Applying PCA, BCT and Imposters using word features

Introduction

In this notebook we will apply Principal Component Analysis, Bootstrap Consensus Tree and the `imposters()` method to make the results a little bit more interpretable. This is a supplement material to the main analysis which will be conducted with character 4-grams and character 5-grams.

```
# import the neccessary libraries
# if a library is not installed run the following command: `install.packages(package to install)`
library(stylo)
```

```
##
## ### stylo version: 0.7.4 ###
##
## If you plan to cite this software (please do!), use the following reference:
##     Eder, M., Rybicki, J. and Kestemont, M. (2016). Stylometry with R:
##     a package for computational text analysis. R Journal 8(1): 107-121.
##     <https://journal.r-project.org/archive/2016/RJ-2016-007/index.html>
##
## To get full BibTeX entry, type: citation("stylo")
```

```
# library(gplots)
# library(pheatmap)
```

```
set.seed(100) # random seed for reproducibility
```

Setting the working directory

```
setwd("../../analysis/word_features/")
getwd()
```

```
## [1] "/Users/paschalis/Documents/MA_DH/Thesis/seneca_stylometry/analysis/word_features"
```

Preparation of the data

Importing the corpus and tokenisation

In this step we import the corpus that we are going to use and consequently we tokenize it. The tokenisation follows the rules of the parameter `Latin.corr`. This is done because a lot of texts do not distinguish "u/v" and by setting this parameter to `Latin.corr` we take care of this variation in the letters. Moreover, we change uppercase letters to lowercase.

We should clarify that since applying PCA and BCT to a dataset with a big number of authors and works might cause some overlapping and might make the interpretation of the plots impossible; we decided to shrink the dataset to authors that lived very close to Seneca's the Younger time, such as Lucan. To test Ferri's hypothesis that *Silvae* by Statius might work as a terminus ante quem for *Octavia*, we have also included Statius works. Due to their small size, *Satires* by Persius are excluded from the dataset; this will allow us to extract some samples from the texts to balance very large texts with "normal" size texts.

```r
raw.corpus <- load.corpus(files = "all", corpus.dir = "../verse_corpus_pca_bct/",
                          encoding = "UTF-8")

tokenized.corpus <- txt.to.words.ext(raw.corpus, corpus.lang = "Latin.corr",
                                     preserve.case = FALSE)

# make samples
sliced.corpus <- make.samples(tokenized.corpus,
                              sampling = "random.sampling",
                              number.of.samples = 2,
                              sample.size = 3000)
```

```
## luc_phars_1.txt
##  - text length (in words): 4375
##  - nr. of random samples: 2
##  - sample length: 3000
## luc_phars_10.txt
##  - text length (in words): 3506
##  - nr. of random samples: 2
##  - sample length: 3000
## luc_phars_2.txt
##  - text length (in words): 4643
##  - nr. of random samples: 2
##  - sample length: 3000
## luc_phars_3.txt
##  - text length (in words): 4763
##  - nr. of random samples: 2
##  - sample length: 3000
## luc_phars_4.txt
##  - text length (in words): 5153
##  - nr. of random samples: 2
##  - sample length: 3000
## luc_phars_5.txt
##  - text length (in words): 5181
##  - nr. of random samples: 2
##  - sample length: 3000
## luc_phars_6.txt
##  - text length (in words): 5163
##  - nr. of random samples: 2
##  - sample length: 3000
## luc_phars_7.txt
##  - text length (in words): 5589
##  - nr. of random samples: 2
##  - sample length: 3000
## luc_phars_8.txt
##  - text length (in words): 5618
##  - nr. of random samples: 2
##  - sample length: 3000
## luc_phars_9.txt
##  - text length (in words): 7074
##  - nr. of random samples: 2
##  - sample length: 3000
## sen_ag.txt
##  - text length (in words): 5447
##  - nr. of random samples: 2
##  - sample length: 3000
## sen_her_f.txt
##  - text length (in words): 7495
##  - nr. of random samples: 2
##  - sample length: 3000
## sen_her_o.txt
##  - text length (in words): 11157
##  - nr. of random samples: 2
##  - sample length: 3000
## sen_med.txt
##  - text length (in words): 5557
##  - nr. of random samples: 2
##  - sample length: 3000
## sen_oct.txt
##  - text length (in words): 5093
##  - nr. of random samples: 2
##  - sample length: 3000
## sen_oed.txt
##  - text length (in words): 5764
##  - nr. of random samples: 2
##  - sample length: 3000
## sen_phaed.txt
##  - text length (in words): 7063
##  - nr. of random samples: 2
##  - sample length: 3000
## sen_phoen.txt
##  - text length (in words): 4072
##  - nr. of random samples: 2
##  - sample length: 3000
## sen_thy.txt
##  - text length (in words): 6160
##  - nr. of random samples: 2
##  - sample length: 3000
## sen_tro.txt
##  - text length (in words): 6671
##  - nr. of random samples: 2
##  - sample length: 3000
## stat_achill.txt
##  - text length (in words): 7205
##  - nr. of random samples: 2
##  - sample length: 3000
## stat_silv_1.txt
##  - text length (in words): 5226
##  - nr. of random samples: 2
##  - sample length: 3000
## stat_silv_2.txt
##  - text length (in words): 4965
##  - nr. of random samples: 2
##  - sample length: 3000
## stat_silv_3.txt
##  - text length (in words): 5097
##  - nr. of random samples: 2
##  - sample length: 3000
## stat_silv_4.txt
##  - text length (in words): 4328
##  - nr. of random samples: 2
##  - sample length: 3000
## stat_silv_5.txt
##  - text length (in words): 5489
##  - nr. of random samples: 2
##  - sample length: 3000
## stat_theb_1.txt
##  - text length (in words): 4527
##  - nr. of random samples: 2
##  - sample length: 3000
## stat_theb_10.txt
##  - text length (in words): 6026
##  - nr. of random samples: 2
##  - sample length: 3000
## stat_theb_11.txt
##  - text length (in words): 4970
##  - nr. of random samples: 2
##  - sample length: 3000
## stat_theb_12.txt
##  - text length (in words): 5237
##  - nr. of random samples: 2
##  - sample length: 3000
## stat_theb_2.txt
##  - text length (in words): 4755
##  - nr. of random samples: 2
##  - sample length: 3000
## stat_theb_3.txt
##  - text length (in words): 4681
##  - nr. of random samples: 2
##  - sample length: 3000
## stat_theb_4.txt
##  - text length (in words): 5392
##  - nr. of random samples: 2
##  - sample length: 3000
## stat_theb_5.txt
##  - text length (in words): 4915
##  - nr. of random samples: 2
##  - sample length: 3000
## stat_theb_6.txt
##  - text length (in words): 6002
##  - nr. of random samples: 2
##  - sample length: 3000
## stat_theb_7.txt
##  - text length (in words): 5273
##  - nr. of random samples: 2
##  - sample length: 3000
## stat_theb_8.txt
##  - text length (in words): 4945
##  - nr. of random samples: 2
##  - sample length: 3000
## stat_theb_9.txt
##  - text length (in words): 5829
##  - nr. of random samples: 2
##  - sample length: 3000
```

```
help("make.samples")
```

Remove the pronouns

It was decided to remove the pronouns, since some pronouns are connected to the genre of the text.

```
corpus.no.pronouns <- delete.stop.words(sliced.corpus,
                                        stop.words = stylo.pronouns(corpus.lang = "Latin.corr"))
```

Extracting the features

The final step before proceeding to the method per se is to extract the features that we want to use and add them to a table with frequencies. In our case, we want to extract word 1grams (i.e., simple words (aka tokens)).

```
corpus.w.grams <- txt.to.features(corpus.no.pronouns,
                                  ngram.size = 1,
                                  features = "w")

freq.features.word.grams <- make.frequency.list(corpus.w.grams,
                                                head = 3000)

freqs.word.grams <- make.table.of.frequencies(corpus.w.grams,
                                              features = freq.features.word.grams,
                                              relative = T)
```

```
## processing  76  text samples
## .......
## combining frequencies into a table...
```

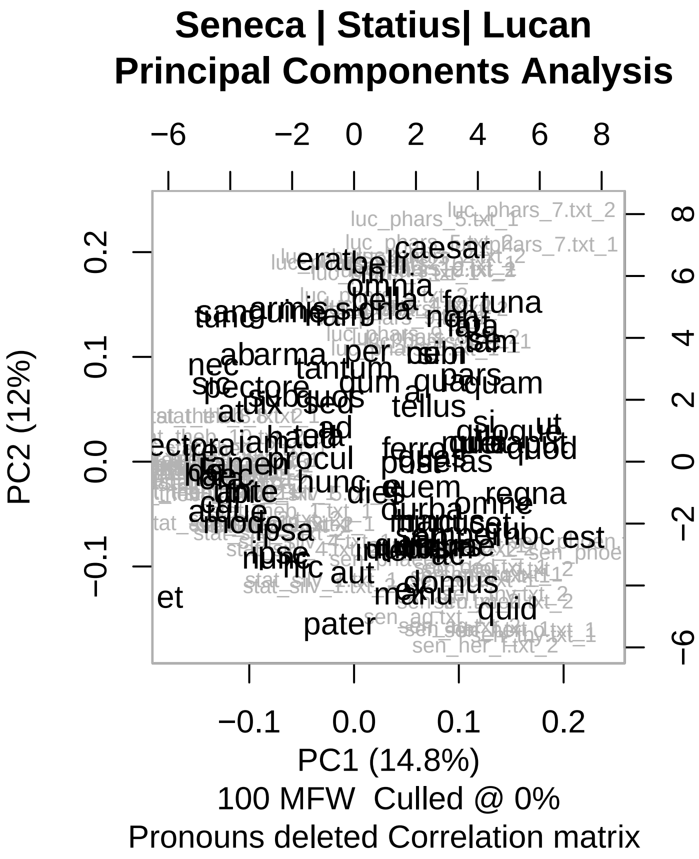
Methods

Principal Component Analysis

We will run two separate experiments with PCA; one will be using a correlation plot to visualize the results and on will be using a covariance plot.

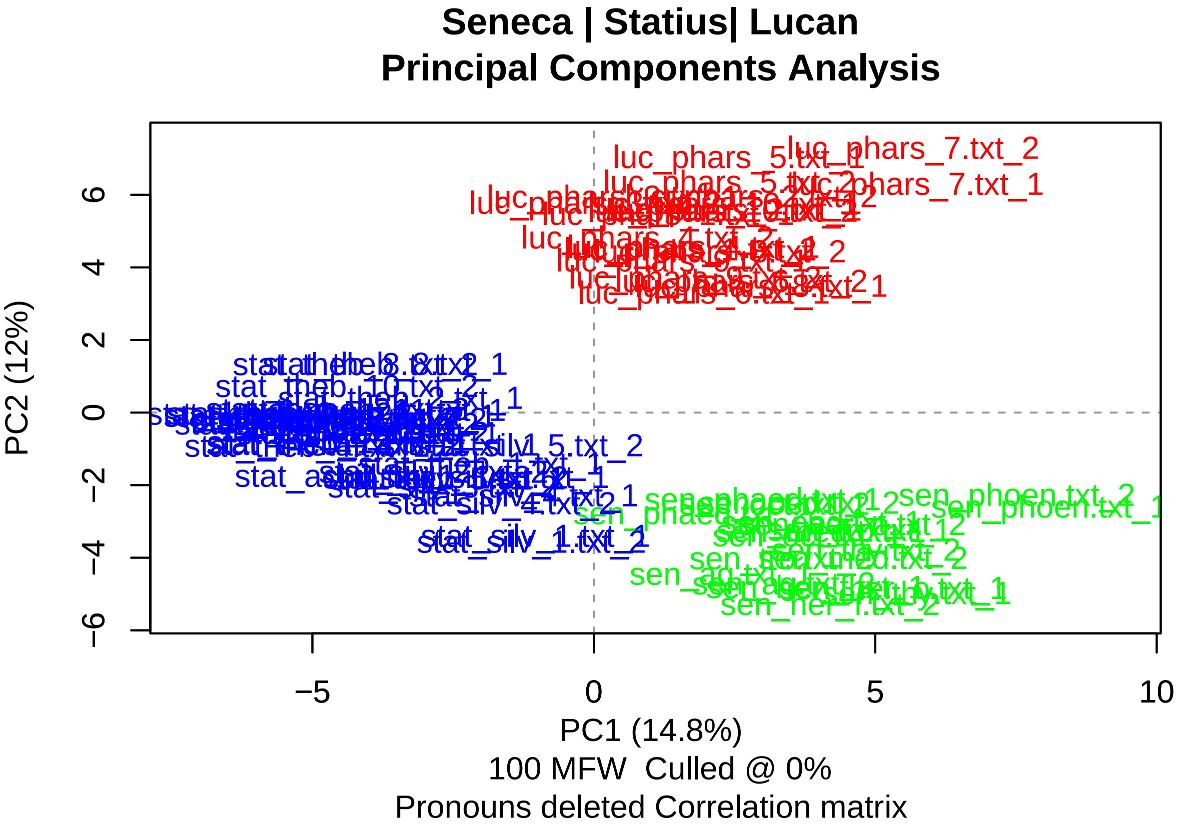
```
# PCA 100 | no culling to obtain a sufficient number of features
results_pca_4grams_cor = stylo(frequencies = freqs.word.grams,
                               analysis.type = "PCR",
                               mfw.min = 100, mfw.max = 100, #look at this small number of words becaus
                               distance.measure = "eder",
                               custom.graph.title = "Seneca | Statius| Lucan",
                               write.png.file = T,
                               pca.visual.flavour = "loadings", # too many words if set to 1000 or more
                               gui = T)
```

```
## using current directory...

## Warning in delete.stop.words(table.with.all.freqs, pronouns): chosen stop words were not found in the
##   please check the language, lower/uppercase issues, etc.

##

## culling @ 0  available features (words) 3000

## MFW used:

## 100
## Processing metadata...
##
##
## Assigning plot colors according to file names...
##
##
```

