1_Software_Clusterisation

As we discussed, I've filtered the software out of the top 9. As far as I can judge it might have notable implications further.

Added the binaries such as:
- cloudnessBinary
- mobileBinary

Relabeled Q4-Q7 such as:
- stationaryQuantity
- mobileQuantity
- cloudnessQuantity

```
## SOLVED: remake the table with a `q` per `a2`

merged = merged %>%
  mutate(id = 1:nrow(merged)) %>%
  group_by(a2, q) %>%
  mutate(QuantPerDevice = n()) %>%
  spread(key = q, value = QuantPerDevice, fill = 0) %>% select(-id) %>%
  ungroup() %>% group_by(a2) %>%
  mutate(stationaryQuantity = sum(Q4), # feature: quantity, software on desktop/laptop
         mobileQuantitiy = sum(Q5 + Q6), # feature: quantity, software on mobile/laptop devices
         cloudnessQuantity = sum(Q7)) %>% # feature: quantity, software is used on the web (cloud)
  select(-c(Q4, Q5,Q6, Q7))

# feaure: whether software is principally used on several devices
merged$mobileBinary = ifelse(merged$mobileQuantitiy > 1, 1, 0)
# feature: whether software is principally possess web version
merged$cloudnessBinary  = ifelse(merged$cloudnessQuantity > 1, 1, 0)
```

Adding Q8_v2_1:Q16 questions with are ordinal. We treat them close to numerical in order to make aggregation for the software peaces.

Pre-MDS data preparation, scaling

One of the first plot I showed was MDS without any scaling for Occupations. Due to the fact that some of the Occupations are simple over represented it might have (and probably did) affected the interpretability. Though, current groups of software clustered a bit better.

```
## Scaling Before MDS, results are even worse
## UPD: scaling should be accomplished for all of the vars, another fix

# pre_dist_multi_final %>% colnames()

## scaling only for occupations since they represent absolute frequencies

pre_dist_multi_final_no_scaling = pre_dist_multi_final
pre_dist_multi_final[,-c(1,2)] %<>% apply(2, function(X) scale(X))

## pre_dist_multi_final[,-c(1:3)] %<>% apply(2, function(X) scale(X, center = FALSE))
```

MDS includes all

I tried several distance calculation metrics (‘euclidean’ ‘maximum’ ‘manhattan’ ‘minkowski’ ‘canberra’ ‘binary’). Manhattan is good, though, canberra - seems to better divide groups. The difference between them is that canberra is better suited for discrete, while euclidean is the de facto default for continuous numeric. I wrapped MDS, kmeans and visualisation into function - you can try different distances, it is the **_mds_with_Q8**.

Figure 1 is the option I suggest to stop on for now

```
## [1] "Nothing to exlude"
```

