FRS workshop

September 2022

Contents

1 Benefit Comparison – Year: 2020-21

1.1 Create Both1

Create a new dataset called 'both1' from merging the FRS adult dataset and FRS benefits dataset by their common variables (sernum, benunit and person)

In this case, we choose to keep the numerical version of categorical variables

```
library(foreign)    ### Library necessary to open SPSS files
library(dplyr)      ### Advanced data manipulation functions
library(Hmisc)      ### Extra statistical functions

setwd("/home/piet/Dropbox/work/UKDS/FRS workshop/UKDA-8948-spss/spss/spss25/")
a21<-read.spss("adult.sav",to.data.frame = T,
use.value.labels = F )%>%
              select(SERNUM,BENUNIT,PERSON,GROSS4)
b21<-read.spss("benefits.sav",to.data.frame = T, use.value.labels = F)%>%
             select(SERNUM,BENUNIT,PERSON, BENEFIT,BENAMT)
both1<-merge(a21,b21,by=c("SERNUM","BENUNIT","PERSON"),all.x=F,all.y=F)

names(both1)<-tolower(names(both1)) ### Changes names to lowercase
```

1.2 Benefit frequencies

Gives the grossed frequency, percentage, cumulative frequency and cumulative percentage of each benefit

```
tmptab<-xtabs(gross4~benefit,both1,drop.unused.levels=T)
cbind(
"Freq"= tmptab, "Pct"=round(100*prop.table(tmptab),1),
"Cum. freq."=cumsum(tmptab), "Cum. pct"= round(cumsum(100*prop.table(tmptab)),2)
)
```

```
##         Freq  Pct Cum. freq. Cum. pct
## 1     682377  1.4     682377     1.40
## 2     605697  1.2    1288074     2.65
## 3    6207126 12.8    7495200    15.41
## 4    1006961  2.1    8502161    17.48
## 5   11157837 22.9   19659998    40.43
## 6      53610  0.1   19713608    40.54
## 8      81338  0.2   19794946    40.70
## 9     142150  0.3   19937096    41.00
## 10      8515  0.0   19945611    41.01
## 12    813335  1.7   20758946    42.69
## 13    736470  1.5   21495416    44.20
## 14    252917  0.5   21748333    44.72
## 15    123434  0.3   21871767    44.97
## 16   1425025  2.9   23296792    47.90
## 19    244486  0.5   23541278    48.41
## 21     16183  0.0   23557461    48.44
## 22     18430  0.0   23575891    48.48
## 24     10906  0.0   23586797    48.50
## 32      2449  0.0   23589246    48.51
## 33     22348  0.0   23611594    48.55
## 34      7461  0.0   23619055    48.57
```

```
## 37       2297  0.0    23621352    48.57
## 61       1300  0.0    23622652    48.57
## 62   11361568 23.4    34984220    71.94
## 65      32452  0.1    35016672    72.00
## 66      57514  0.1    35074186    72.12
## 69      58364  0.1    35132550    72.24
## 70     138664  0.3    35271214    72.53
## 78       6089  0.0    35277303    72.54
## 81       4545  0.0    35281848    72.55
## 82      13705  0.0    35295553    72.58
## 83       1306  0.0    35296859    72.58
## 90     825829  1.7    36122688    74.28
## 91    1431594  2.9    37554282    77.22
## 94    2532242  5.2    40086524    82.43
## 95    2774958  5.7    42861482    88.14
## 96    2175755  4.5    45037237    92.61
## 97    1767003  3.6    46804240    96.24
## 98     303494  0.6    47107734    96.87
## 99       9577  0.0    47117311    96.89
## 110    678344  1.4    47795655    98.28
## 111    831922  1.7    48627577    99.99
## 115      3754  0.0    48631331   100.00
```

1.3 Distribution of Universal Credit, State Pension , Personal Independence Payment (Daily Living), Personal Independence Payment (PIP) Mobility

Examines the distribution of grossed Universal Credit benefit, grossed State Pension benefit amounts, grossed Personal Independence Payment (Daily Living benefit), grossed Personal Independence Payment (PIP) Mobility benefit amounts, including an assessment of normality and discovery of outliers

xtabs() produces the tables; *prop.table()* computes the proportions, *round(…,1)* rounds the result to one decimal. R is programmed with parsimony in mind: we only get the ourput we asked for. This is why we need to specify each indicator we are interested in. We can then gather them all in a single table with *cbind()*. We select the subset of respondents on UC with the *filter()* command. The pipe *%>%* operators allow us to combine several command one after the other.

```
#knitr::kable(
round(t(both1%>%
      filter(benefit==95  | benefit==5 | benefit==96 |
            benefit==97)%>%group_by(benefit)%>%
        summarise(
  "Observations"=length(benamt),
  "Mean"=wtd.mean(benamt,gross4),
  "Median"=wtd.quantile(benamt,gross4,probs=.5),
  "Std Deviation"=sqrt(wtd.var(benamt,gross4)),
  "Variance"=wtd.var(benamt,gross4),
  "Mode"=as.numeric(names(sort(-table(benamt)))[1]),
  "Range"=max(benamt)-min(benamt),
  "IQR"=wtd.quantile(benamt,gross4,probs=.75)-wtd.quantile(benamt,gross4,probs=.25),
"Skewness"=Weighted.Desc.Stat::w.skewness(benamt,gross4),
"Kurtosis"=Weighted.Desc.Stat::w.kurtosis(benamt,gross4)
)),2) #)
```

```
##                   [,1]      [,2]    [,3]    [,4]
```

```
## benefit            5.00     95.00  96.00  97.00
## Observations  5573.00    642.00 678.00 566.00
## Mean            157.73    180.86  72.79  46.75
## Median          159.25    169.07  59.70  62.25
## Std Deviation    48.85    108.03  15.25  18.94
## Variance       2386.61  11670.59 232.61 358.81
## Mode            175.00      0.00  59.70  62.25
## Range           475.00    758.23  65.55  38.65
## IQR              46.15    160.21  29.45  38.65
## Skewness          0.30      0.49  -0.04  -0.40
## Kurtosis          2.10      0.39  -1.26  -1.84
```

Weighted quantiles

```
#knitr::kable(
### Computing the results first, and storing them in tmp
tmp<-both1%>%filter(benefit==95| benefit==5 | benefit==96 | benefit==97) %>%
  group_by(benefit)%>%
summarise(
"ben nr"=wtd.quantile(benamt,gross4,probs=c(.01,.05,.1,.25,.5,.75,.9,.95,.99))
) #)

### tmp is a long table with each set of nine quantiles
### added one after the other (36 rows).
### The code below makes it easier to read

data.frame(
  "Quantile"=c(.01,.05,.1,.25,.5,.75,.9,.95,.99),
  "UC"=      tmp[tmp$benefit==95,2],
  "Pension"= tmp[tmp$benefit==5,2],
  "PIP (DL)"=tmp[tmp$benefit==96,2],
  "PIP (M)"= tmp[tmp$benefit==97,2])
```

```
##   Quantile    ben.nr ben.nr.1 ben.nr.2 ben.nr.3
## 1     0.01   0.00000    36.49    59.70    23.60
## 2     0.05   0.00000    80.00    59.70    23.60
## 3     0.10  50.43912    87.00    59.70    23.60
## 4     0.25  94.33085   134.81    59.70    23.60
## 5     0.50 169.07014   159.25    59.70    62.25
## 6     0.75 254.54071   180.96    89.15    62.25
## 7     0.90 333.69863   211.25    89.15    62.25
## 8     0.95 362.00000   238.58    89.15    62.25
## 9     0.99 432.29622   297.50    89.15    62.25
```

Extreme values

We have chosen to collate the results with *cbind()* in this case, but each part can be run separately if needed.

```
#knitr::kable(
round(
cbind(
    both1%>%filter(benefit==95 | benefit==5 | benefit==96 | benefit==97)%>%
      select(sernum,person,benamt,benefit)%>%
  group_by(benefit)%>%slice_min(order_by = benamt,n=5,with_ties = FALSE),
  both1%>%filter(benefit==95 | benefit==5 | benefit==96 | benefit==97)%>%
    select(sernum,person,benamt,benefit)%>%
    group_by(benefit)%>%slice_max(order_by = benamt,n=5,with_ties = FALSE)
```

```
)
  ,1)#)
```

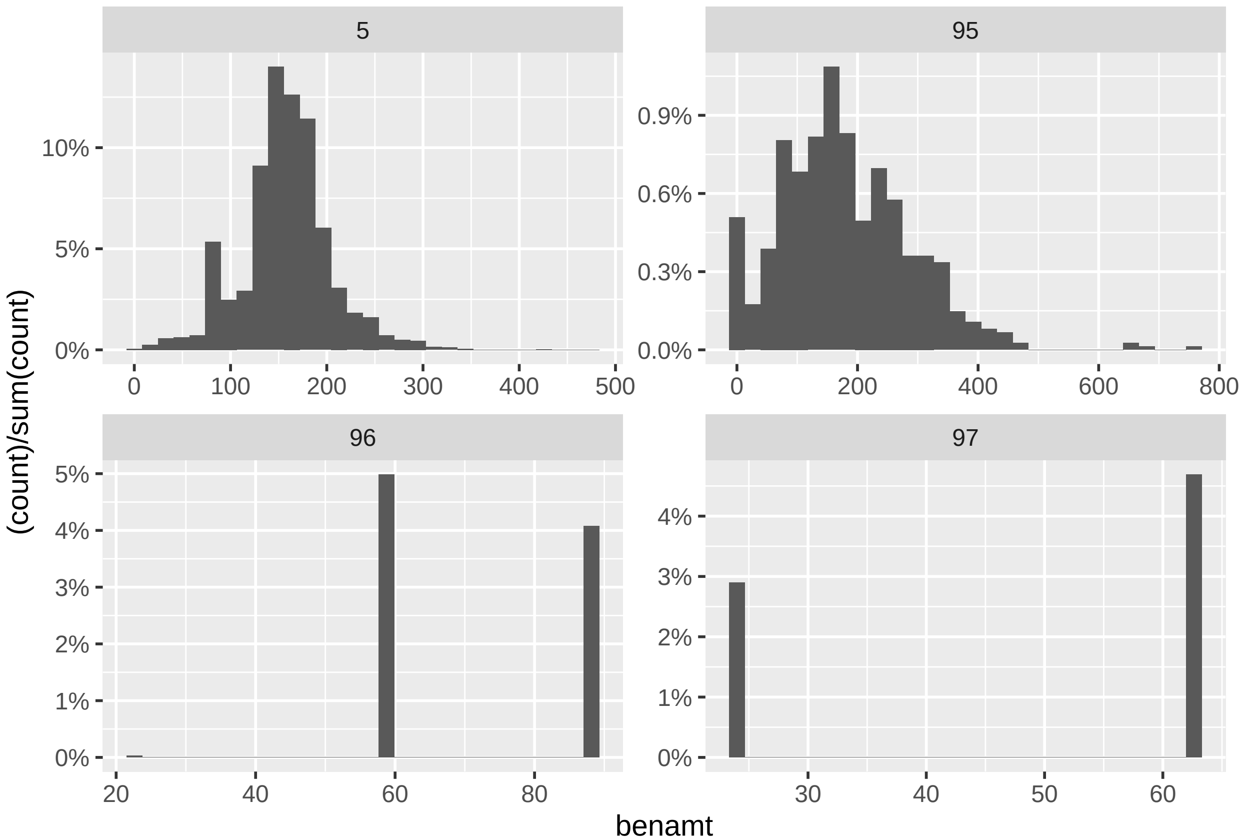
```
## # A tibble: 20 x 8
##    sernum...1 person...2 benamt...3 benefit...4 sernum~1 perso~2 benam~3 benef~4
##         <dbl>      <dbl>      <dbl>       <dbl>    <dbl>   <dbl>   <dbl>   <dbl>
##  1       7709          1        0             5     1200       1   475         5
##  2       1890          2        1             5     5630       1   425         5
##  3       3811          1        1             5      204       1   425.        5
##  4       2589          1        6.2           5     7363       1   355.        5
##  5       3995          1        8.6           5     4913       2   352.        5
##  6       1038          1        0            95     2992       1   758.       95
##  7       1560          1        0            95     2229       1   679.       95
##  8       1607          1        0            95     4292       1   646.       95
##  9       1808          2        0            95     7751       1   645.       95
## 10       1896          1        0            95     9571       1   480.       95
## 11       3237          1       23.6          96     1015       2    89.2      96
## 12       5924          1       23.6          96     1041       1    89.2      96
## 13      10007          2       59.7          96     1065       1    89.2      96
## 14       1006          1       59.7          96     1163       1    89.2      96
## 15       1015          1       59.7          96     1172       2    89.2      96
## 16       1041          1       23.6          97    10007       2    62.2      97
## 17       1120          1       23.6          97     1006       1    62.2      97
## 18       1172          2       23.6          97     1015       2    62.2      97
## 19       1191          1       23.6          97      105       1    62.2      97
## 20       1199          1       23.6          97     1065       1    62.2      97
## # ... with abbreviated variable names 1: sernum...5, 2: person...6,
## #   3: benamt...7, 4: benefit...8
```

