Hilfreiche Ideen aus R

1. Motivation
2. Labels, names,
   Vektor, Matrix etc
3. Stapel und retour,
   unlist und relist
4. aplpack – Stamm & Blatt

1. Motivation

Martin Barghoorn APL-Online
November 2020

2. Labels, names, Vektor, Matrix etc

APL2

```
nam←'eins' 'zwei' 'drei' 'vier' 'funf' 'sech
s' 'sieb' 'acht' 'neun' 'zehn'
dat←7+⍳10
      dat
8 9 10 11 12 13 14 15 16 17
```

Verbinden in APL2, geht es auch anders?

```
vek←nam,[.5]dat
      vek
 eins zwei drei vier funf sechs sieb acht neun zehn
    8    9   10   11   12    13   14   15   16   17
```

Suchen und Finden der Daten im Vektor

```
   ⍴vek
2 10
```

Suche: Welche Zahl steht bei label vier?

```
 (,(vek[1;]∈⊂'vier')/vek)[2]
11      ⍝  die 11 wurde gefunden
 ((vek[1;]∈⊂'vier')/vek[2;])
11      ⍝  Variante
Speicherbedarf
 4 ⎕AT 'vek'
752  81  ⍝  752 byte
```

Frage: Geht es im APL auch einfacher?

Wie geht das bei  zuerst Vektordefinition

```
> nam<-strsplit("eins zwei drei vier funf sechs sieb
acht neun zehn", " ")
> nam
[[1]]
 [1] "eins"  "zwei"  "drei"  "vier"  "funf"  "sechs"
"sieb"  "acht"  "neun"  "zehn"
> length(nam)
[1] 1
```

Noch entlisten, strsplit erzeugt Liste

```
> nam<-unlist(nam) ; length(nam)
[1] 10
> dat<-7+1:10 ; dat          # Daten mit Index
 [1]  8  9 10 11 12 13 14 15 16 17
```

Verbinden von nam und dat

```
> vek<-dat                    # Kopie
> names(vek)<-nam             # Zauberei
> length(vek)
[1] 10                        # innig verbunden
> vek
 eins  zwei  drei  vier  funf sechs  sieb  acht  neun  zehn
    8     9    10    11    12    13    14    15    16    17
```

Frage: Welche Zahl steht bei label vier?

```
> vek['vier']
vier
  11
```

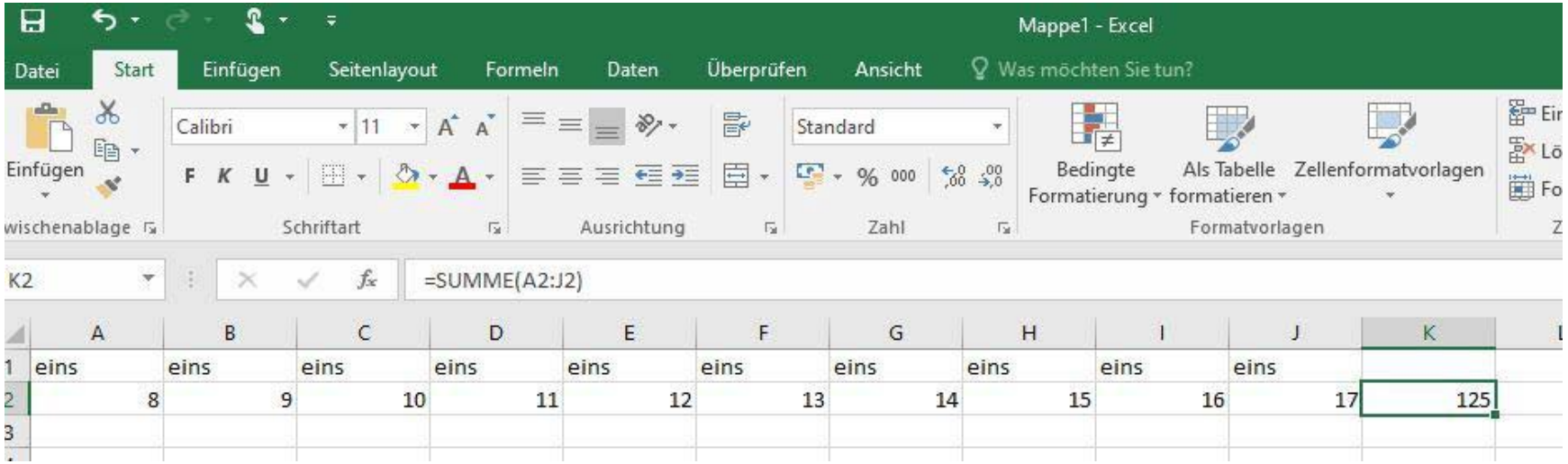
```
> length(vek['vier'])
[1] 1                  # Skalar
```

Sofort losrechnen mit den selektierten Daten

```
vek['vier']+99         # finden und rechnen
vier
 110
> sum(vek) ; mean(vek)
[1] 125
[1] 12.5
> summary(vek)              # kleine Statistik
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   8.00   10.25   12.50   12.50   14.75   17.00
```