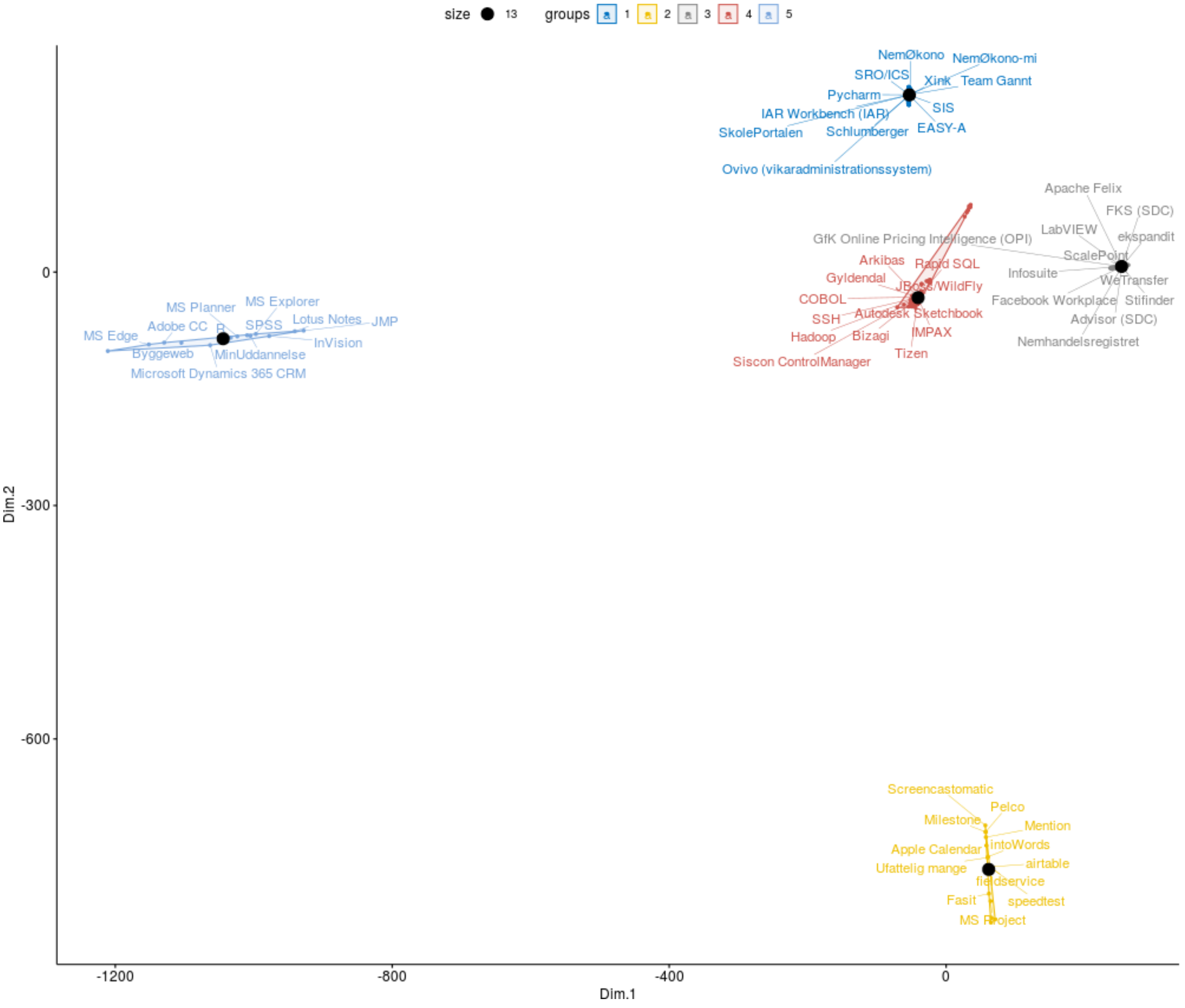


Figure 1: MDS with all constructs:devices, reprogrammability, occupation

Different distance used, what I don’t like here is that Apple Contacts and Calendar are in a different places.

```
p = flexible_clustering(pre_dist_multi_final, method = "manhattan", clust_n = 5, labels_col = 12,
                        column_exclude_start = 0, column_exclude_end = 0, seed = 11)$plot
```

```
## [1] "Nothing to exlude"
```

```
p
```

