A Danish Diabetes Register in R: Register, DK population status, LABKA, RMPS and Complications

SDCC
March 2021

Version 8

Compiled Thursday 11th March, 2021, 09:51
from: V:\SDC\469DRIVE\DMreg\tex/DMreg2018-R.tex

Bendix Carstensen Senior statistician, Clinical Epidemiology
Steno Diabetes Center Copenhagen, Gentofte, Denmark
& Department of Biostatistics, University of Copenhagen
bendix.carstensen@regionh.dk b@bxc.dk
http://BendixCarstensen.com

Contents

Chapter 1

R-version of the register

```
> library(Epi)
> library(tidyverse)
> library(haven)
> source( '../../util/elapsed.R' )
```

1.1 SAS-version of DMreg converted to R

```
> start()
```

```
------------------------------------------
Code: E:/workdata/707655/DMreg/r/mkDMreg.rnw
Time: 2020-08-29 at 14:39:15
------------------------------------------
```

We have created the DMreg as a SAS-file; the entire process is available in the document http://bendixcarstensen/DMreg/DMreg2018.pdf. Also available as v:\sdc\469drive\DMreg\tex\DMreg2018.pdf; it should have a creation date earlier than that of this document, but not too much earlier.

In this section we document the conversion of the SAS-version of the register to an R-version with variables defined as factors where necessary, with date variables converted to cal.yr and a logical ordering of the variables.

We first read the SAS-version of the register and rename PNR to PNR remove unwanted attributes and convert dates to cal.yr:

```
> system.time(
+ DMreg <- as.data.frame(read_sas("e:/workdata/707655/DMreg/data/DMreg.sas7bdat")) )
   user  system elapsed
   3.64    0.07    9.52
```

We want the variable labels for convenience so we get the variable labels from the attributes extracted by read_sas:

```
> cbind( vlabs <- sapply(DMreg, FUN = function(x) attr(x, "label")) )
       [,1]
pnr    "Person-id"
sex    "Sex"
```

```
doBth  "Date of birth"
doDM   "Date of inclusion"
doLast "Date of latest criterion"
doDth  "Date of death"
DMtp   "Type of DM"
dvdtyp "Type from DVDD"
nprtyp "Type from NPR"
only1  "Only one criterion"
hasdvd "has DVDD record"
inCr   "Incl. criterion"
do2nd  "Date of 2nd of Ins/OAD/NPR"
doNPR  "Date of 1st NPR"
doNPR2 "Date of 2nd NPR"
doOAD  "Date of 1st OAD"
doOAD2 "Date of 2nd OAD"
doIns  "Date of 1st Ins"
doIns2 "Date of 2nd Ins"
doPod  "Date of Podiatry"
doDia  "Date of diaBase"
doDVD  "Date of DVDD"
```

Thus, `vlabs` is now a character vector with *values* that are the labels of the variables, and with a `names` attribute that is the variable names.

We do not want to carry variable attributes around with data frame, and we want the date variables in `cal.yr` format:

```
> for(vn in names(DMreg))
+ for(at in c("label","format.sas") ) attr(DMreg[,vn], at) <- NULL
> DMreg <- cal.yr(DMreg)
```

Further, we define factors as needed. Note that `dvdtyp` and `nprtyp` will have missing values — they are character variables and one value that occur is `""` (a zero-length character string), which, when not mentioned in the `levels` argument, will become a missing value for the factor.

```
> DMreg <- transform(DMreg,
+                    sex = factor(sex),
+                   DMtp = factor(DMtp),
+                   inCr = factor(inCr),
+                  only1 = factor(only1,  labels=c("N","Y")),
+                 hasdvd = factor(hasdvd, labels=c("N","Y")),
+                 dvdtyp = factor(dvdtyp, levels=c("NA"   ,"T1","T2"),
+                                         labels=c("undef","T1","T2")),
+                 nprtyp = factor(nprtyp, levels=c("NA"   ,"T1","T2"),
+                                         labels=c("undef","T1","T2")))
> str(DMreg, v = 0)
'data.frame':        485989 obs. of  22 variables:
 $ pnr   : chr   ...
 $ sex   : Factor w/ 2 levels "M","W": NULL ...
 $ doBth : 'cal.yr' num  NULL ...
 $ doDM  : 'cal.yr' num  NULL ...
 $ doLast: 'cal.yr' num  NULL ...
 $ doDth : 'cal.yr' num  NULL ...
 $ DMtp  : Factor w/ 2 levels "T1","T2": NULL ...
 $ dvdtyp: Factor w/ 3 levels "undef","T1","T2": NULL ...
 $ nprtyp: Factor w/ 3 levels "undef","T1","T2": NULL ...
```

```
 $ only1 : Factor w/ 2 levels "N","Y": NULL ...
 $ hasdvd: Factor w/ 2 levels "N","Y": NULL ...
 $ inCr  : Factor w/ 12 levels "Dia","DVD","I-I",..: NULL ...
 $ do2nd : 'cal.yr' num  NULL ...
 $ doNPR : 'cal.yr' num  NULL ...
 $ doNPR2: 'cal.yr' num  NULL ...
 $ doOAD : 'cal.yr' num  NULL ...
 $ doOAD2: 'cal.yr' num  NULL ...
 $ doIns : 'cal.yr' num  NULL ...
 $ doIns2: 'cal.yr' num  NULL ...
 $ doPod : 'cal.yr' num  NULL ...
 $ doDia : 'cal.yr' num  NULL ...
 $ doDVD : 'cal.yr' num  NULL ...
```

Finally, we save the register *and* the vector `vlabs` with the variable labels in the same file; it is a handy feature of `save`, that you can save several R-objects in one file, in this case `DMreg.Rda`; the `load` command will then load all objects stored in the file (`v=T` causes `load` to print the objects it loads.)

```
> system.time(
+ save(DMreg, vlabs, file="e:/workdata/707655/DMreg/data/DMreg.Rda") )
   user  system elapsed
   3.34    0.03    4.47
> system.time( load( file="e:/workdata/707655/DMreg/data/DMreg.Rda", v=T) )
Loading objects:
  DMreg
  vlabs
   user  system elapsed
   0.58    0.01    0.59
```

1.1.1 The diabetes drug register

There is also a version of the diabetes register where persons are included only on the basis of diabetes drug purchase. They are included at the date of the `second` drug purchase, but where the type of diabetes is taken from the `DMreg`. It is a subset of the `DMreg`.

```
> system.time(
+ DMdreg <- as.data.frame(read_sas("e:/workdata/707655/DMreg/data/DMdreg.sas7bdat")) )
   user  system elapsed
   1.49    0.02    3.61
```

We want the variable labels for convenience so we get the variable labels from the attributes extracted by `read_sas`:

```
> cbind( vlabs <- sapply(DMdreg, FUN = function(x) attr(x, "label")) )
        [,1]
pnr     "Person id"
sex     "Sex"
DMtp    "Type of DM"
doBth   "Date of birth"
doDM    "Date of inclusion"
doDth   "Date of death"
inCr    "Incl. criterion"
doOAD   "Date of 1st OAD"
doIns   "Date of 1st Ins"
lastOAD "Date of last OAD"
lastIns "Date of last Ins"
```

Thus, `vlabs` is now a character vector with *values* that are the labels of the variables, and with a `names` attribute that is the variable names.

We do not want to carry variable attributes around with data frame, and we want the date variables in `cal.yr` format:

```
> for(vn in names(DMdreg))
+ for(at in c("label","format.sas") ) attr(DMdreg[,vn], at) <- NULL
> DMdreg <- cal.yr(DMdreg)
```

Further, we define factors:

```
> DMdreg <- transform(DMdreg,
+                      sex = factor(sex),
+                     DMtp = factor(DMtp),
+                     inCr = factor(inCr))
> str(DMdreg, v = 0)
'data.frame':        440687 obs. of  11 variables:
 $ pnr    : chr   ...
 $ sex    : Factor w/ 2 levels "M","W": NULL ...
 $ DMtp   : Factor w/ 2 levels "T1","T2": NULL ...
 $ doBth  : 'cal.yr' num  NULL ...
 $ doDM   : 'cal.yr' num  NULL ...
 $ doDth  : 'cal.yr' num  NULL ...
 $ inCr   : Factor w/ 4 levels "I-I","I-O","O-I",..: NULL ...
 $ doOAD  : 'cal.yr' num  NULL ...
 $ doIns  : 'cal.yr' num  NULL ...
 $ lastOAD: 'cal.yr' num  NULL ...
 $ lastIns: 'cal.yr' num  NULL ...


> system.time(
+ save(DMdreg, vlabs, file="e:/workdata/707655/DMreg/data/DMdreg.Rda") )
   user  system elapsed
   1.63    0.01    2.11
> system.time(  load( file="e:/workdata/707655/DMreg/data/DMdreg.Rda", v=T) )
Loading objects:
  DMdreg
  vlabs
   user  system elapsed
   0.37    0.02    0.39


------------------------------------------
Code: E:/workdata/707655/DMreg/r/mkDMreg.rnw
Ends: 2020-08-29 at 14:39:40
Time elapsed:       00:00:25
------------------------------------------
```

1.2 Reading and using the R-version of the `DMreg`

```
----------------------------------------------
Code: E:/workdata/707655/DMreg/r/readDMreg.rnw
Time: 2020-08-29 at 14:40:37
----------------------------------------------
```

The details of *creating* the R-version of the `DMreg` is in section 1.1.

The R-code from this section is available as the file `E:\workdata\707655\DMreg\r\readDMreg.R`—you will most likely want some of this at the top of your program.

We can load the register and the variable labels—note the `v=TRUE` argument to `load` that lists the objects you are loading, and the `v=0` argument to `str` that prints the structure of `DMreg` without listing individual data values, allowing you to export the resulting document from DST (`v=0` suppresses the listing of data points; it lists 0 values of each variable):

```
> system.time(
+ load(file = "e:/workdata/707655/DMreg/data/DMreg.Rda", v = TRUE) )
Loading objects:
  DMreg
  vlabs
   user  system elapsed
   1.05    0.03    1.33
> str(DMreg, v=0)
'data.frame':        485989 obs. of  22 variables:
 $ pnr   : chr   ...
 $ sex   : Factor w/ 2 levels "M","W": NULL ...
 $ doBth : 'cal.yr' num  NULL ...
 $ doDM  : 'cal.yr' num  NULL ...
 $ doLast: 'cal.yr' num  NULL ...
 $ doDth : 'cal.yr' num  NULL ...
 $ DMtp  : Factor w/ 2 levels "T1","T2": NULL ...
 $ dvdtyp: Factor w/ 3 levels "undef","T1","T2": NULL ...
 $ nprtyp: Factor w/ 3 levels "undef","T1","T2": NULL ...
 $ only1 : Factor w/ 2 levels "N","Y": NULL ...
 $ hasdvd: Factor w/ 2 levels "N","Y": NULL ...
 $ inCr  : Factor w/ 12 levels "Dia","DVD","I-I",..: NULL ...
 $ do2nd : 'cal.yr' num  NULL ...
 $ doNPR : 'cal.yr' num  NULL ...
 $ doNPR2: 'cal.yr' num  NULL ...
 $ doOAD : 'cal.yr' num  NULL ...
 $ doOAD2: 'cal.yr' num  NULL ...
 $ doIns : 'cal.yr' num  NULL ...
 $ doIns2: 'cal.yr' num  NULL ...
 $ doPod : 'cal.yr' num  NULL ...
 $ doDia : 'cal.yr' num  NULL ...
 $ doDVD : 'cal.yr' num  NULL ...
> cbind(vlabs)
       vlabs
pnr    "Person-id"
sex    "Sex"
doBth  "Date of birth"
doDM   "Date of inclusion"
doLast "Date of latest criterion"
doDth  "Date of death"
DMtp   "Type of DM"
dvdtyp "Type from DVDD"
nprtyp "Type from NPR"
only1  "Only one criterion"
hasdvd "has DVDD record"
inCr   "Incl. criterion"
do2nd  "Date of 2nd of Ins/OAD/NPR"
```

```
doNPR  "Date of 1st NPR"
doNPR2 "Date of 2nd NPR"
doOAD  "Date of 1st OAD"
doOAD2 "Date of 2nd OAD"
doIns  "Date of 1st Ins"
doIns2 "Date of 2nd Ins"
doPod  "Date of Podiatry"
doDia  "Date of diaBase"
doDVD  "Date of DVDD"
```

The character vector `vlabs` holds the long labels of the variables; its *names* attribute is the a vector variable names in the `DMreg`. Note also that for practical use, you may not need more than the first 6 or 7 variables, so for parsimony of your code as well as decency in behaviour toward other using the DST servers you could do:

```
> DMreg <- DMreg[,1:6]
```

1.2.1 Things to note when using the DMreg:

- Do not put anything in the folder `E:\workdata\707655\DMreg` or any of its sub-folders.
- `pnr` is of class `character`. It must remain so, numerical values are inaccurate on a computer.
- Keep the factors in the `DMreg` that are defined.
- There is a point of *not* having the `pnr` as a factor, it saves no space to have a factor with as many levels as records in the data frame, whereas it is a good idea to have the variables with few levels as factors. Moreover, if you make a subsets of a data frame where `pnr` is a factor, the subset will carry along the entire set of 400,000+ levels.
- Do not rename the variables from the `DMreg`, that would be a prescription of confusion.

