# Results and visualisations of Evert et al. (2017)

Stefan Evert

8 Oct 2016

## Contents



# 1 Data sets and setup

This document collects all experiments and plots for the paper on Burrows Delta submitted to *Digital Scholarship in the Humanities*.

Load relative frequencies and z-scores for the German, English and French data set. For technical reasons, the data structures store the transposed document-term matrices $\mathbf{F}^T$ and $\mathbf{Z}^T$

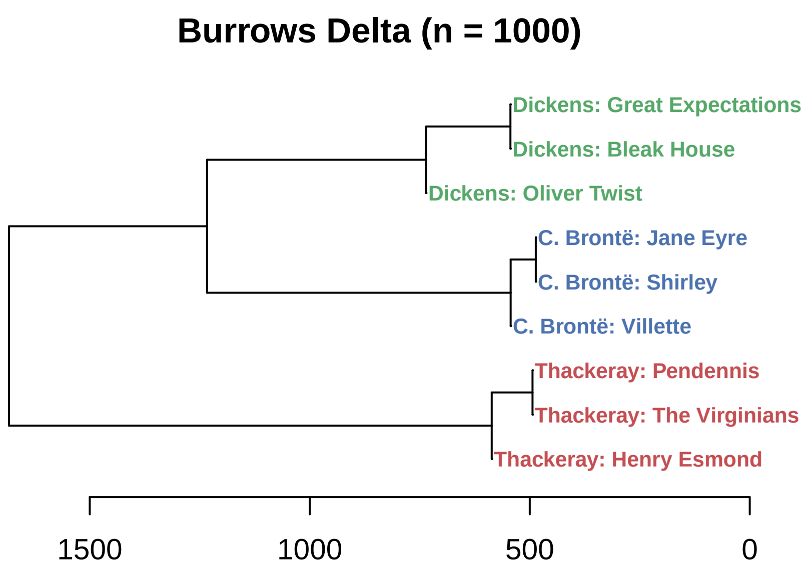
- $\mathbf{F}^T$ is available under the names `FreqDE$S`, `FreqEN$S` and `FreqFR$S`
- $\mathbf{Z}^T$ is available under the names `zDE`, `zEN` and `zFR`
- absolute frequencies $n_{D_j} \cdot f_i(D_j)$ can be found in `FreqDE$M`, `FreqEN$M`, `FreqFR$M`

Callback functions for feature transformations are defined with suitable defaults: `clamp2` clamps outliers with $|z| > 2$, `ternarize3` uses theoretical 33% quantiles, and `crosstern3` adds a cross-over after 150 mfw.

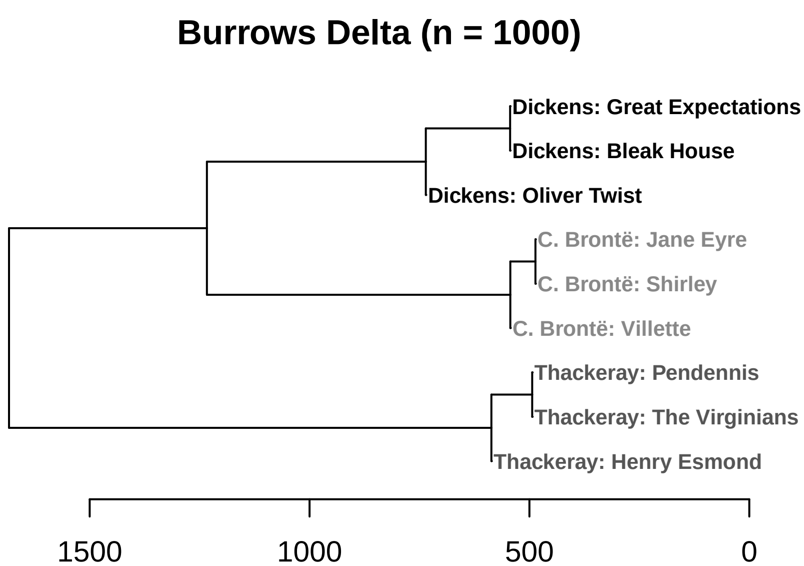
# 2 Illustrations

## 2.1 Example dendrogram (three authors from English corpus)

As an illustration, show hierarchical clustering of 9 English novels (by Charlotte Brontë, Charles Dickens and William Makepeace Thackeray). This is based on Burrows Delta with $n_w = 1000$.