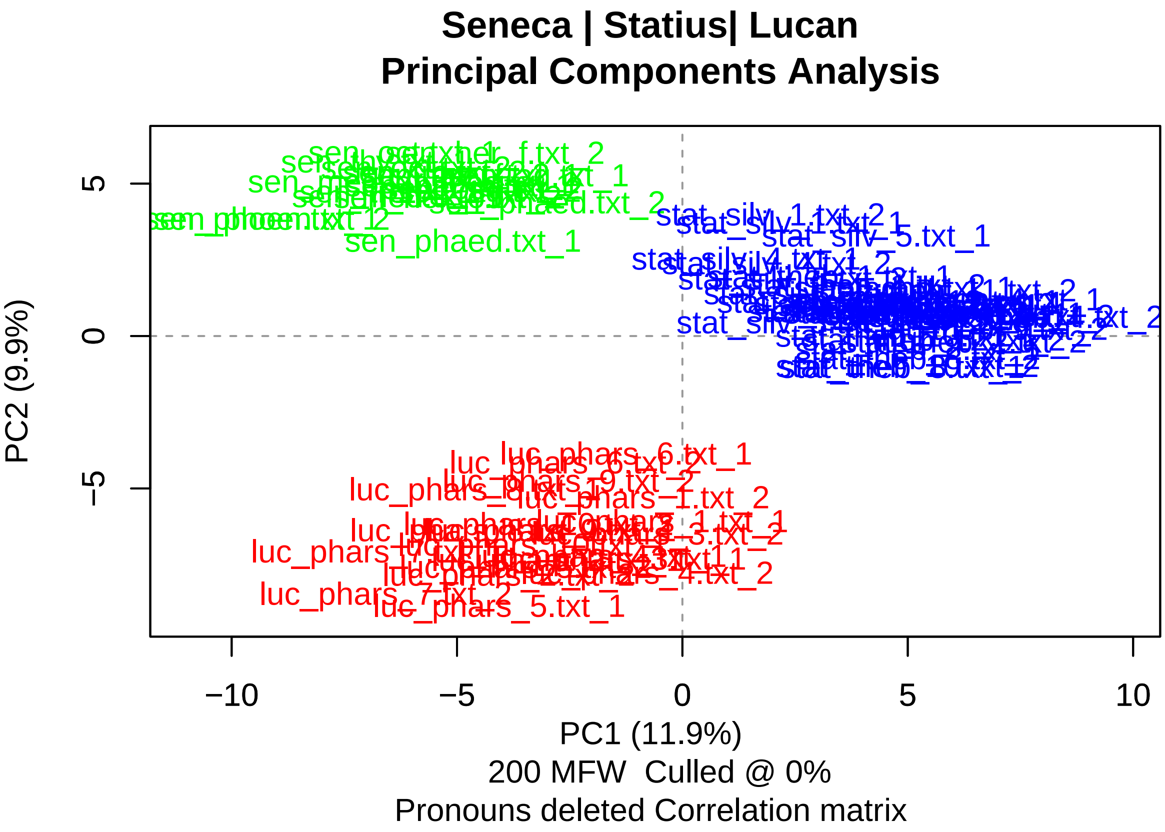


```
# same visualisation without the words, only with the filename
# include a bigger range of MFW to show the robustness of the results
# PCA 100-1500 MFW | no culling to obtain a sufficient number of features
results_pca_4grams_cor = stylo(frequencies = freqs.word.grams,
                               analysis.type = "PCR",
                               mfw.min = 100, mfw.max = 1500, increment = 100,
                               distance.measure = "eder",
                               custom.graph.title = "Seneca | Statius| Lucan",
                               write.png.file = T,
                               pca.visual.flavour = "classic",
                               gui = T)
```

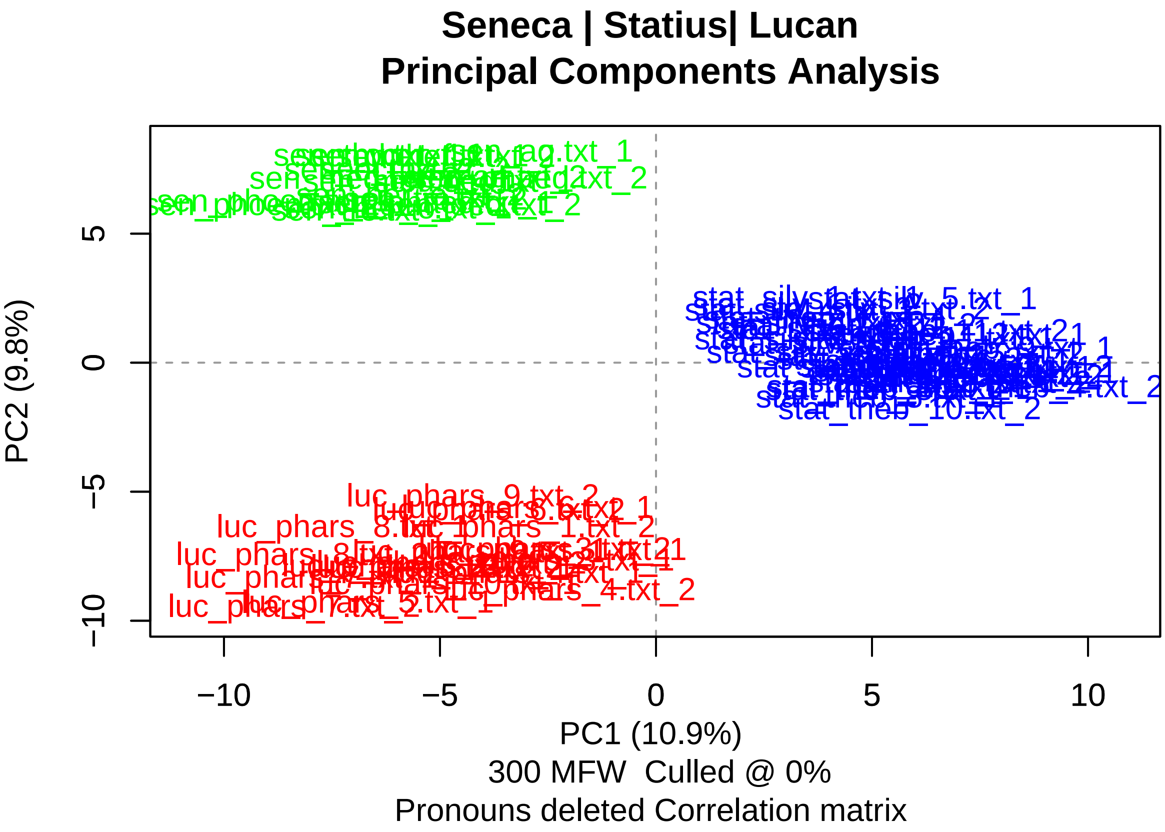
```
## using current directory...

## Warning in delete.stop.words(table.with.all.freqs, pronouns): chosen stop words were not found in the
##   please check the language, lower/uppercase issues, etc.

##

## culling @ 0  available features (words) 3000

## MFW used:

## 100
## Processing metadata...
##
##
## Assigning plot colors according to file names...
##
## 200
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



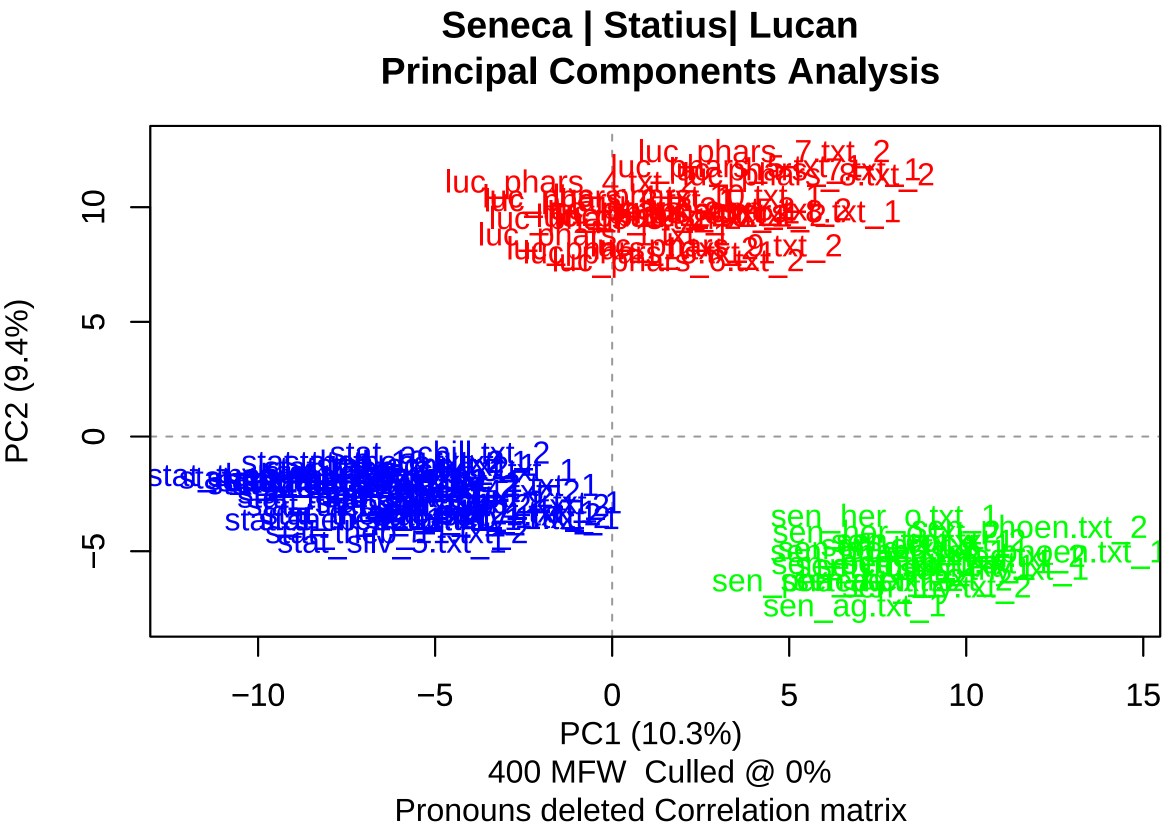
```
## 300
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



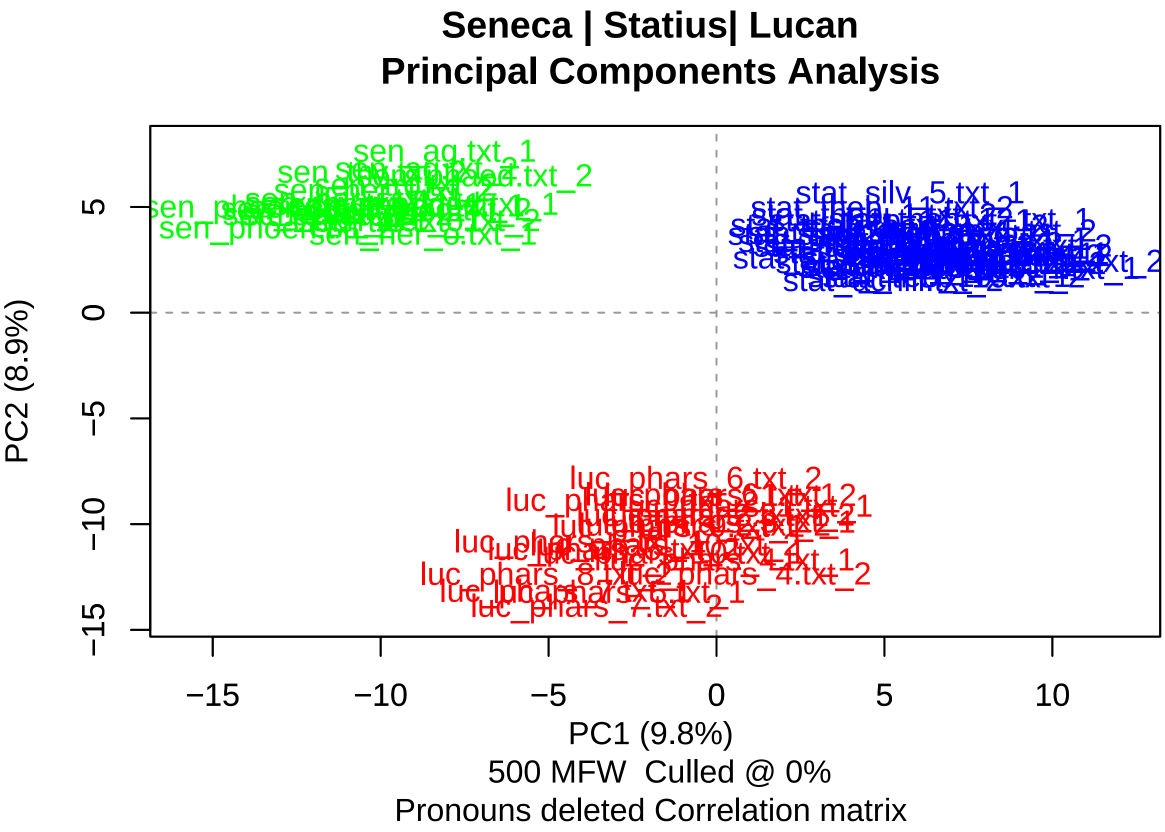
```
## 400
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



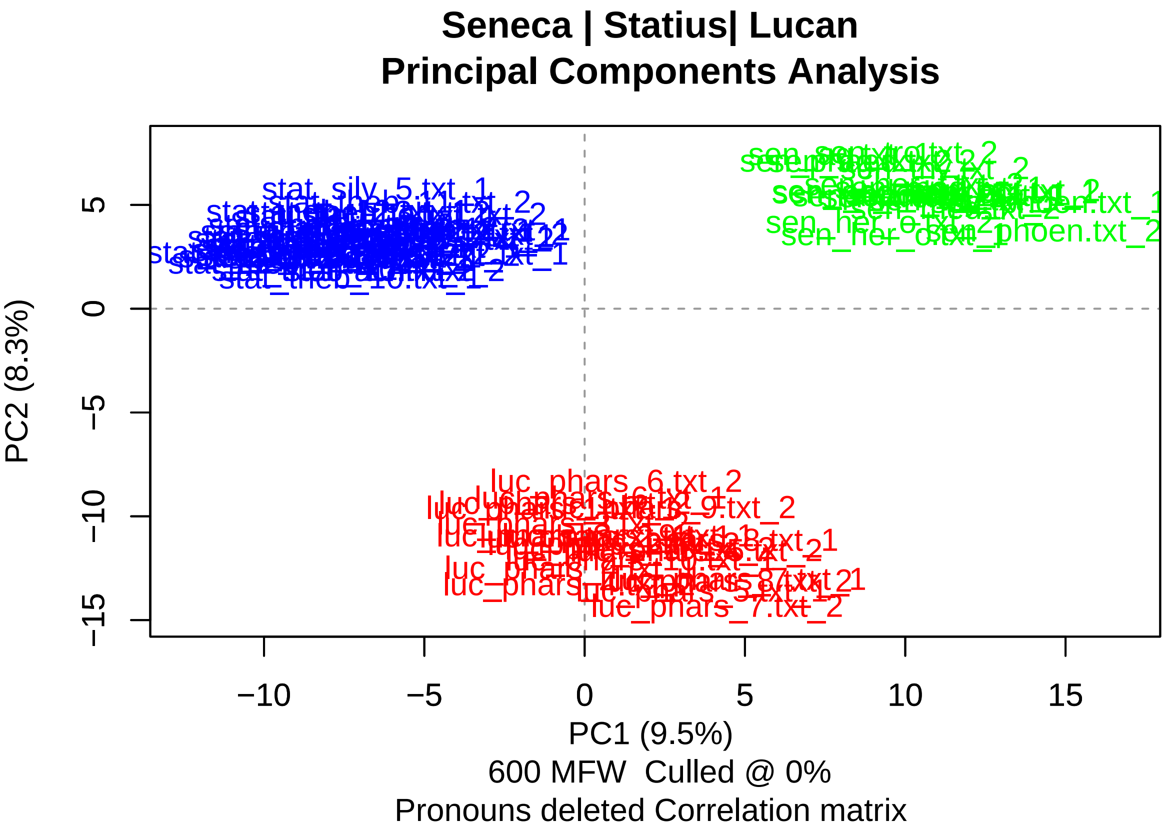
```
## 500
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