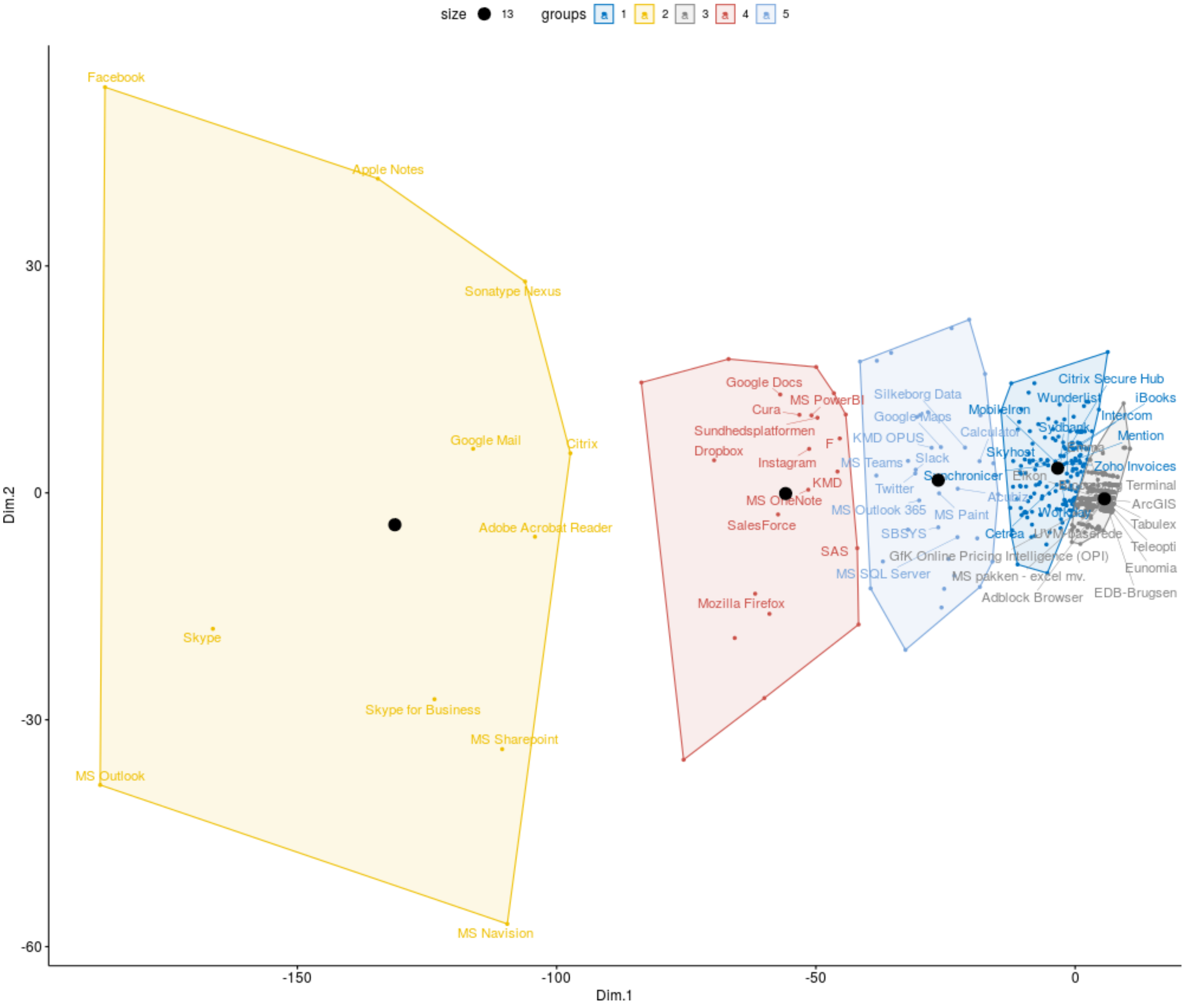


Figure 2: Mapping Software: Apple Products Appeared in Different Clusters

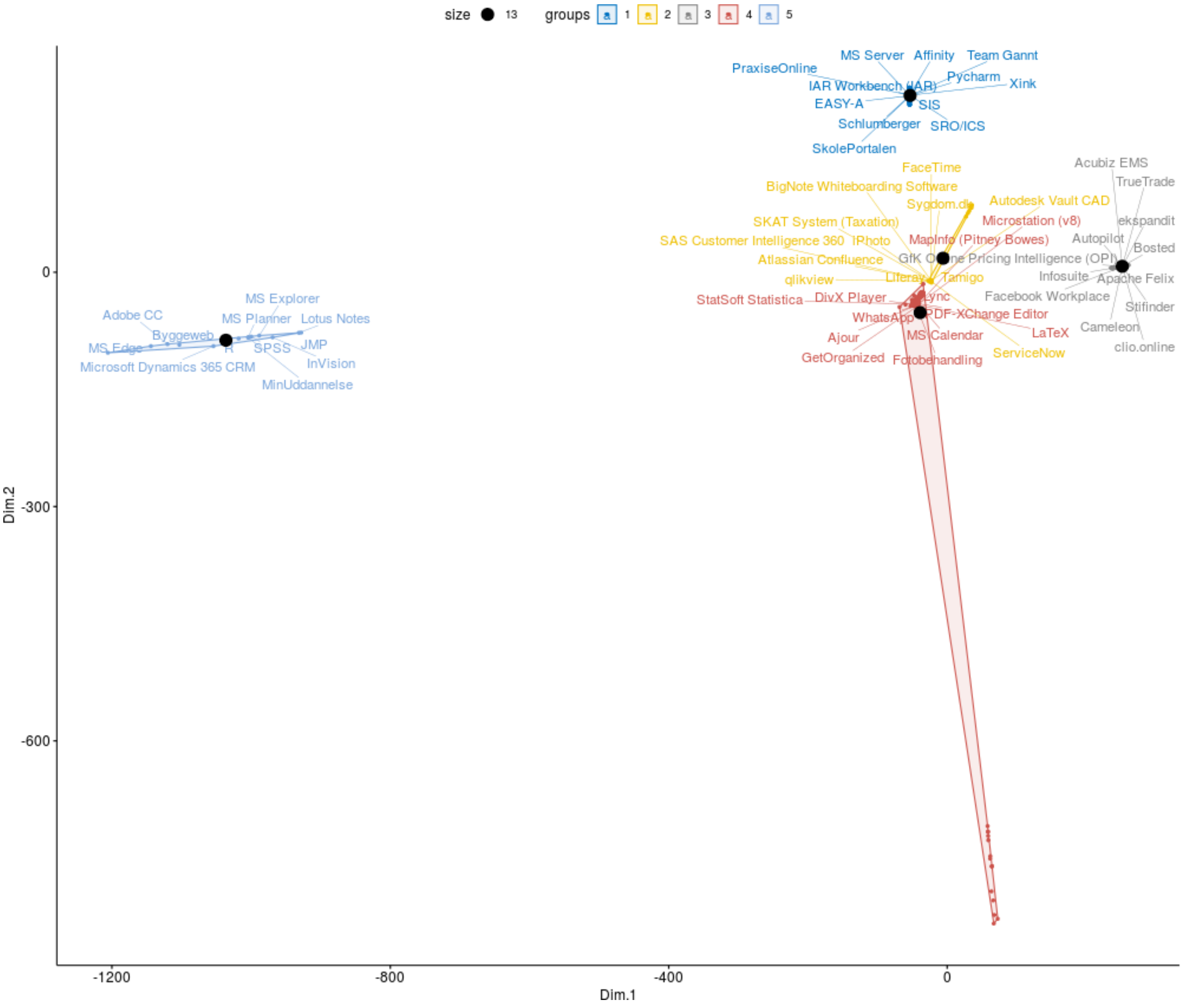
Others (just take to be aware of alternatives)

MDS: reprogrammability and devices used

I put it here just to compare with the previous one, I like it much less than the previous one. Meanwhile, it might be that I just overlooked something important.