1.4 Histograms

Universal credit

The line below creates the histogram.

```
ggplot(both1%>%
      filter(benefit==95 | benefit==5 | benefit==96 | benefit==97),
      aes(x=benamt))+ ### Base plot specification
geom_histogram(aes(y = (..count..)/sum(..count..)))+ ### Converts y axis to percent
scale_y_continuous(labels = scales::percent)+ ### Prints y axis as percent
facet_wrap(~benefit,scale="free")  ### Each histogram has its own plot
```



1.5 Distribution of Universal Credit ex. £0

Note There is no need to create a new dataset, we can keep on working with both1

Examines the distribution of grossed Universal Credit benefit amounts excluding £0 observations, including an assessment of normality and discovery of outliers

In the syntax below we exclude cases with £0 benefits by adding *benamt>0* to the *filter()* command.

```
round(t(both1%>%filter(benefit==95 & benamt>0)%>%summarise(
  "Observations"=length(benamt),
  "Mean"=wtd.mean(benamt,gross4),
  "Median"=wtd.quantile(benamt,gross4,probs=.5),
  "Std Deviation"=sqrt(wtd.var(benamt,gross4)),
  "Variance"=wtd.var(benamt,gross4),
  "Mode"=as.numeric(names(sort(-table(benamt)))[1]),
  "Range"=max(benamt)-min(benamt),
  "IQR"=wtd.quantile(benamt,gross4,probs=.75)-wtd.quantile(benamt,gross4,probs=.25),
"Skewness"=Weighted.Desc.Stat::w.skewness(benamt,gross4),
"Kurtosis"=Weighted.Desc.Stat::w.kurtosis(benamt,gross4)
)),2)
```

```
##                    [,1]
## Observations     611.00
## Mean             191.99
## Median           173.02
## Std Deviation    101.25
## Variance       10251.55
## Mode              94.33
```

```
## Range          757.32
## IQR            140.74
## Skewness         0.65
## Kurtosis         0.68
```

Weighted quantiles

```
both1%>%filter(benefit==95 & benamt>0)%>%select(benamt,gross4)%>%summarise(
  "Quantiles"=
round(wtd.quantile(benamt,gross4,probs=c(.01,.05,.1,.25,.5,.75,.9,.95,.99)),2)
)
```

```
##   Quantiles
## 1     21.60
## 2     51.99
## 3     74.04
## 4    116.40
## 5    173.02
## 6    257.15
## 7    335.31
## 8    362.81
## 9    432.30
```

Extreme values

```
cbind(
  both1%>%filter(benefit==95 & benamt>0)%>%select(sernum,benunit,person,benamt)%>%
  slice_min(order_by = benamt,n=5,with_ties = FALSE),
  both1%>%filter(benefit==95 & benamt>0)%>%select(sernum,benunit,person,benamt)%>%
    slice_max(order_by = benamt,n=5,with_ties = FALSE)
)
```

```
##   sernum benunit person    benamt sernum benunit person   benamt
## 1   4183       2      3 0.9182466   2992       1      1 758.2346
## 2   7667       4      4 1.8802192   2229       1      1 679.3828
## 3   6337       1      1 3.3784110   4292       1      1 645.5872
## 4   6814       1      1 3.5349041   7751       1      1 645.1706
## 5   5498       1      1 3.6499726   9571       1      1 479.5600
```

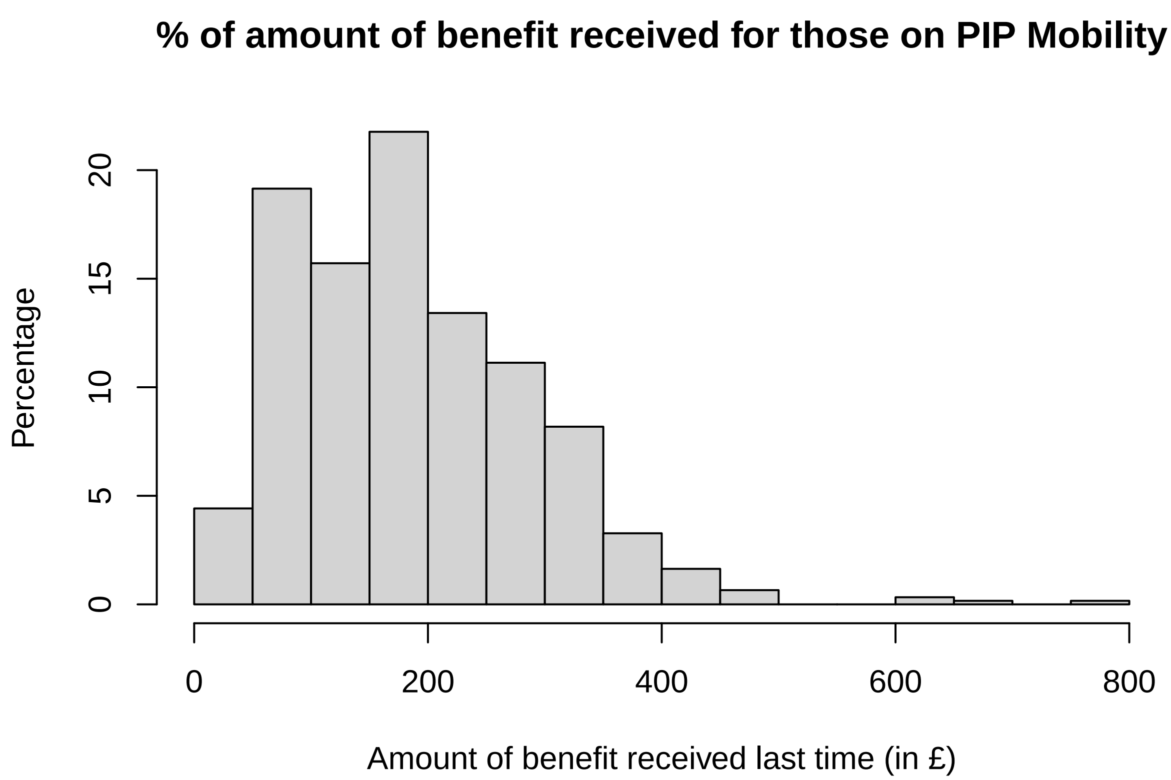
Histogram of Universal Credit (without 0s)

The syntax below also select a subset of the data, with square brackets as an alternative to the *filter()* command used above. We also use an alternative command to create the histogram.

```
z<-hist(both1$benamt[both1$benefit==95 & both1$benamt>0],freq=F)
```

```
z$density <- z$counts/sum(z$counts) * 100 ### Converting y axis into percentages
```

```
plot(z, freq = FALSE,
     main= '% of amount of benefit received for those on PIP Mobility',
     xlab="Amount of benefit received last time (in £)",
     ylab="Percentage"
)
```



1.6 UC and tenure

Create uc_tenure2

Create a new dataset called 'uc_tenure2' from merging the 'uc_tenure1'(ie both1) dataset and FRS household dataset by their common variable (sernum)

```
setwd("/home/piet/Dropbox/work/UKDS/FRS workshop/UKDA-8948-spss/spss/spss25" )
h21<-read.spss("househol.sav",to.data.frame = T, use.value.labels = T,
add.undeclared.levels="append")%>%select(SERNUM,TENURE)
names(h21)<-tolower(names(h21))
uc_tenure2<-merge(both1,h21,by=c("sernum"),all.x=F,all.y=F)
levels(uc_tenure2$tenure)[2]<-"Bought with mortgage" ### Shorten a category, for nicer display
```

Frequency of Tenure

Gives the grossed frequency, percentage, cumulative frequency and cumulative percentage of each category for the variable 'tenure' for those in receipt of Universal Credit

```
tmptab<-xtabs(gross4~tenure,uc_tenure2,subset=benefit==95,drop.unused.levels=T)
cbind(
"Freq"= tmptab, "Pct"=round(100*prop.table(tmptab),1),
"Cum. freq."=cumsum(tmptab), "Cum. pct"= round(cumsum(100*prop.table(tmptab)),2)
)
```

```
##                         Freq  Pct Cum. freq. Cum. pct
## Owns it outright      243349  8.8     243349     8.77
## Bought with mortgage  282334 10.2     525683    18.94
## Part own, part rent    15214  0.5     540897    19.49
## Rents                2217836 79.9    2758733    99.42
## Rent-free              16225  0.6    2774958   100.00
```

2 Benefit Comparison – Year: 2019-20

Create a new dataset Both3 (we could re-use both1) from merging the FRS adult dataset and FRS benefits dataset by their common variables (sernum, benunit and person)

```
setwd("/home/piet/Dropbox/work/UKDS/FRS workshop/UKDA-8802-spss/spss/spss25/")
a19<-read.spss("adult.sav",to.data.frame = T,use.value.labels = F )%>%
select(SERNUM,BENUNIT,PERSON,GROSS4)
b19<-read.spss("benefits.sav",to.data.frame = T,use.value.labels = F)%>%
select(SERNUM,BENUNIT,PERSON, BENEFIT,BENAMT)
both3<-merge(a19,b19,by=c("SERNUM","BENUNIT","PERSON"),all.x=F,all.y=F)

names(both3)<-tolower(names(both3))
```

2.1 Benefit frequencies

Gives the grossed frequency, percentage, cumulative frequency and cumulative percentage of each benefit

```
tmptab<-xtabs(gross4~benefit,both3,drop.unused.levels=T)
cbind(
"Freq"= tmptab,
"Pct"=round(100*prop.table(tmptab),1),
"Cum. freq."=cumsum(tmptab),
"Cum. pct"= round(cumsum(100*prop.table(tmptab)),2)
)
```

```
##         Freq  Pct Cum. freq. Cum. pct
## 1    1003954  2.1    1003954     2.08
## 2     816758  1.7    1820712     3.77
## 3    6260230 12.9    8080942    16.71
## 4    1128441  2.3    9209383    19.05
## 5   11302238 23.4   20511621    42.42
## 6      57256  0.1   20568877    42.54
## 8      58179  0.1   20627056    42.66
## 9     114847  0.2   20741903    42.90
## 10     15110  0.0   20757013    42.93
## 12    814195  1.7   21571208    44.62
## 13    662945  1.4   22234153    45.99
## 14    241009  0.5   22475162    46.49
## 15    152041  0.3   22627203    46.80
## 16   1244643  2.6   23871846    49.37
## 19    378065  0.8   24249911    50.16
## 21     13961  0.0   24263872    50.19
## 22     28481  0.1   24292353    50.24
## 24     10723  0.0   24303076    50.27
## 31     10803  0.0   24313879    50.29
## 32      1771  0.0   24315650    50.29
## 33     60499  0.1   24376149    50.42
## 34      7752  0.0   24383901    50.43
## 35     12546  0.0   24396447    50.46
## 36      9893  0.0   24406340    50.48
## 37      5510  0.0   24411850    50.49
## 61      6878  0.0   24418728    50.51
## 62  11479871 23.7   35898599    74.25
```

```
## 65      44592  0.1   35943191    74.34
## 66      67486  0.1   36010677    74.48
## 69      92289  0.2   36102966    74.67
## 70     130467  0.3   36233433    74.94
## 78      10190  0.0   36243623    74.96
## 81       5167  0.0   36248790    74.97
## 82      14097  0.0   36262887    75.00
## 83       5313  0.0   36268200    75.01
## 90    1035774  2.1   37303974    77.16
## 91    1870822  3.9   39174796    81.03
## 94    2860448  5.9   42035244    86.94
## 95    1376924  2.8   43412168    89.79
## 96    1577237  3.3   44989405    93.05
## 97    1176666  2.4   46166071    95.49
## 98      50744  0.1   46216815    95.59
## 99     349210  0.7   46566025    96.31
## 102     45448  0.1   46611473    96.41
## 103    136546  0.3   46748019    96.69
## 104     54263  0.1   46802282    96.80
## 105     28931  0.1   46831213    96.86
## 106     42498  0.1   46873711    96.95
## 107     18598  0.0   46892309    96.99
## 108     36122  0.1   46928431    97.06
## 109     95016  0.2   47023447    97.26
## 110     77910  0.2   47101357    97.42
## 111    381552  0.8   47482909    98.21
## 112    191637  0.4   47674546    98.61
## 113    618294  1.3   48292840    99.89
## 114     55232  0.1   48348072   100.00
```