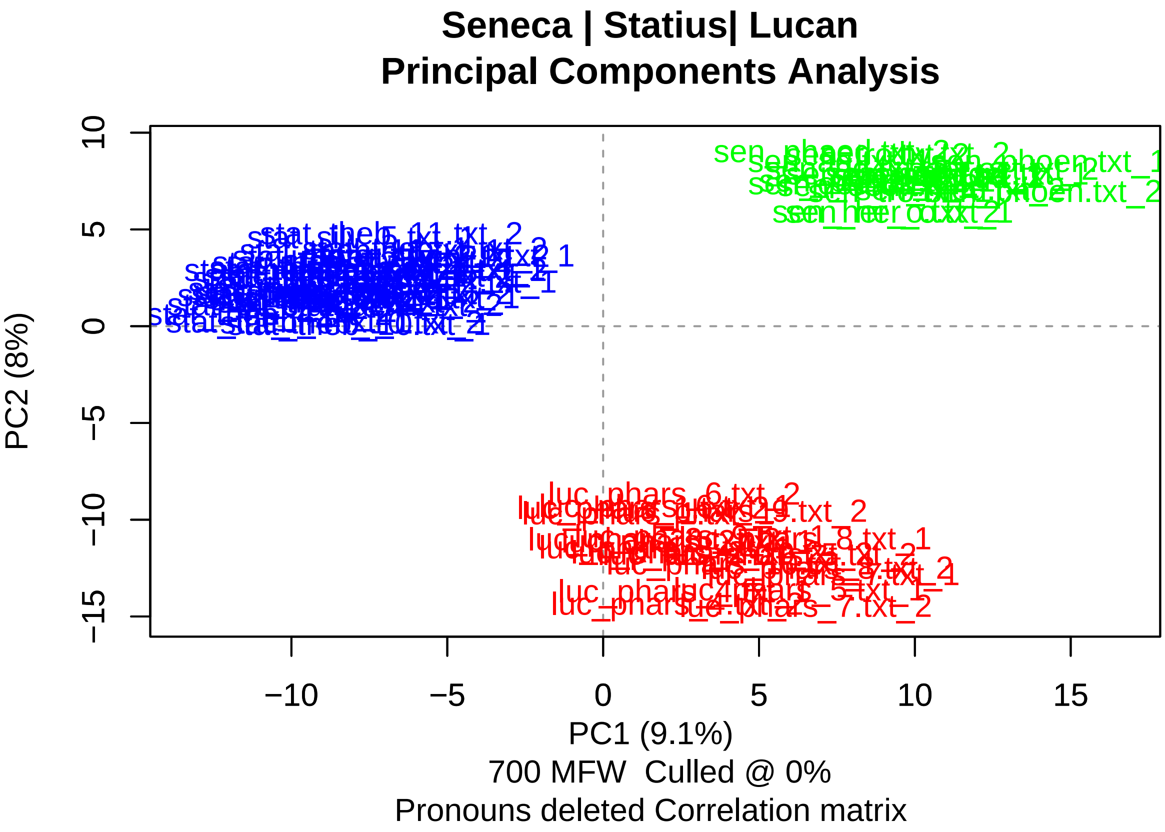
```
## 600
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



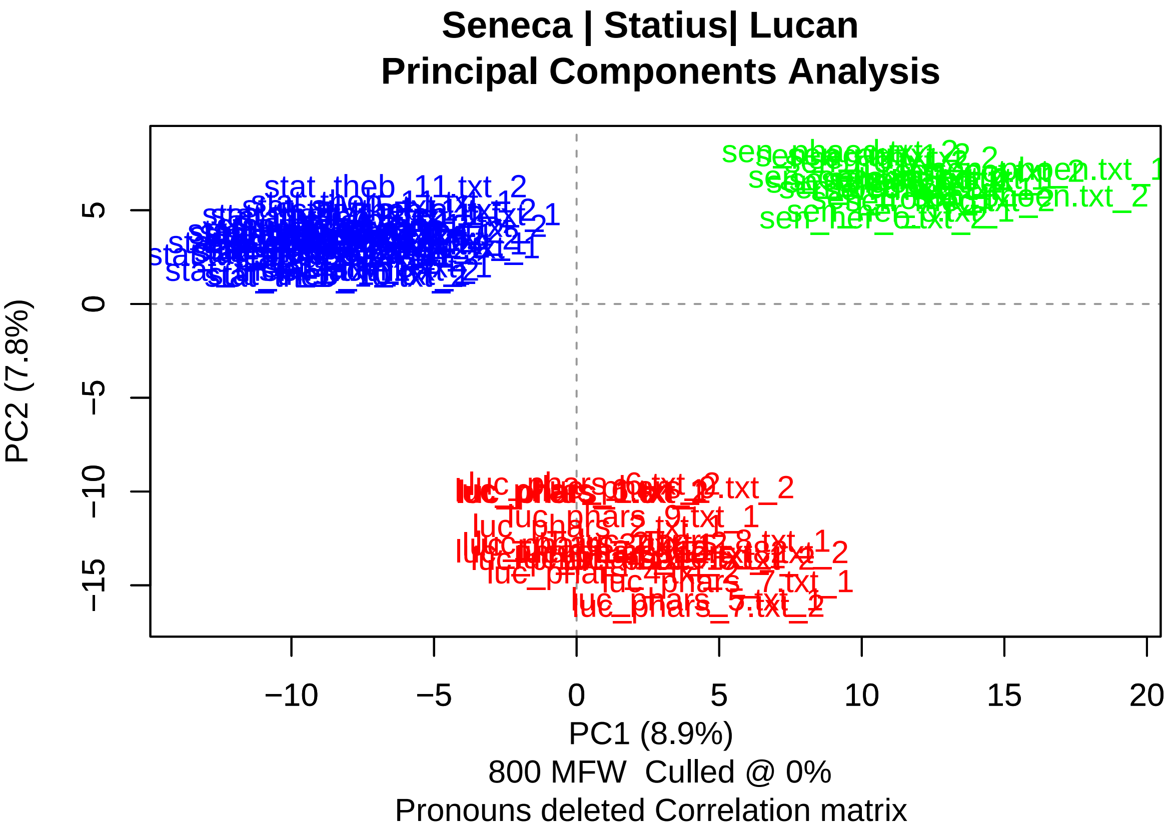
```
## 700
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



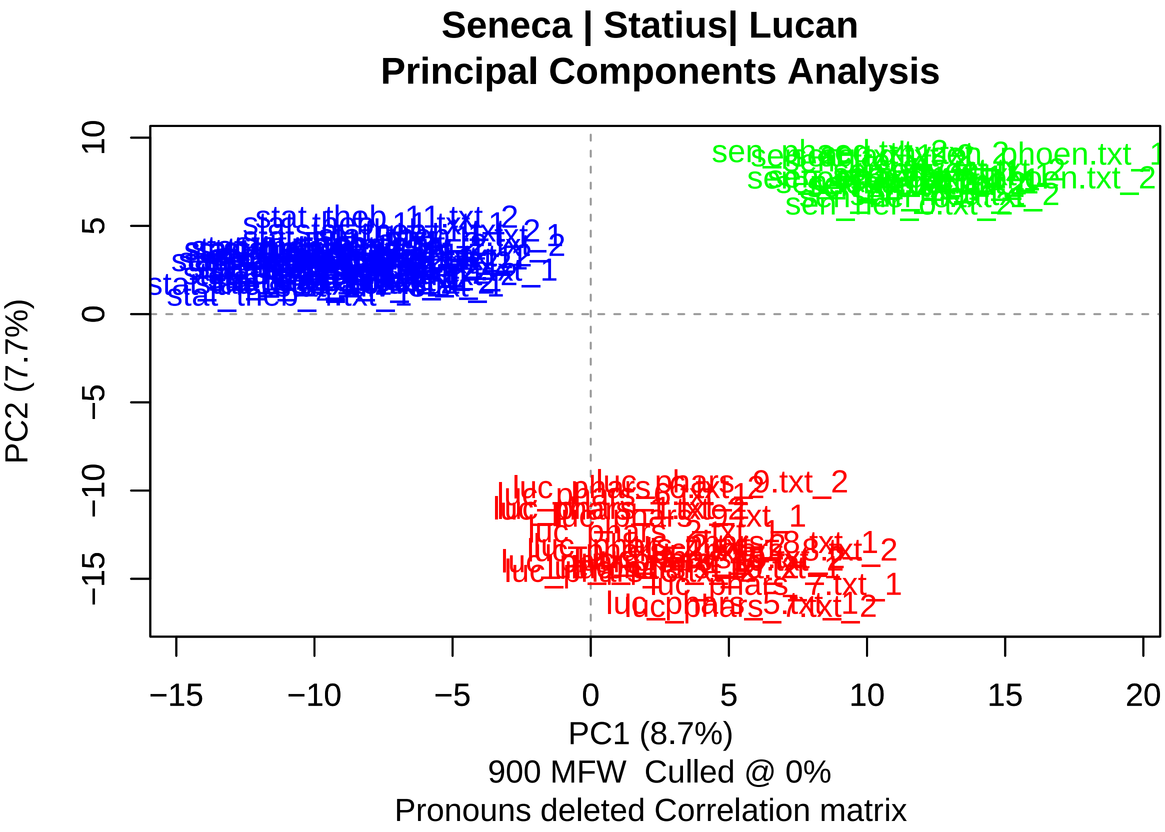
```
## 800
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



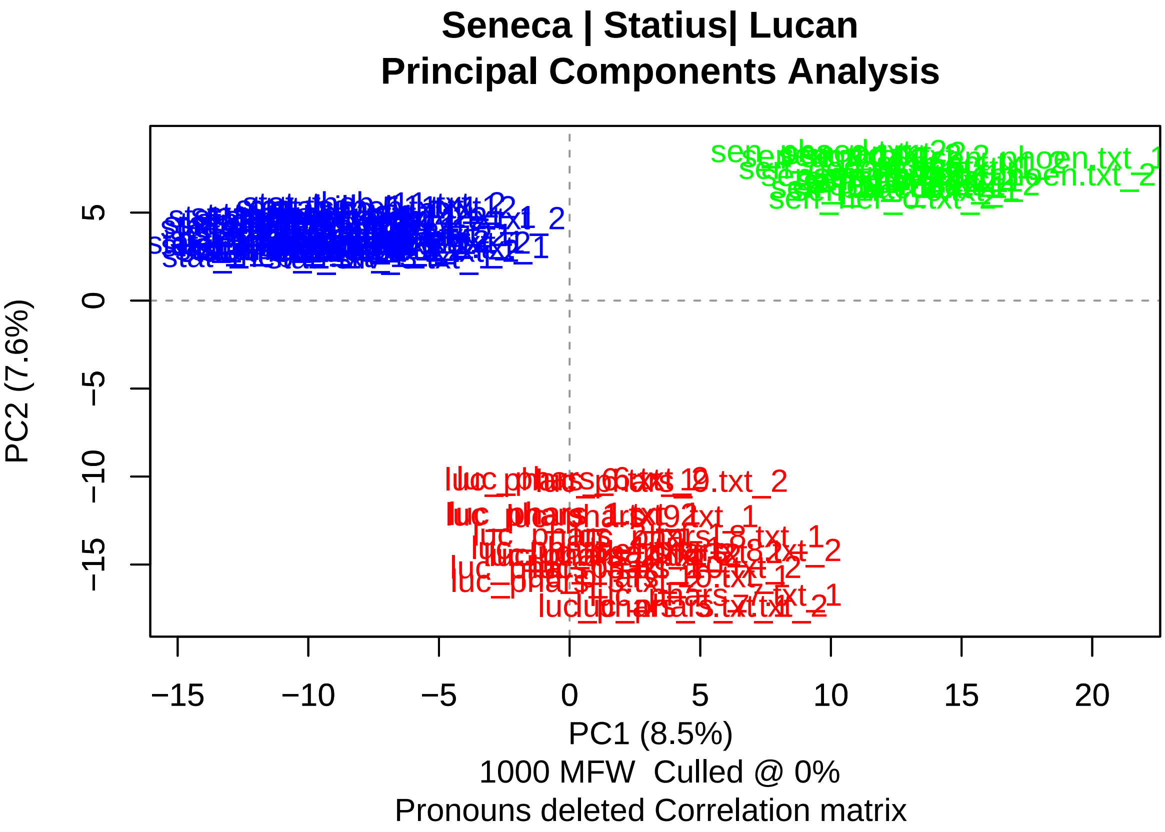
```
## 900
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