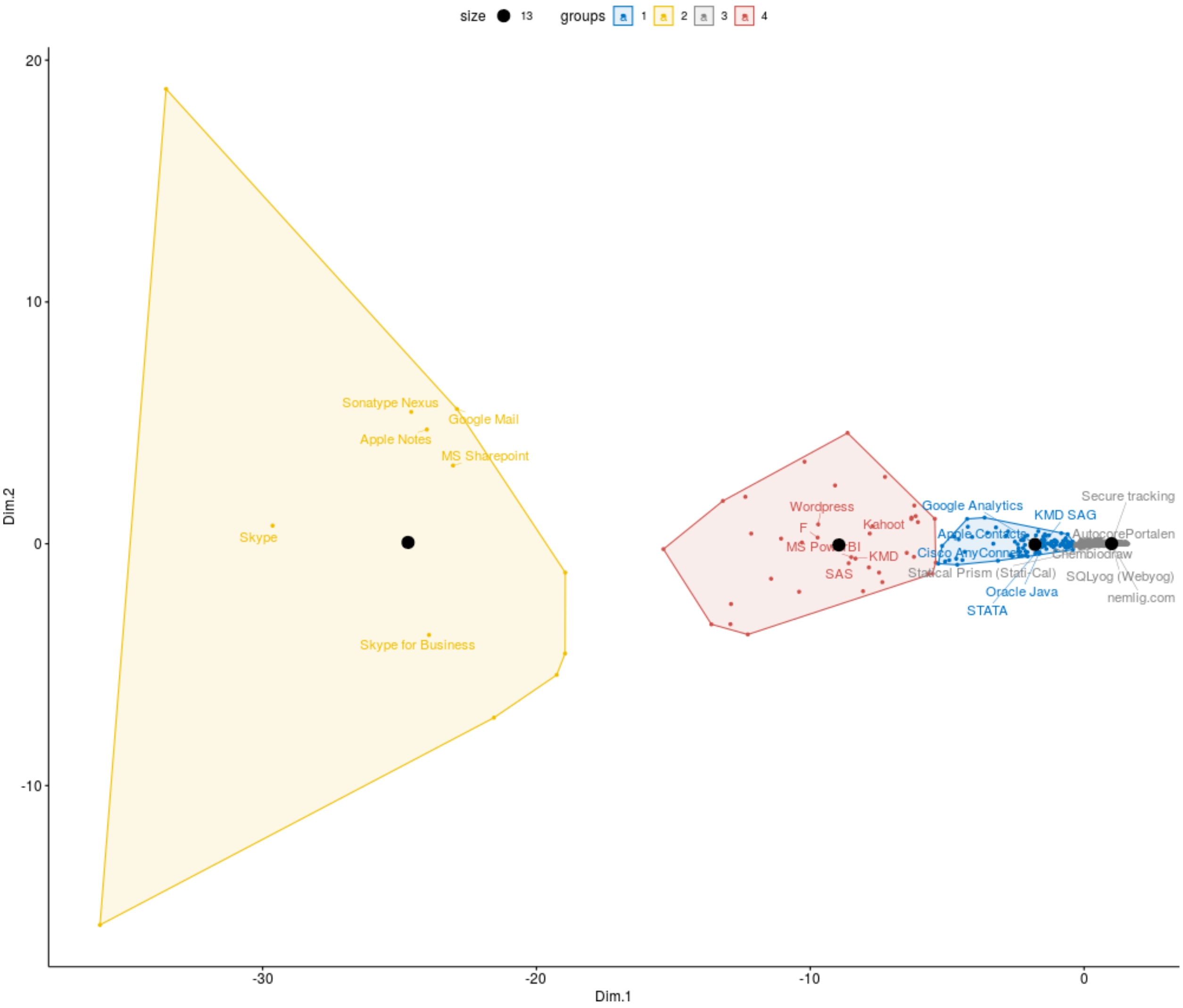
Canberra distances

```
## pre_dist_multi_final %>% colnames

p = flexible_clustering(pre_dist_multi_final, method = "canberra", clust_n = 5, labels_col = 12,
                        column_exclude_start = 20, column_exclude_end = 29, additional_exclude = c("clou
                        seed = 11)$plot

p
```



Euclidean distances

```
p = flexible_clustering(pre_dist_multi_final, method = "euclidean", clust_n = 4, labels_col = 6,
                        column_exclude_start = 20, column_exclude_end = 29, additional_exclude = c("clou
                        seed = 11)$plot


p
```