2.2 Distribution of Universal Credit, State Pension , Personal Independence Payment (Daily Living), Personal Independence Payment (PIP) Mobility

Examines the distribution of grossed Universal Credit benefit, State Pension benefit amounts, Personal Independence Payment (Daily Living benefit), Personal Independence Payment (PIP) Mobility benefit amounts, including an assessment of normality and discovery of outliers

```
round(t(both3%>%filter(benefit==95  | benefit==5 | benefit==96 | benefit==97)%>%
      group_by(benefit)%>%summarise(
  "Observations"=length(benamt),
  "Mean"=wtd.mean(benamt,gross4),
  "Median"=wtd.quantile(benamt,gross4,probs=.5),
  "Std Deviation"=sqrt(wtd.var(benamt,gross4)),
  "Variance"=wtd.var(benamt,gross4),
  "Mode"=as.numeric(names(sort(-table(benamt)))[1]),
  "Range"=max(benamt)-min(benamt),
  "IQR"=wtd.quantile(benamt,gross4,probs=.75)-wtd.quantile(benamt,gross4,probs=.25),
"Skewness"=Weighted.Desc.Stat::w.skewness(benamt,gross4),
"Kurtosis"=Weighted.Desc.Stat::w.kurtosis(benamt,gross4)
)),2) #)
```

```
##                   [,1]    [,2]    [,3]   [,4]
## benefit           5.00   95.00   96.00  97.00
## Observations   9055.00  919.00 1204.00 905.00
## Mean            150.15  162.92   71.43  44.93
```

```
## Median         150.38  151.03   58.70   61.20
## Std Deviation   46.76   95.73   14.37   18.84
## Variance      2186.20 9164.64  206.47  355.08
## Mode           150.00    0.00   58.70   61.20
## Range          433.76  507.00   28.95   96.70
## IQR             45.65  148.92   28.95   38.00
## Skewness         0.14    0.50    0.24   -0.27
## Kurtosis         1.33    0.00   -1.94   -1.86
```

Weighted quantiles

```
#knitr::kable(
### Computing the results first, and storing them in tmp
tmp<-both3%>%filter(benefit==95| benefit==5 | benefit==96 | benefit==97) %>%
  group_by(benefit)%>% summarise(
 "ben#"=wtd.quantile(benamt,gross4,probs=c(.01,.05,.1,.25,.5,.75,.9,.95,.99))
) #)

### tmp is a long table with each set of nine quantiles
### added one after the other (36 rows).
### The code below makes it easier to read

data.frame(
  "Quantile"=c(.01,.05,.1,.25,.5,.75,.9,.95,.99),
  "UC"=      tmp[tmp$benefit==95,2],
  "Pension"= tmp[tmp$benefit==5,2],
  "PIP (DL)"=tmp[tmp$benefit==96,2],
  "PIP (M)"= tmp[tmp$benefit==97,2])
```

```
##   Quantile      ben.   ben..1 ben..2 ben..3
## 1     0.01   0.00000  34.0500  58.70   23.2
## 2     0.05  28.76197  75.0000  58.70   23.2
## 3     0.10  46.02740  80.8000  58.70   23.2
## 4     0.25  79.17344 129.2000  58.70   23.2
## 5     0.50 151.03016 150.3775  58.70   61.2
## 6     0.75 228.09213 174.8500  87.65   61.2
## 7     0.90 287.06885 204.4300  87.65   61.2
## 8     0.95 326.41803 227.6700  87.65   61.2
## 9     0.99 428.95082 279.7500  87.65   61.2
```

Extreme values

We have chosen to collate the results with *cbind()* in this case, but each part can be run separately if needed.

```
#knitr::kable(
round(
cbind(
    both3%>%filter(benefit==95 | benefit==5 | benefit==96 | benefit==97)%>%
      select(sernum,person,benamt,benefit)%>%
  group_by(benefit)%>%slice_min(order_by = benamt,n=5,with_ties = FALSE),
  both3%>%filter(benefit==95 | benefit==5 | benefit==96 | benefit==97)%>%
    select(sernum,person,benamt,benefit)%>%
    group_by(benefit)%>%slice_max(order_by = benamt,n=5,with_ties = FALSE)
)
  ,1)#)
```

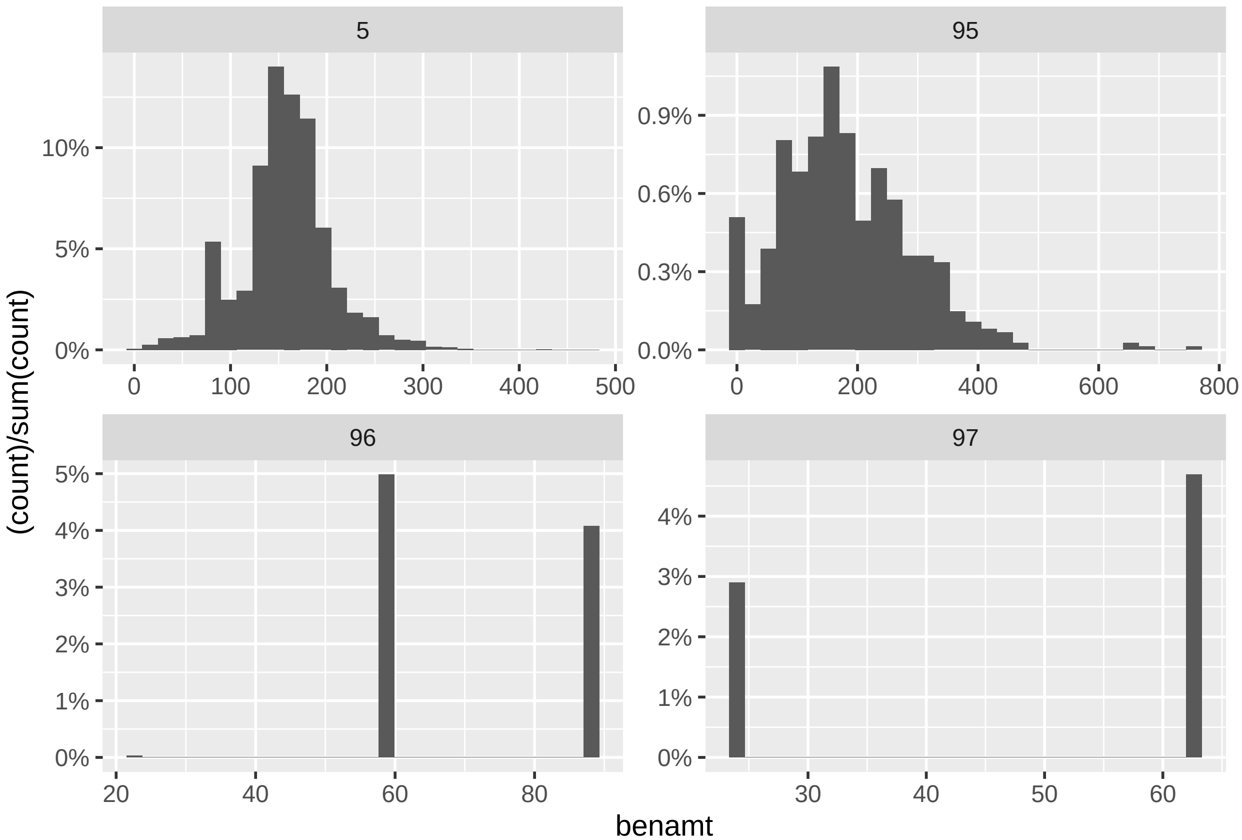
```
## # A tibble: 20 x 8
```

```
##    sernum...1 person...2 benamt...3 benefit...4 sernum~1 perso~2 benam~3 benef~4
##         <dbl>      <dbl>      <dbl>       <dbl>    <dbl>   <dbl>   <dbl>   <dbl>
##  1       9257          3          0           5     2727       2   434.        5
##  2       3302          2          0.1         5    13813       1   388         5
##  3      10263          2          0.2         5    17339       1   380.        5
##  4      12153          1          0.2         5    17521       1   372.        5
##  5      14864          1          0.8         5     2544       1   354         5
##  6      10810          5          0          95    14585       2   507        95
##  7      11967          1          0          95    14561       1   483        95
##  8      12635          2          0          95    10708       1   471        95
##  9      12884          1          0          95    10705       1   470.       95
## 10      15995          1          0          95     7993       1   452.       95
## 11      10018          1         58.7        96    10019       2    87.7      96
## 12      10023          1         58.7        96     1002       1    87.7      96
## 13      10050          1         58.7        96    10031       1    87.7      96
## 14      10081          1         58.7        96    10035       1    87.7      96
## 15      10158          1         58.7        96    10096       1    87.7      96
## 16      10018          1         23.2        97    11765       1   120.       97
## 17      10019          2         23.2        97    10035       1    61.2      97
## 18       1002          1         23.2        97    10096       1    61.2      97
## 19      10031          1         23.2        97    10098       1    61.2      97
## 20      10050          1         23.2        97    10123       1    61.2      97
## # ... with abbreviated variable names 1: sernum...5, 2: person...6,
## #   3: benamt...7, 4: benefit...8
```

2.3 Histograms

Universal credit

```
ggplot(both1%>%
      filter(benefit==95 | benefit==5 | benefit==96 | benefit==97),
      aes(x=benamt))+ ### Base plot specification
geom_histogram(aes(y = (..count..)/sum(..count..)))+ ### Converts y axis to percent
scale_y_continuous(labels = scales::percent)+ ### Prints y axis as percent
facet_wrap(~benefit,scale="free")  ### Each histogram has its own plot
```



2.4 Create Both2 – Distribution of Universal Credit ex. £0

(*Note There is no need for this step in R, we can keep on working with both3*)

Examines the distribution of grossed Universal Credit benefit amounts excluding £0 observations, including an assessment of normality and discovery of outliers

```
round(t(both3%>%filter(benefit==95 & benamt>0)%>%summarise(
  "Observations"=length(benamt),
  "Mean"=wtd.mean(benamt,gross4),
  "Median"=wtd.quantile(benamt,gross4,probs=.5),
  "Std Deviation"=sqrt(wtd.var(benamt,gross4)),
  "Variance"=wtd.var(benamt,gross4),
  "Mode"=as.numeric(names(sort(-table(benamt)))[1]),
  "Range"=max(benamt)-min(benamt),
  "IQR"=wtd.quantile(benamt,gross4,probs=.75)-wtd.quantile(benamt,gross4,probs=.25),
"Skewness"=Weighted.Desc.Stat::w.skewness(benamt,gross4),
"Kurtosis"=Weighted.Desc.Stat::w.kurtosis(benamt,gross4)
)),2)
```

```
##                    [,1]
## Observations     902.00
## Mean             166.64
## Median           155.15
## Std Deviation     93.56
## Variance        8754.38
## Mode              72.94
## Range            506.71
```

```
## IQR            138.39
## Skewness         0.55
## Kurtosis         0.05
```

Weighted quantiles

```
both3%>%filter(benefit==95 & benamt>0)%>%select(benamt,gross4)%>%summarise(
  "Quantiles"=
round(wtd.quantile(benamt,gross4,probs=c(.01,.05,.1,.25,.5,.75,.9,.95,.99)),2)
)
```

```
##   Quantiles
## 1      6.56
## 2     38.84
## 3     52.93
## 4     92.05
## 5    155.15
## 6    230.45
## 7    288.46
## 8    326.48
## 9    428.95
```

Extreme values

```
cbind(
  both3%>%filter(benefit==95 & benamt>0)%>%select(sernum,benunit,person,benamt)%>%
  slice_min(order_by = benamt,n=5,with_ties = FALSE),
  both3%>%filter(benefit==95 & benamt>0)%>%select(sernum,benunit,person,benamt)%>%
    slice_max(order_by = benamt,n=5,with_ties = FALSE)
)
```