1.3 Tabular overview of incidence and prevalence in the DMreg

We can get an overview of the number of cases in the register, by date of inclusion, sex and type of diabetes.

```
> with( DMreg, ftable(addmargins(table(floor(pmax(doDM,1995)),
+                                      sex,
+                                      DMtp,
+                                      exclude=NULL) ),
+                     row.vars=1) )
     sex       M                    W                  Sum
     DMtp     T1     T2    Sum     T1     T2    Sum     T1     T2    Sum

1995       12384  30259  42643   9613  31046  40659  21997  61305  83302
1996         681   6166   6847    519   5285   5804   1200  11451  12651
1997         688   5861   6549    490   4892   5382   1178  10753  11931
1998         664   6535   7199    460   5256   5716   1124  11791  12915
1999         592   6727   7319    402   5669   6071    994  12396  13390
```

```
2000          598   6588   7186    390   5534   5924    988  12122  13110
2001          589   6804   7393    414   5407   5821   1003  12211  13214
2002          607   8056   8663    394   7265   7659   1001  15321  16322
2003          547   9147   9694    387   7607   7994    934  16754  17688
2004          508   9292   9800    397   7733   8130    905  17025  17930
2005          518   8161   8679    378   6438   6816    896  14599  15495
2006          555   8161   8716    384   5889   6273    939  14050  14989
2007          562   8680   9242    387   6695   7082    949  15375  16324
2008          558   9865  10423    373   7573   7946    931  17438  18369
2009          572  10730  11302    366   7686   8052    938  18416  19354
2010          526  11847  12373    371   8664   9035    897  20511  21408
2011          511  15479  15990    357  13068  13425    868  28547  29415
2012          496  12792  13288    322  10093  10415    818  22885  23703
2013          488  10189  10677    363   8053   8416    851  18242  19093
2014          485   9877  10362    361   7407   7768    846  17284  18130
2015          496  10054  10550    390   7760   8150    886  17814  18700
2016          517  10695  11212    381   7956   8337    898  18651  19549
2017          508  10430  10938    364   8105   8469    872  18535  19407
2018          495  10739  11234    326   8040   8366    821  18779  19600
Sum         25145 243134 268279  18589 199121 217710  43734 442255 485989
```

The prevalent cases as of 2019-1-1 by age, sex and type of diabetes can be also be derived on the fly:

```
> with( subset(DMreg, doDM < 2019 & (doDth > 2019 | is.na(doDth))),
+       ftable(addmargins(table(cut(2019 - doBth,
+                                   breaks = seq(0, 120, 5),
+                                   right = FALSE),
+                               sex,
+                               DMtp,
+                               exclude = NULL)),
+              row.vars=1) )
```

This table is not printed because because it has small numbers in it. But there is a remedy for that.

1.3.1 Readable tables that can be sent from DST

Readable tables with large numbers require position commas for readability; we want to write 485,989 instead of 485989.

Moreover, thin tables will benefit from having 0s printed as "."—or some other character requiring minimal ink.

Finally, if you want to send a table home from the DST server you must omit counts smaller than 4, for example by replacing them by "*".

The two first facilities are available in the functions `fCp`, and `fCtable`, whereas all three are available in the functions `rCtable` and `rCp`. The code for these functions is available in the file `elapsed.R`, which is read by:

```
> source( "e:/workdata/707655/util/elapsed.R" )
```

We can then print the table of the prevalent cases of DM as of 2019-1-1 by age, sex and diabetes type:

```
> with( subset( DMreg, doDM < 2019 & (doDth > 2019 | is.na(doDth)) ),
+       rCtable(addmargins(table(cut(2019-doBth,
+                                    breaks = seq(0,120,5),
+                                    right = FALSE),
+                                sex,
+                                DMtp,
+                                exclude = NULL)),
+               row.vars = 1, w = 7) )
```

sex	M			W			Sum		
DMtp	T1	T2	Sum	T1	T2	Sum	T1	T2	Sum
[0,5)	48	*	49	34	5	39	82	6	88
[5,10)	261	*	264	260	*	263	521	6	527
[10,15)	614	9	623	557	20	577	1,171	29	1,200
[15,20)	901	40	941	766	123	889	1,667	163	1,830
[20,25)	1,057	180	1,237	901	342	1,243	1,958	522	2,480
[25,30)	1,195	461	1,656	920	598	1,518	2,115	1,059	3,174
[30,35)	1,137	980	2,117	788	914	1,702	1,925	1,894	3,819
[35,40)	1,139	1,939	3,078	798	1,489	2,287	1,937	3,428	5,365
[40,45)	1,408	3,726	5,134	1,011	3,600	4,611	2,419	7,326	9,745
[45,50)	1,593	6,870	8,463	1,120	5,717	6,837	2,713	12,587	15,300
[50,55)	1,758	12,124	13,882	1,243	9,113	10,356	3,001	21,237	24,238
[55,60)	1,513	16,000	17,513	1,000	11,471	12,471	2,513	27,471	29,984
[60,65)	1,308	19,539	20,847	954	13,503	14,457	2,262	33,042	35,304
[65,70)	1,035	22,179	23,214	790	15,559	16,349	1,825	37,738	39,563
[70,75)	906	27,138	28,044	729	19,244	19,973	1,635	46,382	48,017
[75,80)	525	19,197	19,722	395	15,973	16,368	920	35,170	36,090
[80,85)	262	11,738	12,000	253	11,712	11,965	515	23,450	23,965
[85,90)	76	5,228	5,304	120	7,031	7,151	196	12,259	12,455
[90,95)	18	1,631	1,649	23	3,063	3,086	41	4,694	4,735
[95,100)	*	226	227	9	718	727	10	944	954
[100,105)	.	11	11	.	72	72	.	83	83
[105,110)	.	*	*	.	9	9	.	10	10
[110,115)	.	*	*	.	.	.	.	*	*
[115,120)	.	.	.	.	.	.	.	.	.
Sum	16,755	149,223	165,978	12,671	120,279	132,950	29,426	269,502	298,928

The last argument to `rCtable`, `w=7` determines the width of the columns in the resulting table.

You will note that the 0s have been replaced by a "." and numbers 1, 2 and 3 by a "*".

The functions `rCtable` and `fCtable`

have their funny names because they use `ftable` to layout the table (so arguments `row.vars` and `col.vars` from `ftable` apply), and use `formatC` to print the numbers with position markers. That explains the names `fCtable` and `fCp`; the names `rCtable` and `rCp` are versions that restrict entries to be at least 4.

The argument `w` (default 11) gives the width of the table entries, `d` (default 0) gives the number of digits after the decimal point, `z` (default ".") gives the character to print instead of 0, and the argument `klim` (only for `rCp` and `rCtable`, default 4) gives the smallest admissible number printed.

The function `rCtable` is of course only relevant for tables of counts.

1.3.2 Reading and using the diabetes drug register

For illustration we also load the register based on drugs purchases alone:

```
> system.time(
+ load(file = "e:/workdata/707655/DMreg/data/DMdreg.Rda", v = TRUE) )
Loading objects:
  DMdreg
  vlabs
   user  system elapsed
   0.39    0.00    0.43
> str(DMdreg, v=0)
'data.frame':        440687 obs. of  11 variables:
 $ pnr    : chr   ...
 $ sex    : Factor w/ 2 levels "M","W": NULL ...
 $ DMtp   : Factor w/ 2 levels "T1","T2": NULL ...
 $ doBth  : 'cal.yr' num  NULL ...
 $ doDM   : 'cal.yr' num  NULL ...
 $ doDth  : 'cal.yr' num  NULL ...
 $ inCr   : Factor w/ 4 levels "I-I","I-O","O-I",..: NULL ...
 $ doOAD  : 'cal.yr' num  NULL ...
 $ doIns  : 'cal.yr' num  NULL ...
 $ lastOAD: 'cal.yr' num  NULL ...
 $ lastIns: 'cal.yr' num  NULL ...
> cbind(vlabs)
        vlabs
pnr     "Person id"
sex     "Sex"
DMtp    "Type of DM"
doBth   "Date of birth"
doDM    "Date of inclusion"
doDth   "Date of death"
inCr    "Incl. criterion"
doOAD   "Date of 1st OAD"
doIns   "Date of 1st Ins"
lastOAD "Date of last OAD"
lastIns "Date of last Ins"
```

We can then print the table of the prevalent cases of drug-treated DM as of 2019-1-1 by age, sex and diabetes type:

```
> with( subset( DMdreg, doDM < 2019 & (doDth > 2019 | is.na(doDth)) ),
+       rCtable(addmargins(table(cut(2019-doBth,
+                                    breaks = seq(0,120,5),
+                                     right = FALSE),
+                                sex,
+                                DMtp,
+                                exclude = NULL)),
+               row.vars = 1, w = 7) )
          sex        M                       W                     Sum
          DMtp      T1      T2     Sum      T1      T2     Sum      T1      T2     Sum

[0,5)                46       .      46      31       .      31      77       .      77
[5,10)              253       .     253     255       .     255     508       .     508
[10,15)             604       4     608     554       9     563   1,158      13   1,171
[15,20)             898      26     924     761     105     866   1,659     131   1,790
```

[20,25)	1,048	142	1,190	888	265	1,153	1,936	407	2,343
[25,30)	1,180	375	1,555	913	424	1,337	2,093	799	2,892
[30,35)	1,128	880	2,008	778	640	1,418	1,906	1,520	3,426
[35,40)	1,133	1,816	2,949	795	1,014	1,809	1,928	2,830	4,758
[40,45)	1,401	3,534	4,935	1,004	3,251	4,255	2,405	6,785	9,190
[45,50)	1,588	6,519	8,107	1,114	5,322	6,436	2,702	11,841	14,543
[50,55)	1,754	11,544	13,298	1,242	8,494	9,736	2,996	20,038	23,034
[55,60)	1,513	15,213	16,726	1,000	10,651	11,651	2,513	25,864	28,377
[60,65)	1,303	18,446	19,749	953	12,323	13,276	2,256	30,769	33,025
[65,70)	1,034	20,689	21,723	790	13,982	14,772	1,824	34,671	36,495
[70,75)	904	24,987	25,891	728	16,946	17,674	1,632	41,933	43,565
[75,80)	523	17,469	17,992	395	13,943	14,338	918	31,412	32,330
[80,85)	262	10,462	10,724	253	10,084	10,337	515	20,546	21,061
[85,90)	76	4,559	4,635	120	5,840	5,960	196	10,399	10,595
[90,95)	18	1,404	1,422	23	2,482	2,505	41	3,886	3,927
[95,100)	*	186	187	9	549	558	10	735	745
[100,105)	.	8	8	.	56	56	.	64	64
[105,110)	.	*	*	.	8	8	.	9	9
[110,115)	.	.	.	.	.	.	.	.	.
[115,120)	.	.	.	.	.	.	.	.	.
Sum	16,667	138,264	154,931	12,606	106,388	118,994	29,273	244,652	273,925

Note that the resulting factor from the `cut` function has levels all the way to 120, even if no persons are over 110.

```
---------------------------------------------
Code: E:/workdata/707655/DMreg/r/readDMreg.rnw
Ends: 2020-08-29 at 14:40:41
Time elapsed:       00:00:04
---------------------------------------------
```

Chapter 2

Auxiliary data files in R-format

The following sections describe data files that were created using SAS-code which is available the relevant chapters in the main document, `V:\SDC\469DRIVE\DMreg\tex\DMreg2018.pdf` `00-base` (population files), `00y-base` (status file), `00-labka` (LABKA files), `00-rmps` (prescription files), `10-labcomp` and `10-compl` (complications files).

2.1 The population file

First the paraphernalia:

```
----------------------------------------
Code: E:/workdata/707655/DMreg/r/mkPop.rnw
Time: 2020-09-02 at 14:57:24
----------------------------------------
```

The file we produce contains the `pnr`, sex, dates of birth and death from the population files and the reclassified causes of death and country of birth from the cause of death files. We retrieve data from the SAS-files `POP` (population) and `COD` (cause of death):

```
> system.time(pop <- as.data.frame(read_sas("../data/pop.sas7bdat")))
   user  system elapsed
  36.19    0.53   69.50
> attr( pop$pnr, "label"      ) <- NULL
> attr( pop$pnr, "format.sas" ) <- NULL
> str(pop, v = 0)
'data.frame':        7631979 obs. of  6 variables:
 $ pnr  : chr   ...
 $ sex  : chr   ...
  ..- attr(*, "label")= chr  ...
 $ doBth: Date, format:  ...
 $ doDth: Date, format:  ...
 $ whBth: chr   ...
  ..- attr(*, "label")= chr  ...
 $ dSrc : chr   ...
  ..- attr(*, "label")= chr  ...
 - attr(*, "label")= chr  ...
> system.time(cod <- as.data.frame(read_sas("../data/cod.sas7bdat")))
   user  system elapsed
   7.09    0.03   15.43
```

```
> attr( cod$pnr, "label"      ) <- NULL
> attr( cod$pnr, "format.sas" ) <- NULL
> str(cod, v = 0)
'data.frame':        1211314 obs. of  10 variables:
 $ pnr  : chr   ...
 $ doDth: Date, format:  ...
 $ cod4 : chr   ...
  ..- attr(*, "label")= chr  ...
 $ codX : chr   ...
  ..- attr(*, "label")= chr  ...
 $ codD : chr   ...
  ..- attr(*, "label")= chr  ...
 $ daar : chr   ...
  ..- attr(*, "label")= chr  ...
 $ daa1 : chr   ...
  ..- attr(*, "label")= chr  ...
  ..- attr(*, "format.sas")= chr  ...
 $ daa2 : chr   ...
  ..- attr(*, "label")= chr  ...
  ..- attr(*, "format.sas")= chr  ...
 $ daa3 : chr   ...
  ..- attr(*, "label")= chr  ...
  ..- attr(*, "format.sas")= chr  ...
 $ daa4 : chr   ...
  ..- attr(*, "label")= chr  ...
  ..- attr(*, "format.sas")= chr  ...
 - attr(*, "label")= chr  ...
```