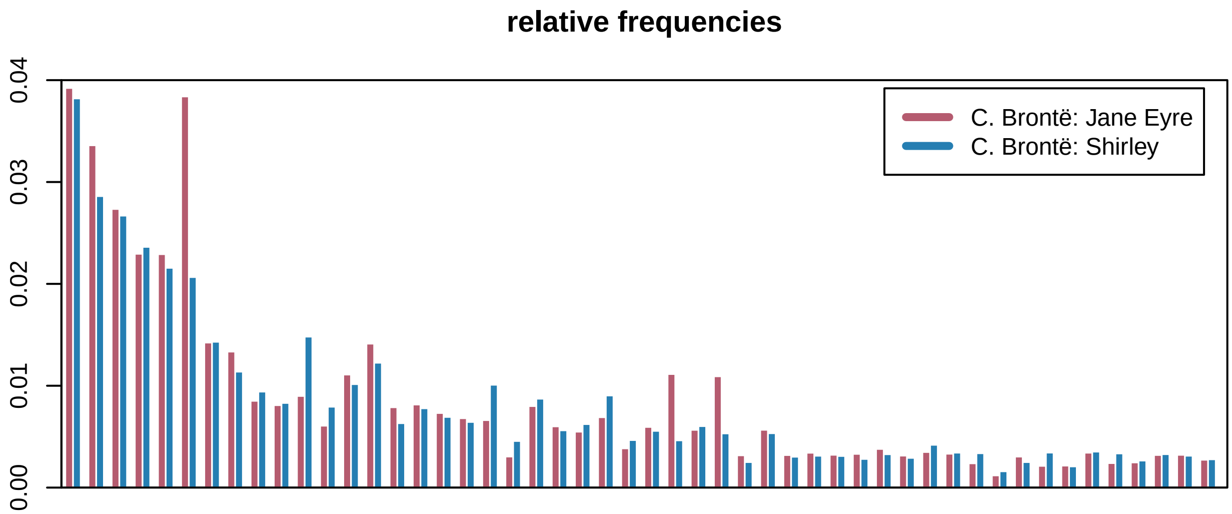

Black-and-white version for the paper:



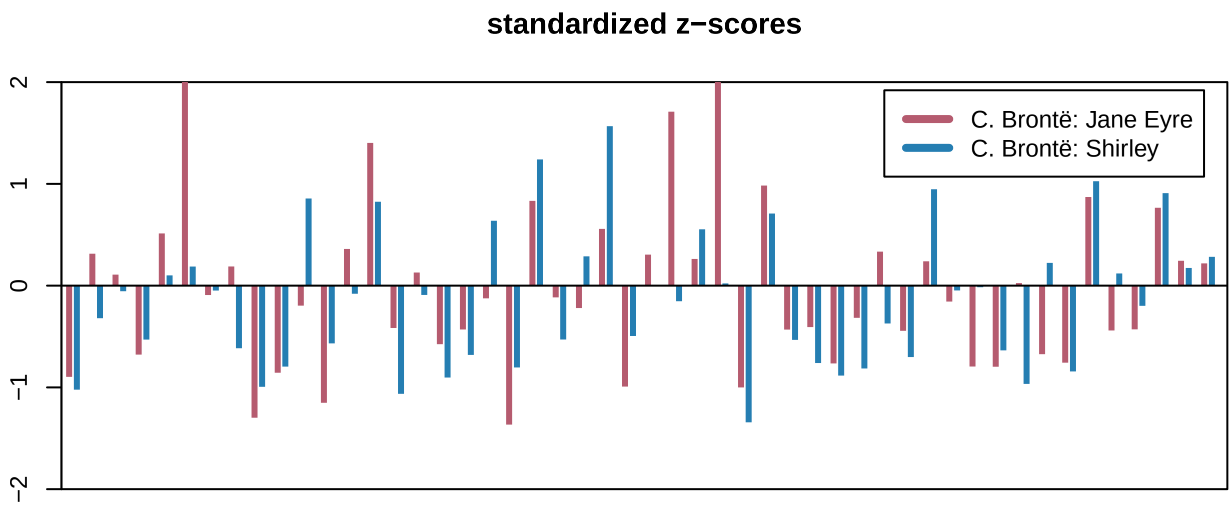

## 2.2 Spike plots as an illustration of Delta vectors

We use the same texts as in the clustering example above. Charlotte Brontë seems to write in a fairly consistent style, while there are enormous differences between the three novels by Charles Dickens.