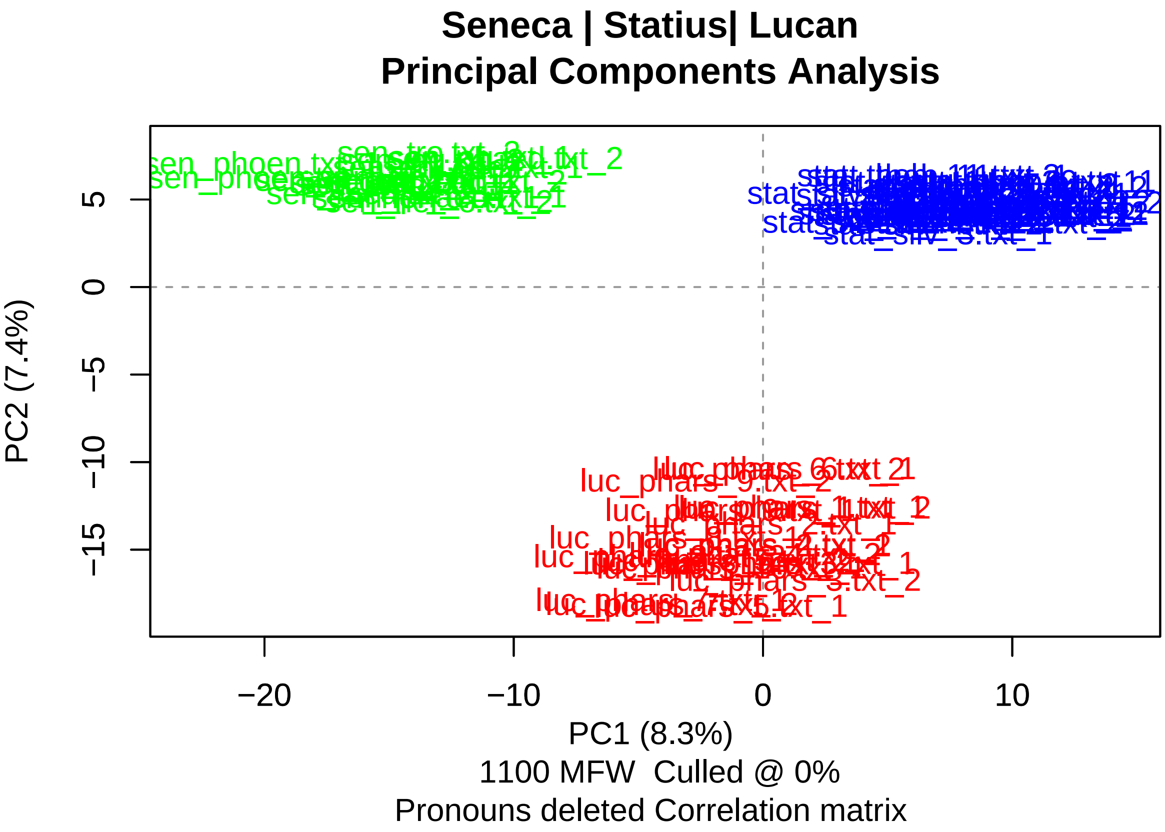
```
## 1000
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



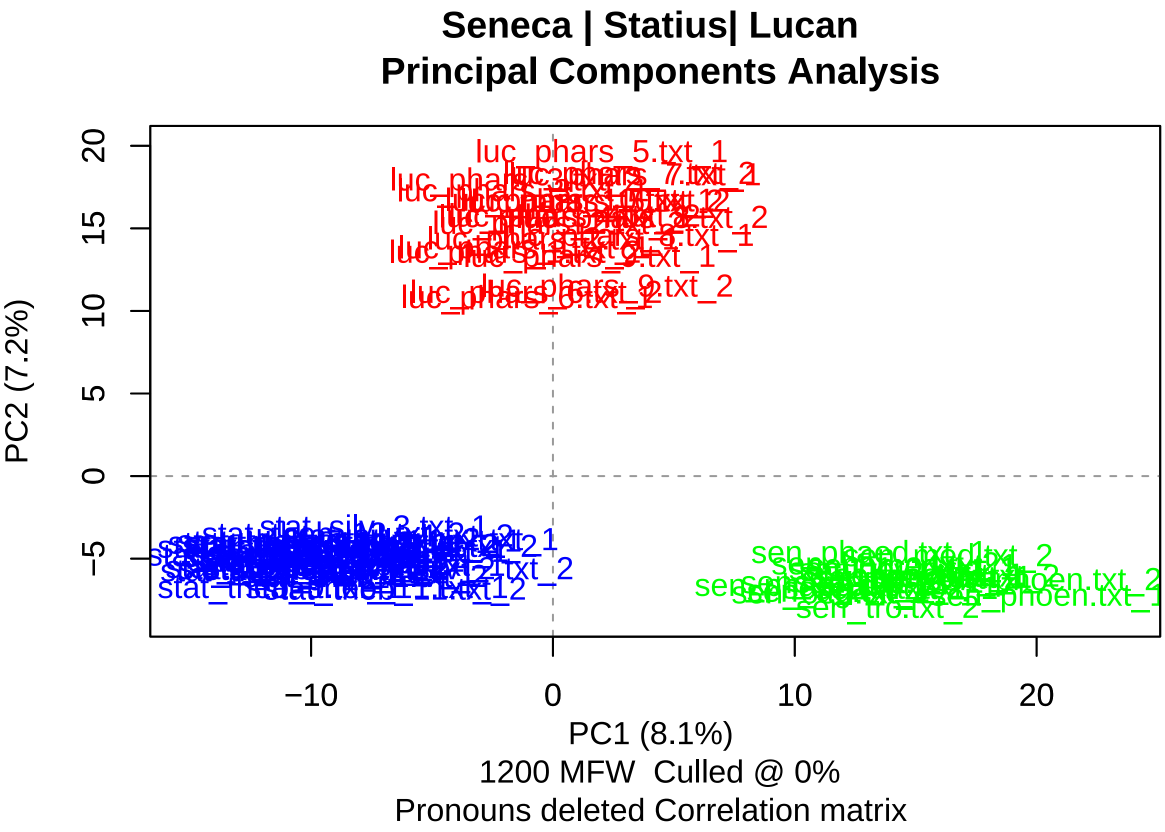
```
## 1100
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



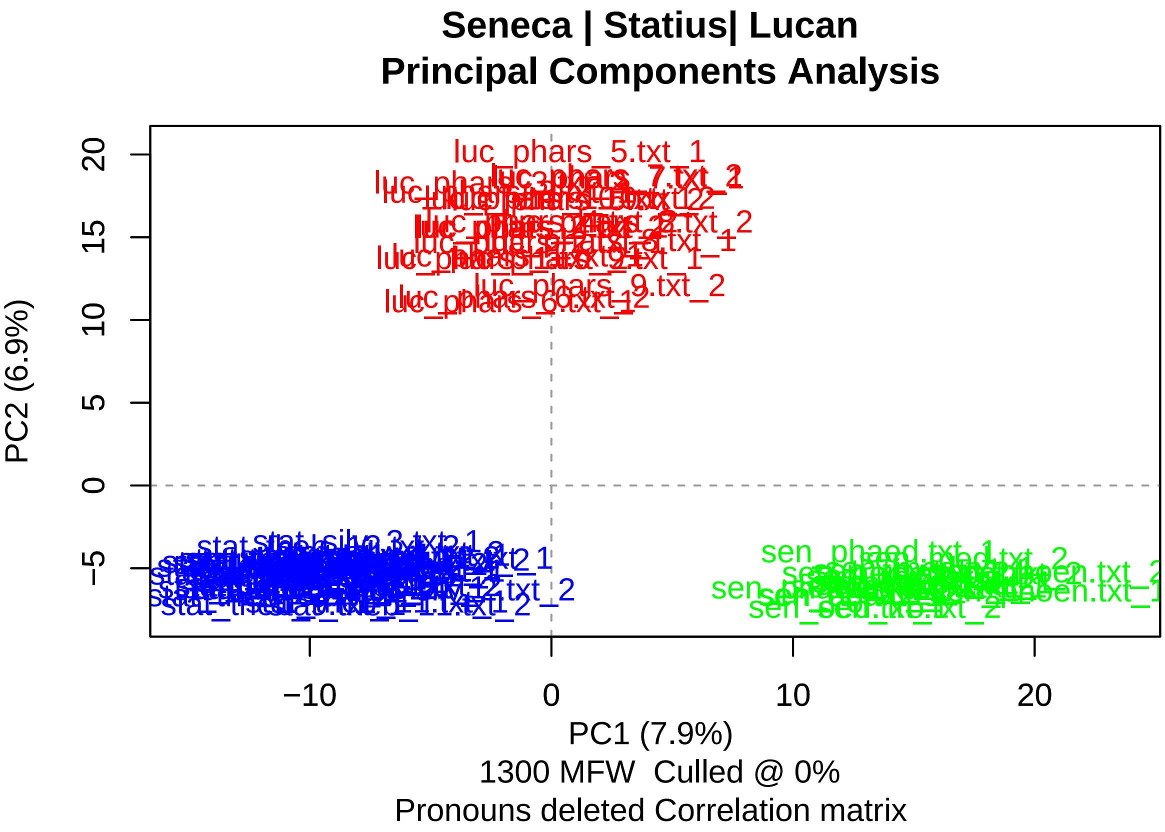
```
## 1200
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



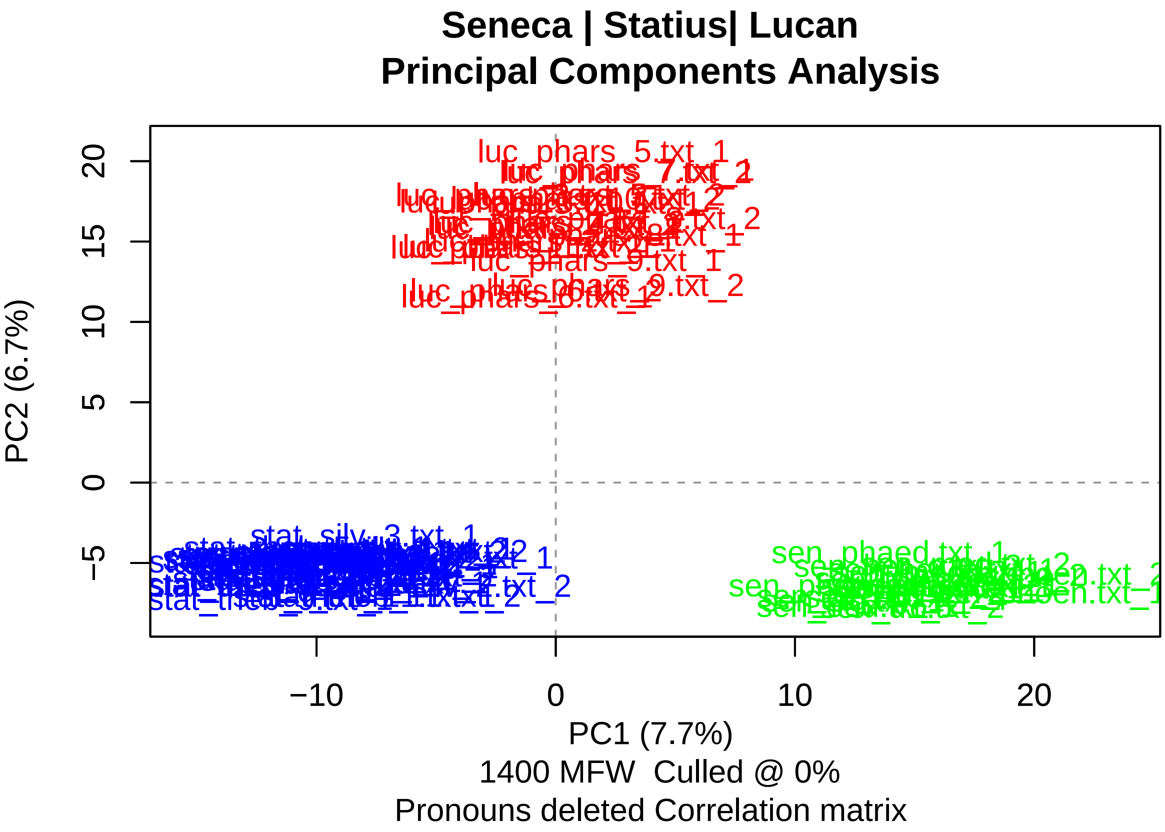
```
## 1300
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



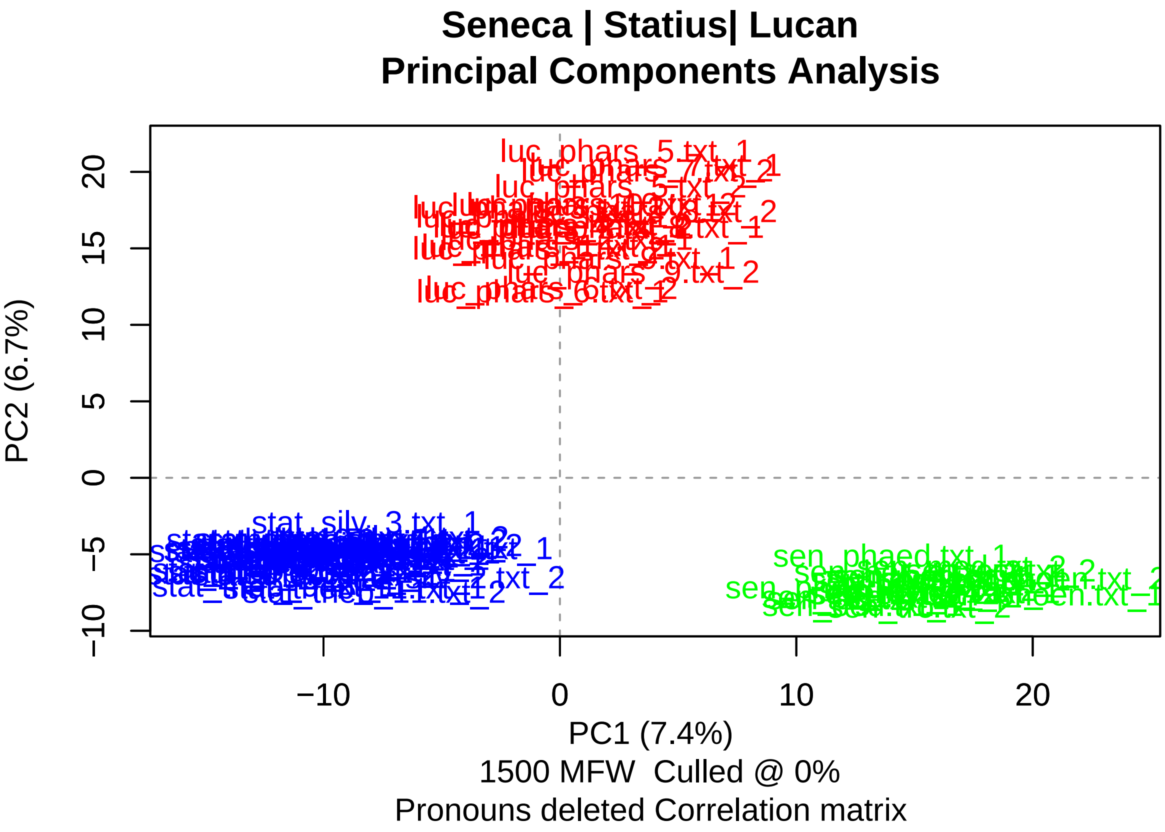
```
## 1400
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



