smallest with 228 cases

```
system.time(
synbig3 <- syn.strata(SD2011a, strata = c("sex","agegr"),
method = bigmethod, minstratumsize = 200)
)
```

```
CAUTION: There should be at least 450 observations (100 +
10 * no. of variables used in prediction).

m = 1, strata = MALE_16-24

-----------------------------------------------------

Sample(s) of size 341 will be generated from original data
of size 346

 user  system elapsed
  53.01    0.07   53.14
```

.

Variables with lots of categories

- Can lead to memory problems
- Options
  - Simplify the predictor matrix
  - If suitable use or make nested categories
- In SD2011
  - Largest number of categories is eduspec (educational specialty) 27 categories
  - Reduce what it is predicted from to 3 variables
  - And what it predicts to 4

Reducing predictor matrix

```
newpm <- synbig0$predictor.matrix ##  make new predictor matrix
newpm["eduspec",] # predicted from these
#
# change so just predicted from edu and agegr and socprof
#
newpm["eduspec", ] <- 0
newpm["eduspec",c("edu","agegr","socprof") ] <- 1

newpm[,"eduspec"] # and is a predictor for these
#
# change so just predictor for smoke englang alacbuse and workab
#
newpm[,"eduspec" ] <- 0
newpm[c("englang","alcabuse","smoke","workab"),"eduspec"] <- 1

system.time(
synbig1_eduspec <- syn(SD2011, method = bigmethod, predictor.matrix =
newpm,
               cont.na = list(income = -8, unempdur = -8, nofriend = -
8, nociga = -8), models = TRUE)
)system.time(

synbig3 <- syn.strata(SD2011a, strata = c("sex","agegr"),method =
bigmethod, minstratumsize = 200,. . )
+ )

# Cut synthesis time to less than half
```

Using nested categories

- Some categories are hierarchical
- E.g. classifications of occupations, causes of death, diagnoses
- Use the larger class to relate to other variables and the nested class only relates to the larger one
- Nested variables are synthesised as bootstrap samples
- Example from I-CeM data

Synthesising nested variables

occlab1

```
"WORKING IN AND ABOUT, AND WORKING AND DEALING IN THE PRODUCTS OF, MINES AND QUARRIES"
"BLANK"
"PERSONS ENGAGED IN AGRICULTURE"
"PERSONS WORKING AND DEALING IN DRESS"
```

Occlab3

```
"LIMESTONE QUARRIER"
"BLANK"
"WOODMAN"
"DRESSMAKERS“
```

```
method[names(method)=="occlab2"]<-"nested.occlab1"
method[names(method)=="occlab3"]<-"nested.occlab2"
```

Methods taking a group of variables

- Need to be at start of synthesis
- If any other variables, these are built up on conditional models
- Two methods
- catall – cross tabulation of all variables
- ipf – iterative proportional fitting of log-linear models
- Designed for categorical variables, but numeric variables will be grouped

Example – needs development version of synthpop

JUST USING FIRST 9 variables
"sex" "agegr" "placesize" "region" "edu"
"eduspec" "socprof" "unempdur" "income"

```
newmethod <- bigmethod
newmethod[1,3:10] <- "catall"

system.time(
synbig4 <- syn(ninevars, catall.structzero =
struct.zero,method = newmethod[-2], numtocat =
c("unempdur","income"),seed = 78976,cont.na =
list(income = -8, unempdur = -8, nofriend = -8,
nociga = -8), models = TRUE)
) #  68 million cells
```

Fitted in under 1 minute

Methods taking a group of variables

- Catall will preserve all relationships between variables
- ipf maintains pairwise relationships as default, but you can also specify which higher margins of the table you want
- But it may not do as well as CART models for complex relationships
- These methods may have advantages in having demonstrable disclosure protection