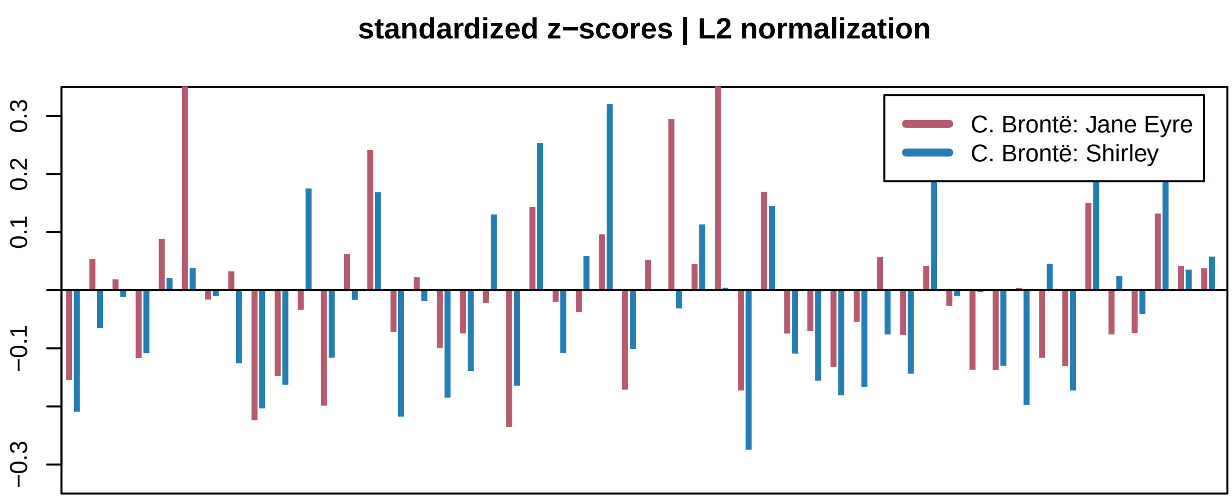
```
novels.spike <- c("jane", "shirley")
idx.spike <- match(novels.spike, FreqEN$rows$title)
labels.spike <- text.labels[novels.spike]
spike.plot(FreqEN$S[idx.spike, 1:50], lwd=3, yaxs="i", ylim=c(0, 0.04),
           legend=labels.spike, main="relative frequencies")
```



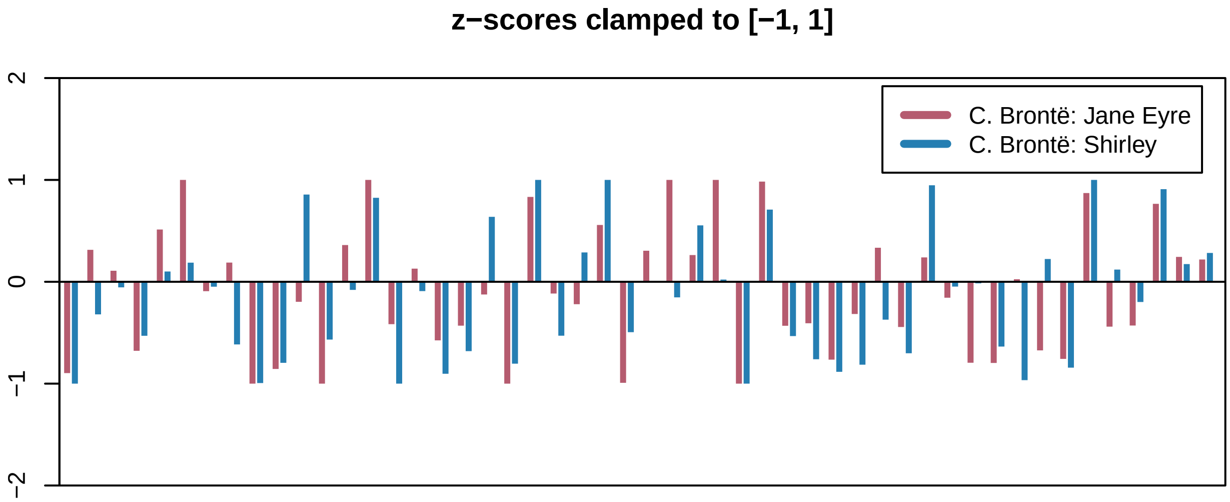

```
spike.plot(zEN[idx.spike, 1:50], lwd=3, yaxs="i", ylim=c(-2, 2),
           legend=labels.spike, main="standardized z-scores")
```