```
## 1500
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```

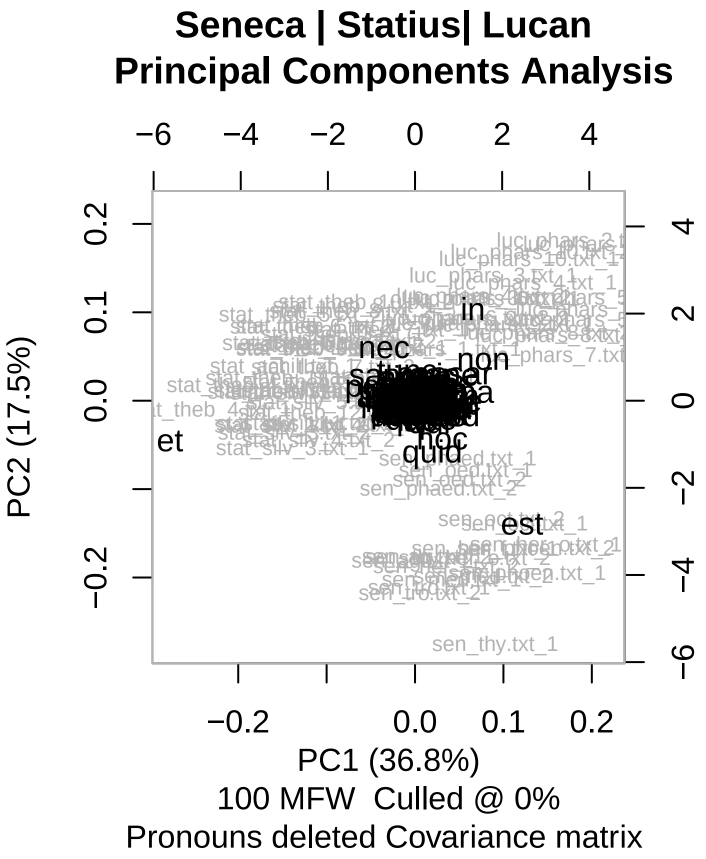

Seneca | Statius| Lucan
Principal Components Analysis
PC2 (6.7%)
20
15
10
5
0
−5
−10
0
10
20
PC1 (7.7%)
1400 MFW Culled @ 0%
Pronouns deleted Correlation matrix
luc_phars_5.txt_1
luc_phars_7.txt_1
luc_phars_9.txt_1
luc_phars_9.txt_2
luc_phars_6.txt_1
stat_silv_3.txt_1
sen_phaed.txt_1

##


Seneca | Statius| Lucan
Principal Components Analysis
PC2 (6.7%)
20
15
10
5
0
−5
−10
−10
0
10
20
PC1 (7.4%)
1500 MFW Culled @ 0%
Pronouns deleted Correlation matrix
luc_phars_5.txt_1
luc_phars_7.txt_1
luc_phars_9.txt_1
luc_phars_9.txt_2
luc_phars_6.txt_1
stat_silv_3.txt_1
sen_phaed.txt_1

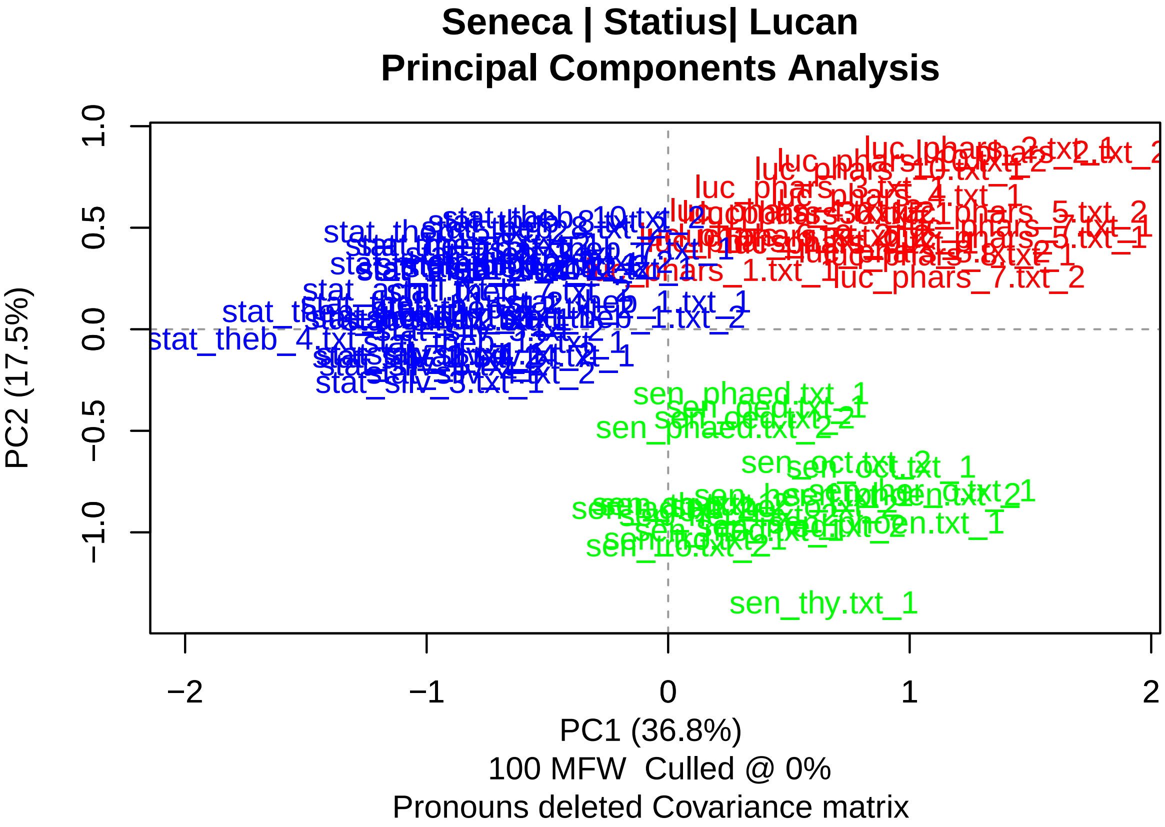
```r
# apply the same technique but this time using a covariance plot instead of a correlation
# first one will be technical, the second one will be without the words
# using a broader range of words to test the robustness of the results
results_pca_4grams_cov_1 = stylo(frequencies = freqs.word.grams,
                                 analysis.type = "PCR",
                                 mfw.min = 100, mfw.max = 100,
                                 distance.measure = "eder",
                                 custom.graph.title = "Seneca | Statius| Lucan",
                                 write.png.file = T,
                                 pca.visual.flavour = "loadings",
                                 gui = T)
```

```
## using current directory...

## Warning in delete.stop.words(table.with.all.freqs, pronouns): chosen stop words were not found in the
##   please check the language, lower/uppercase issues, etc.

##

## culling @ 0  available features (words) 3000

## MFW used:

## 100
## Processing metadata...
##
##
## Assigning plot colors according to file names...
##
##
```

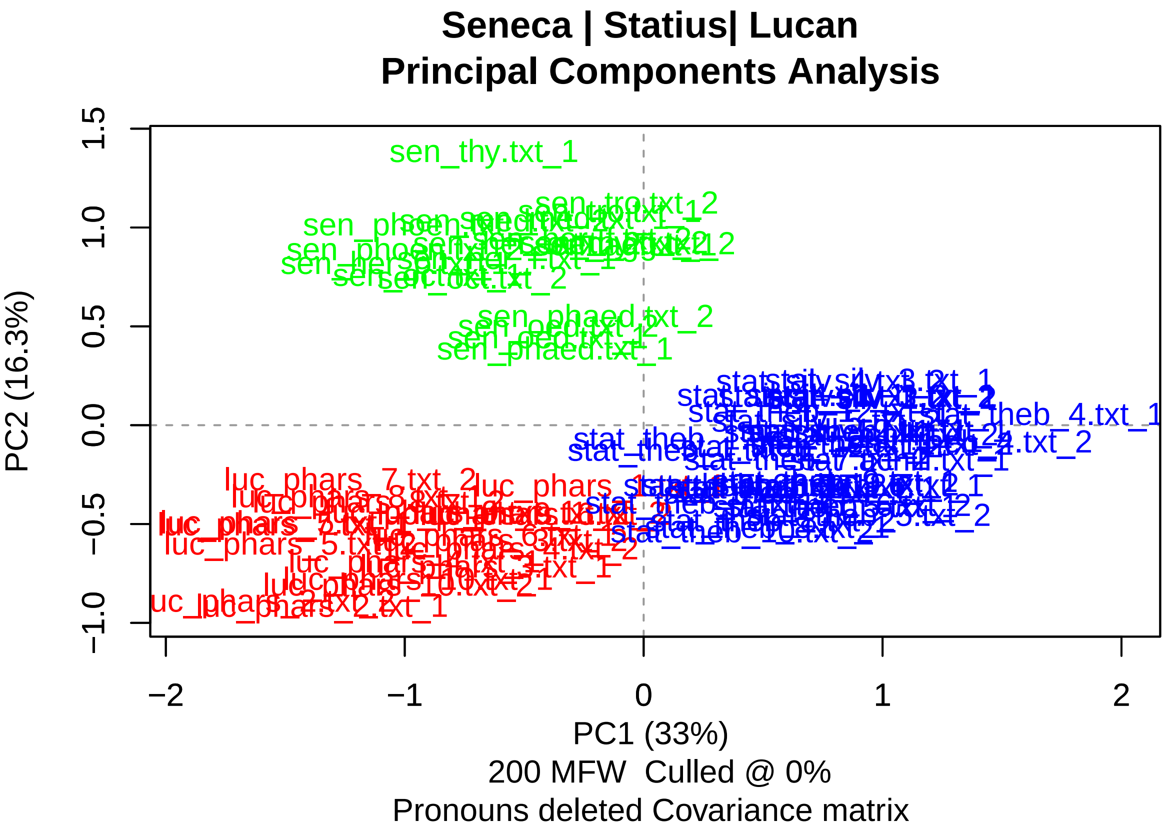


```r
# same visualization without the words, only with the filename
# include a bigger range of MFW to show the robustness of the results
# PCA 100-1500 MFW | no culling to obtain a sufficient number of features
results_pca_4grams_cov_2 = stylo(frequencies = freqs.word.grams,
                                 analysis.type = "PCR",
                                 mfw.min = 100, mfw.max = 1500, increment = 100,
                                 distance.measure = "eder",
                                 custom.graph.title = "Seneca | Statius| Lucan",
                                 write.png.file = T,
                                 pca.visual.flavour = "classic",
                                 gui = T)
```