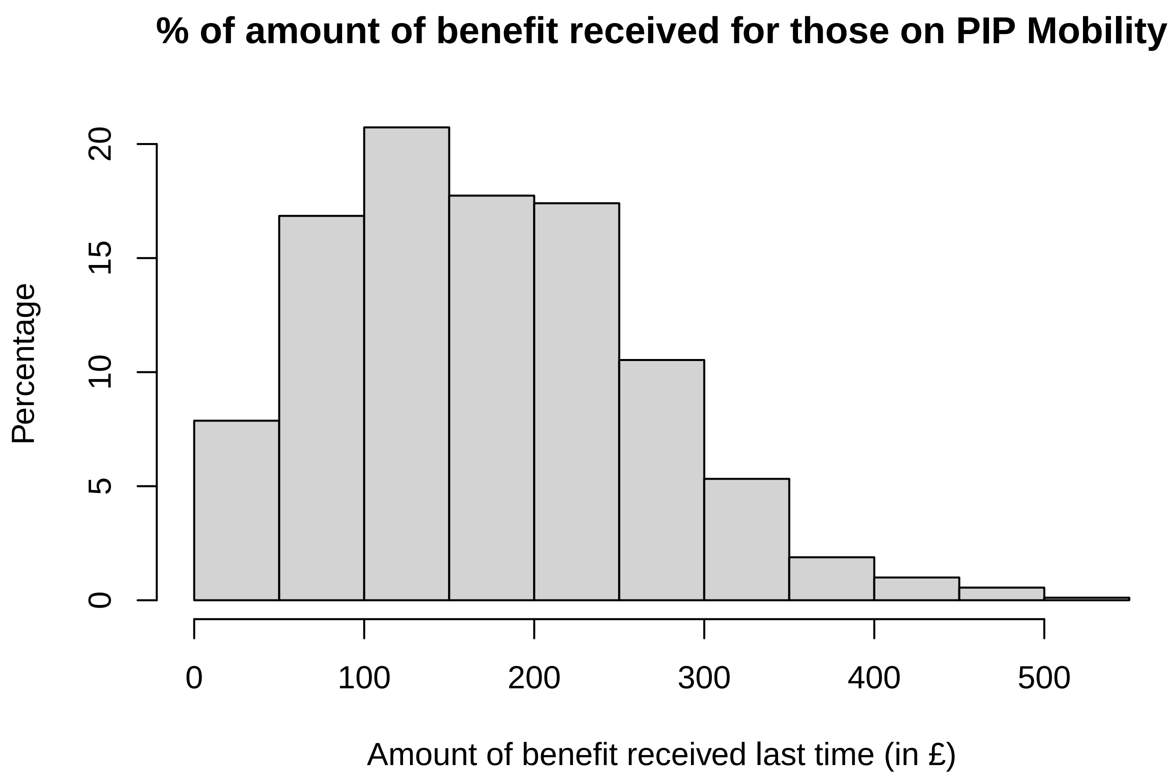
```
##   sernum benunit person    benamt sernum benunit person   benamt
## 1   5504       1      1 0.2853699  14585       1      2 507.0000
## 2   2748       1      1 0.3544110  14561       1      1 482.9564
## 3  14014       1      1 0.4268852  10708       1      1 470.9692
## 4  16360       2      2 1.1429508  10705       1      1 470.5400
## 5   5746       1      2 1.4504918   7993       1      1 451.7502
```

** Histogram of Universal Credit (without 0s)**

```
z<-hist(both3$benamt[both3$benefit==95 & both3$benamt>0],freq=F)
```

```
z$density <- z$counts/sum(z$counts) * 100 ### Converting y axis into percentages
```

```
plot(z, freq = FALSE,
     main= '% of amount of benefit received for those on PIP Mobility',
     xlab="Amount of benefit received last time (in £)",
     ylab="Percentage"
)
```



2.5 UC and tenure

Create tenure4 Create a new dataset called 'uc_tenure4' from merging the 'uc_tenure1'(ie both3) dataset and FRS household dataset by their common variable (sernum)**

```
setwd("/home/piet/Dropbox/work/UKDS/FRS workshop/UKDA-8948-spss/spss/spss25" )
h21<-read.spss("househol.sav",to.data.frame = T, use.value.labels = T,
add.undeclared.levels="append")%>%select(SERNUM,TENURE)
names(h21)<-tolower(names(h21))
uc_tenure4<-merge(both3,h21,by=c("sernum"),all.x=F,all.y=F)
levels(uc_tenure4$tenure)[2]<-"Bought with mortgage"
```

Frequency of Tenure

Gives the grossed frequency, percentage, cumulative frequency and cumulative percentage of each category for the variable 'tenure' for those in receipt of Universal Credit

```
tmptab<-xtabs(gross4~tenure,uc_tenure4,subset=benefit==95,drop.unused.levels=T)
cbind(
"Freq"= tmptab,
"Pct"=round(100*prop.table(tmptab),1),
"Cum. freq."=cumsum(tmptab),
"Cum. pct"= round(cumsum(100*prop.table(tmptab)),2)
)
```

```
##                        Freq  Pct Cum. freq. Cum. pct
## Owns it outright     324217 46.0     324217    45.99
## Bought with mortgage 176384 25.0     500601    71.02
## Part own, part rent    1142  0.2     501743    71.18
## Rents                198607 28.2     700350    99.35
## Rent-free              4561  0.6     704911   100.00
```

3 Statistically significant differences between 2019-20 and 2020-21

3.1 Create Combine_years

Create a new appended dataset called 'combine_years' that contains the 'ad_ben_1920' dataset and 'ad_ben_2021' dataset (ie Both1 and Both3) A new variable called 'year' is created that assigns '1920' to the case if the year is 2019-20 and '2021' if the year is 2020-21

```
combine_years<-rbind(
cbind(both1,year=2021),
cbind(both3,year=1920)
)
```

[Provides grossed analysis of variance and Kruskal-Wallis for Universal Credit benefit amounts by year]

3.2 ANOVA UC amount by year

```
summary(lm(benamt~as.factor(year),data=combine_years%>%filter(benefit==95),weights=gross4))
```

```
##
## Call:
## lm(formula = benamt ~ as.factor(year), data = combine_years %>%
##     filter(benefit == 95), weights = gross4)
##
## Weighted Residuals:
##      Min       1Q   Median       3Q      Max
## -18448.5  -3073.7   -438.3   2896.9  26313.4
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          162.921      4.579  35.582  < 2e-16 ***
## as.factor(year)2021   17.938      5.601   3.203  0.00139 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5373 on 1559 degrees of freedom
## Multiple R-squared:  0.006537,   Adjusted R-squared:  0.0059
## F-statistic: 10.26 on 1 and 1559 DF,  p-value: 0.001388
```

```
### We need to use the Survey package in order to access the weighted rank test functions
### (Wilcoxon/Kruskal-Wallis)
library(survey)
```

```
### We also need to declare the survey design, in this case we will only specify the
### grossing weights
design <- svydesign(ids = ~0, data=combine_years%>%filter(benefit==95),
weights=combine_years%>%filter(benefit==95)%>%select(gross4))

svyranktest(benamt ~ as.factor(year), design = design)
```

```
##
##  Design-based KruskalWallis test
##
## data:  benamt ~ as.factor(year)
## t = 2.7011, df = 1559, p-value = 0.006987
## alternative hypothesis: true difference in mean rank score is not equal to 0
```

```
## sample estimates:
## difference in mean rank score
##                    0.04835153
```

3.3 ANOVA State pension benefit amount by year

[Provides grossed analysis of variance and Kruskal-Wallis Rank Sums for State Pension benefit amounts by year]

```
summary(lm(benamt~as.factor(year),data=combine_years%>%filter(benefit==5),weights=gross4))
```

```
##
## Call:
## lm(formula = benamt ~ as.factor(year), data = combine_years %>%
##     filter(benefit == 5), weights = gross4)
##
## Weighted Residuals:
##      Min       1Q   Median       3Q      Max
## -14600.3   -776.5     -4.8    818.9  30003.1
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)         150.1536     0.5573 269.438   <2e-16 ***
## as.factor(year)2021   7.5813     0.7907   9.589   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1874 on 14626 degrees of freedom
## Multiple R-squared:  0.006247,   Adjusted R-squared:  0.006179
## F-statistic: 91.94 on 1 and 14626 DF,  p-value: < 2.2e-16
```

```
design <- svydesign(ids = ~0, data=combine_years%>%filter(benefit==5),
weights=combine_years%>%filter(benefit==5)%>%select(gross4))

svyranktest(benamt ~ as.factor(year), design = design)
```

```
##
##  Design-based KruskalWallis test
##
## data:  benamt ~ as.factor(year)
## t = 8.5945, df = 14626, p-value < 2.2e-16
## alternative hypothesis: true difference in mean rank score is not equal to 0
## sample estimates:
## difference in mean rank score
##                    0.05344663
```

3.4 ANOVA PIP Daily Living

[Provides grossed analysis of variance and Kruskal-Wallis Rank Sums for Personal Independence Payment (PIP) Daily Living benefit amounts by year]

```
summary(lm(benamt~as.factor(year),data=combine_years%>%filter(benefit==96),weights=gross4))
```

```
##
## Call:
## lm(formula = benamt ~ as.factor(year), data = combine_years %>%
```

```
##     filter(benefit == 96), weights = gross4)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -5017.9  -490.6  -319.5   590.1  1871.6
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          71.4256     0.5296 134.861   <2e-16 ***
## as.factor(year)2021   1.3692     0.6956   1.968   0.0492 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 665.1 on 1880 degrees of freedom
## Multiple R-squared:  0.002057,   Adjusted R-squared:  0.001526
## F-statistic: 3.875 on 1 and 1880 DF,  p-value: 0.04917
```

```
design <- svydesign(ids = ~0, data=combine_years%>%filter(benefit==96),
weights=combine_years%>%filter(benefit==96)%>%select(gross4))

svyranktest(benamt ~ as.factor(year), design = design)
```

```
##
##  Design-based KruskalWallis test
##
## data:  benamt ~ as.factor(year)
## t = 16.83, df = 1880, p-value < 2.2e-16
## alternative hypothesis: true difference in mean rank score is not equal to 0
## sample estimates:
## difference in mean rank score
##                     0.2546178
```

3.5 ANOVA PIP mobility

/Provides grossed analysis of variance and Kruskal-Wallis Rank Sums for Personal Independence Payment (PIP) Mobility benefit amounts by year/

```
summary(lm(benamt~as.factor(year),data=combine_years%>%filter(benefit==97),weights=gross4))
```

```
##
## Call:
## lm(formula = benamt ~ as.factor(year), data = combine_years %>%
##     filter(benefit == 97), weights = gross4)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -2361.6  -790.0   413.2   625.3  1773.5
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          44.9285     0.7801  57.595   <2e-16 ***
## as.factor(year)2021   1.8233     1.0068   1.811   0.0704 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 846.2 on 1469 degrees of freedom
```

```
## Multiple R-squared:  0.002227,   Adjusted R-squared:  0.001548
## F-statistic: 3.279 on 1 and 1469 DF,  p-value: 0.07036
```

```
design <- svydesign(ids = ~0, data=combine_years%>%filter(benefit==97),
weights=combine_years%>%filter(benefit==97)%>%select(gross4))

svyranktest(benamt ~ as.factor(year), design = design)
```

```
##
##  Design-based KruskalWallis test
##
## data:  benamt ~ as.factor(year)
## t = 16.198, df = 1469, p-value < 2.2e-16
## alternative hypothesis: true difference in mean rank score is not equal to 0
## sample estimates:
## difference in mean rank score
##                      0.268868
```

4 Wellbeing variables

4.1 Comparison between 2019-20 and 2020-21

Creating a 'combine_years'dataset from the merged individual and households datasets from 2019-20 and 2020-21

```
setwd("/home/piet/Dropbox/work/UKDS/FRS workshop/")
combine_years<-merge(
rbind(
cbind(read.spss("UKDA-8948-spss/spss/spss25/adult.sav",to.data.frame = T,
use.value.labels = F)%>%
select(SERNUM,BENUNIT,PERSON,GROSS4,DISD07,HAPPYWB,ANXIOUS,LIFESAT,MEANING),year=1920),

cbind(read.spss("UKDA-8802-spss/spss/spss25/adult.sav",to.data.frame = T,
use.value.labels = F )%>%
select(SERNUM,BENUNIT,PERSON,GROSS4,DISD07,HAPPYWB,ANXIOUS,LIFESAT,MEANING),year=2021)
),

t<-rbind(
cbind(read.spss("UKDA-8948-spss/spss/spss25/househol.sav",to.data.frame = T,
use.value.labels = F)%>%
select(SERNUM,HHINCBND),year=1920),

cbind(read.spss("UKDA-8802-spss/spss/spss25/househol.sav",to.data.frame = T,
use.value.labels = F )%>%
select(SERNUM,HHINCBND),year=2021)
),

by=c("SERNUM","year"),all.x=F,all.y=F)

names(combine_years)<-tolower(names(combine_years))
```