There is not the same number of deaths in the two files; and even the two dates of death do not always match:

```
> table(!is.na(pop$doDth))
  FALSE    TRUE
6298570 1333409
> table(!is.na(cod$doDth))
   TRUE
1211314
> jj <- left_join( pop[,c("pnr","doDth","doBth","sex")],
+                  cod[,c("pnr","doDth","cod4","codX")],
+                  by = "pnr")
> jj <- cal.yr(jj)
> dim(jj) ; cbind(sapply(jj, function(x) class(x)[1]))
[1] 7631979       7
        [,1]
pnr     "character"
doDth.x "cal.yr"
doBth   "cal.yr"
sex     "character"
doDth.y "cal.yr"
cod4    "character"
codX    "character"
> tt <- table(cod = floor(jj$doDth.y),
+             pop = floor(jj$doDth.x),
+             exclude=NULL )
> tt <- tt / ifelse(tt>10^5, 1000, 1)
> rCtable( tt[,  1:10 ], w=6 )
```

```
     pop   1995   1996   1997   1998   1999   2000   2001   2002   2003   2004
cod
1995         283      .      .      .      .      .      .      .      .      .
1996           . 60,375      .      .      .      .      .      .      .      .
1997           .      . 59,531      .      .      .      .      .      .      .
1998           .      .      . 57,965      .      .      .      .      .      .
1999           .      .      .      . 58,452      .      .      .      .      .
2000           .      .      .      .      . 56,661      .      .      .      .
2001           .      .      .      .      .      . 57,326      .      .      .
2002           .      .      .      .      .      .      . 58,250      .      .
2003           .      .      .      .      .      .      .      * 57,069      *
2004           .      .      .      .      .      .      .      .      . 55,098
2005           .      .      .      .      .      .      .      .      .      .
2006           .      .      .      .      .      .      .      .      .      .
2007           .      .      .      .      .      .      .      .      .      .
2008           .      .      .      .      .      .      .      .      .      .
2009           .      .      .      .      .      .      .      .      .      .
2010           .      .      .      .      .      .      .      .      .      .
2011           .      .      .      .      .      .      .      .      .      .
2012           .      .      .      .      .      .      .      .      .      .
2013           .      .      .      .      .      .      .      .      .      .
2014           .      .      .      .      .      .      .      .      .      .
2015           .      .      .      .      .      .      .      .      .      .
2016           .      .      .      .      .      .      .      .      .      .
2017           .      .      .      .      .      .      .      .      .      .
2018           .      .      .      .      .      .      .      .      .      .
NA             .     23     49    443    515    853    812    418    469    475
```

```
> rCtable( tt[, 11:20 ], w=6 )
```

```
     pop   2005   2006   2007   2008   2009   2010   2011   2012   2013   2014
cod
1995           .      .      .      .      .      .      .      .      .      .
1996           .      .      .      .      .      .      .      .      .      .
1997           .      .      .      .      .      .      .      .      .      .
1998           .      .      .      .      .      .      .      .      .      .
1999           .      .      .      .      .      .      .      .      .      .
2000           .      *      7      8      6      .      4      .      *      *
2001           .      .      *      *      *     61     98      8      *      7
2002           .      .      *      *      *      *      .      *      .      .
2003           .      .      *      *      *      .      .      .      8      .
2004           .      .      5      *      *      *      *      .      .     12
2005      54,387      8      *      7      *      .      .      .      .      .
2006           . 55,083     11      5     11      *      .      .      .      .
2007           .      * 55,021     46      6      .      .      .      .      .
2008           .      .      * 53,742     62      *      .      *      *      .
2009           .      .      *      * 54,227     27      .      .      *      .
2010           .      .      .      *      * 53,860    100     52      8      8
2011           .      .      .      .      .      5 51,790     61      4      7
2012           .      .      .      .      .      .     11 51,502    231     21
2013           .      .      .      .      .      .      *     10 51,703     59
2014           .      .      .      .      .      .      *      .     13 50,790
2015           .      .      .      .      .      *      .      .      *      *
2016           .      .      .      .      .      .      .      .      .      *
2017           .      .      .      .      .      .      .      .      .      .
2018           .      .      .      .      .      .      .      .      .      .
NA           517    521    573    621    640    649    641    702    709    784
```

```
> rCtable( tt[,-(1:20)], w=9 )
```

```
     pop      2015      2016      2017      2018      2019        NA
cod
1995            .         .         .         .         .         .
1996            .         .         .         .         .         .
1997            .         .         .         .         .         .
1998            .         .         .         .         .         .
1999            .         .         .         .         .         .
2000            .         .         .         .         .         .
2001            6         *         7         .         .         .
2002            .         .         .         .         .         .
2003            .         .         .         .         .         .
2004            .         .         .         .         .         .
2005           14         .         .         .         .         .
2006            .        14         .         .         .         .
2007            .         .        23         .         .         .
2008            *         .         .         .         .         .
2009            .         .         .         .         .         .
2010            6        12         6         .         .         .
2011            5         *         5         .         .         .
2012           10        10         *         .         .         .
2013            4        15         .         .         .         .
2014           60         4        13         .         .         .
2015       51,843        65         *         .         .         .
2016           20    52,009        32         .         .         .
2017            *         8    52,605         .         .         .
2018            .         .         *         .         .         .
NA            741       802       829    55,729    53,909     6,299
```

We see that the cause of death file (`doDth.y`) only have dates of death till 2017 incl. and that there is a tendency that discrepancies are concentrated around dates of death from `cod` being 10 years earlier the date of death in the `pop` file. So we conclude that the date of death obtained from the `pop` file is the correct one; essentially we ascribe discrepancies to misrecordings of dates of death on death certificates. Also this is more complete cover the years 2018 and 2019 too.

```
> pop <- data.frame(pnr = jj$pnr,
+                   sex = factor(jj$sex),
+                 doBth = jj$doBth,
+                 doDth = jj$doDth.x,
+                  cod4 = factor(jj$cod4),
+                  codX = factor(jj$codX),
+      stringsAsFactors = FALSE)
> with( pop, rCtable( addmargins(table(has.cod = !is.na(codX),
+                                      has.doDth = !is.na(doDth) )), w=9 ) )
        has.doDth     FALSE      TRUE       Sum
has.cod
FALSE             6,298,570   122,424 6,420,994
TRUE                      . 1,210,985 1,210,985
Sum               6,298,570 1,333,409 7,631,979
> with( pop, rCtable( addmargins(table(codX, cod4)), w=9 ) )
       cod4       Can       CVD       Oth       Res       Sum
codX
Cancer        338,209         .         .         .   338,209
CVD                 .   382,590         .         .   382,590
Diab                .         .       777         .       777
Digest              .         .    57,347         .    57,347
```

```
Extern              .         .    56,702         .    56,702
Infect              .         .    17,628         .    17,628
Other               .         .   209,946         .   209,946
Renal               .         .     8,785         .     8,785
Respir              .         .         .   128,362   128,362
Urinal              .         .    10,639         .    10,639
Sum           338,209   382,590   361,824   128,362 1,210,985
```

Note the there are very few deaths from diabetes; this is because this cause of death has been been taken as the secondary or tertiary cause if the first or two first recorded causes were diabetes. This Is specific for this project because we are primarily interested in comparing mortality between diabetes patients and other persons, and so diabetes as a cause of death in itself is not relevant, we would want to see the underlying cause(s) instead.

```
> str(pop, v=0)
'data.frame':        7631979 obs. of  6 variables:
 $ pnr  : chr   ...
 $ sex  : Factor w/ 2 levels "M","W": NULL ...
 $ doBth: 'cal.yr' num  NULL ...
 $ doDth: 'cal.yr' num  NULL ...
 $ cod4 : Factor w/ 4 levels "Can","CVD","Oth",..: NULL ...
 $ codX : Factor w/ 10 levels "Cancer","CVD",..: NULL ...
> save(pop,        file = "../data/pop.Rda")
> system.time(load(file = "../data/pop.Rda"))
   user  system elapsed
   5.73    0.10    5.84


----------------------------------------
Code: E:/workdata/707655/DMreg/r/mkPop.rnw
Ends: 2020-09-02 at 14:59:46
Time elapsed:       00:02:21
----------------------------------------
```

2.2 The status file

```
----------------------------------------
Code: E:/workdata/707655/DMreg/r/mkStat.rnw
Time: 2020-08-29 at 16:27:48
----------------------------------------
```

The status file is classified by `pnr` and `yr`, each record representing a person's status at 1 January of the year `yr`. There are records for *all* residents in Denmark. The status variables are place of residence, family income and highest achieved education.

2.2.1 Converting the SAS dataset to `.Rda` format

We now read the SAS dataset and convert it to an R-dataset for easier (and quicker) access:

```
> system.time(popstat <- read_sas("../data/popstat.sas7bdat"))
```

```
   user  system elapsed
 791.34    4.84 1758.00
> names(popstat) <- tolower( names(popstat) )
> for(v in names(popstat))
+    {
+    attr(popstat[,v], "label") <- NULL
+    attr(popstat[,v], "format.sas") <- NULL
+    }
> str(popstat, v = 0)
Classes 'tbl_df', 'tbl' and 'data.frame':        131784868 obs. of  8 variables:
 $ pnr  : chr   ...
  ..- attr(*, "label")= chr  ...
  ..- attr(*, "format.sas")= chr  ...
 $ kom  : chr   ...
  ..- attr(*, "label")= chr  ...
  ..- attr(*, "format.sas")= chr  ...
 $ reg  : chr   ...
  ..- attr(*, "label")= chr  ...
  ..- attr(*, "format.sas")= chr  ...
 $ yr   : num  NULL ...
  ..- attr(*, "label")= chr  ...
 $ find : num  NULL ...
  ..- attr(*, "label")= chr  ...
 $ udd  : num  NULL ...
  ..- attr(*, "label")= chr  ...
 $ udddk: num  NULL ...
  ..- attr(*, "label")= chr  ...
  ..- attr(*, "format.sas")= chr  ...
 $ eduen: num  NULL ...
  ..- attr(*, "label")= chr  ...
  ..- attr(*, "format.sas")= chr  ...
 - attr(*, "label")= chr  ...
```

We then read the file with the character formats for geography and education (generated in the SAS program `00y-base.sas`) in order to get the correct factor levels for `kom`, `reg`, `udddk` and `eduen`:

```
> labs <-  read_sas("../data/statlabels.sas7bdat")
> labs$START <- as.numeric(labs$START)
> table( labs$FMTNAME )
AUDD_HOVED_L5L5_T AUDD_LEVEL_L4L4_T          KOM_V4_T          REG_V4_T
               15                 9                99                 6
> ( kl <- labs[grep("KOM"  ,labs$FMTNAME),c("LABEL","START")] )
# A tibble: 99 x 2
   LABEL         START
   <chr>         <dbl>
 1 København       101
 2 Frederiksberg   147
 3 Ballerup        151
 4 Brøndby         153
 5 Dragør          155
 6 Gentofte        157
 7 Gladsaxe        159
 8 Glostrup        161
 9 Herlev          163
10 Albertslund     165
# ... with 89 more rows
```

```
> ( rl <- labs[grep("REG"  ,labs$FMTNAME),c("LABEL","START")][-1,] )
```

```
# A tibble: 5 x 2
  LABEL       START
  <chr>       <dbl>
1 Nordjylland    81
2 Midtjylland    82
3 Syddanmark     83
4 Hovedstaden    84
5 Sjælland       85
```

```
> ( ul <- labs[grep("HOVED",labs$FMTNAME),c("LABEL","START")] )
```

```
# A tibble: 15 x 2
   LABEL                                    START
   <chr>                                    <dbl>
 1 Førskoleuddannelser                          5
 2 Grundskole                                  10
 3 Forberedende uddannelser                    15
 4 Gymnasiale uddannelser                      20
 5 Danskundervisning ved sprogcentre           25
 6 Erhvervsfaglige grundforløb                 29
 7 Erhvervsfaglige uddannelser                 30
 8 Adgangsgivende uddannelsesforløb            35
 9 Arbejdsmarkedsuddannelser, AMU              39
10 Korte videregående uddannelser, KVU         40
11 Mellemlange videregående uddannelser, MVU   50
12 Bacheloruddannelser, BACH                   60
13 Lange videregående uddannelser, LVU         70
14 Ph.d. og forskeruddannelser                 80
15 Uoplyst mv.                                 90
```

```
> ( el <- labs[grep("LEVEL",labs$FMTNAME),c("LABEL","START")] )
```

```
# A tibble: 9 x 2
  LABEL                     START
  <chr>                     <dbl>
1 Early childhood education     0
2 Primary                       1
3 Lower secondary               2
4 Upper secondary               3
5 Short cycle tertiary          5
6 Bachelor or equivalent        6
7 Master or equivalent          7
8 Doctoral or equivalent        8
9 Not elsewhere classified      9
```

These are the in turn used to define the relevant variables as factors:

```
> system.time(
+ popstat <-
+ mutate( popstat,
+         kom = factor(  kom, levels = kl$START, labels = kl$LABEL),
+         reg = factor(  reg, levels = rl$START, labels = rl$LABEL),
+       udddk = factor(udddk, levels = ul$START, labels = ul$LABEL),
+       eduen = factor(eduen, levels = el$START, labels = el$LABEL) ) )
```

```
   user  system elapsed
 234.00    1.58  235.58
```

2.2.2 Creating income deciles

Income levels change over a period as long as the the one covered by these data (1996–2018, 22 years), so we construct a factor of deciles of income for each year, `findec`, family `income decile`. We finally `ungroup` the tibble before we save it:

```
> system.time(
+ popstat %>%
+   group_by(yr) %>%
+   mutate( findec = cut( find,
+                         quantile( find,
+                                   0:10/10,
+                                   na.rm=TRUE ),
+                         labels = paste(1:10) ) ) %>%
+   ungroup() -> popstat )
   user  system elapsed
  38.99    4.58   43.56
> attr( popstat$findec, "label" ) <- "fam.indk. decil pr. yr"
```

2.2.3 Saving the file for future use

Finally, we save the data as an R-file and load it again to demonstrate the time it likely takes to load it. First we rearrange the order of the variables to make it more logical.

```
> popstat <- popstat[,c(1,4,2,3,5,9,6:8)]
> str( popstat, v=0 )
Classes 'tbl_df', 'tbl' and 'data.frame':        131784868 obs. of  9 variables:
 $ pnr   : chr   ...
  ..- attr(*, "label")= chr  ...
  ..- attr(*, "format.sas")= chr  ...
 $ yr    : num  NULL ...
  ..- attr(*, "label")= chr  ...
 $ kom   : Factor w/ 99 levels "København","Frederiksberg",..: NULL ...
 $ reg   : Factor w/ 5 levels "Nordjylland",..: NULL ...
 $ find  : num  NULL ...
  ..- attr(*, "label")= chr  ...
 $ findec: Factor w/ 10 levels "1","2","3","4",..: NULL ...
  ..- attr(*, "label")= chr  ...
 $ udd   : num  NULL ...
  ..- attr(*, "label")= chr  ...
 $ udddk : Factor w/ 15 levels "Førskoleuddannelser",..: NULL ...
 $ eduen : Factor w/ 9 levels "Early childhood education",..: NULL ...
> system.time( save(popstat, file="../data/popstat.Rda") )
   user  system elapsed
 496.22    3.95  578.44
> rm( popstat )
> system.time( load(         file="../data/popstat.Rda") )
   user  system elapsed
 106.09    1.36  167.44
```

So the R-version of the `popstat` dataset loads about 10 times faster than the SAS-version, and it is properly equipped with factors for residence (`kom`, `reg`), income decile (`findec`) and educational level (`udddk`, `eduen`).

```
----------------------------------------
Code: E:/workdata/707655/DMreg/r/mkStat.rnw
Ends: 2020-08-29 at 17:14:24
Time elapsed:       00:46:37
----------------------------------------
```

2.3 The LABKA database

```
> library( Epi )
> library( tidyverse )
> library( haven )
> source("E:/workdata/707655/util/elapsed.r")
> setwd("E:/workdata/707655/DMreg/r")
> start()
--------------------------------
Home: E:/workdata/707655/DMreg/r
Time: 2020-06-22 15:26:00
--------------------------------
```

LABKA measurements are in a very large file, 346 mil. records, 146 Gb, so we have read the file and created 26 sas-files with separate measurements in the folder `E:\workdata\707655\DMreg\data\labka`.