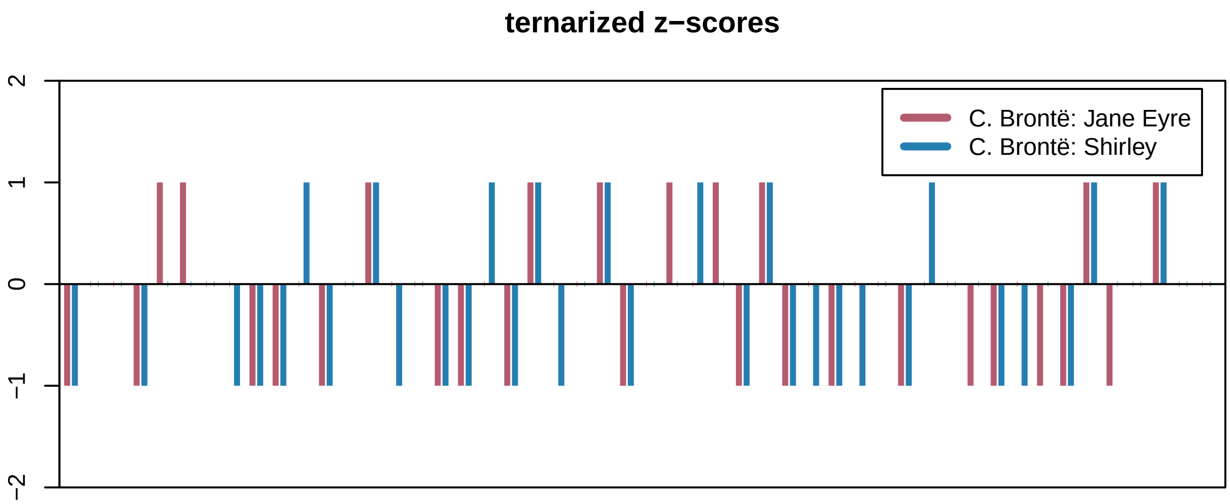

```
spike.plot(normalize.rows(zEN[idx.spike, 1:50]), lwd=3, yaxs="i", ylim=c(-.35, .35),
           legend=labels.spike, main="standardized z-scores | L2 normalization")
```



```r
spike.plot(clamp(zEN[idx.spike, 1:50], min=-1, max=1), lwd=3, yaxs="i", ylim=c(-2, 2),
           legend=labels.spike, main="z-scores clamped to [-1, 1]")
```