Version Vektor MS-Excel

```
object.size(vek)          # R Speicherbedarf
704 bytes
```



Nun die nächste Dimension, Matrix

```
> mat<-matrix(1:12, dim<-
3:4, dimnames=list(NULL,
c("IMI", "ATA", "PERSIL",
"FEWA")))
# keine Zeilenlabels
```

```
> mat                                  # Matrix mat
     IMI ATA PERSIL FEWA
[1,]   1   4      7   10
[2,]   2   5      8   11
[3,]   3   6      9   12
> dim(mat)
[1] 3 4
> mat[,'PERSIL']                       # Spalte PERSIL
[1] 7 8 9
> mat[,'PERSIL'] *17   # suchen und losrechnen
[1] 119 136 153
> mat[,'PERSIL']<-mat[,'PERSIL']*17  # Zuweisung
# selective Assigment
```

```
> mat                # nach update
     IMI ATA PERSIL FEWA
[1,]   1   4    119   10
[2,]   2   5    136   11
[3,]   3   6    153   12
```

In gleicher Weise funktioniert es auch in der nächsten Dimension: beim array

3. Stapeln und retour

Stacking vectors concatenates multiple vectors into a single vector along with a factor indicating where each observation originated. Unstacking reverses this operation.

Stack work only with data.frame

```
> data<-as.data.frame(mat)
> data
  IMI ATA PERSIL FEWA
1   1   4    119   10
2   2   5    136   11
3   3   6    153   12
> data$PERSIL              # nur bei dataframe möglich
[1] 119 136 153
```

stack und unstack

```
> stack(data)
   values    ind
1       1    IMI
2       2    IMI
3       3    IMI
4       4    ATA
5       5    ATA
6       6    ATA
7     119 PERSIL
8     136 PERSIL
9     153 PERSIL
10     10   FEWA
11     11   FEWA
12     12   FEWA
```

```
> unstack(stack(data))
  IMI ATA PERSIL FEWA
1   1   4    119   10
2   2   5    136   11
3   3   6    153   12
# Data retour
```

unlist und relist

```
> vektor<-unlist(data) ; vektor
IMI1  IMI2   IMI3    ATA1    ATA2  ATA3 PERSIL1 PERSIL2 PERSIL3   FEWA1   FEWA2   FEWA3
  1     2      3       4       5     6     119     136     153      10      11      12
> relist(vektor, skeleton=data)
   IMI    ATA PERSIL   FEWA   <NA>   <NA>   <NA>
<NA>   <NA>   <NA>   <NA>   <NA>
     1      2      3      4      5      6    119
136    153     10     11     12
> relist(vektor, skeleton=mat)    # alte Matrix
     IMI ATA PERSIL FEWA
[1,]   1   4    119   10
[2,]   2   5    136   11
[3,]   3   6    153   12
```

4. aplpack – Stamm & Blatt

Datenimport Umfragedaten

```
> umfrage<-
read.csv2("d:/Lehre/Statistik3/umfrage.csv")
> dim(umfrage)
[1] 3471   17
> stich<-sample(3471, 300)
> reduct<-umfrage[stich,]
> attach(reduct)
> library(aplpack)          # laden von aplpack
> MAENNLICH=subset(GRO, GESCHL=="MAENNLICH")
> WEIBLICH=subset(GRO, GESCHL=="WEIBLICH")
> stem.leaf.backback(MAENNLICH, WEIBLICH)
```

stem.leaf.backback (MAENNLICH, WEIBLICH)

```
                MAENNLICH       WEIBLICH

                                | 14* |
                                |   t |
                                |   f |
                                |   s |
                                | 14. |9                                  1
                                | 15* |00                                 3
                                |   t |2333                               7
                                |   f |445                               10
                                |   s |667777                            16
     1                        9| 15. |88888888999                       27
     2                        0| 16* |0000000001111                     40
     4                       32|   t |233333333333333                   55
     5                        5|   f |4444444444445555555              (19)
     9                     7766|   s |666677777777777                   66
    14                    99988| 16. |88888888888889999                 51
    29           1100000000000000| 17* |000000000111                      34
    47       3333332222222222222| t |2222222                           22
    58            55555444444| f |44555                                 15
    73          777776666666666| s |6666                                10
  (26)  99999888888888888888888888| 17. |8888                            6
    60            110000000000000| 18* |00                                2
    46                 3332222222| t |
    36                  555554444| f |
    27          7777776666666666| s |
    11                       98| 18. |
     9                     1000| 19* |
     5                    33222| t |
                               | f |
                               | s |
                               | 19. |
                               | 20* |

n:                        160       140
NAs:                        1         0
```

Resumée

Die Verbindung zwischen Text und Zahlen sollte im APL modernisiert werden, um die Arbeit zu erleichtern.

Ich bedanke mich herzlich für Ihre Aufmerksamkeit!

Martin@Barghoorn.com