Gives the grossed frequency, percentage, cumulative frequency and cumulative percentage of each category for each wellbeing variable Happiness 2019-20 and 2020-21

```
tmptab<-xtabs(gross4~happywb+year,combine_years) ### creates the table

### Computes Frequencies and percentages
  cbind(
"Freq"= tmptab,
"Pct"=round(100*prop.table(tmptab),2)
  )
```

```
##       1920    2021  1920  2021
## 0   354257  410207  0.47  0.54
## 1   304245  241372  0.40  0.32
## 2   578424  636717  0.76  0.84
## 3  1117826  931021  1.47  1.22
## 4  1386922 1290415  1.82  1.69
## 5  3269500 3114612  4.29  4.09
## 6  3617610 3109194  4.75  4.08
## 7  6483098 6175554  8.51  8.11
## 8  8905319 9858331 11.69 12.95
## 9  5467150 6359517  7.18  8.35
## 10 5653133 6882457  7.42  9.04
```

```r
#### Cumulated frequencies and percentages
cbind(
"Cum. freq."=cumsum(as.data.frame.matrix(tmptab)),
"Cum. pct"= round(cumsum(as.data.frame.matrix(100*prop.table(tmptab,2))),1)
)
```

```
##    Cum. freq..1920 Cum. freq..2021 Cum. pct.1920 Cum. pct.2021
## 0           354257          410207           1.0           1.1
## 1           658502          651579           1.8           1.7
## 2          1236926         1288296           3.3           3.3
## 3          2354752         2219317           6.3           5.7
## 4          3741674         3509732          10.1           9.0
## 5          7011174         6624344          18.9          17.0
## 6         10628784         9733538          28.6          25.0
## 7         17111882        15909092          46.1          40.8
## 8         26017201        25767423          70.1          66.1
## 9         31484351        32126940          84.8          82.4
## 10        37137484        39009397         100.0         100.0
```

Anxiety 2019-20

```r
tmptab<-xtabs(gross4~anxious+year,combine_years) ### creates the table

### Computes Frequencies and percents
  cbind(
"Freq"= tmptab,
"Pct"=round(100*prop.table(tmptab),2)
  )
```

```
##        1920     2021  1920  2021
## 0  11311437 13180739 14.86 17.31
## 1   3354055  3680690  4.41  4.83
## 2   5132901  5117912  6.74  6.72
## 3   3184121  3279182  4.18  4.31
## 4   2344513  2302439  3.08  3.02
## 5   3824487  3684135  5.02  4.84
## 6   2184132  2000579  2.87  2.63
## 7   2162332  2176659  2.84  2.86
## 8   2051803  1996550  2.69  2.62
## 9    843754   807702  1.11  1.06
## 10   730299   783908  0.96  1.03
```

```r
#### Cumulated frequencies and percentages
cbind(
"Cum. freq."=cumsum(as.data.frame.matrix(tmptab)),
"Cum. pct"= round(cumsum(as.data.frame.matrix(100*prop.table(tmptab,2))),1)
)
```

```
##    Cum. freq..1920 Cum. freq..2021 Cum. pct.1920 Cum. pct.2021
## 0         11311437        13180739          30.5          33.8
## 1         14665492        16861429          39.5          43.2
## 2         19798393        21979341          53.3          56.3
## 3         22982514        25258523          61.9          64.7
## 4         25327027        27560962          68.2          70.7
## 5         29151514        31245097          78.5          80.1
## 6         31335646        33245676          84.4          85.2
```

```
## 7          33497978        35422335           90.2           90.8
## 8          35549781        37418885           95.8           95.9
## 9          36393535        38226587           98.0           98.0
## 10         37123834        39010495          100.0          100.0
```

Life satisfaction 2019-20

```
tmptab<-xtabs(gross4~lifesat+year,combine_years) ### creates the table

### Computes Frequencies and percents
  cbind(
"Freq"= tmptab,
"Pct"=round(100*prop.table(tmptab),2)
  )
```

```
##         1920     2021  1920  2021
## 0     307607   303214  0.40  0.40
## 1     318731   358313  0.42  0.47
## 2     350467   372074  0.46  0.49
## 3     736660   574953  0.97  0.76
## 4    1068417   829821  1.40  1.09
## 5    3147252  2865933  4.13  3.76
## 6    3417905  2851165  4.49  3.74
## 7    7315920  6909701  9.61  9.08
## 8   10890719 12414550 14.30 16.31
## 9    4875114  5991986  6.40  7.87
## 10   4694164  5544646  6.17  7.28
```

```
#### Cumulated frequencies and percentages
cbind(
"Cum. freq."=cumsum(as.data.frame.matrix(tmptab)),
"Cum. pct"= round(cumsum(as.data.frame.matrix(100*prop.table(tmptab,2))),1)
)
```

```
##    Cum. freq..1920 Cum. freq..2021 Cum. pct.1920 Cum. pct.2021
## 0           307607          303214           0.8           0.8
## 1           626338          661527           1.7           1.7
## 2           976805         1033601           2.6           2.6
## 3          1713465         1608554           4.6           4.1
## 4          2781882         2438375           7.5           6.2
## 5          5929134         5304308          16.0          13.6
## 6          9347039         8155473          25.2          20.9
## 7         16662959        15065174          44.9          38.6
## 8         27553678        27479724          74.2          70.4
## 9         32428792        33471710          87.4          85.8
## 10        37122956        39016356         100.0         100.0
```

Life Meaningfulness 2019-20 and 2020-21

```
tmptab<-xtabs(gross4~meaning+year,combine_years)

  cbind(
"Freq"= tmptab,
"Pct"=round(100*prop.table(tmptab),2)
  )
```

```
##         1920     2021  1920  2021
```

```
## 0    192049   233193  0.25  0.31
## 1     98069   114816  0.13  0.15
## 2    286656   293746  0.38  0.39
## 3    458307   505038  0.60  0.66
## 4    797482   631245  1.05  0.83
## 5   2703368  2440702  3.56  3.21
## 6   2583599  2554497  3.40  3.36
## 7   6589179  6468443  8.67  8.51
## 8  10907043 11573508 14.34 15.22
## 9   6039536  6664005  7.94  8.76
## 10  6445424  7458907  8.48  9.81
```

```
#### Cumulated frequencies and percentages
cbind(
"Cum. freq."=cumsum(as.data.frame.matrix(tmptab)),
"Cum. pct"= round(cumsum(as.data.frame.matrix(100*prop.table(tmptab,2))),1)
)
```

```
##    Cum. freq..1920 Cum. freq..2021 Cum. pct.1920 Cum. pct.2021
## 0           192049          233193           0.5           0.6
## 1           290118          348009           0.8           0.9
## 2           576774          641755           1.6           1.6
## 3          1035081         1146793           2.8           2.9
## 4          1832563         1778038           4.9           4.6
## 5          4535931         4218740          12.2          10.8
## 6          7119530         6773237          19.2          17.4
## 7         13708709        13241680          36.9          34.0
## 8         24615752        24815188          66.3          63.7
## 9         30655288        31479193          82.6          80.8
## 10        37100712        38938100         100.0         100.0
```

Examines the distribution of happiness in 2019-20 and 2020-21, including an assessment of normality and discovery of outliers

```
round(t(combine_years%>%filter(!is.na(happywb))%>%group_by(year)%>%summarise(
  "Observations"=length(happywb),
  "Mean"=wtd.mean(happywb,gross4),
  "Median"=wtd.quantile(happywb,gross4,probs=.5),
  "Std Deviation"=sqrt(wtd.var(happywb,gross4)),
  "Variance"=wtd.var(happywb,gross4),
  "Mode"=as.numeric(names(sort(-table(happywb)))[1]),
  "Range"=max(happywb)-min(happywb),
  "IQR"=wtd.quantile(happywb,gross4,probs=.75)-wtd.quantile(happywb,gross4,probs=.25),
"Skewness"=Weighted.Desc.Stat::w.skewness(happywb,gross4),
"Kurtosis"=Weighted.Desc.Stat::w.kurtosis(happywb,gross4)
)),2)
```

```
##                   [,1]     [,2]
## year           1920.00  2021.00
## Observations  12923.00 26262.00
## Mean              7.29     7.48
## Median            8.00     8.00
## Std Deviation     2.13     2.12
## Variance          4.53     4.49
## Mode              8.00     8.00
## Range            10.00    10.00
```

```
## IQR                3.00     2.00
## Skewness          -0.95    -1.09
## Kurtosis           0.84     1.20
```

Weighted quantiles

```
cbind(
combine_years%>%filter(year==1920 & !is.na(happywb))%>%select(happywb,gross4)%>%summarise(
  "Quantiles"=
round(wtd.quantile(happywb,gross4,probs=c(.01,.05,.1,.25,.5,.75,.9,.95,.99)),2)
),
combine_years%>%filter(year==1920 & !is.na(happywb))%>%select(happywb,gross4)%>%summarise(
  "Quantiles"=
round(wtd.quantile(happywb,gross4,probs=c(.01,.05,.1,.25,.5,.75,.9,.95,.99)),2)
)
)
```

```
##   Quantiles Quantiles
## 1         1         1
## 2         3         3
## 3         4         4
## 4         6         6
## 5         8         8
## 6         9         9
## 7        10        10
## 8        10        10
## 9        10        10
```

Extreme values

```
cbind(
  combine_years%>%filter(year==1920 & !is.na(happywb))%>%select(happywb)%>%
  slice_min(order_by = happywb,n=5,with_ties = FALSE),
  combine_years%>%filter(year==2021 & !is.na(happywb))%>%select(happywb)%>%
    slice_max(order_by = happywb,n=5,with_ties = FALSE)
)
```

```
##   happywb happywb
## 1       0      10
## 2       0      10
## 3       0      10
## 4       0      10
## 5       0      10
```

Examines the distribution of anxiety in 2019-20 and 2020-21, including an assessment of normality and discovery of outliers

```
round(t(combine_years%>%filter(!is.na(anxious))%>%group_by(year)%>%summarise(
  "Observations"=length(anxious),
  "Mean"=wtd.mean(anxious,gross4),
  "Median"=wtd.quantile(anxious,gross4,probs=.5),
  "Std Deviation"=sqrt(wtd.var(anxious,gross4)),
  "Variance"=wtd.var(anxious,gross4),
  "Mode"=as.numeric(names(sort(-table(anxious)))[1]),
  "Range"=max(anxious)-min(anxious),
  "IQR"=wtd.quantile(anxious,gross4,probs=.75)-wtd.quantile(anxious,gross4,probs=.25),
"Skewness"=Weighted.Desc.Stat::w.skewness(anxious,gross4),
```

```r
"Kurtosis"=Weighted.Desc.Stat::w.kurtosis(anxious,gross4)
)),2)
```

```
##                    [,1]     [,2]
## year            1920.00  2021.00
## Observations   12920.00 26259.00
## Mean               3.00     2.81
## Median             2.00     2.00
## Std Deviation      2.89     2.88
## Variance           8.34     8.31
## Mode               0.00     0.00
## Range             10.00    10.00
## IQR                5.00     5.00
## Skewness           0.65     0.76
## Kurtosis          -0.71    -0.57
```

Weighted quantiles

```r
cbind(
combine_years%>%filter(year==1920 & !is.na(anxious))%>%select(anxious,gross4)%>%summarise(
  "Quantiles"=
round(wtd.quantile(anxious,gross4,probs=c(.01,.05,.1,.25,.5,.75,.9,.95,.99)),2)
),
combine_years%>%filter(year==1920 & !is.na(anxious))%>%select(anxious,gross4)%>%summarise(
  "Quantiles"=
round(wtd.quantile(anxious,gross4,probs=c(.01,.05,.1,.25,.5,.75,.9,.95,.99)),2)
)
)
```

```
##   Quantiles Quantiles
## 1         0         0
## 2         0         0
## 3         0         0
## 4         0         0
## 5         2         2
## 6         5         5
## 7         7         7
## 8         8         8
## 9        10        10
```

Extreme values

```r
cbind(
  combine_years%>%filter(year==1920 & !is.na(anxious))%>%select(anxious)%>%
  slice_min(order_by = anxious,n=5,with_ties = FALSE),
  combine_years%>%filter(year==2021 & !is.na(anxious))%>%select(anxious)%>%
    slice_max(order_by = anxious,n=5,with_ties = FALSE)
)
```

```
##   anxious anxious
## 1       0      10
## 2       0      10
## 3       0      10
## 4       0      10
## 5       0      10
```

Examines the distribution of life satisfaction in 2019-20 and 2020-21, including an assessment

of normality and discovery of outliers

```
round(t(combine_years%>%filter(!is.na(lifesat))%>%group_by(year)%>%summarise(
  "Observations"=length(lifesat),
  "Mean"=wtd.mean(lifesat,gross4),
  "Median"=wtd.quantile(lifesat,gross4,probs=.5),
  "Std Deviation"=sqrt(wtd.var(lifesat,gross4)),
  "Variance"=wtd.var(lifesat,gross4),
  "Mode"=as.numeric(names(sort(-table(lifesat)))[1]),
  "Range"=max(lifesat)-min(lifesat),
  "IQR"=wtd.quantile(lifesat,gross4,probs=.75)-wtd.quantile(lifesat,gross4,probs=.25),
"Skewness"=Weighted.Desc.Stat::w.skewness(lifesat,gross4),
"Kurtosis"=Weighted.Desc.Stat::w.kurtosis(lifesat,gross4)
)),2)
```

```
##                   [,1]     [,2]
## year           1920.00  2021.00
## Observations  12922.00 26266.00
## Mean              7.35     7.55
## Median            8.00     8.00
## Std Deviation     1.95     1.91
## Variance          3.79     3.65
## Mode              8.00     8.00
## Range            10.00    10.00
## IQR               3.00     2.00
## Skewness         -1.08    -1.27
## Kurtosis          1.65     2.28
```