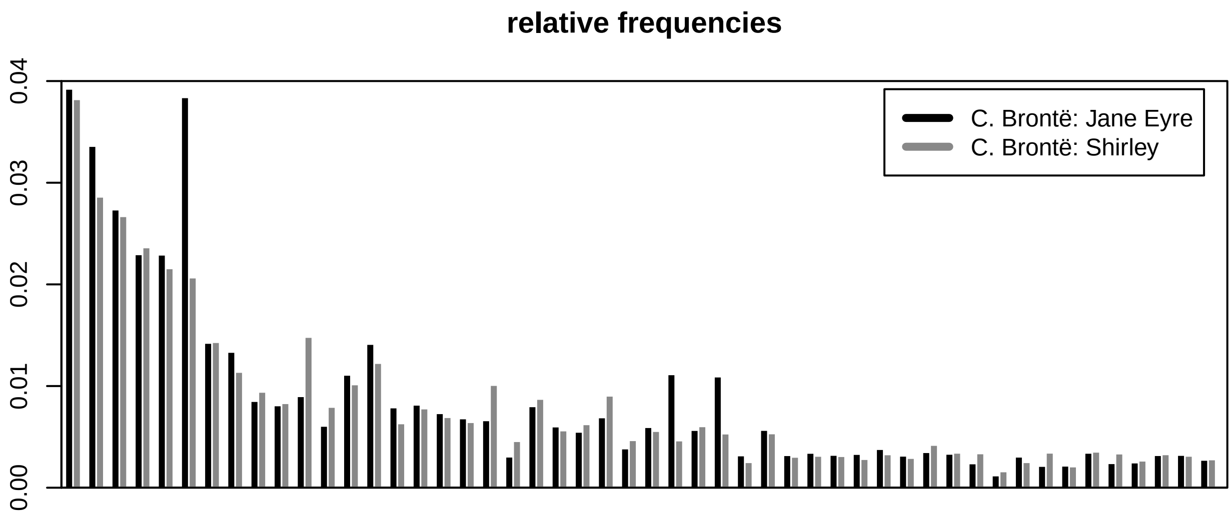


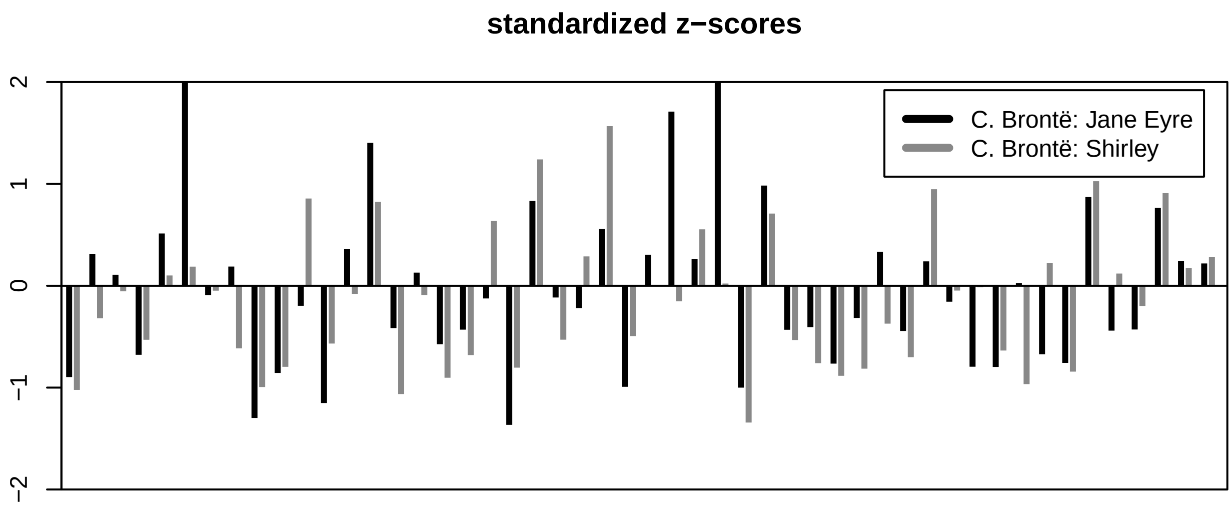

```r
spike.plot(ternarize3(zEN[idx.spike, 1:50]), lwd=3, yaxs="i", ylim=c(-2, 2),
           legend=labels.spike, main="ternarized z-scores")
```




For spike plots that appear in the published paper, we also need to provide suitable b/w versions.