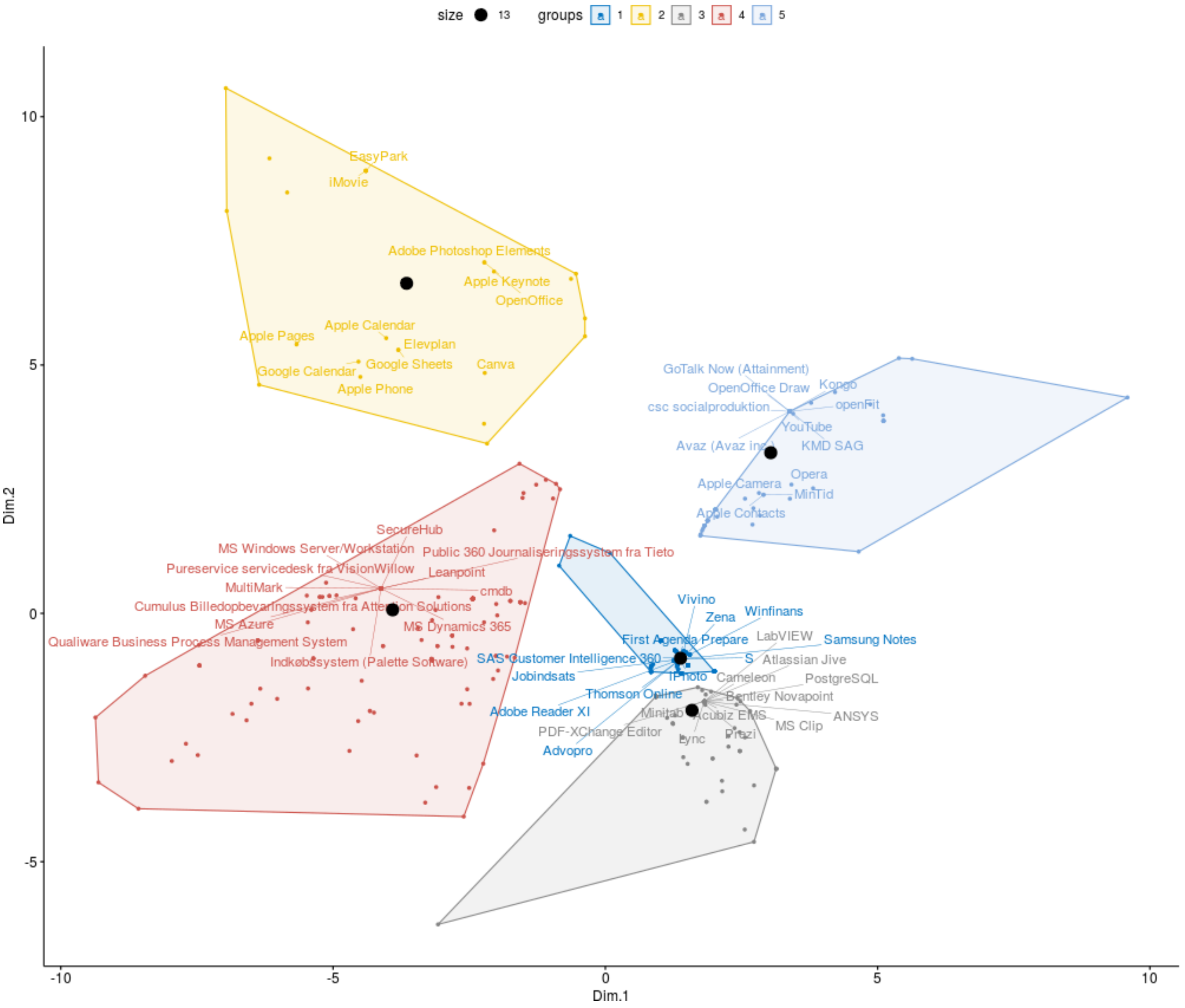
MDS: only Occupation

This one is a bit better.

```
p = flexible_clustering(pre_dist_multi_final, method = "canberra", clust_n = 5, labels_col = 12,
                        column_exclude_start = 3, column_exclude_end = 20,
                        #additional_exclude = c("cloudnessBinary","mobileBinary"),
                        seed = 11)$plot
```

```
## [1] "Nothing to exlude"
```

```
p
```



Inspection

This table gives an overview of the most typical software items for each cluster. Raw frequencies are useful to identify goodness of clusters (some contains much less than others, probably, should think about decreasing to 4). Scaled might be useful for the description of clusters using info about professions (health guys use MS Word and few other software items and that's all). Something similar is given for the respondents in LPA.

```
## options(width = 80)

inspect = flexible_clustering(pre_dist_multi_final, method = "canberra", clust_n = 5, labels_col = 12,
                              column_exclude_start = 0, column_exclude_end = 0, seed = 11)
```

```
## [1] "Nothing to exlude"
```

```
inner_join(cbind(inspect$fit, labels = inspect$labels) %>%
             select(labels, groups),
           pre_dist_multi_final_no_scaling, by = c("labels" = "a")) %>%
  arrange(groups) %>% select(-RecordNo) %>%
  group_by(groups) %>% summarise_all(funs(if(is.numeric(.)) sum(.) else first(.))) %>%
  dplyr::select(-labels, -c(stationaryQuantity:Q16_agr))  %>%
  knitr::kable(digits = 2) %>% kable_styling(bootstrap_options = "striped", full_width = T)
```