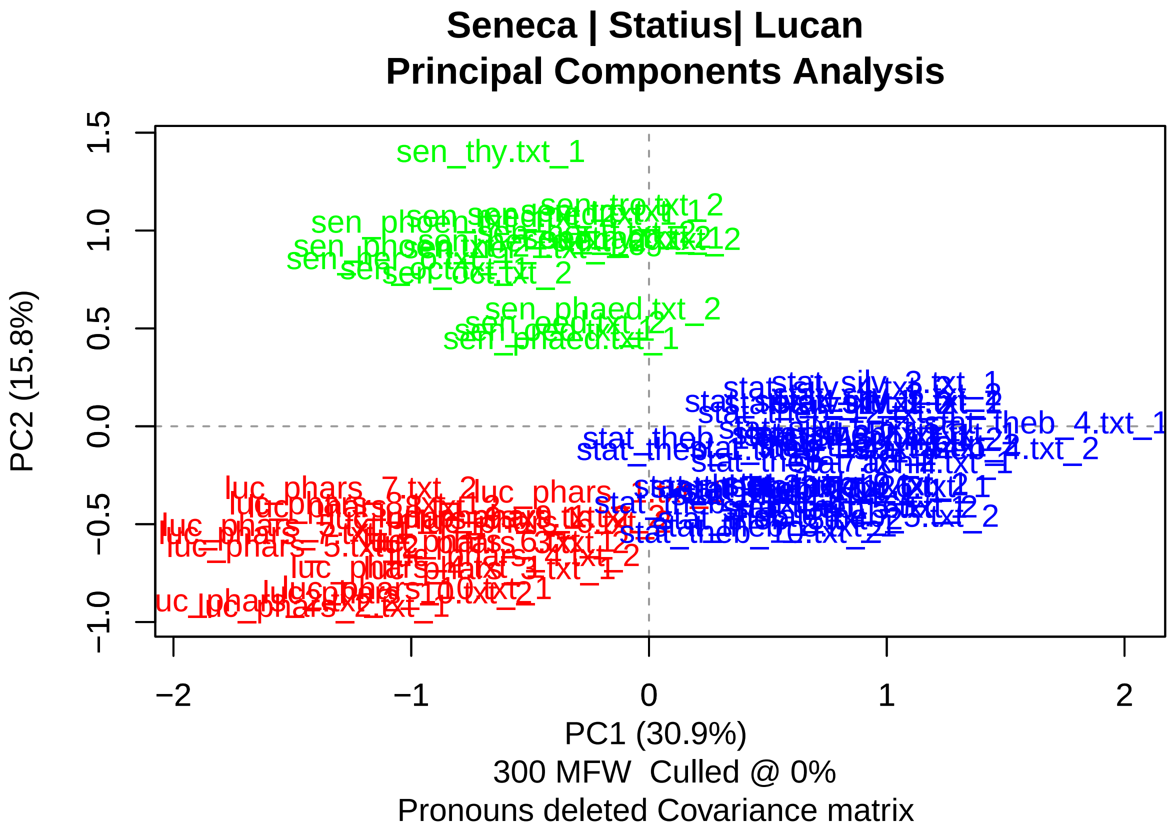
```
## using current directory...

## Warning in delete.stop.words(table.with.all.freqs, pronouns): chosen stop words were not found in th
##   please check the language, lower/uppercase issues, etc.

##

## culling @ 0  available features (words) 3000

## MFW used:

## 100
## Processing metadata...
##
##
## Assigning plot colors according to file names...
##
## 200
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



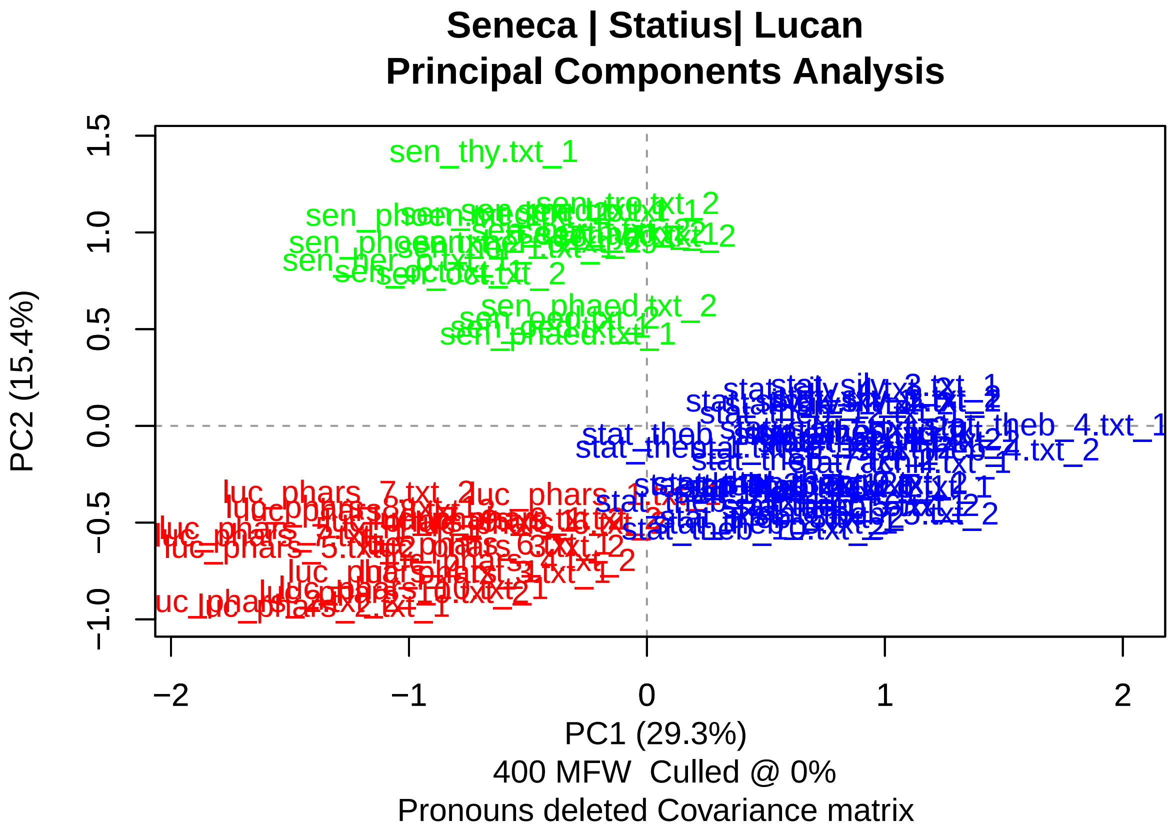
```
## 300
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



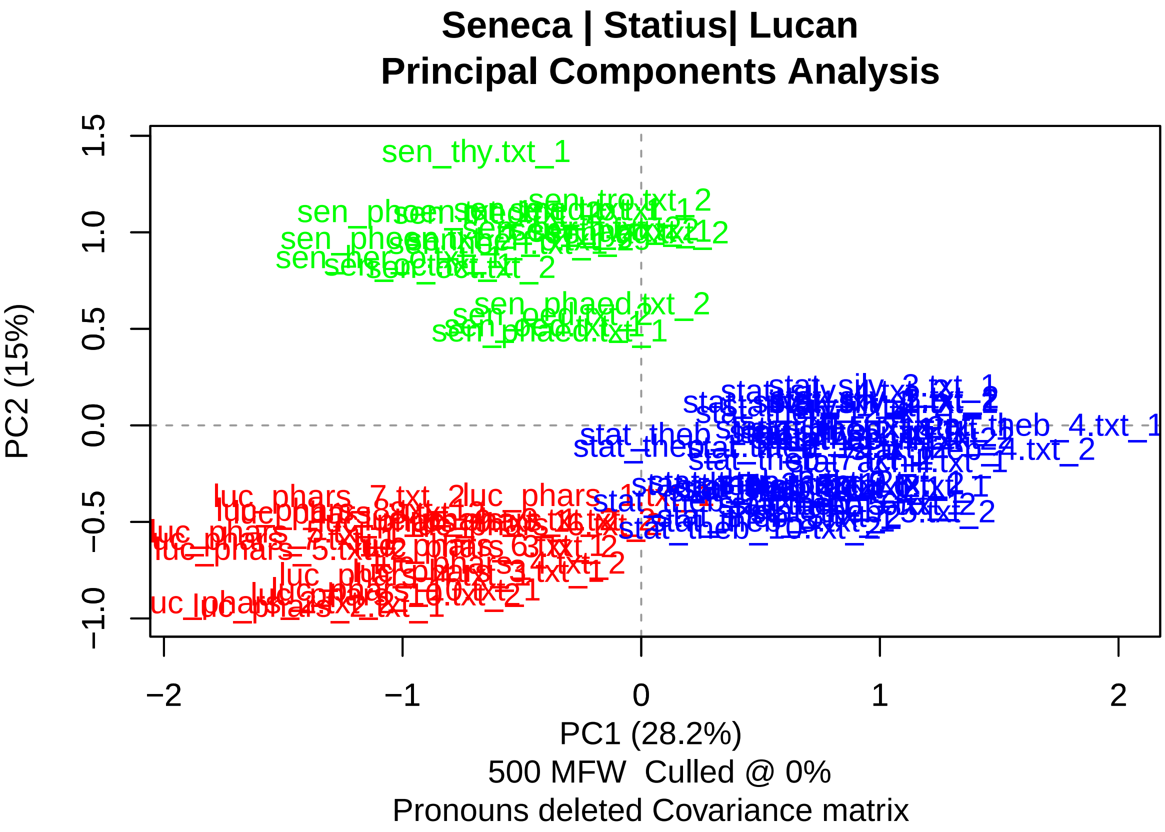
```
## 400
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



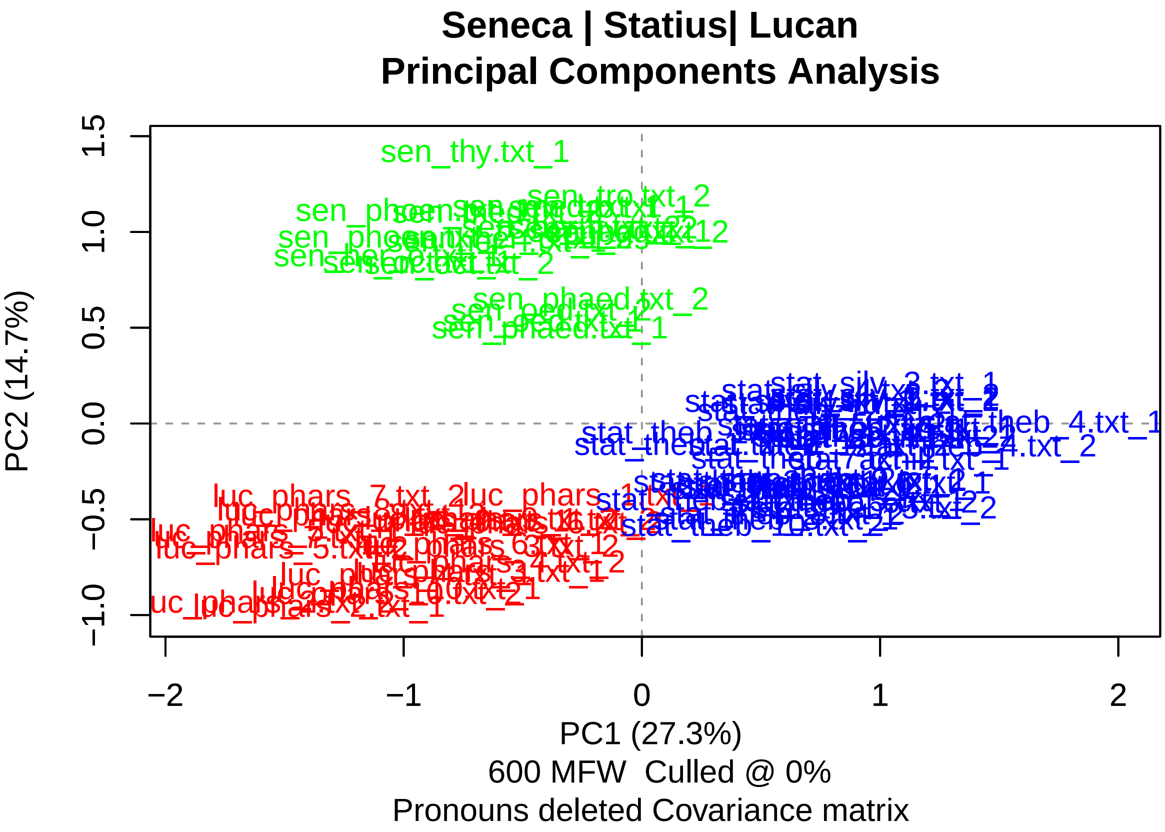
```
## 500
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



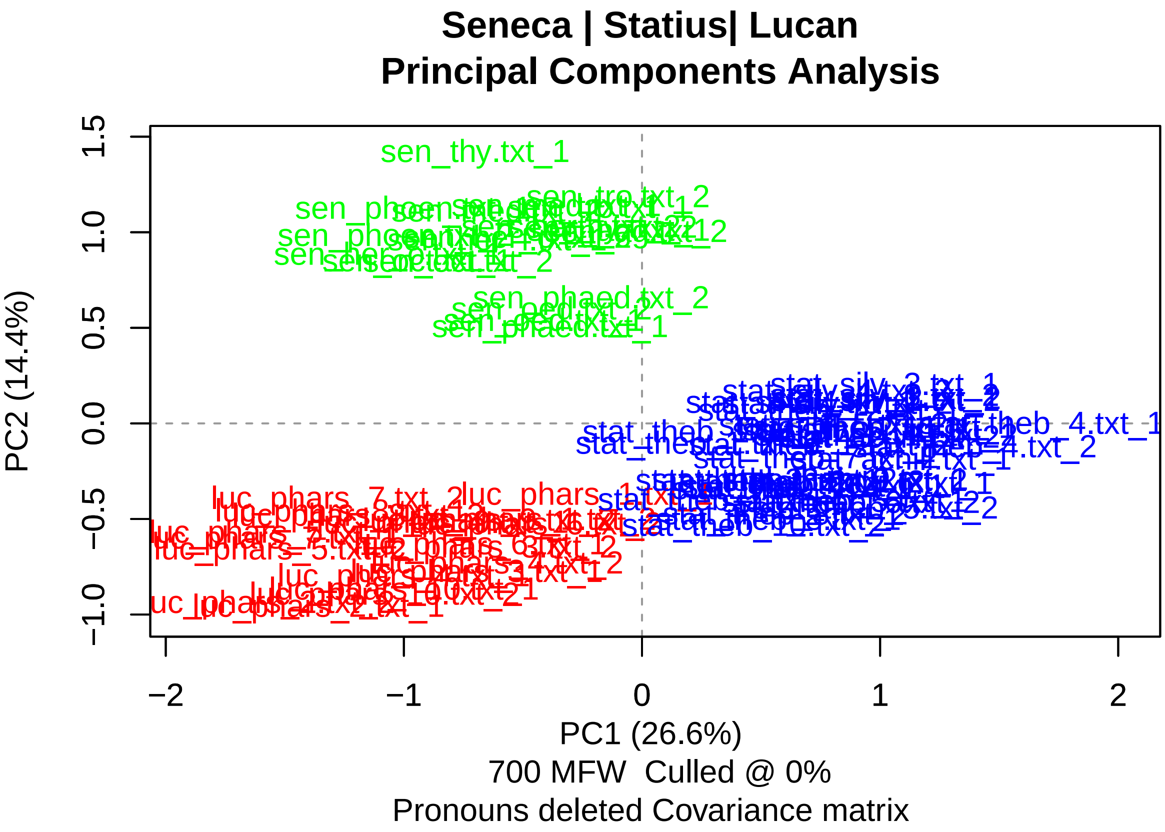
```
## 600
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



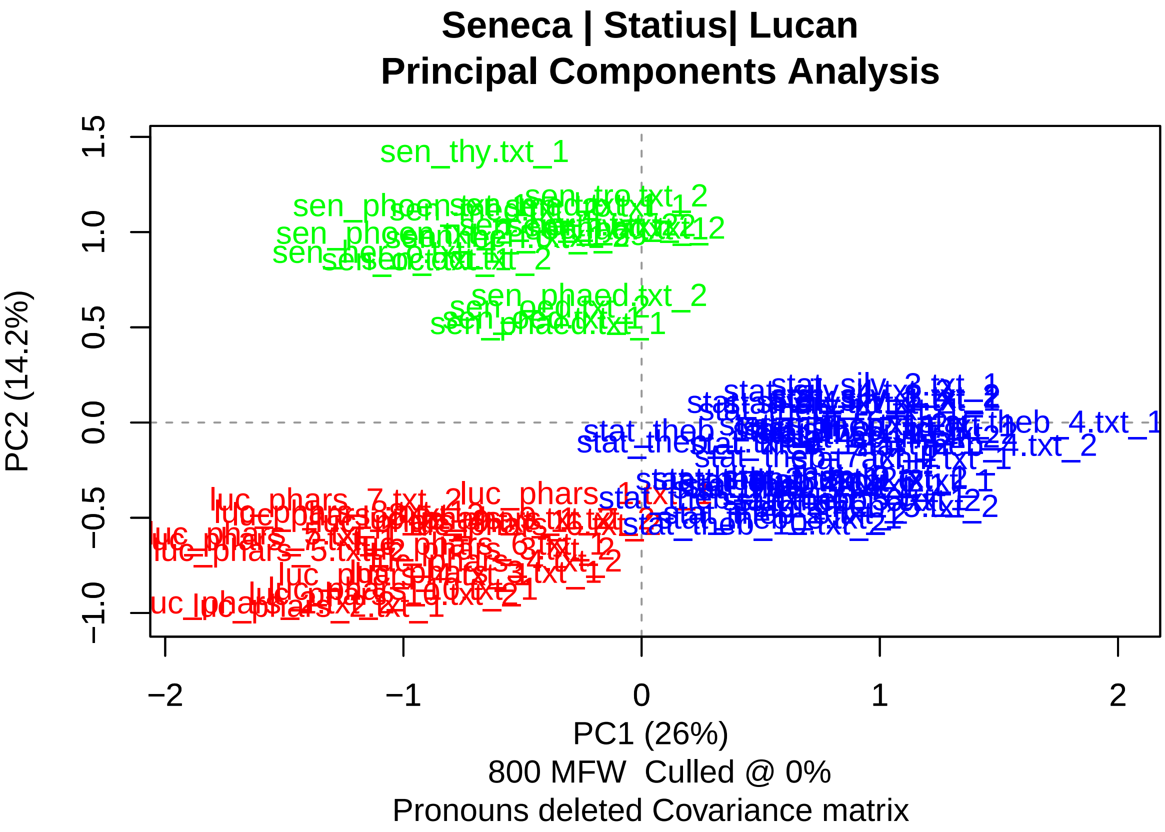
```
## 700
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