2.3.1 SAS splitting of the LABKA data.

2.3.2 Converting to `.Rda`

The SAS program `00-labka` contains the names and the labels of the files, so we read the SAS-code and extract the file names and the labels for use in the R-files:

```
> ll <- read.table( "../sas/00-labka.sas", sep="/" )[,1]
> ll <- read.table( "../sas/00-labka.log", sep="/" )[,1]
> ll <- grep( "label", ll, value=TRUE )
> dot <- sapply( strsplit(ll,""), function(x) which(x==".") )
> eql <- sapply( strsplit(ll,""), function(x) which(x=="=") )
> rbr <- sapply( strsplit(ll,""), function(x) which(x==")") )
> nam <- gsub(" ", "", substr( ll, dot+1, dot+4 ) )
> lab <-               substr( ll, eql+2, rbr-2 )
> nam <- tolower( nam )
> names( lab ) <- nam
> cbind( lab )
     lab
hba1 "Hba1c"
gluc "Glukose"
glu0 "Glukose 0"
gl30 "Glukose 30"
g120 "Glukose 120"
tchl "Total kolesterol"
ldl  "LDL kolesterol"
hdl  "HDL kolesterol"
vldl "VLDL kolesterol"
trig "Triglycerid"
```

```
plcr "Plasma Kreatinin"
uacr "Ualbcrea"
pota "Kalium"
sodi "Natrium"
tsh  "TSH"
cpep "C-peptid/Proinsulin"
crp  "CRP"
gad  "GAD65"
egfr "eGFR"
gfr  "GFR"
alat "ALAT"
alcp "Basisk fosfatase"
cobl "Cobalamin"
trmb "Trombocytter"
leuc "Leucocytter"
hmgb "Hæmoglobin"
```

We now have the filenames (without extension) — note all filenames are in lower case; they are in the `names` attribute of the `lab` vector of labels of the various types of measurements.

Then we read the SAS-files, coerce them to `data.frames`, strip the disturbing attributes of the variabels, assigns the proper label to the `label` attribute of the data frame. It is then assigned to a object with the proper name and subsequently saved in an R-file with the correct name.

```
> for( fn in names(lab) )
+    {
+ cat( fn, " start at", format( Sys.time(), "%T" ) )
+ xx <- read_sas( paste0("../data/labka/", fn, ".sas7bdat") )
+ xx <- as.data.frame( xx )
+ for( i in names(xx) ) attr( xx[,i], "format.sas" ) <- NULL
+ attr( xx$SAMPLINGTIME, "units" ) <- NULL
+ attr( xx, "label" ) <- lab[fn]
+ assign( fn, xx )
+ system.time(
+ save( list = fn,
+       file = paste0("e:/workdata/707655/DMreg/data/labka/", fn, ".Rda" ) ) )
+ cat( " end at", format( Sys.time(), "%T" ),
+         "dim=", paste( fCp(dim(xx)), collapse=" by" ), "\n" )
+ rm( list = fn )
+    }
hba1  start at 15:26:01 end at 15:37:31 dim= 21,261,038 by        7
gluc  start at 15:37:31 end at 15:42:18 dim= 8,736,053 by        7
glu0  start at 15:42:18 end at 15:42:51 dim=   874,845 by        7
gl30  start at 15:42:51 end at 15:42:52 dim=    11,395 by        7
g120  start at 15:42:52 end at 15:42:54 dim=    61,892 by        7
tchl  start at 15:42:54 end at 15:49:05 dim= 10,463,522 by        7
ldl  start at 15:49:05 end at 15:54:52 dim= 9,875,421 by        7
hdl  start at 15:54:52 end at 16:01:12 dim= 10,083,655 by        7
vldl  start at 16:01:12 end at 16:02:06 dim= 1,492,139 by        7
trig  start at 16:02:06 end at 16:08:08 dim= 10,356,568 by        7
plcr  start at 16:08:08 end at 16:25:24 dim= 31,617,208 by        7
uacr  start at 16:25:24 end at 16:26:37 dim= 2,085,164 by        7
pota  start at 16:26:37 end at 16:42:26 dim= 30,207,229 by        7
sodi  start at 16:42:26 end at 16:57:45 dim= 30,186,282 by        7
tsh  start at 16:57:45 end at 17:02:50 dim= 11,495,628 by        7
cpep  start at 17:02:50 end at 17:02:56 dim=   164,936 by        7
```

```
crp  start at 17:02:56 end at 17:11:33 dim= 20,723,651 by        7
gad  start at 17:11:33 end at 17:11:35 dim=    28,416 by        7
egfr  start at 17:11:35 end at 17:23:50 dim= 28,742,105 by        7
gfr  start at 17:23:50 end at 17:23:52 dim=     2,409 by        7
alat  start at 17:23:52 end at 17:32:22 dim= 20,540,099 by        7
alcp  start at 17:32:22 end at 17:38:52 dim= 15,495,551 by        7
cobl  start at 17:38:52 end at 17:41:04 dim= 5,324,860 by        7
trmb  start at 17:41:04 end at 17:49:42 dim= 21,039,994 by        7
leuc  start at 17:49:42 end at 18:00:17 dim= 25,630,130 by        7
hmgb  start at 18:00:17 end at 18:12:44 dim= 30,419,252 by        7
```

Thus, for example if you need the cobalamin measurements you just do:

```
> system.time(
+ load( "e:/workdata/707655/DMreg/data/labka/cobl.Rda", v=T ) )
Loading objects:
  cobl
   user  system elapsed
   8.53    0.06    8.77
> str( cobl, v=0 )
'data.frame':        5324860 obs. of  7 variables:
 $ pnr                : chr   ...
 $ SAMPLINGDATE       : Date, format:  ...
 $ SAMPLINGTIME       : 'hms' num   ...
 $ ANALYSISCODE       : chr   ...
 $ LABORATORIUM_IDCODE: chr   ...
 $ VALUE              : chr   ...
 $ UNIT               : chr   ...
 - attr(*, "label")= Named chr  ...
  ..- attr(*, "names")= chr  ...
> attr( cobl, "label" )
       cobl
"Cobalamin"
> fCp( object.size( cobl ) )
[1] 416,553,960
```

The last use of `attr` is necessary because `v=0` also cuts the the first (and only) element of the `label` attribute, so if you want a human readable label this is what to do.

```
----------------------
2020-06-22 at 18:12:56
Time elapsed: 02:46:55
----------------------
```

2.4 The RMPS database

```
> library( Epi )
> library( tidyverse )
> library( haven )
> source("E:/workdata/707655/util/elapsed.r")
> setwd("E:/workdata/707655/DMreg/r")
> start()
```

```
----------------------------------------
Code: E:/workdata/707655/DMreg/r/mkRMPS.rnw
Time: 2020-11-30 at 17:20:56
----------------------------------------
```

The RMPS (prescription regiuster) is very large, each year some 30 mil. records, so it is a tall order to read this by R, so we read it by SAS and created a number of SAS-files, each with a subgroup of drugs, see `00-rmps`. These are then read in turn by R and saved in R-format.