groups	Business and administration	Business, economics, and administration	Chief executives, senior officials, and legislators	General management	Health	Hospitality and retail	ICT	Legal, social, and cultural	Science and engineering	Teaching
1	21	20	6	23	30	2	34	6	12	32
2	1	0	0	8	4	0	19	3	9	12
3	19	16	2	14	38	2	21	20	24	27
4	191	99	59	245	226	22	535	157	246	289
5	7	5	0	4	3	1	6	8	15	7

```
inner_join(cbind(inspect$fit, labels = inspect$labels) %>%
             select(labels, groups),
           pre_dist_multi_final, by = c("labels" = "a")) %>%
  arrange(groups) %>% select(-RecordNo) %>%
  group_by(groups) %>% summarise_all(funs(if(is.numeric(.)) mean(.) else first(.))) %>%
  dplyr::select(-labels, -c(stationaryQuantity:Q16_agr)) %>%
  ## formattable()
  knitr::kable(digits = 2) %>% kable_styling(bootstrap_options = "striped", full_width = T)
```

groups	Business and administration	Business, economics, and administration	Chief executives, senior officials, and legislators	General management	Health	Hospitality and retail	ICT	Legal, social, and cultural	Science and engineering	Teaching
1	-0.12	0.00	-0.07	-0.15	-0.07	-0.06	-0.26	-0.24	-0.27	-0.14
2	-0.22	-0.26	-0.19	0.26	-0.03	-0.13	0.45	0.01	0.36	0.42
3	-0.17	-0.10	-0.16	-0.24	-0.07	-0.08	-0.34	-0.13	-0.21	-0.21
4	0.07	0.02	0.07	0.09	0.04	0.03	0.15	0.08	0.09	0.08
5	0.39	0.45	-0.19	0.01	-0.05	0.24	-0.10	0.66	1.04	0.16

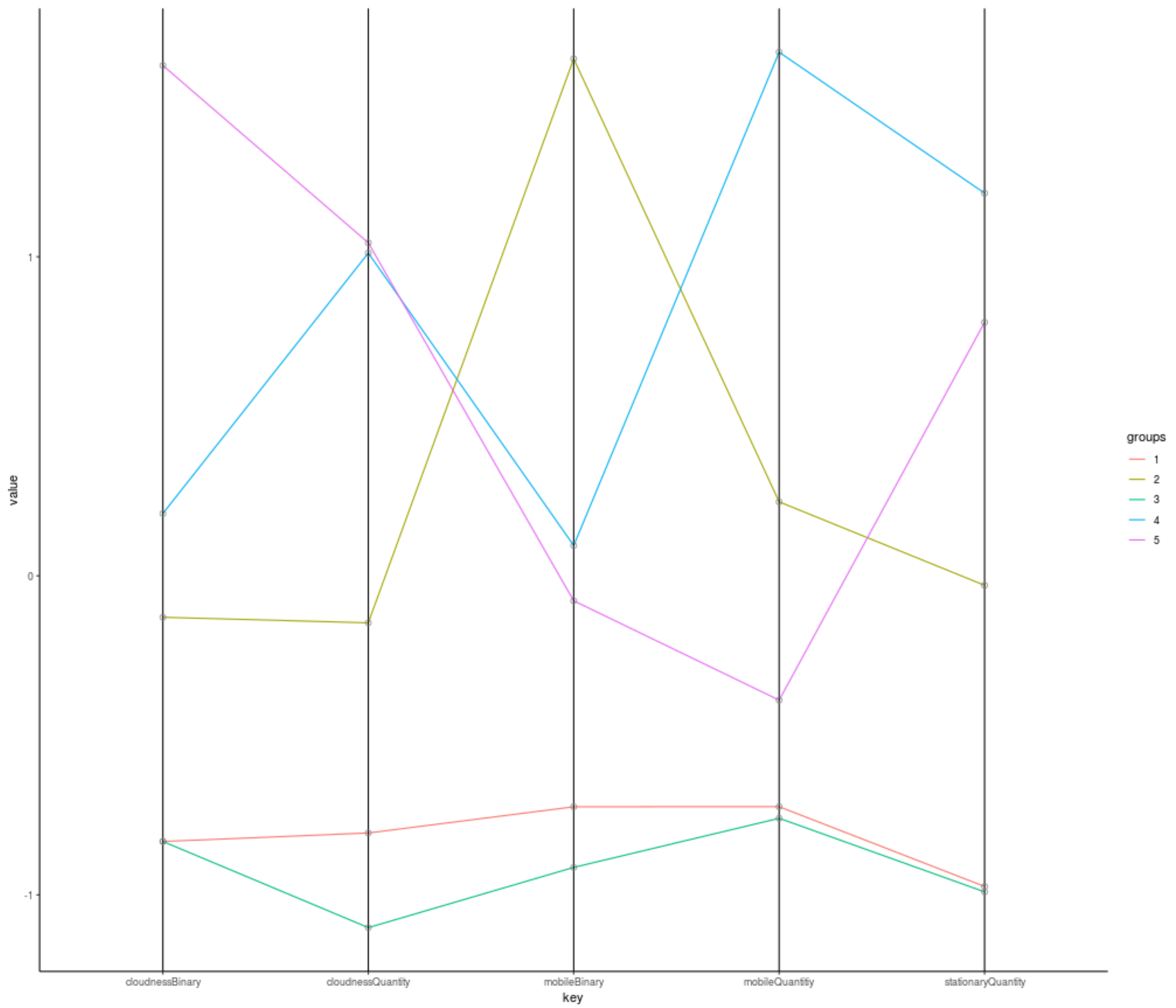
```
## inner_join(cbind(inspect$fit, labels = inspect$labels) %>%
##              ## filter(labels %in% inspect$selectedLables) %>%
##              select(labels, groups),
##            pre_dist_multi_final_no_scaling, by = c("labels" = "a")) %>% arrange(groups) %>% select(-
##   group_by(groups) %>% summarise_all(funs(if(is.numeric(.)) sum(.) else first(.))) %>% View



# cbind(inspect$fit, labels = inspect$labels) %>% filter(labels %in% c("R", "MS Planner"))
```