Weighted quantiles

```
cbind(
combine_years%>%filter(year==1920 & !is.na(lifesat))%>%select(lifesat,gross4)%>%summarise(
  "Quantiles"=
round(wtd.quantile(lifesat,gross4,probs=c(.01,.05,.1,.25,.5,.75,.9,.95,.99)),2)
),
combine_years%>%filter(year==1920 & !is.na(lifesat))%>%select(lifesat,gross4)%>%summarise(
  "Quantiles"=
round(wtd.quantile(lifesat,gross4,probs=c(.01,.05,.1,.25,.5,.75,.9,.95,.99)),2)
)
)
```

```
##   Quantiles Quantiles
## 1         1         1
## 2         4         4
## 3         5         5
## 4         6         6
## 5         8         8
## 6         9         9
## 7        10        10
## 8        10        10
## 9        10        10
```

Extreme values

```
cbind(
  combine_years%>%filter(year==1920 & !is.na(lifesat))%>%select(lifesat)%>%
  slice_min(order_by = lifesat,n=5,with_ties = FALSE),
```

```
  combine_years%>%filter(year==2021 & !is.na(lifesat))%>%select(lifesat)%>%
    slice_max(order_by = lifesat,n=5,with_ties = FALSE)
)
```

```
##   lifesat lifesat
## 1       0      10
## 2       0      10
## 3       0      10
## 4       0      10
## 5       0      10
```

Examines the distribution of meaning in 2019-20 and 2020-21, including an assessment of normality and discovery of outliers

```
round(t(combine_years%>%filter(!is.na(meaning))%>%group_by(year)%>%summarise(
  "Observations"=length(meaning),
  "Mean"=wtd.mean(meaning,gross4),
  "Median"=wtd.quantile(meaning,gross4,probs=.5),
  "Std Deviation"=sqrt(wtd.var(meaning,gross4)),
  "Variance"=wtd.var(meaning,gross4),
  "Mode"=as.numeric(names(sort(-table(meaning)))[1]),
  "Range"=max(meaning)-min(meaning),
  "IQR"=wtd.quantile(meaning,gross4,probs=.75)-wtd.quantile(meaning,gross4,probs=.25),
"Skewness"=Weighted.Desc.Stat::w.skewness(meaning,gross4),
"Kurtosis"=Weighted.Desc.Stat::w.kurtosis(meaning,gross4)
)),2)
```

```
##                   [,1]     [,2]
## year           1920.00  2021.00
## Observations  12911.00 26219.00
## Mean              7.72     7.83
## Median            8.00     8.00
## Std Deviation     1.81     1.81
## Variance          3.28     3.28
## Mode              8.00     8.00
## Range            10.00    10.00
## IQR               2.00     2.00
## Skewness         -1.10    -1.23
## Kurtosis          1.83     2.33
```

Weighted quantiles

```
cbind(
combine_years%>%filter(year==1920 & !is.na(meaning))%>%select(meaning,gross4)%>%summarise(
  "Quantiles"=
round(wtd.quantile(meaning,gross4,probs=c(.01,.05,.1,.25,.5,.75,.9,.95,.99)),2)
),
combine_years%>%filter(year==1920 & !is.na(meaning))%>%select(meaning,gross4)%>%summarise(
  "Quantiles"=
round(wtd.quantile(meaning,gross4,probs=c(.01,.05,.1,.25,.5,.75,.9,.95,.99)),2)
)
)
```

```
##   Quantiles Quantiles
## 1         2         2
## 2         5         5
```

```
## 3          5          5
## 4          7          7
## 5          8          8
## 6          9          9
## 7         10         10
## 8         10         10
## 9         10         10
```

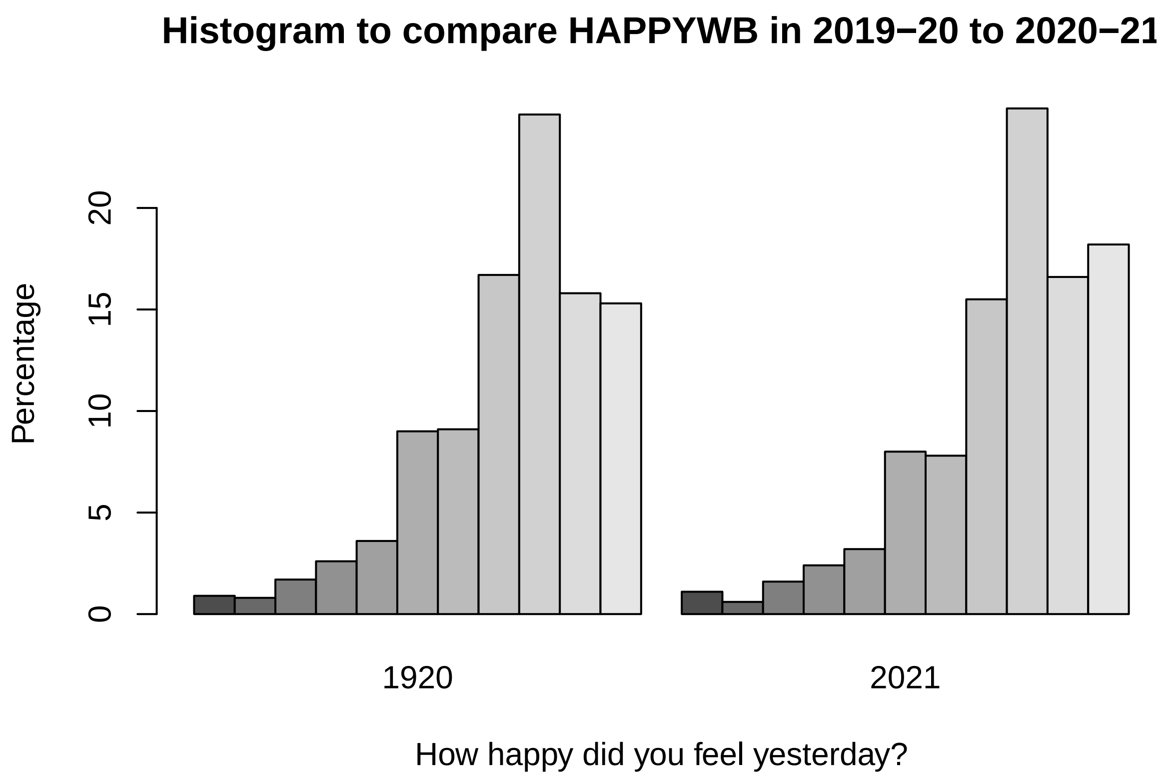
Extreme values

```
cbind(
  combine_years%>%filter(year==1920 & !is.na(meaning))%>%select(meaning)%>%
  slice_min(order_by = meaning,n=5,with_ties = FALSE),
  combine_years%>%filter(year==2021 & !is.na(meaning))%>%select(meaning)%>%
    slice_max(order_by = meaning,n=5,with_ties = FALSE)
)
```

```
##   meaning meaning
## 1       0      10
## 2       0      10
## 3       0      10
## 4       0      10
## 5       0      10
```

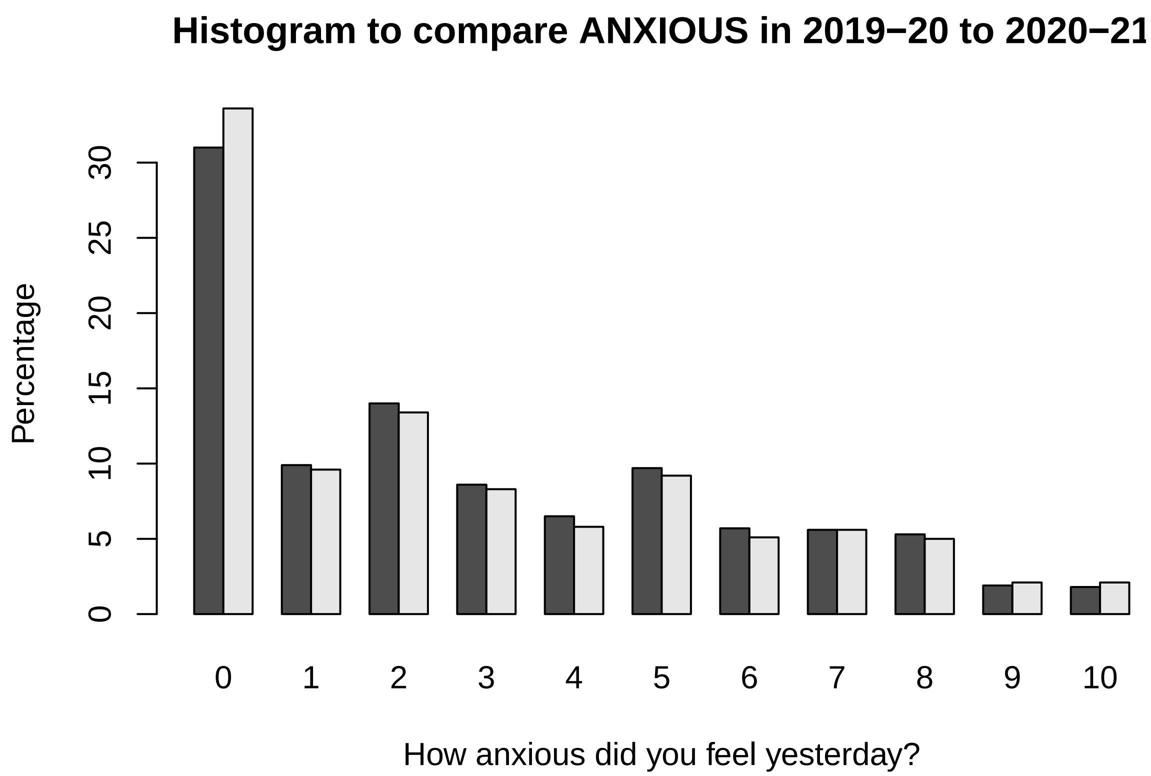
```
### First getting the happiness tables (we could have reused
### the frequency table we created  above)
t<-
round(                             ### Rounding result to 1 decimal
100*prop.table(                    ### Computing proportions, then multiplying by to get %
xtabs(~happywb+year, combine_years) ### basic frequency table
,2)
,1)

### The  plotting
barplot(t, beside = T,
     main="Histogram to compare HAPPYWB in 2019-20 to 2020-21",
     xlab="How happy did you feel yesterday?",
     ylab="Percentage"
     )
```



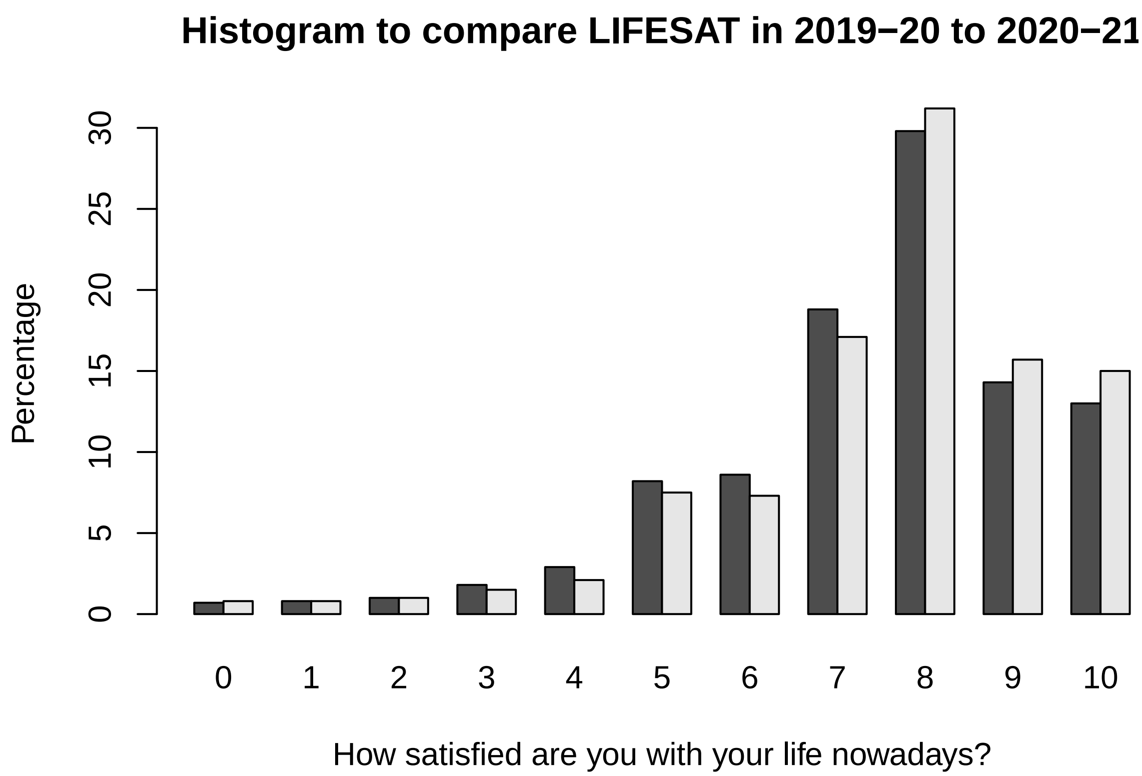

```
### First getting the anxiety  table
t<-
round(                              ### Rounding result to 1 decimal
100*prop.table(                     ### Computing proportions, then multiplying by to get %
xtabs(~year+anxious, combine_years) ### basic frequency table
,1)
,1)

### The actual plotting
barplot(t, beside = T,
     main="Histogram to compare ANXIOUS in 2019-20 to 2020-21",
     xlab="How anxious did you feel yesterday?",
     ylab="Percentage"
     )
```