2.4.1 SAS splitting of the ATC data.

```
1                                  "Program: 00-rmps.sas"  19:08 Thursday, December 3, 2020

NOTE: Copyright (c) 2016 by SAS Institute Inc., Cary, NC, USA.
NOTE: SAS (r) Proprietary Software 9.4 (TS1M5)
      Licensed to FORSKNING 1, Site 50800722.
NOTE: This session is executing on the X64_SR12R2  platform.



NOTE: Updated analytical products:

      SAS/STAT 14.3

NOTE: Additional host information:

 X64_SR12R2 WIN 6.3.9600  Server

NOTE: SAS initialization used:
      real time           0.10 seconds
      cpu time            0.12 seconds


NOTE: AUTOEXEC processing beginning; file is E:\workdata\707655\DMreg\sas\optslibs.sas.


NOTE: AUTOEXEC processing completed.

1          %let fr = 1995 ;
2          %let to = 2019 ;
3
4          %macro getmed ;
5          *-------------------------------------------------------------------------;
6          data lipid renal card blpr plate oad ins ;
7            set %do i = &fr. %to &to. ;
8                grund.lmdb&i.         ( keep = pnr atc eksd doso apk packsize strnum
8        ! strunit )
9                grund.lmdb&i._brutto  ( keep = pnr atc eksd doso apk packsize strnum
9        ! strunit )
10               %end ; ;
11           if substr(atc, 1, 4) in ("A10A" ) then output ins ;
12           if substr(atc, 1, 4) in ("A10B" ) then output oad ;
13           if substr(atc, 1, 5) in ("B01AC") then output plate ;
14           if substr(atc, 1, 4) in ("C01D" ) then output card ;
15           if substr(atc, 1, 3) in ("C02","C03","C07","C08")
16                                                then output blpr ;
17           if substr(atc, 1, 3) in ("C09"  ) then output renal ;
18           if substr(atc, 1, 3) in ("C10"  ) then output lipid ;
19         run ;
20         %mend ;
```

```
21          %getmed ;

NOTE: There were 13552545 observations read from the data set GRUND.LMDB1995.
NOTE: There were 3568979 observations read from the data set GRUND.LMDB1995_BRUTTO.
NOTE: There were 13987953 observations read from the data set GRUND.LMDB1996.
NOTE: There were 3784012 observations read from the data set GRUND.LMDB1996_BRUTTO.
NOTE: There were 14470331 observations read from the data set GRUND.LMDB1997.
NOTE: There were 3973525 observations read from the data set GRUND.LMDB1997_BRUTTO.
NOTE: There were 15235400 observations read from the data set GRUND.LMDB1998.
NOTE: There were 4248450 observations read from the data set GRUND.LMDB1998_BRUTTO.
NOTE: There were 15540101 observations read from the data set GRUND.LMDB1999.
NOTE: There were 4483510 observations read from the data set GRUND.LMDB1999_BRUTTO.
NOTE: There were 15733948 observations read from the data set GRUND.LMDB2000.
NOTE: There were 4653099 observations read from the data set GRUND.LMDB2000_BRUTTO.
NOTE: There were 16595791 observations read from the data set GRUND.LMDB2001.
NOTE: There were 5022086 observations read from the data set GRUND.LMDB2001_BRUTTO.
NOTE: There were 17666883 observations read from the data set GRUND.LMDB2002.
NOTE: There were 5459492 observations read from the data set GRUND.LMDB2002_BRUTTO.
NOTE: There were 18878804 observations read from the data set GRUND.LMDB2003.
NOTE: There were 6000805 observations read from the data set GRUND.LMDB2003_BRUTTO.
NOTE: There were 20449486 observations read from the data set GRUND.LMDB2004.
NOTE: There were 6588662 observations read from the data set GRUND.LMDB2004_BRUTTO.
NOTE: There were 21663811 observations read from the data set GRUND.LMDB2005.
NOTE: There were 7075022 observations read from the data set GRUND.LMDB2005_BRUTTO.
NOTE: There were 23033327 observations read from the data set GRUND.LMDB2006.
NOTE: There were 7567168 observations read from the data set GRUND.LMDB2006_BRUTTO.
NOTE: There were 24324181 observations read from the data set GRUND.LMDB2007.
NOTE: There were 8030396 observations read from the data set GRUND.LMDB2007_BRUTTO.
NOTE: There were 25484004 observations read from the data set GRUND.LMDB2008.
NOTE: There were 8533368 observations read from the data set GRUND.LMDB2008_BRUTTO.
NOTE: There were 26040637 observations read from the data set GRUND.LMDB2009.
NOTE: There were 8758122 observations read from the data set GRUND.LMDB2009_BRUTTO.
NOTE: There were 26874842 observations read from the data set GRUND.LMDB2010.
NOTE: There were 9053925 observations read from the data set GRUND.LMDB2010_BRUTTO.
NOTE: There were 27476210 observations read from the data set GRUND.LMDB2011.
NOTE: There were 9309185 observations read from the data set GRUND.LMDB2011_BRUTTO.
NOTE: There were 27720576 observations read from the data set GRUND.LMDB2012.
NOTE: There were 9463003 observations read from the data set GRUND.LMDB2012_BRUTTO.
NOTE: There were 27670851 observations read from the data set GRUND.LMDB2013.
NOTE: There were 9401555 observations read from the data set GRUND.LMDB2013_BRUTTO.
NOTE: There were 27612777 observations read from the data set GRUND.LMDB2014.
NOTE: There were 9292871 observations read from the data set GRUND.LMDB2014_BRUTTO.
NOTE: There were 27468960 observations read from the data set GRUND.LMDB2015.
NOTE: There were 9137469 observations read from the data set GRUND.LMDB2015_BRUTTO.
NOTE: There were 27365352 observations read from the data set GRUND.LMDB2016.
NOTE: There were 9011666 observations read from the data set GRUND.LMDB2016_BRUTTO.
NOTE: There were 27161833 observations read from the data set GRUND.LMDB2017.
NOTE: There were 8836650 observations read from the data set GRUND.LMDB2017_BRUTTO.
NOTE: There were 26976587 observations read from the data set GRUND.LMDB2018.
NOTE: There were 8643481 observations read from the data set GRUND.LMDB2018_BRUTTO.
NOTE: There were 27259310 observations read from the data set GRUND.LMDB2019.
NOTE: There were 8553674 observations read from the data set GRUND.LMDB2019_BRUTTO.
NOTE: The data set WORK.LIPID has 49609142 observations and 8 variables.
NOTE: The data set WORK.RENAL has 68956839 observations and 8 variables.
NOTE: The data set WORK.CARD has 10103914 observations and 8 variables.
NOTE: The data set WORK.BLPR has 143262020 observations and 8 variables.
NOTE: The data set WORK.PLATE has 51752610 observations and 8 variables.
NOTE: The data set WORK.OAD has 50384170 observations and 8 variables.
NOTE: The data set WORK.INS has 25235372 observations and 8 variables.
NOTE: DATA statement used (Total process time):
      real time           17:20.17
      cpu time            4:22.06
```

```
22
23         %macro sortmed( dsn, lab ) ;
24         proc sort data = &dsn.
25              out = drdat.&dsn. ( label = "&lab." )
26              nodupkey ;
27           by pnr atc eksd doso apk packsize ;
28         run ;
29         proc contents  data = drdat.&dsn.  varnum ; run ;
30         proc tabulate  data = drdat.&dsn.  missing  noseps ;
31           class atc ;
32           table all atc, n * f=comma10. / rts = 60  box = "&lab."  ;
33           format atc $ATC_L1L1_KT. ;
34         run ;
35         %mend ;
36
37         %sortmed( ins  , %str(Insulines) ) ;

NOTE: There were 25235372 observations read from the data set WORK.INS.
NOTE: 12815356 observations with duplicate key values were deleted.
NOTE: The data set DRDAT.INS has 12420016 observations and 8 variables.
NOTE: PROCEDURE SORT used (Total process time):
      real time           7.82 seconds
      cpu time            14.46 seconds



NOTE: PROCEDURE CONTENTS used (Total process time):
      real time           0.04 seconds
      cpu time            0.03 seconds

NOTE: The PROCEDURE CONTENTS printed page 1.


NOTE: There were 12420016 observations read from the data set DRDAT.INS.
NOTE: The PROCEDURE TABULATE printed page 2.
NOTE: PROCEDURE TABULATE used (Total process time):
      real time           2.95 seconds
      cpu time            1.40 seconds


38         %sortmed( oad  , %str(Oral antidiabetic drugs) ) ;

NOTE: There were 50384170 observations read from the data set WORK.OAD.
NOTE: 25548530 observations with duplicate key values were deleted.
NOTE: The data set DRDAT.OAD has 24835640 observations and 8 variables.
NOTE: PROCEDURE SORT used (Total process time):
      real time           19.99 seconds
      cpu time            31.82 seconds



NOTE: PROCEDURE CONTENTS used (Total process time):
      real time           0.01 seconds
      cpu time            0.00 seconds

NOTE: The PROCEDURE CONTENTS printed page 3.


NOTE: There were 24835640 observations read from the data set DRDAT.OAD.
NOTE: The PROCEDURE TABULATE printed page 4.
```

```
NOTE: PROCEDURE TABULATE used (Total process time):
      real time           5.44 seconds
      cpu time            2.95 seconds


39         %sortmed( plate, %str(Platelets) ) ;

NOTE: There were 51752610 observations read from the data set WORK.PLATE.
NOTE: 11714298 observations with duplicate key values were deleted.
NOTE: The data set DRDAT.PLATE has 40038312 observations and 8 variables.
NOTE: PROCEDURE SORT used (Total process time):
      real time           22.73 seconds
      cpu time            34.79 seconds



NOTE: PROCEDURE CONTENTS used (Total process time):
      real time           0.01 seconds
      cpu time            0.00 seconds

NOTE: The PROCEDURE CONTENTS printed page 5.


NOTE: There were 40038312 observations read from the data set DRDAT.PLATE.
NOTE: The PROCEDURE TABULATE printed page 6.
NOTE: PROCEDURE TABULATE used (Total process time):
      real time           8.80 seconds
      cpu time            4.17 seconds


40         %sortmed( card , %str(Cardiac stimulus) ) ;

NOTE: There were 10103914 observations read from the data set WORK.CARD.
NOTE: 2352541 observations with duplicate key values were deleted.
NOTE: The data set DRDAT.CARD has 7751373 observations and 8 variables.
NOTE: PROCEDURE SORT used (Total process time):
      real time           4.33 seconds
      cpu time            5.89 seconds



NOTE: PROCEDURE CONTENTS used (Total process time):
      real time           0.01 seconds
      cpu time            0.00 seconds

NOTE: The PROCEDURE CONTENTS printed page 7.


NOTE: There were 7751373 observations read from the data set DRDAT.CARD.
NOTE: The PROCEDURE TABULATE printed page 8.
NOTE: PROCEDURE TABULATE used (Total process time):
      real time           1.65 seconds
      cpu time            0.53 seconds


41         %sortmed( blpr , %str(Blood pressure lowering) ) ;

NOTE: There were 143262020 observations read from the data set WORK.BLPR.
NOTE: 31975359 observations with duplicate key values were deleted.
NOTE: The data set DRDAT.BLPR has 111286661 observations and 8 variables.
NOTE: PROCEDURE SORT used (Total process time):
      real time           1:09.89
```

```
      cpu time            1:40.84


NOTE: PROCEDURE CONTENTS used (Total process time):
      real time           0.01 seconds
      cpu time            0.00 seconds

NOTE: The PROCEDURE CONTENTS printed page 9.


NOTE: There were 111286661 observations read from the data set DRDAT.BLPR.
NOTE: The PROCEDURE TABULATE printed page 10.
NOTE: PROCEDURE TABULATE used (Total process time):
      real time           22.43 seconds
      cpu time            10.76 seconds


42         %sortmed( renal, %str(Renal related drugs) ) ;

NOTE: There were 68956839 observations read from the data set WORK.RENAL.
NOTE: 17116848 observations with duplicate key values were deleted.
NOTE: The data set DRDAT.RENAL has 51839991 observations and 8 variables.
NOTE: PROCEDURE SORT used (Total process time):
      real time           25.61 seconds
      cpu time            40.01 seconds



NOTE: PROCEDURE CONTENTS used (Total process time):
      real time           0.01 seconds
      cpu time            0.00 seconds

NOTE: The PROCEDURE CONTENTS printed page 11.


NOTE: There were 51839991 observations read from the data set DRDAT.RENAL.
NOTE: The PROCEDURE TABULATE printed page 12.
NOTE: PROCEDURE TABULATE used (Total process time):
      real time           8.79 seconds
      cpu time            5.17 seconds


43         %sortmed( lipid, %str(Lipid lowering drugs) ) ;

NOTE: There were 49609142 observations read from the data set WORK.LIPID.
NOTE: 13384998 observations with duplicate key values were deleted.
NOTE: The data set DRDAT.LIPID has 36224144 observations and 8 variables.
NOTE: PROCEDURE SORT used (Total process time):
      real time           17.16 seconds
      cpu time            27.72 seconds



NOTE: PROCEDURE CONTENTS used (Total process time):
      real time           0.01 seconds
      cpu time            0.01 seconds

NOTE: The PROCEDURE CONTENTS printed page 13.


NOTE: There were 36224144 observations read from the data set DRDAT.LIPID.
```

```
NOTE: The PROCEDURE TABULATE printed page 14.
NOTE: PROCEDURE TABULATE used (Total process time):
      real time           6.09 seconds
      cpu time            3.20 seconds


44
45         * A data frame with all ATC codes ;
46         proc format  library = dsfmt.sundhed
47                      cntlout = drdat.atcnam ( keep = fmtname start label type ) ;
48           select $ATC_L1L1_KT ;
49         run ;

NOTE: PROCEDURE FORMAT used (Total process time):
      real time           0.03 seconds
      cpu time            0.00 seconds

NOTE: The data set DRDAT.ATCNAM has 6605 observations and 4 variables.

NOTE: SAS Institute Inc., SAS Campus Drive, Cary, NC USA 27513-2414
NOTE: The SAS System used:
      real time           21:04.27
      cpu time            9:06.04
```

2.4.2 The ATC codes with names

We extract the ATC codes and corrsponding names from the DST format file and saved it as a character vector with a names attribute equal to the ATC codes:

```
> atcnam <- read_sas( paste0("../data/rmps/atcnam.sas7bdat") )
> names(atcnam) <- tolower(names(atcnam))
> atclab <- atcnam$label
> names(atclab) <- atcnam$start
> cbind(atclab[1:10])
        [,1]
A       "A  Fordøjelsesorganer og stofskifte"
A01     "A01  midler mod sygdomme i mundhulen"
A01A    "A01A  stomatologica"
A01AA   "A01AA  midler mod caries"
A01AA01 "A01AA01  natriumfluorid"
A01AA02 "A01AA02  natriummonofluorophosphat"
A01AA03 "A01AA03  olaflur"
A01AA04 "A01AA04  stannofluorid"
A01AA30 "A01AA30  kombinationer"
A01AA51 "A01AA51  natriumfluorid, komb."
```

This means that we can subset the `atclab` by ATC-codes to mimic the format machinery of SAS.

2.4.3 Converting to `.Rda`

The SAS program `00-labka` contains the names and the labels of the files, so we read the SAS-code and extract the file names and the labels for use in the R-files:

```
>  whf <- shell("dir ..\\data\\rmps", intern=TRUE)
>  whf <- gsub(".sas7bdat", "",
```

```
+              fgrep("s7b",
+                    unlist(strsplit(fgrep('s7b', whf), " ")))
> (whf <- whf[whf != "atcnam"])
[1] "lipid" "renal" "blpr"  "plate" "oad"   "ins"   "card"
```

We now have the filenames (without extension) — note all filenames are in lower case.

Then we read the SAS-files, coerce them to `data.frames`, strip the disturbing attributes of the variabels, assigns the proper label to the `label` attribute of the data frame. It is then assigned to a object with the proper name and subsequently saved in an R-file with the same name. For convenience the `atcnam` is saved in the same file too.

```
> for( fn in whf )
+    {
+ cat( "\n----------------------------------------\n",
+      fn, " start at", format( Sys.time(), "%T" ) )
+ system.time(
+ xx <- read_sas( paste0("../data/rmps/", fn, ".sas7bdat") ) )
+ names(xx) <- tolower(names(xx))
+ xx <- cal.yr(as.data.frame(xx))
+ for( i in names(xx) ) attr( xx[,i], "format.sas" ) <- NULL
+ for( i in names(xx) ) attr( xx[,i], "label" ) <- NULL
+ xx <- mutate(xx, doso = factor(doso),
+                  atc = factor(atc))
+ attr(xx, "label") <- fn
+ assign( fn, xx )
+ system.time(
+ save( list = c(fn,'atclab'),
+       file = paste0("e:/workdata/707655/DMreg/data/rmps/", fn, ".Rda" ) ) )
+ cat(", ended at", format(Sys.time(), "%T"), '---',
+     fn, "has", fCp(nrow(xx)), "rows.\n" )
+ tt <- table(xx$atc)
+ names(tt) <- atclab[names(tt)]
+ print(fCp(cbind(tt), w=15))
+ rm( list = fn )
+    }
----------------------------------------
 lipid  start at 17:20:57, ended at 17:27:06 --- lipid has 36,224,144 rows.
                                      tt
C10AB01  clofibrat                             2,509
C10AB02  bezafibrat                           66,208
C10AB04  gemfibrozil                         440,350
C10AC01  colestyramin                        242,024
C10AC02  colestipol                           33,399
C10AC04  colesevelam                          16,502
C10AD06  acipimox                             58,308
C10AD52  nicotinsyre, kombinationer           11,406
C10AX09  ezetimib                            709,052
C10AX13  evolocumab                              397
C10AX14  alirocumab                              257
C10BA02  simvastatin og ezetimib              36,399
C10BA05  atorvastatin og ezetimibe            13,410
C10AA01  simvastatin                      23,812,032
C10AA02  lovastatin                          267,986
C10AA03  pravastatin                         801,622
C10AA04  fluvastatin                         214,862
C10AA05  atorvastatin                      7,497,815
```

```
C10AA06  cerivastatin                        28,686
C10AA07  rosuvastatin                     1,970,920

----------------------------------------
 renal  start at 17:27:07, ended at 17:36:28 --- renal has 51,839,991 rows.
                                                  tt
C09BA01  captopril og diuretica                          21,547
C09BA02  enalapril og diuretica                       3,389,813
C09BA03  lisinopril og diuretica                        641,404
C09BA04  perindopril og diuretica                       216,476
C09BA05  ramipril og diuretica                          992,664
C09BA07  benazepril og diuretica                          1,463
C09BA15  zofenopril og diuretica                              *
C09BB02  enalapril og lercanidipin                          144
C09BB04  perindopril og amlodipin                         1,070
C09CA01  losartan                                     9,951,770
C09CA02  eprosartan                                     106,520
C09CA03  valsartan                                      530,140
C09CA04  irbesartan                                     630,108
C09CA06  candesartan                                  1,903,466
C09CA07  telmisartan                                    351,176
C09CA08  olmesartanmedoxomil                             64,397
C09DA01  losartan og diuretica                        5,840,604
C09DA02  eprosartan og diuretica                         38,101
C09DA03  valsartan og diuretica                         520,123
C09DA04  irbesartan og diuretica                        366,289
C09DA06  candesartan og diuretica                       565,938
C09DA07  telmisartan og diuretica                       201,219
C09DA08  olmesartanmedoxomil og diuretica                29,621
C09DB01  valsartan og amlodipin                          92,273
C09DX01  valsartan, amlodipin og hydrochlorthiazid        7,045
C09DX04  valsartan og sacubitril                         24,206
C09XA02  aliskiren                                       78,870
C09XA52  aliskiren og hydrochlorthiazid                   3,651
C09AA01  captopril                                    1,061,868
C09AA02  enalapril                                   11,872,578
C09AA03  lisinopril                                   1,301,750
C09AA04  perindopril                                  1,549,005
C09AA05  ramipril                                     7,144,279
C09AA06  quinapril                                      148,082
C09AA07  benazepril                                      41,489
C09AA09  fosinopril                                      67,771
C09AA10  trandolapril                                 2,078,574
C09AA13  moexipril                                        4,486
C09AA15  zofenopril                                           9

----------------------------------------
 blpr  start at 17:36:30, ended at 17:56:25 --- blpr has 111,286,661 rows.
                                                      tt
C02AB01  methyldopa (L-form)                              78,791
C02AC01  clonidin                                         15,387
C02AC02  guanfacin                                         8,490
C02AC05  moxonidin                                       530,709
C02CA01  prazosin                                        171,794
C02CA04  doxazosin                                       887,113
C02CA06  urapidil                                            230
C02DB02  hydralazin                                       21,267
```