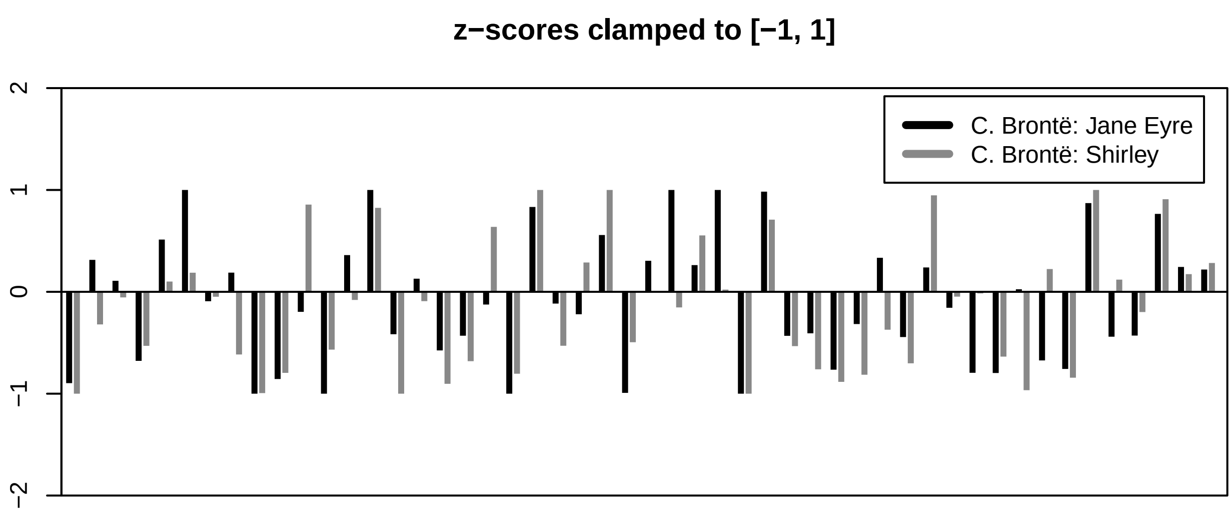
```r
par.save <- par(mfrow=c(2,1), mar=c(1,4,4,2)+.1)
spike.plot(FreqEN$S[idx.spike, 1:50], lwd=3, yaxs="i", ylim=c(0, 0.04),
           col=grayscale.pal, lty="solid", legend=labels.spike, main="relative frequencies")
spike.plot(zEN[idx.spike, 1:50], lwd=3, yaxs="i", ylim=c(-2, 2),
           col=grayscale.pal, lty="solid", legend=labels.spike, main="standardized z-scores")
```



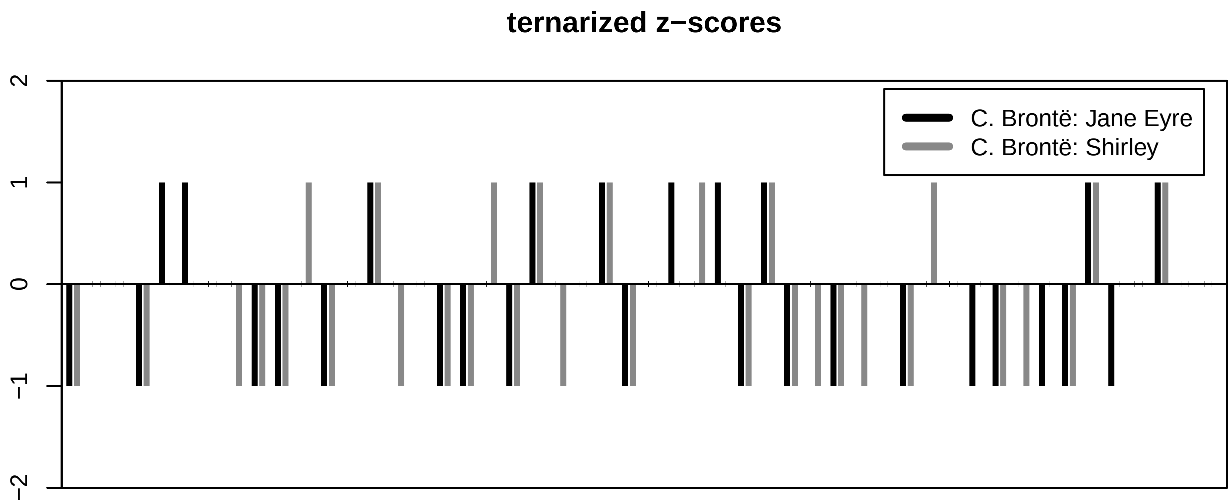




```
par(par.save)
```

```
spike.plot(clamp(zEN[idx.spike, 1:50], min=-1, max=1), lwd=3, yaxs="i", ylim=c(-2, 2),
           col=grayscale.pal, lty="solid", legend=labels.spike, main="z-scores clamped to [-1, 1]")
```



```r
spike.plot(ternarize3(zEN[idx.spike, 1:50]), lwd=3, yaxs="i", ylim=c(-2, 2),
           col=grayscale.pal, lty="solid", legend=labels.spike, main="ternarized z-scores")
```