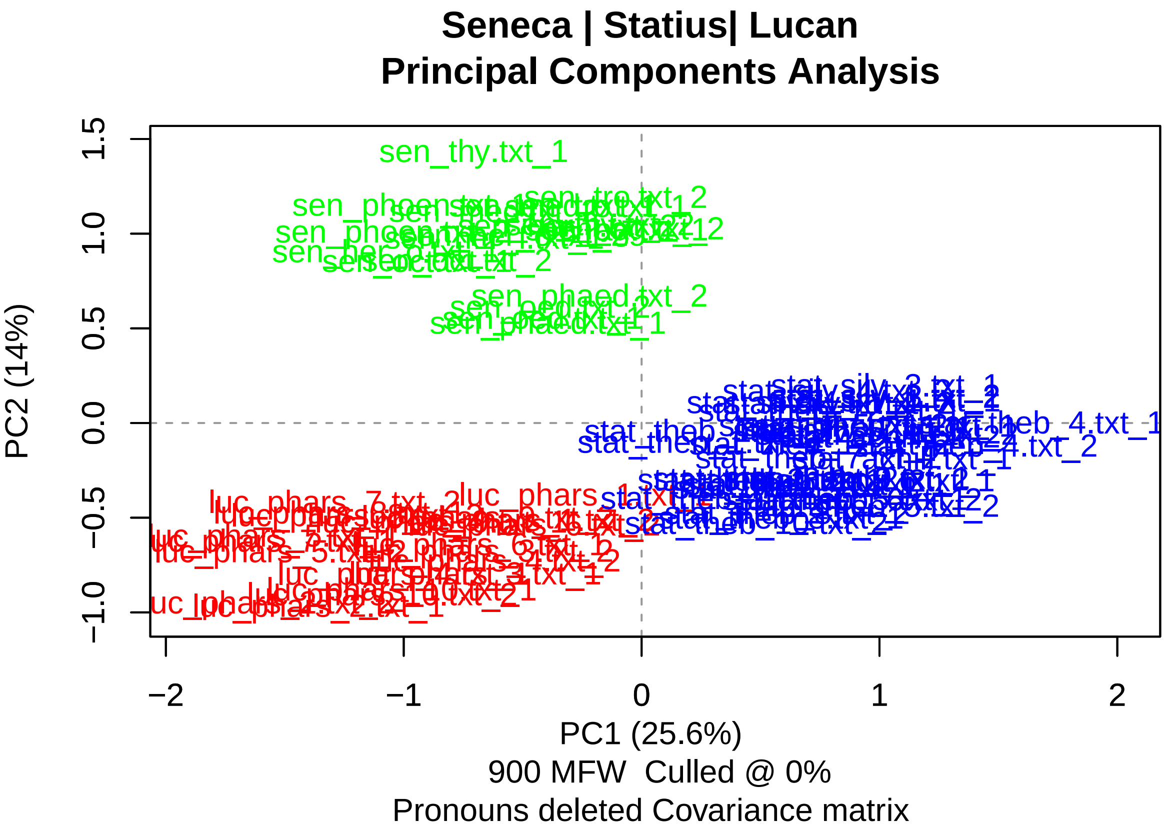
```
## 800
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



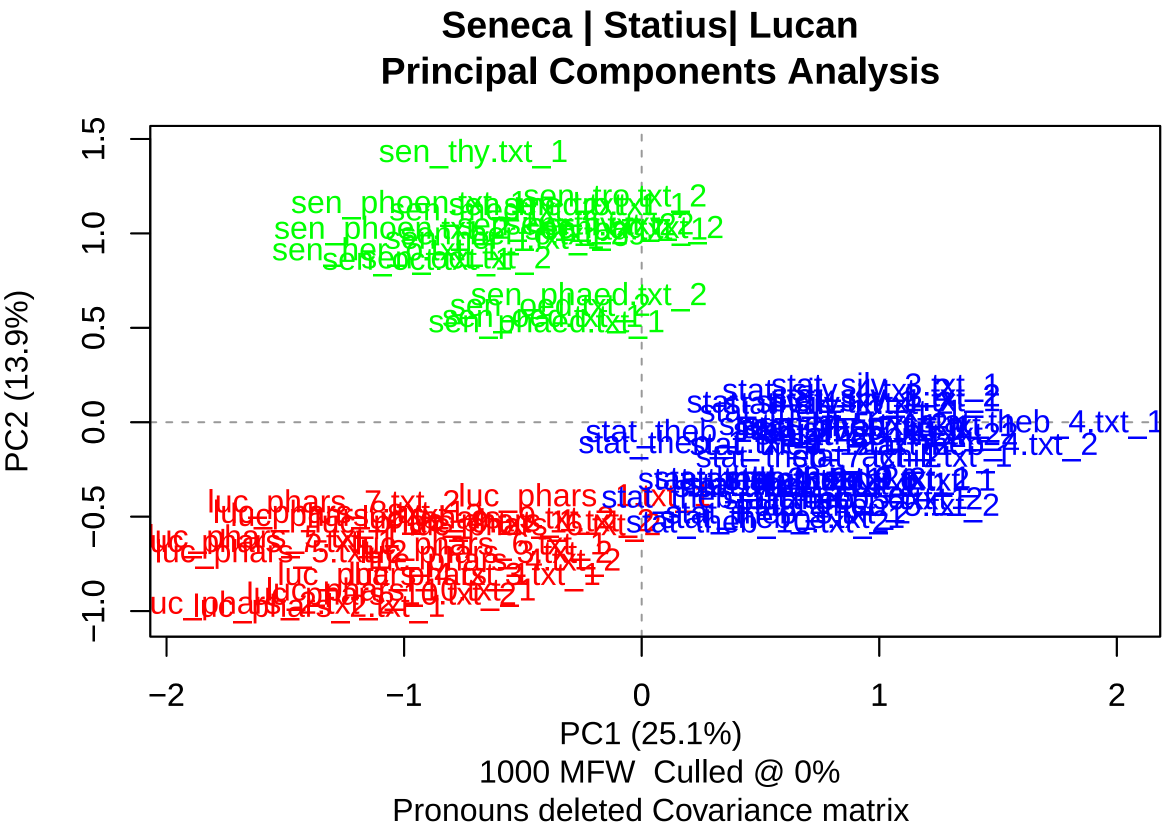
```
## 900
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



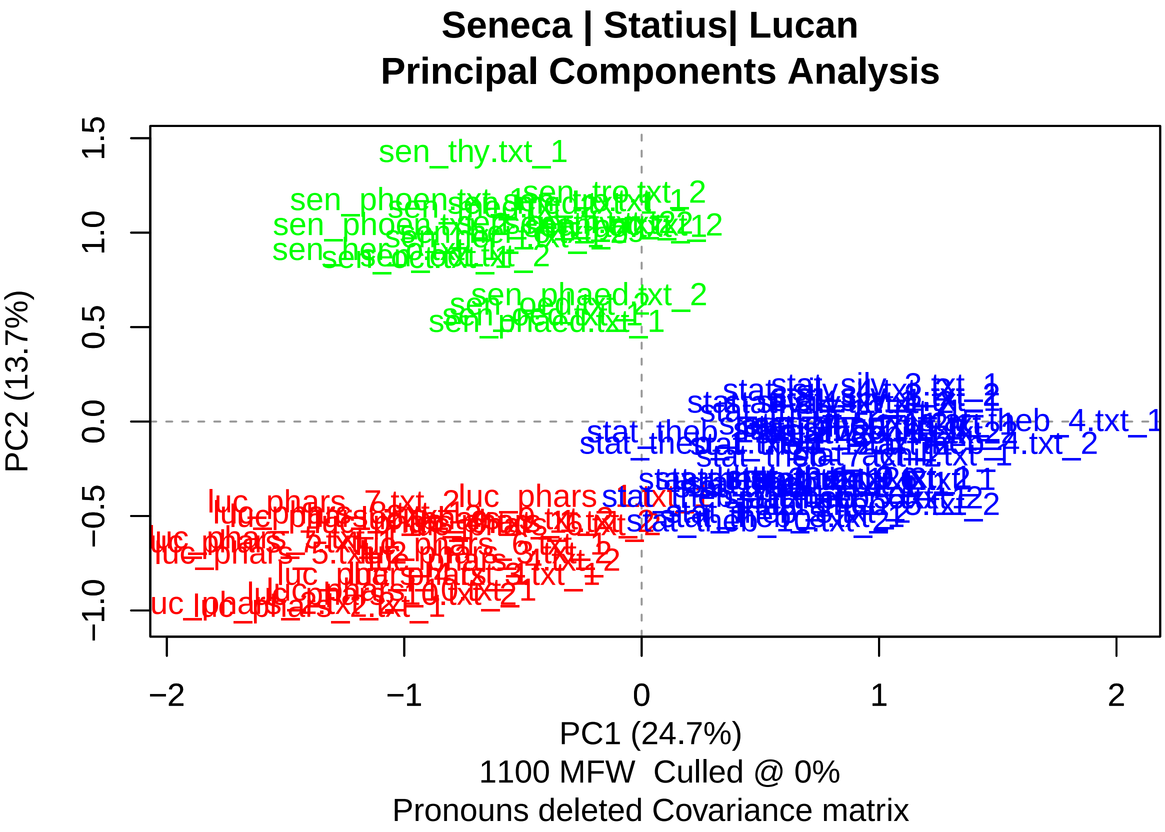
```
## 1000
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



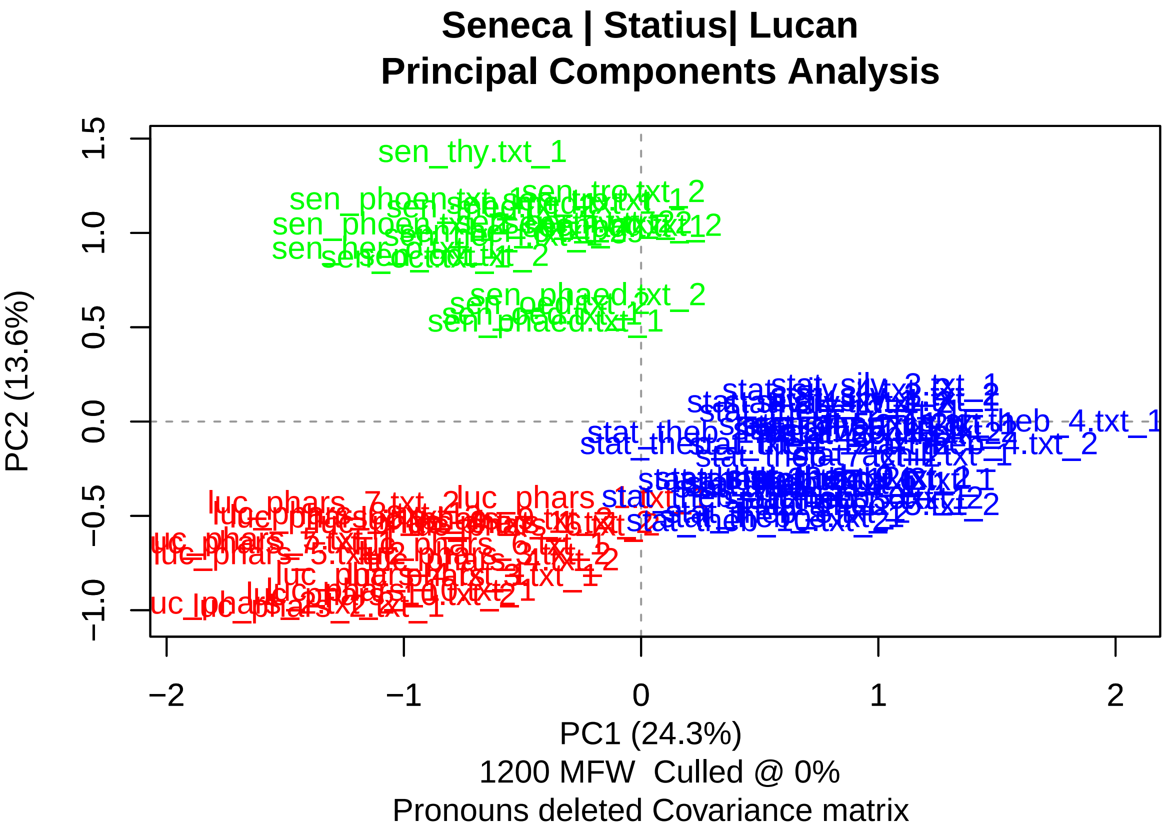
```
## 1100
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