C02DG01	pinacidil	13,463
C02KX01	bosentan	9
C03AB01	bendroflumethiazid og kalium	19,549,044
C03AB02	hydroflumethiazid og kalium	76,922
C03BA02	quinethazon	5
C03BA03	clopamid	78,024
C03BA04	chlortalidon	15,957
C03BA05	mefrusid	15,114
C03BA11	indapamid	440,994
C03BB03	clopamid og kalium	20,787
C03CA01	furosemid	18,919,186
C03CA02	bumetanid	533,607
C03CB02	bumetanid og kalium	163,788
C03DA01	spironolacton	4,036,377
C03DA02	kaliumcanrenoat	23
C03DA04	eplerenon	94,715
C03DB01	amilorid	75,587
C03EA01	hydrochlorthiazid og kaliumbesparende midler	2,640,092
C03EB01	furosemid og kaliumbesparende midler	163,392
C03EB02	bumetanid og kaliumbesparende midler	24,075
C03XA01	tolvaptan	300
C03AA01	bendroflumethiazid	389,460
C03AA02	hydroflumethiazid	2,728
C03AA03	hydrochlorthiazid	34,872
C03AA04	chlorthiazid	705
C03AA05	polythiazid	653
C07AB02	metoprolol	19,778,720
C07AB03	atenolol	2,794,121
C07AB04	acebutolol	38,068
C07AB05	betaxolol	34,958
C07AB07	bisoprolol	1,905,921
C07AB12	nebivolol	157,923
C07AG01	labetalol	170,741
C07AG02	carvedilol	2,964,750
C07BA06	timolol og thiazider	4,281
C07BB02	metoprolol og thiazider	189,079
C07BB12	nebivolol og thiazider	293
C07CA03	pindolol og andre diuretica	12,051
C07CB03	atenolol og andre diuretica	278,270
C07FB02	metoprolol og felodipin	84,897
C07AA01	alprenolol	10,363
C07AA02	oxprenolol	9,282
C07AA03	pindolol	267,063
C07AA05	propranolol	2,942,756
C07AA06	timolol	41,138
C07AA07	sotalol	984,445
C07AA16	tertatolol	261
C08CA01	amlodipin	18,366,665
C08CA02	felodipin	2,739,986
C08CA03	isradipin	231,827
C08CA04	nicardipin	3,199
C08CA05	nifedipin	1,047,128
C08CA06	nimodipin	1,600
C08CA08	nitrendipin	79,464
C08CA09	lacidipin	135,478
C08CA10	nilvadipin	4,617
C08CA13	lercanidipin	1,431,894

```
C08CX01  mibefradil                                          2,806
C08DA01  verapamil                                       2,980,463
C08DA51  verapamil, kombinationer                           20,668
C08DB01  diltiazem                                       2,567,825

----------------------------------------
 plate  start at 17:56:27, ended at 18:02:22 --- plate has 40,038,312 rows.
                           tt
B01AC04  clopidogrel              4,590,749
B01AC06  acetylsalicylsyre       29,123,745
B01AC07  dipyridamol              4,902,586
B01AC11  iloprost                        10
B01AC13  abciximab                        1
B01AC22  prasugrel                   40,135
B01AC24  ticagrelor                 244,015
B01AC30  kombinationer            1,137,071

----------------------------------------
 oad  start at 18:02:23, ended at 18:05:08 --- oad has 24,835,640 rows.
                                      tt
A10BA02  metformin                         14,364,484
A10BB01  glibenclamid                       1,297,916
A10BB03  tolbutamid                           429,862
A10BB04  glibornurid                              117
A10BB07  glipizid                             676,259
A10BB09  gliclazid                            600,567
A10BB12  glimepirid                         3,086,352
A10BD03  metformin og rosiglitazon            115,313
A10BD04  glimepirid og rosiglitazon               282
A10BD07  metformin og sitagliptin             417,011
A10BD08  metformin og vildagliptin            370,837
A10BD09  pioglitazon og alogliptin                 69
A10BD10  metformin og saxagliptin                 584
A10BD11  metformin og linagliptin               3,556
A10BD13  metformin og alogliptin               14,760
A10BD15  metformin og dapagliflozin            18,616
A10BD16  metformin og canagliflozin               319
A10BD19  linagliptin og empagliflozin             166
A10BD20  metformin og empagliflozin            37,186
A10BD21  saxagliptin og dapagliflozin             840
A10BD23  metformin og ertugliflozin                42
A10BD24  sitagliptin og ertugliflozin              77
A10BF01  acarbose                             112,815
A10BG02  rosiglitazon                          41,604
A10BG03  pioglitazon                           21,390
A10BH01  sitagliptin                          558,588
A10BH02  vildagliptin                         106,869
A10BH03  saxagliptin                           46,005
A10BH04  alogliptin                            14,884
A10BH05  linagliptin                          154,538
A10BJ01  exenatid                              46,891
A10BJ02  liraglutid                         1,396,293
A10BJ03  lixisenatid                            3,498
A10BJ05  dulaglutid                            42,479
A10BJ06  semaglutid                           152,063
A10BK01  dapagliflozin                        213,992
A10BK02  canagliflozin                         15,632
```

```
A10BK03  empagliflozin                          244,622
A10BK04  ertugliflozin                              175
A10BX02  repaglinid                             227,996
A10BX03  nateglinid                                  91

----------------------------------------
 ins  start at 18:05:09, ended at 18:06:53 --- ins has 12,420,016 rows.
                                            tt
A10AB01  insulin (human)                         1,555,753
A10AB04  insulin lispro                            130,656
A10AB05  insulin aspart                          2,145,054
A10AB06  insulin glulisin                           33,548
A10AC01  insulin (human)                         3,295,514
A10AD01  insulin (human)                         1,013,463
A10AD04  insulin lispro                             20,486
A10AD05  insulin aspart                          1,367,252
A10AD06  insulin degludec og insulin aspart            621
A10AE01  insulin (human)                               186
A10AE04  insulin glargin                         1,454,486
A10AE05  insulin detemir                           969,839
A10AE06  insulin degludec                          410,037
A10AE54  insulin glargin og lixisenatid                160
A10AE56  insulin degludec og liraglutid             22,961

----------------------------------------
 card  start at 18:06:53, ended at 18:08:43 --- card has 15,187,155 rows.
                           tt
C01DA02  glyceryltrinitrat         2,481,209
C01DA08  isosorbiddinitrat         1,799,896
C01DA14  isosorbidmononitrat       3,269,014
C01DX16  nicorandil                  201,254
C01AA05  digoxin                   7,435,782
```

2.4.4 Retrieving the data for each ATC group

Thus, for example if you need the lipid lowering drugs you just do:

```
> system.time(
+ load( "e:/workdata/707655/DMreg/data/rmps/lipid.Rda", v=T ) )
Loading objects:
  lipid
  atclab
   user  system elapsed
  23.06    0.33   25.66
> str( lipid, v=0 )
'data.frame':        36224144 obs. of  8 variables:
 $ pnr     : chr   ...
 $ eksd    : 'cal.yr' num  NULL ...
 $ apk     : num  NULL ...
 $ doso    : Factor w/ 221 levels "","0000000","0000001",..: NULL ...
 $ atc     : Factor w/ 20 levels "C10AB01","C10AB02",..: NULL ...
 $ strnum  : num  NULL ...
 $ strunit : chr   ...
 $ packsize: num  NULL ...
 - attr(*, "label")= chr  ...
```

```
> attr(lipid, "label")
[1] "lipid"
> fCp( object.size(lipid) )
[1] 2,105,797,376
```

The last use of `attr` is necessary because `v=0` also cuts the the first (and only) element of the `label` attribute, so if you want a human readable label this is what to do.

```
-----------------------------------------
Code: E:/workdata/707655/DMreg/r/mkRMPS.rnw
Ends: 2020-11-30 at 18:09:13
Time elapsed:       00:48:17
-----------------------------------------
```

2.5 The complications files

First the paraphernalia:

```
> library( Epi )
> library( tidyverse )
> library( haven )
> source("E:/workdata/707655/util/elapsed.r")
> setwd("E:/workdata/707655/DMreg/r")
> start()
-----------------------------------------
Code: E:/workdata/707655/DMreg/r/mkCompl.rnw
Time: 2021-03-10 at 10:50:46
-----------------------------------------
```

Complications occurring in the entire population (*i.e.* not only among diabetes patients) have been gathered in two SAS files

`wcompl`: One record per person with at least one complication, key is `pnr`, and with further 34 variables, namely the date of first occurrence of each of the 34 (grpups of) complications.

`rcompl`: One record per person and recurrent complication (`HpoG`, `Keto`, `MI`, `Str`), the key is (`pnr`, `compl`, `doC`), and there are no other variables in the dataset.

2.5.1 The SAS programs generating the complications files

The first program, `10-labcomp` defines complications based on the lab-measurements in LABKA and DVDD. Since some measurements may actually be the same between the two data bases, we exclude any measurement less than 4 days after a previous one of the same kind.

We compute eGFR based on the plasma creatinine measurement and the sex and age of the persons at the date of measurement.

The diagnoses of moderate, severe and end stage kidney disease (`ModL`, `SevL`, `ESRL`) are defined as two measurements of eGFR below 60, 30, resp. 15 with at least 60 days interval. Correspondingly, diagnoses of micro- and macro-albuminuria (`MicA`, `MacA`) are defined as two

measurements of albumin/creatinine ratio (mg/g) or albumin excretion (mg/24 h) above 30 resp. 300 with at least 60 days interval. Both for kidney disease and albuminuria, the date of the complication is defined as the date of the second measurement beyond the threshold.

Note that there are also kidney complications based exclusively on diagnosis and procedure codes (`ModL`, `SevL`, `ESRL`)

The first of each of the dates where these criteria are met are stored in a SAS data set `DMdat.micompl` with key (`pnr`,`compl`), and only non-key variable `doC`, date of first occurrence of the complication.

The second program, `10-compl` extracts complications from NPR, (from all types of records), and for each type of complication takes the first of these within each person.

The full listings of `.log` and `.lst` files from these SAS-programs (plus quite a few m ore) are in the document `v:\sdc\469drive\DMreg\tex\DMreg2018.pdf`

2.5.2 Converting SAS datasets to `.Rda` format

We now read the SAS datasets and convert them to R-datasets for easier access:

```
> system.time(wcompl <- read_sas("../data/wcompl.sas7bdat"))
   user  system elapsed
  34.38    0.79   49.59
> system.time(rcompl <- read_sas("../data/rcompl.sas7bdat"))
   user  system elapsed
   4.95    0.04    5.58
> names(wcompl) ; fCp(object.size(wcompl))
 [1] "PNR"    "doHypD" "doTCI"  "doAFib" "doIHDx" "doIStr" "doMicA" "doMI"   "doMAtD"
[10] "doModC" "doSevL" "doModL" "doESRD" "doHF"   "doHpoG" "doHStr" "doMedA" "doMinA"
[19] "doUppA" "doReti" "doNeur" "doMacA" "doESRL" "doKeto" "doSevC" "doCVD"  "doCbVD"
[28] "doDNef" "doNefL" "doNefr" "doAmp"  "doIHD"  "doStr"  "doACD"  "doMajA"
[1] 643,211,472
> names(rcompl) ; fCp(object.size(rcompl))
[1] "pnr"   "compl" "doC"
[1] 96,008,784
```