Understanding Clusters

Parallel Coordinates Plot

This is simple implementation of parallel coordinates. Horizontal lines here are simply means of clusters. This way, it gives an overview of what each cluster is about. Since values are scaled, less than 0 - less than mean, above 0 - above the mean. I included only variables connected with devices.

```
parallelCoordsDf = inner_join(cbind(inspect$fit, labels = inspect$labels) %>%
             ## filter(labels %in% inspect$selectedLables) %>%
             select(labels, groups),
             pre_dist_multi_final_no_scaling, by = c("labels" = "a"))


parallelCoordsDf$groups %<>% as.character()
parallelCoordsDf$RecordNo %<>% as.character()
parallelCoordsDf %<>% group_by(groups) %>% summarise_all(funs(if(is.numeric(.)) mean(.) else first(.)))


theme_set(theme_classic())

ggplot(parallelCoordsDf %>%
         mutate(ID = 1:n()) %>%
         mutate_if(is.numeric, scale) %>% ## FIXME: make global scale before aggregation
         gather(key, value, c(4:8)),
       aes(key, value, group=groups, colour = groups)) +
  geom_line() +
  geom_vline(xintercept = 1:5) +
  geom_point(size=2, shape=21, colour="grey50") +
  scale_fill_manual(values=c("black","white"))
```



The main point is that mostly Management Systems (CRM systems), Planning Systems and Software used to Support Collaboration (examples) and software for software development (tautology** and programming languages are plotted.

Only 12 software pieces, which are closer to the core of each cluster are plotted.

I am describing clusters from Figure 1

1 cluster:

- low reprogrammability
- nothing specific in terms of device used
- this cluster includes teachers, general management, business, health and ICT - very diverse in terms of profession
- from another side, it represents respondents who don't care about extensibility of their software toolset

Based on the soft used NemOkono, IAR Workbench, Schlumberger and Team Gannt, especially, Pycharm, it might be about software engineers who do not care about the extensibility of tools used either about the senior developers/managers, who rarely programm (Pycharm), but mostly engaged in the planning activities (Team Gannt - Gannt charts - projects dependencies, seems like IT approach). Teachers probably use this.

Seems like the main logo of this cluster "This tool is just making the job need to be done"

2 cluster:

- small cluster in terms of respondents
- managers, ICT, Teachers
- high scores on customasibility of software (less than in 5th cluster, more than in 4th)
- prevalence of software used on mobile+tablet type of device

Logo: "Intelligent middle-layer"

Software used: - Pelco - CCTV (?) - system administrators , - Fasit - automatisation job workflow system ? management - intoWords - speech-to-text tool, might be used by secretariat - airTable - lightweight and fency web spreadshit tool - Screencastomatic - used for video capturing - Apple Calendar

3 cluster:

- lowest reprogrammability
- nothing specific in terms of device orientation
- diverse in terms of occupations: business + management + health (+) Legal, social and Cultural + Science and Engineering + Teaching

Facebook
WeTransfer
InfoSuite

Software is rather specific (WeTransfer - file sharing) or has very broad focus (Facebook).
While can't say about the software items, should TODO: check those users: intuitively, they might use fewer tools than others.

4 cluster:

The majority of the software items are included in this cluster. I cannot completely understand why mostly items connected with programming languages and open source are mapped on the plot. #TODO: use DALEX or something similar to untangle K-Means clusterisation

- medium repogrammability
- device-specific variables are higher than mean, though, it is the biggest cluster, so, don't think it tells a lot

COBOL - programming language for business
SSH - yeah, well
Autodesk Sketchbook - might be proprietary
TIZEN - open source
Hadoop - should be open sourced
wildfly - open source as well

5 cluster:

- small cluster
- mostly stationary and web applications
- highest reprogrammability
- professions: business, no executives, some users are managers, science + engineering guys

It includes MS Planner, Edge/Explorer, Byggeweb + R and SPSS + Adobe CC (outlier, don't like it here). Mostly this cluster has highest score across reprogrammability. Generally, notable combination of tools with high reprogrammability with proprietary software without any customization capabilities. Should inspect it more closely.