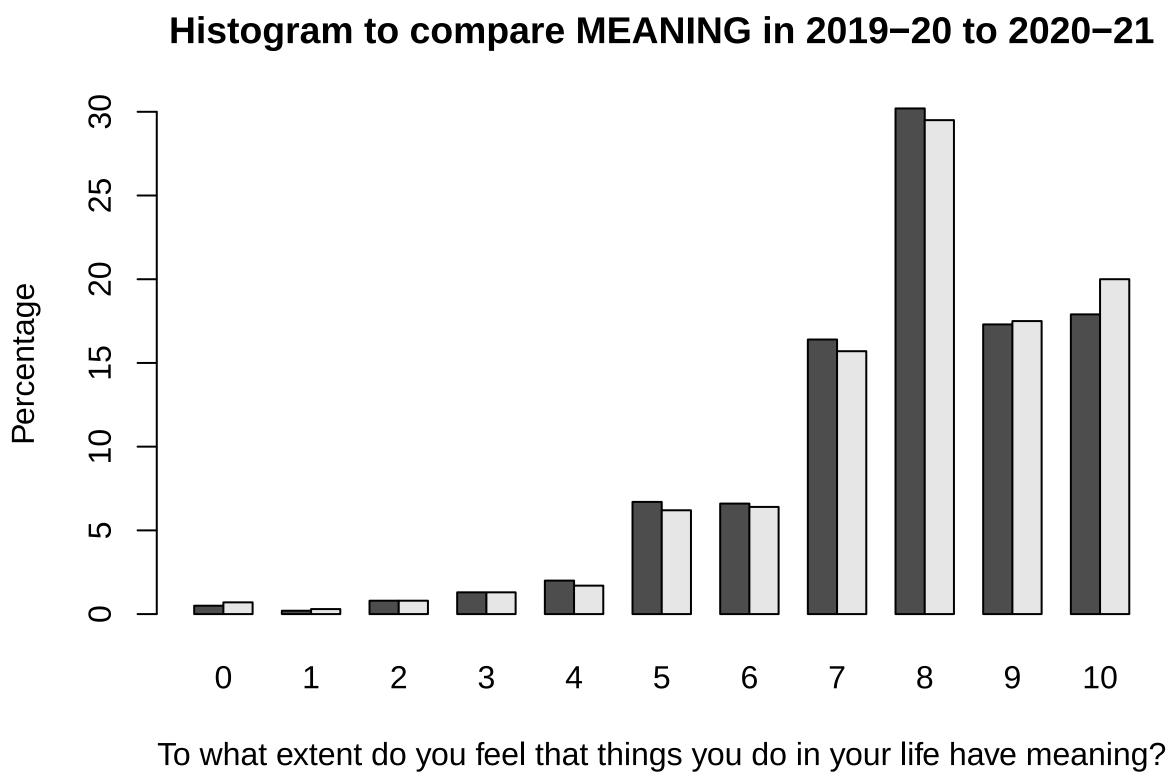

```
### Getting the life satisfaction table
t<-
round(                                 ### Rounding result to 1 decimal
100*prop.table(                        ### Computing proportions, then multiplying by to get %
xtabs(~year+lifesat, combine_years) ### basic frequency table
,1)
,1)

### The actual plotting
barplot(t, beside = T,
     main="Histogram to compare LIFESAT in 2019-20 to 2020-21",
     xlab="How satisfied are you with your life nowadays?",
     ylab="Percentage"
     )
```




```
### Getting the life meaningfulness data
t<-
round(                              ### Rounding result to 1 decimal
100*prop.table(                     ### Computing proportions, then multiplying by to get %
xtabs(~year+meaning, combine_years) ### basic frequency table
,1)
,1)

### The actual plotting
barplot(t, beside = T,
     main="Histogram to compare MEANING in 2019-20 to 2020-21",
     xlab="To what extent do you feel that things you do in your life have meaning?",
     ylab="Percentage"
     )
```



Provides grossed analysis of variance and Wilcoxon Scores (Rank Sums) for wellbeing variables by year

4.1.1 ANOVA happiness by year

ANOVA amounts to running a statistical test of significance of differences in the mean values of a variable across two or more groups. In most cases, linear regression with a single categorical independent (ie X) variable is identical. We will be choosing this approach here, as it is easier to implement in R.

```
summary(
  lm(happywb~as.factor(year),data=combine_years,weights=gross4))
```

```
##
## Call:
## lm(formula = happywb ~ as.factor(year), data = combine_years,
##     weights = gross4)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -743.70  -36.56   19.29   60.68  463.14
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          7.29116    0.01536 474.685   <2e-16 ***
## as.factor(year)2021  0.19046    0.02146   8.875   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 93.6 on 39183 degrees of freedom
##   (11638 observations deleted due to missingness)
## Multiple R-squared:  0.002006,   Adjusted R-squared:  0.001981
```

```
## F-statistic: 78.77 on 1 and 39183 DF,  p-value: < 2.2e-16
```

```r
### We need to use the Survey package in order to access the weighted rank test functions
### (Wilcoxon/Kruskal-Wallis)
library(survey)
```

```r
### We also need to declare the survey design, in this case we will only specify the
### grossing weights
design <- svydesign(ids = ~0, data=combine_years,
weights=combine_years$gross4)

svyranktest(happywb ~ as.factor(year), design = design)
```

```
##
##  Design-based KruskalWallis test
##
## data:  happywb ~ as.factor(year)
## t = 7.5714, df = 39183, p-value = 3.773e-14
## alternative hypothesis: true difference in mean rank score is not equal to 0
## sample estimates:
## difference in mean rank score
##                    0.02985664
```

4.1.2 ANOVA Anxiety amount by year

[Provides grossed analysis of variance and Kruskal-Wallis Rank Sums for State Pension benefit amounts by year]

```r
summary(lm(anxious~as.factor(year),data=combine_years,weights=gross4))
```

```
##
## Call:
## lm(formula = anxious ~ as.factor(year), data = combine_years,
##     weights = gross4)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -517.17  -99.59  -31.00   79.40  747.41
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          2.99605    0.02088 143.491  < 2e-16 ***
## as.factor(year)2021 -0.18385    0.02917  -6.303 2.95e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 127.2 on 39177 degrees of freedom
##   (11644 observations deleted due to missingness)
## Multiple R-squared:  0.001013,	Adjusted R-squared:  0.0009875
## F-statistic: 39.73 on 1 and 39177 DF,  p-value: 2.951e-10
```

```r
svyranktest(anxious ~ as.factor(year), design = design)
```

```
##
##  Design-based KruskalWallis test
##
## data:  anxious ~ as.factor(year)
```

```
## t = -5.1759, df = 39177, p-value = 2.279e-07
## alternative hypothesis: true difference in mean rank score is not equal to 0
## sample estimates:
## difference in mean rank score
##                   -0.02072376
```

4.1.3 ANOVA Life satisfaction by year

[Provides grossed analysis of variance and Kruskal-Wallis Rank Sums for Personal Independence Payment (PIP) Daily Living benefit amounts by year]

```
summary(lm(anxious~as.factor(year),data=combine_years,weights=gross4))
```

```
##
## Call:
## lm(formula = anxious ~ as.factor(year), data = combine_years,
##     weights = gross4)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -517.17  -99.59  -31.00   79.40  747.41
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          2.99605    0.02088 143.491  < 2e-16 ***
## as.factor(year)2021 -0.18385    0.02917  -6.303 2.95e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 127.2 on 39177 degrees of freedom
##   (11644 observations deleted due to missingness)
## Multiple R-squared:  0.001013,   Adjusted R-squared:  0.0009875
## F-statistic: 39.73 on 1 and 39177 DF,  p-value: 2.951e-10
```

```
svyranktest(anxious ~ as.factor(year), design = design)
```

```
##
##  Design-based KruskalWallis test
##
## data:  anxious ~ as.factor(year)
## t = -5.1759, df = 39177, p-value = 2.279e-07
## alternative hypothesis: true difference in mean rank score is not equal to 0
## sample estimates:
## difference in mean rank score
##                   -0.02072376
```

4.1.4 ANOVA life meaning by year

[Provides grossed analysis of variance and Kruskal-Wallis Rank Sums for Personal Independence Payment (PIP) Mobility benefit amounts by year]

```
summary(lm(anxious~as.factor(year),data=combine_years,weights=gross4))
```

```
##
## Call:
## lm(formula = anxious ~ as.factor(year), data = combine_years,
##     weights = gross4)
```

```
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -517.17  -99.59  -31.00   79.40  747.41
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          2.99605    0.02088 143.491  < 2e-16 ***
## as.factor(year)2021 -0.18385    0.02917  -6.303 2.95e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 127.2 on 39177 degrees of freedom
##   (11644 observations deleted due to missingness)
## Multiple R-squared:  0.001013,	Adjusted R-squared:  0.0009875
## F-statistic: 39.73 on 1 and 39177 DF,  p-value: 2.951e-10
```

```
svyranktest(anxious ~ as.factor(year), design = design)
```

```
##
##  Design-based KruskalWallis test
##
## data:  anxious ~ as.factor(year)
## t = -5.1759, df = 39177, p-value = 2.279e-07
## alternative hypothesis: true difference in mean rank score is not equal to 0
## sample estimates:
## difference in mean rank score
##                   -0.02072376
```

4.2 Tables of wellbeing variable scores against other variables

Create a grossed table of life satisfaction score against whether the adult has difficulty with mental health in 2019-20

```
### 2019-20
round(
100*prop.table(
  xtabs(gross4~lifesat+disd07,combine_years%>%filter(year==1920))
  ,2),1)
```

```
##        disd07
## lifesat    1    2    3
##      0   3.4  1.0  0.6
##      1   1.7  0.9  0.9
##      2   4.8  1.0  0.3
##      3   7.3  1.9  0.7
##      4   8.3  3.6  3.1
##      5  18.7 11.3  3.6
##      6  13.4  9.8  8.1
##      7  18.8 18.2 19.3
##      8  14.1 28.3 32.0
##      9   5.8 11.8 15.9
##      10  3.8 12.1 15.4
```

```
### 2020-21
round(
```

```r
100*prop.table(
  xtabs(gross4~lifesat+disd07,combine_years%>%filter(year==2021))
  ,2),1)
```

```
##         disd07
## lifesat    1    2    3
##      0   4.1  1.0  0.4
##      1   2.7  0.8  0.2
##      2   4.4  1.4  0.4
##      3   6.9  1.7  0.6
##      4   7.8  2.4  1.3
##      5  16.7 10.6  4.6
##      6  13.4  8.2  6.0
##      7  17.6 16.7 15.3
##      8  16.2 29.3 35.3
##      9   5.9 13.9 17.7
##      10  4.3 14.1 18.1
```