Complication grouping

The grouping of complications comes from two different places; the format defined the file `complfmt.csv`:

```
> fmt <- read.csv('../fmts/compfmt.csv', as.is = TRUE)
> a2t <- subset(fmt, FMTNAME == "$ab2abtx" & LABEL != 'Other')
> abn <- a2t$LABEL
> names(abn) <- a2t$START
> abb <- subset(fmt, FMTNAME == "$sub2grp" & LABEL != 'Other')
> print(with(abb, table(START, LABEL)), z = '.')
      LABEL
START  Amp CbVD CVD DNef HpoG Keto MAtD NefL Nefr Neur Reti
  AFib   .    .   1    .    .    .    .    .    .    .    .
  ESRD   .    .   .    .    .    .    .    .    1    .    .
  ESRL   .    .   .    .    .    .    .    1    .    .    .
  HF     .    .   1    .    .    .    .    .    .    .    .
```

```
  HpoG   .    .    .    .    1    .    .    .    .    .    .
  HStr   .    1    .    .    .    .    .    .    .    .    .
  HypD   .    .    1    .    .    .    .    .    .    .    .
  IHDx   .    .    1    .    .    .    .    .    .    .    .
  IStr   .    1    .    .    .    .    .    .    .    .    .
  Keto   .    .    .    .    .    1    .    .    .    .    .
  MacA   .    .    .    1    .    .    .    .    .    .    .
  MAtD   .    .    .    .    .    .    1    .    .    .    .
  MedA   1    .    .    .    .    .    .    .    .    .    .
  MI     .    .    1    .    .    .    .    .    .    .    .
  MicA   .    .    .    1    .    .    .    .    .    .    .
  MinA   1    .    .    .    .    .    .    .    .    .    .
  ModC   .    .    .    .    .    .    .    .    1    .    .
  ModL   .    .    .    .    .    .    .    1    .    .    .
  Neur   .    .    .    .    .    .    .    .    .    1    .
  Reti   .    .    .    .    .    .    .    .    .    .    1
  SevC   .    .    .    .    .    .    .    .    1    .    .
  SevL   .    .    .    .    .    .    .    1    .    .    .
  TCI    .    1    .    .    .    .    .    .    .    .    .
  UppA   1    .    .    .    .    .    .    .    .    .    .
> tt <- data.frame(with(abb, table(START, LABEL)))
> tt <- tt[tt$Freq==1,2:1]
> diag <- as.character(tt$START)
> grp  <- as.character(tt$LABEL)
> cbind(abn[grp], abn[diag])
     [,1]
Amp  "Amp: Amputation"
Amp  "Amp: Amputation"
Amp  "Amp: Amputation"
CbVD "CbVD: Cerebrovascular disease"
CbVD "CbVD: Cerebrovascular disease"
CbVD "CbVD: Cerebrovascular disease"
CVD  "CVD: Cardiovascular Disease"
CVD  "CVD: Cardiovascular Disease"
CVD  "CVD: Cardiovascular Disease"
CVD  "CVD: Cardiovascular Disease"
CVD  "CVD: Cardiovascular Disease"
DNef "DNef: Diabetic nephropathy"
DNef "DNef: Diabetic nephropathy"
HpoG "HpoG: Hypoglyceamia"
Keto "Keto: Ketoacidosis"
MAtD "MAtD: Macrovascular atherosclerotic disease"
NefL "NefL: Nephropathy (lab)"
NefL "NefL: Nephropathy (lab)"
NefL "NefL: Nephropathy (lab)"
Nefr "Nefr: Nephropathy"
Nefr "Nefr: Nephropathy"
Nefr "Nefr: Nephropathy"
Neur "Neur: Neuropathy"
Reti "Reti: Retinopathy"
     [,2]
Amp  "MedA: Medium amputation"
Amp  "MinA: Minor amputation"
Amp  "UppA: Upper amputation"
CbVD "HStr: Haemmoragic stroke"
CbVD "IStr: Ischaemic stroke"
CbVD "TCI: Transient cerebral ischaemia"
```

```
CVD  "AFib: Atrial fibrillation"
CVD  "HF: Heart failure"
CVD  "HypD: Hypertensive disease"
CVD  "IHDx: non-MI Ischeamic heart disease"
CVD  "MI: Myocardial Infarction"
DNef "MacA: Macro-abuminuria"
DNef "MicA: Micro-abuminuria"
HpoG "HpoG: Hypoglyceamia"
Keto "Keto: Ketoacidosis"
MAtD "MAtD: Macrovascular atherosclerotic disease"
NefL "ESRL: End-stage CKD (lab)"
NefL "ModL: Moderate CKD (lab)"
NefL "SevL: Severe CKD (lab)"
Nefr "ESRD: End-stage CKD"
Nefr "ModC: Moderate CKD"
Nefr "SevC: Severe CKD"
Neur "Neur: Neuropathy"
Reti "Reti: Retinopathy"
```

```
> print(data.frame(grp, abn[diag]),
+       right = FALSE,
+   row.names = FALSE)
```

```
 grp  abn.diag.
 Amp  MedA: Medium amputation
 Amp  MinA: Minor amputation
 Amp  UppA: Upper amputation
 CbVD HStr: Haemmoragic stroke
 CbVD IStr: Ischaemic stroke
 CbVD TCI: Transient cerebral ischaemia
 CVD  AFib: Atrial fibrillation
 CVD  HF: Heart failure
 CVD  HypD: Hypertensive disease
 CVD  IHDx: non-MI Ischeamic heart disease
 CVD  MI: Myocardial Infarction
 DNef MacA: Macro-abuminuria
 DNef MicA: Micro-abuminuria
 HpoG HpoG: Hypoglyceamia
 Keto Keto: Ketoacidosis
 MAtD MAtD: Macrovascular atherosclerotic disease
 NefL ESRL: End-stage CKD (lab)
 NefL ModL: Moderate CKD (lab)
 NefL SevL: Severe CKD (lab)
 Nefr ESRD: End-stage CKD
 Nefr ModC: Moderate CKD
 Nefr SevC: Severe CKD
 Neur Neur: Neuropathy
 Reti Reti: Retinopathy
```

The other source is the programming statements in the SAS-program `10-compel.sas`:

```
> prg <- read.table('../sas/10-compl.sas', sep='|', as.is=TRUE)[,1]
> ( prg <- prg[grep('min\\(', prg)] )
[1] "  doIHD = min( doIHDx, doMI ) ;"
[2] "  doStr = min( doHStr, doIStr ) ;"
[3] "  doACD = min( doStr, doTCI, doIHD, doMAtD ) ;"
[4] "  doMajA= min( doUppA, doMedA ) ;"
```

```
> vars <- gsub(' ', '',
+         gsub('= min\\(', ',',
+         gsub(') ;', '',
+         gsub('do', '', prg))))
> ( lv <- strsplit(vars, ',') )
[[1]]
[1] "IHD"  "IHDx" "MI"

[[2]]
[1] "Str"  "HStr" "IStr"

[[3]]
[1] "ACD"  "Str"  "TCI"  "IHD"  "MAtD"

[[4]]
[1] "MajA" "UppA" "MedA"
> pr <- function(x) cbind(x[1], x[-1])
> ( xtra <- do.call(rbind, lapply(lv, pr)) )
      [,1]   [,2]
 [1,] "IHD"  "IHDx"
 [2,] "IHD"  "MI"
 [3,] "Str"  "HStr"
 [4,] "Str"  "IStr"
 [5,] "ACD"  "Str"
 [6,] "ACD"  "TCI"
 [7,] "ACD"  "IHD"
 [8,] "ACD"  "MAtD"
 [9,] "MajA" "UppA"
[10,] "MajA" "MedA"
> grp <- c(grp, xtra[,1])
> diag <- c(diag, xtra[,2])
> dgr <- data.frame(grp, abn[diag])
> dgr <- dgr[!duplicated(dgr),]
> oo <- order(dgr[,1],dgr[,2])
```

Once this has been collected, can print the data frames that shows the groupings of the complications:

```
> print(data.frame(Groups = abn[unique(dgr[,1])]),
+       right = FALSE,
+   row.names = FALSE)
 Groups
 Amp: Amputation
 CbVD: Cerebrovascular disease
 CVD: Cardiovascular Disease
 DNef: Diabetic nephropathy
 HpoG: Hypoglyceamia
 Keto: Ketoacidosis
 MAtD: Macrovascular atherosclerotic disease
 NefL: Nephropathy (lab)
 Nefr: Nephropathy
 Neur: Neuropathy
 Reti: Retinopathy
 IHD: Ischeamic heart disease
 Str: Stroke
 ACD: Atherosclerotic cardiovascular disease
 MajA: Major amputation
```

```
> print(dgr[oo,],
+       right = FALSE,
+   row.names = FALSE)
```

```
 grp  abn.diag.
 ACD  IHD: Ischeamic heart disease
 ACD  MAtD: Macrovascular atherosclerotic disease
 ACD  Str: Stroke
 ACD  TCI: Transient cerebral ischaemia
 Amp  MedA: Medium amputation
 Amp  MinA: Minor amputation
 Amp  UppA: Upper amputation
 CbVD HStr: Haemmoragic stroke
 CbVD IStr: Ischaemic stroke
 CbVD TCI: Transient cerebral ischaemia
 CVD  AFib: Atrial fibrillation
 CVD  HF: Heart failure
 CVD  HypD: Hypertensive disease
 CVD  IHDx: non-MI Ischeamic heart disease
 CVD  MI: Myocardial Infarction
 DNef MacA: Macro-abuminuria
 DNef MicA: Micro-abuminuria
 HpoG HpoG: Hypoglyceamia
 IHD  IHDx: non-MI Ischeamic heart disease
 IHD  MI: Myocardial Infarction
 Keto Keto: Ketoacidosis
 MajA MedA: Medium amputation
 MajA UppA: Upper amputation
 MAtD MAtD: Macrovascular atherosclerotic disease
 NefL ESRL: End-stage CKD (lab)
 NefL ModL: Moderate CKD (lab)
 NefL SevL: Severe CKD (lab)
 Nefr ESRD: End-stage CKD
 Nefr ModC: Moderate CKD
 Nefr SevC: Severe CKD
 Neur Neur: Neuropathy
 Reti Reti: Retinopathy
 Str  HStr: Haemmoragic stroke
 Str  IStr: Ischaemic stroke
```

Here is a bit more elaborate print of the groupings:

```
> dd <- sapply(dgr[oo,], as.character)
> wh <- cumsum(2 - duplicated(dd[,1]) )
> xx <- dd[1,,drop=FALSE]
> xx[,1] <- '------'
> xx[,2] <- ''
> xx <- xx[rep(1,wh[length(wh)]+1),]
> xx[wh,] <- dd
```

Complication names

In order to get the long informative names of the complications we read the `.csv` file which is the base for the generating the format used for grouping and labeling of complications. This has the long form of the complications labels:

```
> cnam <- read.csv( "../fmts/compfmt.csv", header=TRUE )
> cnam <- subset( cnam, FMTNAME=="$abb2txt" )
> compl.names <- as.character( cnam$LABEL )
> names( compl.names ) <- cnam$START
> cbind( compl.names )
     compl.names
AtCV "Atherosclerotic CVD"
MAtD "Macrovascular atherosclerotic disease"
ACD  "Atherosclerotic cardiovascular disease"
AFib "Atrial fibrillation"
HF   "Heart failure"
IHD  "Ischeamic heart disease"
IHDx "non-MI Ischeamic heart disease"
MI   "Myocardial Infarction"
HypD "Hypertensive Disease"
HpoG "Hypoglyceamia"
CbVD "Cerebrovascular disease"
Str  "Stroke"
IStr "Ischaemic stroke"
HStr "Haemmoragic stroke"
TCI  "Transient cerebral ischaemia"
Keto "Ketoacidosis"
MajA "Major amputation"
UppA "Upper amputation"
MedA "Medium amputation"
MinA "Minor amputation"
Neur "Neuropathy"
Reti "Retinopathy"
ModC "Moderate CKD"
SevC "Severe CKD"
ESRD "End-stage CKD"
ModL "Moderate CKD (lab)"
SevL "Severe CKD (lab)"
ESRL "End-stage CKD (lab)"
Amp  "Amputation"
CVD  "Cardiovascular Disease"
Nefr "Nephropathy"
NefL "Nephropathy (lab)"
DNef "Diabetic nephropathy"
MicA "Micro-abuminuria"
MacA "Macro-abuminuria"
```

Now `compl.names` is a character vector with the long names of the complications. The *names* attribute of the vector is the abbreviations of the complications used in `fcompl` and `wcompl`; we see that they are all there:

```
> (zz <- sort(match(paste0("do", names(compl.names)), names(wcompl))))
 [1]  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
[29] 30 31 32 33 34 35
> table(diff(zz))

 1
33
```

—if they were not, there would be `NA`s in the vector or the sequence would not be complete. The `table` also demonstrates that there are 28 entries in `zz`.

The point of using the abbreviations as *names* attributes of the `compl.names` is that you can get the official long text by indexing with the abbreviation:

```
> compl.names["CbVD"]
                    CbVD
"Cerebrovascular disease"
```

which is useful when labeling tables and graphs.

Number of recurrent events per person

Here is an account of how many persons have how many recurrences of each of the recurrent types of complications:

```
> nn <- data.frame(N=1)
> for( nm in unique(rcompl$compl) )
+    {
+    dd <- data.frame(with(subset(rcompl, compl == nm), table(table(pnr))))
+    names(dd) <- c("N", nm)
+    dd$N <- as.numeric(as.character(dd$N))
+    nn <- merge(dd, nn, all=TRUE)
+    }
> mm <- as.matrix(nn[,-1])
> row.names(mm) <- nn$N
> rCtable( mm[1:30,] )
```

	Keto	HpoG	MI	Str
1	7,018	26,103	100,743	140,071
2	3,352	12,146	67,154	94,083
3	1,619	4,893	43,110	53,851
4	881	2,609	32,480	32,938
5	532	1,425	24,023	19,394
6	296	917	16,964	11,766
7	192	603	11,385	7,099
8	161	376	7,677	4,314
9	96	299	5,017	2,737
10	75	241	3,557	1,808
11	70	181	2,419	1,148
12	39	122	1,718	813
13	38	88	1,140	534
14	35	94	883	388
15	27	67	657	222
16	27	60	493	171
17	13	32	333	138
18	18	48	272	83
19	13	36	209	65
20	12	28	145	46
21	11	26	111	28
22	10	18	92	22
23	9	17	64	18
24	6	20	55	17
25	10	13	41	10
26	6	14	35	7
27	6	9	25	*
28	4	19	13	4
29	4	9	20	*
30	5	9	11	*

Grooming the data frames

We want to store the datasets as `data.frames`, we remove the `label` and the `format.sas` attributes of the `pnr` variable:

```
> wcompl <- plyr::rename(as.data.frame(wcompl), c("PNR"="pnr"))
> attr(wcompl$pnr, "format.sas") <- NULL
> attr(wcompl$pnr, "label") <- NULL
> attr(wcompl$doIHD, "label") <- NULL
> attr(wcompl$doStr, "label") <- NULL
> attr(wcompl$doACD, "label") <- NULL
> attr(wcompl$doMajA, "label") <- NULL
> rcompl <- as.data.frame(rcompl)
> attr(rcompl$pnr  , "format.sas") <- NULL
> attr(rcompl$pnr  , "label") <- NULL
> attr(rcompl$compl, "label") <- NULL
```

Finally we convert the date variables in data frames to class `cal.yr`:

```
> wcompl <- cal.yr(wcompl)
> rcompl <- cal.yr(rcompl)
```

We save these as R-datasets and document how long it takes to read them back in—note that we save the names vector with each of the files, too.

```
> save(wcompl, compl.names, file = "../data/wcompl.Rda")
> system.time(         load(file = "../data/wcompl.Rda", v=T) )
Loading objects:
  wcompl
  compl.names
   user  system elapsed
   3.70    0.21    3.93
> save(rcompl, compl.names, file = "../data/rcompl.Rda")
> system.time(         load(file = "../data/rcompl.Rda", v=T) )
Loading objects:
  rcompl
  compl.names
   user  system elapsed
   1.95    0.03    1.98
```

Thus these data sets provide for a reading time which is a factor 5–10 smaller then reading from the SAS files.