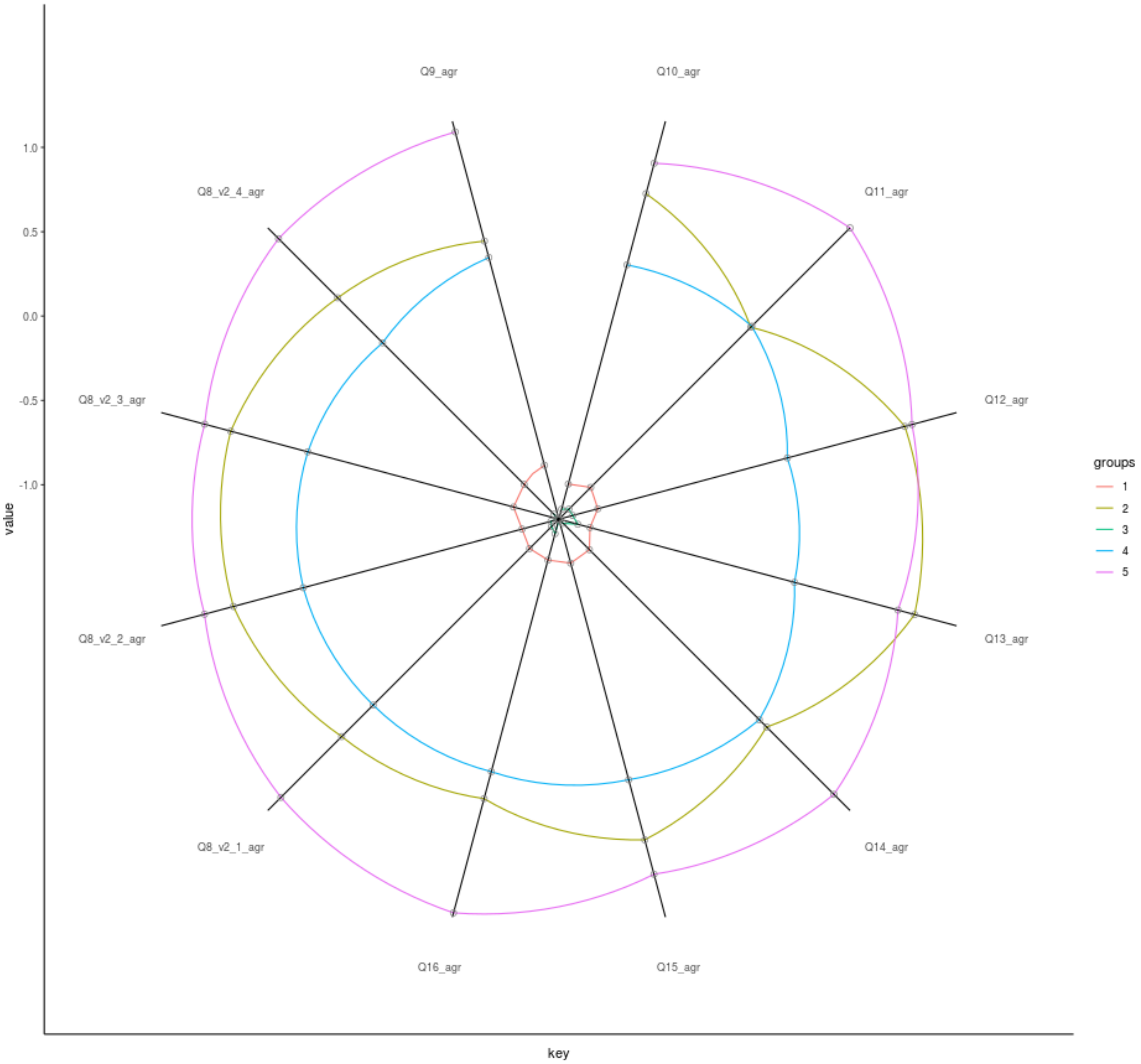
Resume: Either we should figure out a better way to map software considering occupations, either forget about occupation as a measurement for software.

To try: What I started to doing is Latent Profile Analysis (LPA) (sort of Dimensionality Reduction specifically for respondents). Since I didn't completely understand whether it's respondents of software in the focus (still questionable, right?) I started to describe software using respondents and vice versa. Instead, maybe, we should consider use SES-style info in LPA, extract those profiles and try to describe software based on it (it should solve the issue with sparse occupation matrix, since, in clustering Occupations doesn't influence the results that much as expected).

Radar Plot

This is simple implementation of spyder/radar plot. I wasn't a big fan this kind of plot, but I think I like them now. Also, just additional line of code in ggplot2.

```
ggplot(parallelCoordsDf %>%
         mutate(ID = 1:n()) %>%
         mutate_if(is.numeric, scale) %>% ## FIXME: make global scale before aggregation
         gather(key, value, c(9:20)),
       aes(key, value, group=groups, colour = groups)) +
  geom_line() +
  geom_vline(xintercept = 1:12) +
  geom_point(size=2, shape=21, colour="grey50") +
  scale_fill_manual(values=c("black","white")) + coord_polar()
```



TODO: Use hierarchical clustering instead