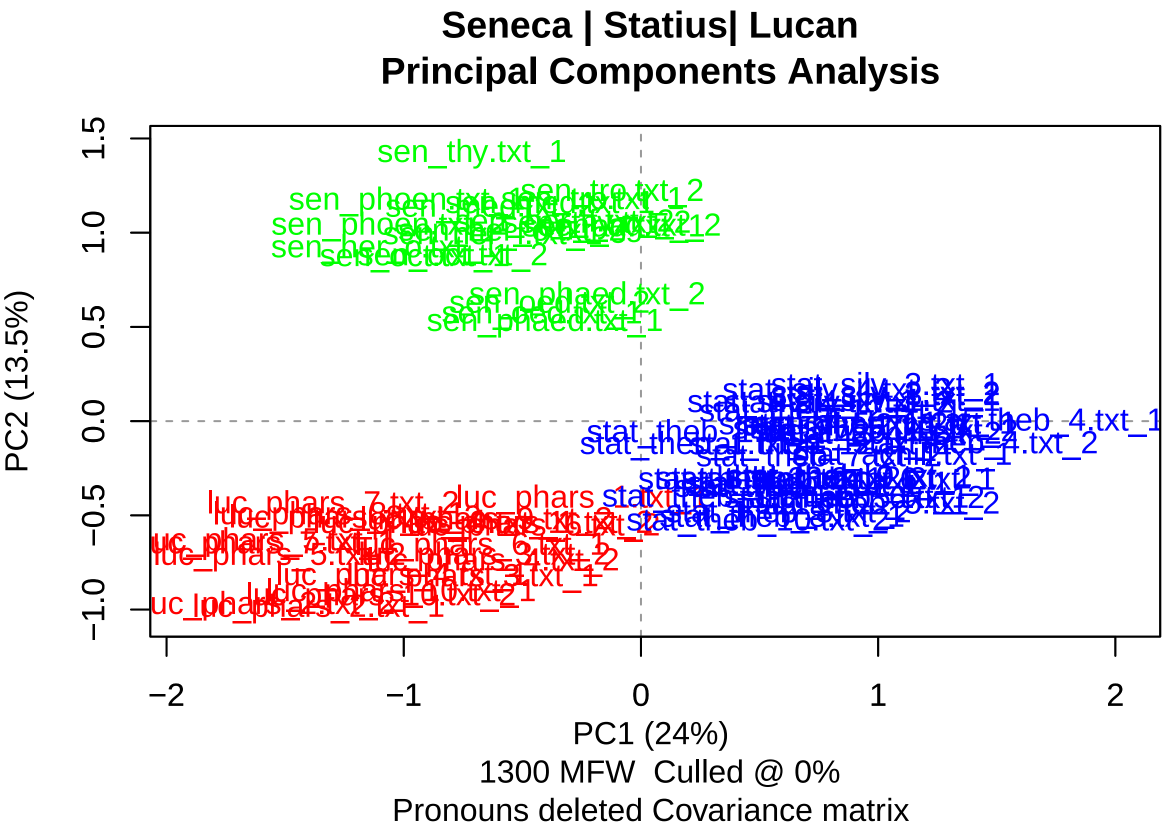
```
## 1200
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



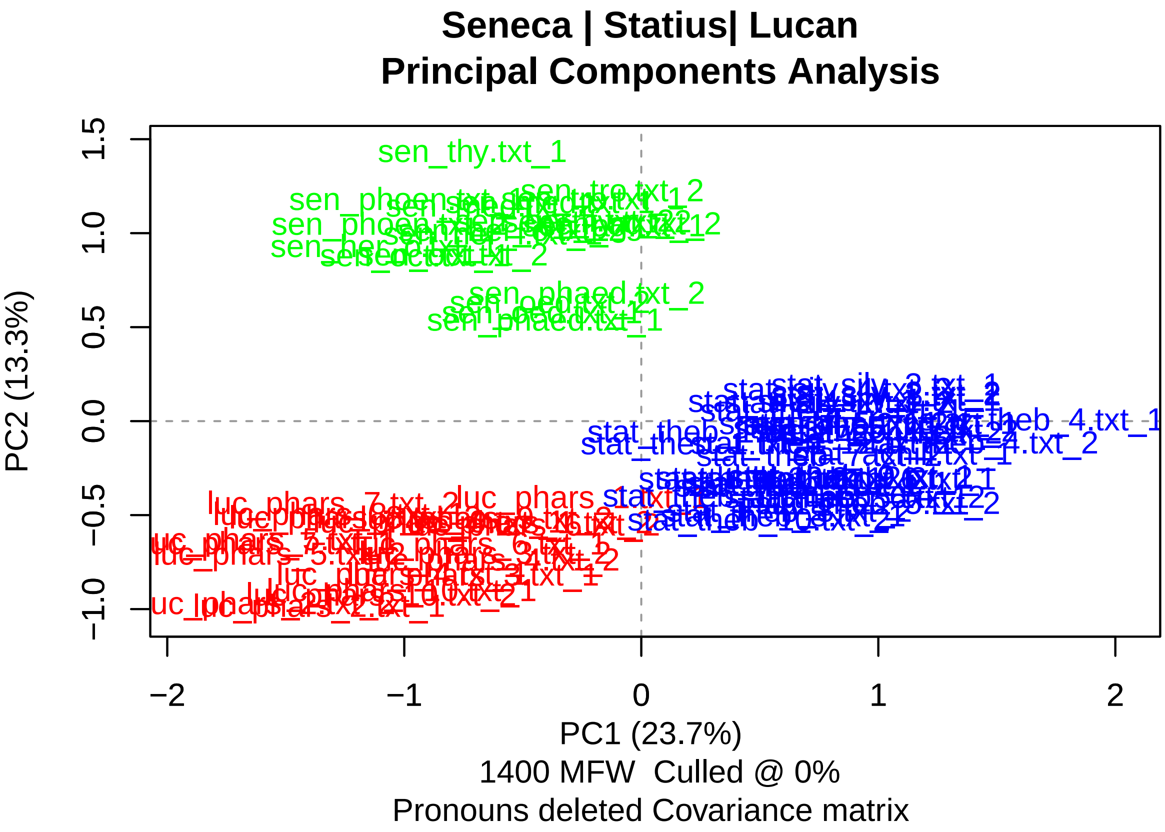
```
## 1300
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



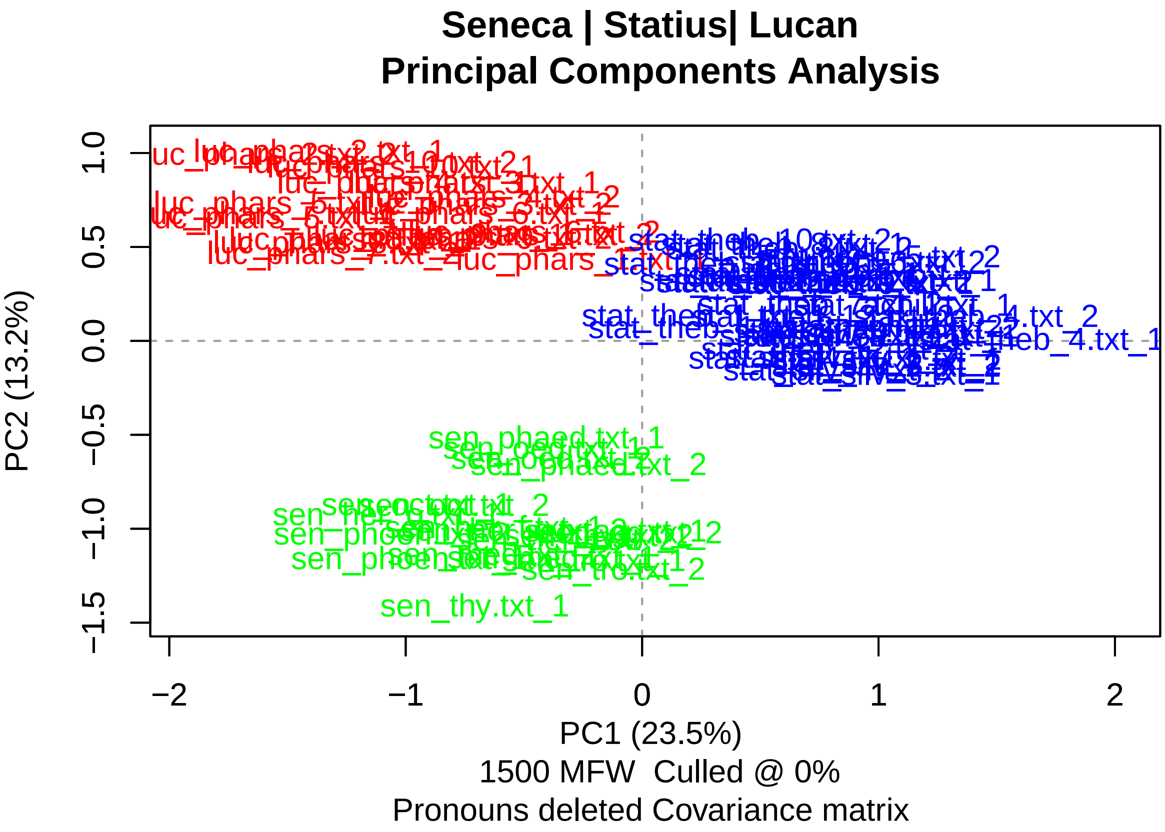
```
## 1400
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



```
## 1500
## Processing metadata...
##
##
## Assigning plot colors according to file names...
```



##



# Apply BCT For this method, we won't use the the sliced corpus (i.e., the corpus with the random samples of

the texts) because the plot get very populated, it looks like a moving fidget spinner, and it very hard to read.

```
corpus.no.pronouns <- delete.stop.words(tokenized.corpus, # if you want the sliced corpus change `token
                                        stop.words = stylo.pronouns(corpus.lang = "Latin.corr"))

corpus.w.grams <- txt.to.features(corpus.no.pronouns,
                                  ngram.size = 1,
                                  features = "w")

freq.features.word.grams <- make.frequency.list(corpus.w.grams,
                                                head = 3000)

freqs.word.grams <- make.table.of.frequencies(corpus.w.grams,
                                              features = freq.features.word.grams,
                                              relative = T)
```

```
## processing  38  text samples
```

```
## ...
## combining frequencies into a table...
```

```
bct.results.words = stylo(frequencies = freqs.word.grams,
                          distance.measure = "eder",
                          analysis.type = "BCT",
                          mfw.min = 100, mfw.max = 1500, increment = 100,
                          consensus.strength = 0.5,
                          write.png.file = T,
                          gui = T)
```

```
## using current directory...
```

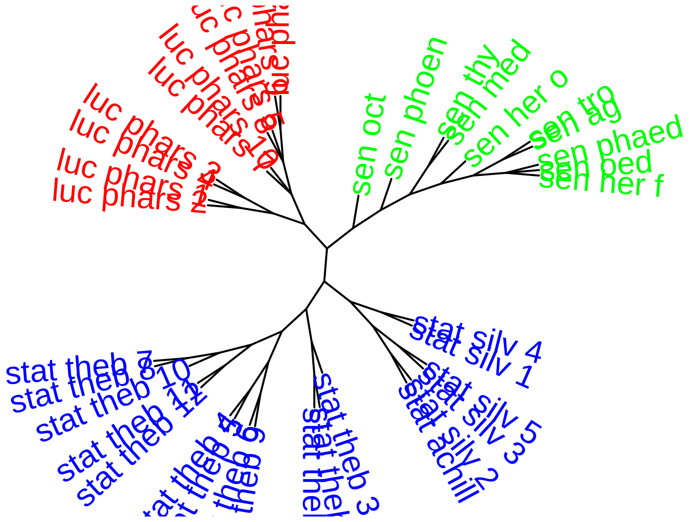
```
## Warning in delete.stop.words(table.with.all.freqs, pronouns): chosen stop words were not found in the
##   please check the language, lower/uppercase issues, etc.
```

```
##
##
## culling @ 0  available features (words) 3000
## Calculating z-scores...
##
## Calculating Eder's Delta distances...
## MFW used:
## 100
## Processing metadata...
##
##
## Assigning plot colors according to file names...
##
## 200
## Processing metadata...
##
##
## Assigning plot colors according to file names...
##
## 300
## Processing metadata...
##
##
```

```
## Assigning plot colors according to file names...
##
## 400
## Processing metadata...
##
##
## Assigning plot colors according to file names...
##
## 500
## Processing metadata...
##
##
## Assigning plot colors according to file names...
##
## 600
## Processing metadata...
##
##
## Assigning plot colors according to file names...
##
## 700
## Processing metadata...
##
##
## Assigning plot colors according to file names...
##
## 800
## Processing metadata...
##
##
## Assigning plot colors according to file names...
##
## 900
## Processing metadata...
##
##
## Assigning plot colors according to file names...
##
## 1000
## Processing metadata...
##
##
## Assigning plot colors according to file names...
##
## 1100
## Processing metadata...
##
##
## Assigning plot colors according to file names...
##
## 1200
## Processing metadata...
##
##
```

```
## Assigning plot colors according to file names...
##
## 1300
## Processing metadata...
##
##
## Assigning plot colors according to file names...
##
## 1400
## Processing metadata...
##
##
## Assigning plot colors according to file names...
##
## 1500
## Processing metadata...
##
##
## Assigning plot colors according to file names...
##
##
```

word_features
Bootstrap Consensus Tree



100−1500 MFW Culled @ 0%
Pronouns deleted Eder's Delta distance Consensus 0.5