Create a grossed table of anxiousness score against the household income band of the adult in 2019-20

```r
### 2019-20
round(
100*prop.table(
  xtabs(gross4~anxious+hhincbnd,combine_years%>%filter(year==1920))
  ,2),1)
```

```
##         hhincbnd
## anxious    1    2    3    4    5    6    7    8    9   10   11
##      0  27.3 32.9 29.6 33.7 31.3 31.3 31.2 25.9 22.9 27.6 27.5
##      1   6.9  6.9  8.3  8.9 10.9  8.7 10.5  8.1 13.8 12.5 10.9
##      2  12.7 10.6 13.2 14.4 14.3 13.3 17.3 16.5 16.5 13.1 16.4
##      3  10.6  7.1  9.6  5.5  8.9  9.8  8.2 10.9  7.7 13.7  9.4
##      4   6.6  6.3  5.6  7.0  6.0  6.9  6.6  8.5  6.2  5.2  5.1
##      5  13.7 11.8 11.0  9.3  9.2  9.5  8.9  9.1  9.4 10.7  9.6
##      6   4.0  5.3  5.7  6.6  5.9  5.3  5.3  6.4  5.7  4.0  9.0
##      7   6.1  6.3  5.4  5.2  5.1  7.0  6.4  5.9  7.5  6.8  5.0
##      8   7.1  7.0  6.1  5.4  4.4  5.9  3.2  5.5  5.6  2.4  4.1
##      9   2.9  3.3  2.9  1.7  1.9  0.8  2.2  2.2  1.7  3.0  1.3
##      10  2.1  2.5  2.6  2.2  2.0  1.5  0.1  1.0  2.9  1.0  1.6
```

```r
### 2020-21
round(
100*prop.table(
  xtabs(gross4~anxious+hhincbnd,combine_years%>%filter(year==2021))
  ,2),1)
```

```
##         hhincbnd
## anxious    1    2    3    4    5    6    7    8    9   10   11
##      0  31.4 33.7 34.2 34.2 33.8 34.7 33.4 35.5 32.0 31.4 33.6
##      1   7.7  6.7  8.7  8.8  9.8 10.3 11.6 12.1 11.5 10.8 13.0
##      2  12.9 12.5 12.2 12.6 14.0 13.9 13.1 16.0 12.1 12.9 14.3
##      3   6.9  7.9  8.0  8.6  8.8  8.2  8.8  7.6 10.5 11.0  9.0
##      4   6.8  5.5  5.7  5.8  5.9  5.9  6.3  4.6  7.1  7.1  6.4
##      5   8.4 10.6  9.8 10.3  9.1  9.4  9.0  8.9  9.5  8.0  7.3
##      6   5.9  4.9  4.9  5.2  5.5  4.5  4.9  4.5  6.3  5.5  5.5
##      7   6.2  6.0  5.6  5.7  5.6  5.2  4.9  5.5  5.6  6.2  4.9
```

```
##       8   7.2  6.4  5.8  5.1  4.5  5.4  4.3  3.0  2.1  3.7  4.0
##       9   2.5  2.8  2.4  1.9  1.5  1.5  1.9  1.7  2.4  2.1  1.5
##       10  4.1  3.1  2.8  1.9  1.4  1.1  1.7  0.7  0.9  1.3  0.6
```

5 Rapid analysis

5.1 FRS households by tenure between 2018-19 and 2020-21

```
setwd("/home/piet/Dropbox/work/UKDS/FRS workshop/" )
tenure1<-read.spss("UKDA-8948-spss/spss/spss25/househol.sav",to.data.frame = T,
use.value.labels = T,
add.undeclared.levels="append")%>%select(SERNUM,TENURE,GROSS4)
names(tenure1)<-tolower(names(tenure1))

tenure2<-read.spss("UKDA-8802-spss/spss/spss25/househol.sav",to.data.frame = T,
use.value.labels = T,
add.undeclared.levels="append")%>%select(SERNUM,TENURE,GROSS4)
names(tenure2)<-tolower(names(tenure2))

tenure3<-read.spss("UKDA-8633-spss/househol.sav",to.data.frame = T,
use.value.labels = T,
add.undeclared.levels="append")%>%select(sernum,TENURE,GROSS4)
names(tenure3)<-tolower(names(tenure3))
```

2020-21

```
tmptab<-xtabs(gross4~tenure,tenure1,drop.unused.levels=T)
cbind(
"Freq"= tmptab,
"Pct"=round(100*prop.table(tmptab),1),
"Cum. freq."=cumsum(tmptab),
"Cum. pct"= round(cumsum(100*prop.table(tmptab)),2)
)
```

```
##                                       Freq  Pct Cum. freq. Cum. pct
## Owns it outright                  10228059 36.3   10228059    36.26
## Buying with the help of a mortgage  7879072 27.9   18107131    64.20
## Part own, part rent                 125263  0.4   18232394    64.64
## Rents                              9715232 34.4   27947626    99.08
## Rent-free                           258704  0.9   28206330   100.00
```

2019-20

```
tmptab<-xtabs(gross4~tenure,tenure2,drop.unused.levels=T)
cbind(
"Freq"= tmptab,
"Pct"=round(100*prop.table(tmptab),1),
"Cum. freq."=cumsum(tmptab),
"Cum. pct"= round(cumsum(100*prop.table(tmptab)),2)
)
```

```
##                                     Freq  Pct Cum. freq. Cum. pct
## Owns it outright                 9952274 35.5    9952274    35.51
## Buying with the help of a mortgage 7849704 28.0   17801978    63.52
## Part own, part rent               187136  0.7   17989114    64.19
## Rents                            9759793 34.8   27748907    99.02
## Rent-free                         270662  1.0   28019569    99.98
## Squatting                           4242  0.0   28023811   100.00
```

2018-19

```
tmptab<-xtabs(gross4~tenure,tenure3,drop.unused.levels=T)
cbind(
"Freq"= tmptab,
"Pct"=round(100*prop.table(tmptab),1),
"Cum. freq."=cumsum(tmptab),
"Cum. pct"= round(cumsum(100*prop.table(tmptab)),2)
)
```

```
##                                        Freq  Pct Cum. freq. Cum. pct
## Owns it outright                    9792127 35.2    9792127    35.19
## Buying with the help of a mortgage  7758708 27.9   17550835    63.07
## Part own, part rent                  139479  0.5   17690314    63.57
## Rents                               9889994 35.5   27580308    99.11
## Rent-free                            245099  0.9   27825407    99.99
## Squatting                              3380  0.0   27828787   100.00
```

5.2 FRS households by Council Tax Band between 2018-19 and 2020-21

```
setwd("/home/piet/Dropbox/work/UKDS/FRS workshop/" )
ctband1<-read.spss("UKDA-8948-spss/spss/spss25/househol.sav",to.data.frame = T,
use.value.labels = T,
add.undeclared.levels="append")%>%select(SERNUM,CTBAND,GROSS4)
names(ctband1)<-tolower(names(ctband1))

ctband2<-read.spss("UKDA-8802-spss/spss/spss25/househol.sav",to.data.frame = T,
use.value.labels = T,
add.undeclared.levels="append")%>%select(SERNUM,CTBAND,GROSS4)
names(ctband2)<-tolower(names(ctband2))

ctband3<-read.spss("UKDA-8633-spss/househol.sav",to.data.frame = T,
use.value.labels = T,
add.undeclared.levels="append")%>%select(sernum,CTBAND,GROSS4)
names(ctband3)<-tolower(names(ctband3))
```

Gives the grossed frequency, percentage, cumulative frequency and cumulative percentage of each category for the variable 'ctband' in 2020-21

```
tmptab<-xtabs(gross4~ctband,ctband1,drop.unused.levels=T)
cbind(
"Freq"= tmptab,
"Pct"=round(100*prop.table(tmptab),1),
"Cum. freq."=cumsum(tmptab),
"Cum. pct"= round(cumsum(100*prop.table(tmptab)),2)
)
```

```
##                                   Freq  Pct Cum. freq. Cum. pct
## Band A                         6310843 23.0    6310843    22.98
## Band B                         5461981 19.9   11772824    42.87
## Band C                         5583009 20.3   17355833    63.19
## Band D                         4469946 16.3   21825779    79.47
## Band E                         2932644 10.7   24758423    90.15
## Band F                         1557968  5.7   26316391    95.82
## Band G                          870044  3.2   27186435    98.99
## Band H                          122557  0.4   27308992    99.43
## Band I                            1262  0.0   27310254    99.44
```

```
## Household not valued separately   154516  0.6   27464770   100.00
```

Gives the grossed frequency, percentage, cumulative frequency and cumulative percentage of each category for the variable 'ctband' in 2019-20

```
tmptab<-xtabs(gross4~ctband,ctband2,drop.unused.levels=T)
cbind(
"Freq"= tmptab,
"Pct"=round(100*prop.table(tmptab),1),
"Cum. freq."=cumsum(tmptab),
"Cum. pct"= round(cumsum(100*prop.table(tmptab)),2)
)
```

```
##                                      Freq  Pct Cum. freq. Cum. pct
## Band A                            6198194 22.7    6198194    22.72
## Band B                            5419568 19.9   11617762    42.58
## Band C                            5525309 20.2   17143071    62.83
## Band D                            4426141 16.2   21569212    79.05
## Band E                            2756851 10.1   24326063    89.15
## Band F                            1512329  5.5   25838392    94.69
## Band G                            1028904  3.8   26867296    98.46
## Band H                             112002  0.4   26979298    98.88
## Band I                               5872  0.0   26985170    98.90
## Household not valued separately    300984  1.1   27286154   100.00
```

Gives the grossed frequency, percentage, cumulative frequency and cumulative percentage of each category for the variable 'ctband' in 2018-19

```
tmptab<-xtabs(gross4~ctband,ctband3,drop.unused.levels=T)
cbind(
"Freq"= tmptab,
"Pct"=round(100*prop.table(tmptab),1),
"Cum. freq."=cumsum(tmptab),
"Cum. pct"= round(cumsum(100*prop.table(tmptab)),2)
)
```

```
##                                      Freq  Pct Cum. freq. Cum. pct
## Band A                            6188935 22.8    6188935    22.84
## Band B                            5394817 19.9   11583752    42.75
## Band C                            5418607 20.0   17002359    62.75
## Band D                            4443995 16.4   21446354    79.15
## Band E                            2669700  9.9   24116054    89.00
## Band F                            1558186  5.8   25674240    94.76
## Band G                            1002356  3.7   26676596    98.46
## Band H                             121146  0.4   26797742    98.90
## Band I                               5918  0.0   26803660    98.92
## Household not valued separately    291550  1.1   27095210   100.00
```

5.3 FRS adult respondents by highest level of qualification between 2018-19 and 2020-21

```
setwd("/home/piet/Dropbox/work/UKDS/FRS workshop/" )
dvhiqual12<-rbind(
  cbind(read.spss("UKDA-8948-spss/spss/spss25/adult.sav",to.data.frame = T,
use.value.labels = F,
add.undeclared.levels="append")%>%select(SERNUM,BENUNIT,PERSON, DVHIQUAL,GROSS4),
```

```r
year=2021),
cbind(read.spss("UKDA-8802-spss/spss/spss25/adult.sav",to.data.frame = T,
use.value.labels = F,
add.undeclared.levels="append")%>%select(SERNUM,BENUNIT,PERSON, DVHIQUAL,GROSS4),
year=2020)
)
names(dvhiqual12)<-tolower(names(dvhiqual12))

### Unfortunately one of the datasets has a different variable name, so we need to deal
### with it separately
dvhiqual3<-cbind(read.spss("UKDA-8633-spss/adult.sav",to.data.frame = T,
use.value.labels = F,
add.undeclared.levels="append")%>%select(sernum,BENUNIT,PERSON, DVHIQUAL,GROSS4),
                 year=2019)
names(dvhiqual3)<-tolower(names(dvhiqual3))

dvhiqual123<-rbind(
  dvhiqual12,dvhiqual3
  )


dvhiqual123<-dvhiqual123%>%mutate(dvhiqual.s=case_when(
dvhiqual>=1 & dvhiqual<=16  ~ 'Level 4 or greater',
dvhiqual>=17 & dvhiqual<=34  ~ 'Level 3',
dvhiqual>=35 & dvhiqual<=54  ~ 'Level 2',
dvhiqual>=55 & dvhiqual<=82  ~ 'Level 1',
dvhiqual>=83 & dvhiqual<=85  ~ 'Entry level',
dvhiqual==86   ~ 'Other',
is.na(dvhiqual==T) ~ 'No reported qualification'
))
```

```r
### Percentages
  cbind(
round(100*prop.table(xtabs(gross4~dvhiqual.s+year,dvhiqual123),2),1),
### Then grossed frequencies
xtabs(gross4~dvhiqual.s+year,dvhiqual123)
)
```

```
##                           2019 2020 2021     2019     2020     2021
## Entry level                0.6  0.7  0.4   306778   338955   183998
## Level 1                    8.7  8.6 10.2  4504197  4424985  5289459
## Level 2                   12.0 11.6 13.0  6191677  6012687  6742296
## Level 3                   18.1 18.7 19.5  9347694  9644517 10059787
## Level 4 or greater        42.6 43.3 40.8 21972516 22390103 21111412
## No reported qualification 15.3 14.5 13.2  7871914  7477719  6844681
## Other                      2.6  2.7  2.9  1359183  1370074  1480014
```