## 3 Evaluation graphs

Standard evaluation is carried out using the `mfw.plot` function, with parameters

- `M`: row matrix of feature vectors
- `gold`: vector of gold standard categories (e.g. authors)
- `params`: data structure of parameter settings for the lines to be drawn (must specify `label`, `col`, `lty`, `method` and optionally `p`, `normalize`, `norm.p`)
- `normalize`, `norm.p`: default normalization applied to feature vectors (or `NA` for none; see `delta.dist` function)
- `clust.method`: clustering method (see `pam.cluster` function)
- `n.clusters`: desired number of clusters (specify range for automatic selection based on silhouette width)
- `transform`: transformation function applied to complete feature matrix
- `skip`: number of mfw to skip (i.e. drop first columns from `M`)

Evaluation steps and corresponding grid for the plots:

```r
## roughly logarithmic steps from 10 to 20000
n.vals <- c(10,15,20,25,32,40,50,65,80,100,125,150,200,250,320,400,500,650,800,1000,
            1250,1500,2000,2500,3200,4000,5000,6500,8000,10000)
draw.grid <- function () { # corresponding grid for plot region
  abline(h=seq(0, 100, 10), col="grey60")
  abline(v=c(10,20,50,100,200,500,1000,2000,5000,10000), col="grey60")
}
```

Data structures defining the lines to be shown in MFW evaluation plots:

```r
param.list <- list(
  list(method="cosine", col=2, lty="solid", label=expression("Cosine Delta")),
  list(method="minkowski", p=0.5, col=5, lty="solid", label=expression(L[1/2]*"-Delta")),
  list(method="manhattan", col=1, lty="solid", label=expression("Burrows "*(L[1])*" Delta")),
  list(method="euclidean", col=3, lty="solid", label=expression("Quadratic "*(L[2])*" Delta")),
  list(method="minkowski", p=4, col=4, lty="solid", label=expression(L[4]*"-Delta")))
param.basic <- c(param.list[c(1,3,4)], list(
  list(method="euclidean", normalize="euclidean", col=4, lty="dashed", label=expression("Quadratic Delta
```

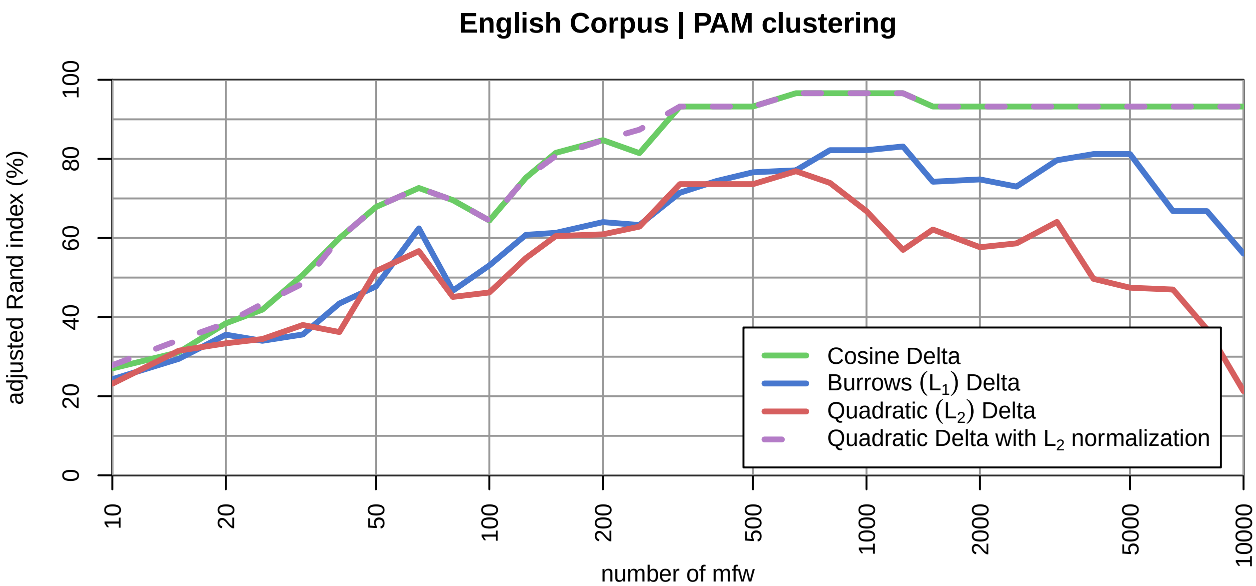
and corresponding b/w version for plots that are included in the published paper (colour indices are intended to index `grayscale.pal`)

```
param.list.bw <- list(
  list(method="cosine", col=2, lty="solid", label=expression("Cosine Delta")),
  list(method="minkowski", p=0.5, col=2, lty="32", label=expression(L[1/2]*"-Delta")),
  list(method="manhattan", col=1, lty="solid", label=expression("Burrows "*(L[1])*" Delta")),
  list(method="euclidean", col=1, lty="32", label=expression("Quadratic "*(L[2])*" Delta")),
  list(method="minkowski", p=4, col=4, lty="solid", label=expression(L[4]*"-Delta")))
param.basic.bw <- c(param.list.bw[c(1,3,4)], list(
  list(method="euclidean", normalize="euclidean", col=1, lty="12", label=expression("Quadratic Delta wi
```