2.5.3 Using the complications data sets

This is the only part of the section you need read. There are two datasets available, both refrring to the *entire population*.

Essentially the the SAS datasets have been read and converted to R-datasets

`wcompl`: One record per person with at least one complication, key is `pnr`, and with further 34 variables, namely the date of first occurrence of each of the 34 (grpups of) complications.

`rcompl`: One record per person and recurrent complication (`HpoG`, `Keto`, `MI`, `Str`), the key is (`pnr`, `compl`, `doC`), and there are no other variables in the dataset.

When loading each of the datasets you do not only get the complications datasets, but also the human readable labels of each of the variables in `compl.names`.

```
> system.time(load(file = "../data/wcompl.Rda", v=T))
Loading objects:
  wcompl
  compl.names
   user  system elapsed
   3.63    0.25    3.87
> names(wcompl)
 [1] "pnr"    "doHypD" "doTCI"  "doAFib" "doIHDx" "doIStr" "doMicA" "doMI"   "doMAtD"
[10] "doModC" "doSevL" "doModL" "doESRD" "doHF"   "doHpoG" "doHStr" "doMedA" "doMinA"
[19] "doUppA" "doReti" "doNeur" "doMacA" "doESRL" "doKeto" "doSevC" "doCVD"  "doCbVD"
[28] "doDNef" "doNefL" "doNefr" "doAmp"  "doIHD"  "doStr"  "doACD"  "doMajA"
> system.time(load(file = "../data/rcompl.Rda", v=T))
Loading objects:
  rcompl
  compl.names
   user  system elapsed
   1.97    0.04    2.02
> names(rcompl)
[1] "pnr"   "compl" "doC"
```

The vector `compl.names` is a *named* character vector, the names are the abbreviations, they can be used as indices, e.g.:

```
> compl.names['CVD']
                   CVD
"Cardiovascular Disease"
```

The grouping of complications are as follows; note that some groups appear both sides.

```
> print(as.data.frame(xx),
+       row.names = FALSE,
+       col.names = FALSE,
+           right = FALSE)
 grp    abn.diag.
 ------
 ACD    IHD: Ischeamic heart disease
 ACD    MAtD: Macrovascular atherosclerotic disease
 ACD    Str: Stroke
 ACD    TCI: Transient cerebral ischaemia
 ------
 Amp    MedA: Medium amputation
 Amp    MinA: Minor amputation
 Amp    UppA: Upper amputation
 ------
 CbVD   HStr: Haemmoragic stroke
 CbVD   IStr: Ischaemic stroke
 CbVD   TCI: Transient cerebral ischaemia
 ------
 CVD    AFib: Atrial fibrillation
 CVD    HF: Heart failure
 CVD    HypD: Hypertensive disease
 CVD    IHDx: non-MI Ischeamic heart disease
```

```
CVD    MI: Myocardial Infarction
------
DNef   MacA: Macro-abuminuria
DNef   MicA: Micro-abuminuria
------
HpoG   HpoG: Hypoglyceamia
------
IHD    IHDx: non-MI Ischeamic heart disease
IHD    MI: Myocardial Infarction
------
Keto   Keto: Ketoacidosis
------
MajA   MedA: Medium amputation
MajA   UppA: Upper amputation
------
MAtD   MAtD: Macrovascular atherosclerotic disease
------
NefL   ESRL: End-stage CKD (lab)
NefL   ModL: Moderate CKD (lab)
NefL   SevL: Severe CKD (lab)
------
Nefr   ESRD: End-stage CKD
Nefr   ModC: Moderate CKD
Nefr   SevC: Severe CKD
------
Neur   Neur: Neuropathy
------
Reti   Reti: Retinopathy
------
Str    HStr: Haemmoragic stroke
Str    IStr: Ischaemic stroke
------
```

2.5.4 Microvascular complications

The dates for microvascular complications are a bit special. The lab-based complications `doMicA`, `doMacA`, `doModL`, `doSevL` and `doESRL` are difficult to use; only information after 2010 is available, and information for Region Midt is absent.

Moreover the lab-based micro-vascular complications are always ordered, such that `doMicA` < `doMacA`, the opposite inequality do not occur:

```
> with(wcompl,
+      ftable(mic = !is.na(doMicA),
+             mac = !is.na(doMacA),
+             doMicA < doMacA,
+             exclude = NULL,
+             col.vars = 1:2))
     mic   FALSE            TRUE
     mac   FALSE    TRUE   FALSE    TRUE

TRUE           0       0       0   18325
NA       1687970   22768  140653       0
```

You will note that there are occurrences of a valid value of `doMacA`, with `NA` for `doMicA`. So the meaning of `doMicA` is "date of microalbuminuria only". If you want the date of "date of *at least* microalbuminuria", you must use for example:

```
> doalMicA <- pmin(doMicA, doMacA, na.rm = TRUE)
```

This is also the case for the lab-based kidney measurements:

```
> with(wcompl,
+      ftable(mod = !is.na(doModL),
+             sev = !is.na(doSevL),
+             esr = !is.na(doESRL),
+             ms = doModL < doSevL,
+             me = doModL < doESRL,
+             se = doSevL < doESRL,
+             exclude = NULL,
+             col.vars = 1:3))
               mod   FALSE                          TRUE
               sev   FALSE           TRUE           FALSE           TRUE
               esr   FALSE    TRUE   FALSE    TRUE   FALSE    TRUE   FALSE    TRUE
ms   me    se
TRUE TRUE  TRUE           0       0       0       0       0       0       0    5066
           NA             0       0       0       0       0       0       0       0
     NA    TRUE           0       0       0       0       0       0       0       0
           NA             0       0       0       0       0       0   42114       0
NA   TRUE  TRUE           0       0       0       0       0       0       0       0
           NA             0       0       0       0       0    2767       0       0
     NA    TRUE           0       0       0    4802       0       0       0       0
           NA       1421870    6202   15825       0  371070       0       0       0
```

—you observe that none of the inequalities appear as `FALSE`.

However, the microvascular complications based on NPR using codes for diagnoses and procedures does not behave like this, the moderate, severe and end-stage complications are independent, and the three dates can occur in any order — all tree inequalities appear as both `TRUE` and `FALSE`:

```
> print(with(wcompl,
+            ftable(mod = !is.na(doModC),
+                   sev = !is.na(doSevC),
+                   esr = !is.na(doESRD),
+                   ms = doModC < doSevC,
+                   me = doModC < doESRD,
+                   se = doSevC < doESRD,
+                   exclude = NULL,
+                   col.vars = 1:3)), zero=".")
                  mod   FALSE                          TRUE
                  sev   FALSE           TRUE           FALSE           TRUE
                  esr   FALSE    TRUE   FALSE    TRUE   FALSE    TRUE   FALSE    TRUE
ms    me    se
FALSE FALSE FALSE           .       .       .       .       .       .       .     114
            TRUE            .       .       .       .       .       .       .     163
            NA              .       .       .       .       .       .       .       .
      TRUE  FALSE           .       .       .       .       .       .       .       .
            TRUE            .       .       .       .       .       .       .     152
            NA              .       .       .       .       .       .       .       .
      NA    FALSE           .       .       .       .       .       .       .       .
            TRUE            .       .       .       .       .       .       .       .
            NA              .       .       .       .       .       .     691       .
TRUE  FALSE FALSE           .       .       .       .       .       .       .     409
            TRUE            .       .       .       .       .       .       .       .
            NA              .       .       .       .       .       .       .       .
```

```
      TRUE  FALSE          .       .       .       .       .       .       .     417
            TRUE           .       .       .       .       .       .       .     502
            NA             .       .       .       .       .       .       .       .
      NA    FALSE          .       .       .       .       .       .       .       .
            TRUE           .       .       .       .       .       .       .       .
            NA             .       .       .       .       .       .    1986       .
NA    FALSE FALSE          .       .       .       .       .       .       .       .
            TRUE           .       .       .       .       .       .       .       .
            NA             .       .       .       .       .   11297       .       .
      TRUE  FALSE          .       .       .       .       .       .       .       .
            TRUE           .       .       .       .       .       .       .       .
            NA             .       .       .       .       .   10171       .       .
      NA    FALSE          .       .       .     183       .       .       .       .
            TRUE           .       .       .     161       .       .       .       .
            NA       1757697   26171    2029       .   57573       .       .       .
```

Finally note that valid dates for the microvascular complications vary considerably by calendar time:

```
> MicA <- data.frame(table(pmax(1995,floor(wcompl$doMicA)))) %>% rename(MicA = Freq)
> MacA <- data.frame(table(pmax(1995,floor(wcompl$doMacA)))) %>% rename(MacA = Freq)
> ModL <- data.frame(table(pmax(1995,floor(wcompl$doModL)))) %>% rename(ModL = Freq)
> SevL <- data.frame(table(pmax(1995,floor(wcompl$doSevL)))) %>% rename(SevL = Freq)
> ESRL <- data.frame(table(pmax(1995,floor(wcompl$doESRL)))) %>% rename(ESRL = Freq)
> ModC <- data.frame(table(pmax(1995,floor(wcompl$doModC)))) %>% rename(ModC = Freq)
> SevC <- data.frame(table(pmax(1995,floor(wcompl$doSevC)))) %>% rename(SevC = Freq)
> ESRD <- data.frame(table(pmax(1995,floor(wcompl$doESRD)))) %>% rename(ESRD = Freq)
> MicA <- mutate(MicA, Var1 = as.numeric(as.character(Var1)))
> MacA <- mutate(MacA, Var1 = as.numeric(as.character(Var1)))
> ModL <- mutate(ModL, Var1 = as.numeric(as.character(Var1)))
> SevL <- mutate(SevL, Var1 = as.numeric(as.character(Var1)))
> ESRL <- mutate(ESRL, Var1 = as.numeric(as.character(Var1)))
> ModC <- mutate(ModC, Var1 = as.numeric(as.character(Var1)))
> SevC <- mutate(SevC, Var1 = as.numeric(as.character(Var1)))
> ESRD <- mutate(ESRD, Var1 = as.numeric(as.character(Var1)))
> when <- full_join(MicA,
+                   MacA) %>%
+         full_join(ModL) %>%
+         full_join(SevL) %>%
+         full_join(ESRL) %>%
+         full_join(ModC) %>%
+         full_join(SevC) %>%
+         full_join(ESRD)
> M <- as.matrix(when[order(when$Var1),])
> M <- ifelse(is.na(M), 0, M)
> rownames(M) <- M[,1]
> M <- addmargins(M, 1)
> # print so that small numbers are masked
> rCtable(M[,-1], w = 7)
         MicA    MacA    ModL    SevL    ESRL    ModC    SevC    ESRD

1995        .       .       .       .       .   1,675       .     366
1996        .       .       .       .       .     732       .     661
1997        .       .       .       .       .     825       .     592
1998        .       .       .       .       .     909       .     593
1999        .       .       .       .       .   1,364       .     841
2000        *       .       .       .       .   1,419       .   1,532
```

2001	*	*	.	.	.	1,702	.	1,784
2002	*	4	.	.	.	2,083	.	1,877
2003	11	*	.	.	.	2,421	.	1,989
2004	28	7	.	.	.	2,675	*	2,165
2005	201	55	.	.	.	2,913	.	2,271
2006	472	89	.	.	.	3,093	.	2,458
2007	1,848	525	.	.	.	3,443	*	2,560
2008	2,170	728	.	.	.	3,571	10	2,529
2009	3,102	970	74	21	22	4,114	7	2,664
2010	5,255	1,474	15,271	2,092	635	4,804	6	2,610
2011	8,852	3,100	25,228	3,520	1,403	6,143	41	2,656
2012	11,814	3,054	28,359	4,183	1,331	5,655	422	2,797
2013	11,182	2,292	24,916	3,791	1,002	6,118	937	2,811
2014	13,707	3,830	65,438	8,526	2,279	5,868	819	2,810
2015	16,124	4,290	62,857	8,633	2,453	5,947	1,027	2,784
2016	21,547	5,712	81,889	12,547	2,960	6,378	1,146	2,911
2017	25,113	6,078	54,332	10,467	2,760	5,387	1,420	2,818
2018	26,562	6,205	45,535	10,103	2,906	4,236	970	2,661
2019	10,985	2,678	17,118	3,924	1,086	.	.	.
Sum	158,978	41,093	421,017	67,807	18,837	83,475	6,807	49,740

Note that this table is a table of persons in the entire Danish population with each of these complications, tabulated by the `first` occurrence of the complication.

```
------------------------------------------
Code: E:/workdata/707655/DMreg/r/mkCompl.rnw
Ends: 2021-03-10 at 10:52:45
Time elapsed:       00:01:59
------------------------------------------
```