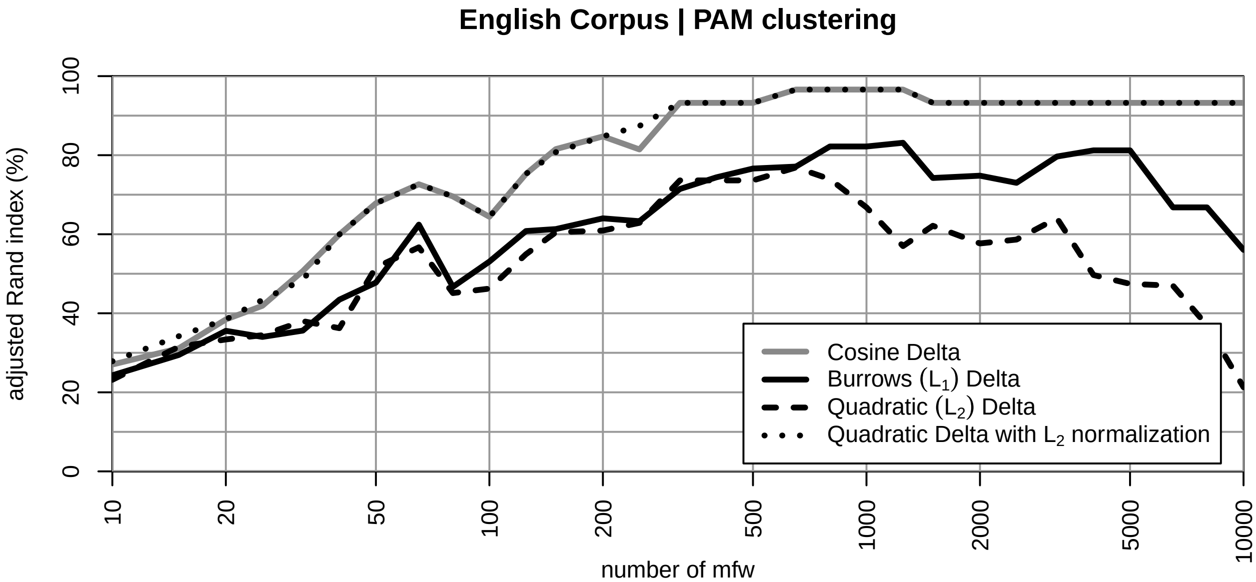
## 3.1 The effect of normalization

Evaluation of $\Delta_B$, $\Delta_Q$ and $\Delta_\angle$ for all three languages, including a $L_2$-normalized version of $\Delta_Q$.

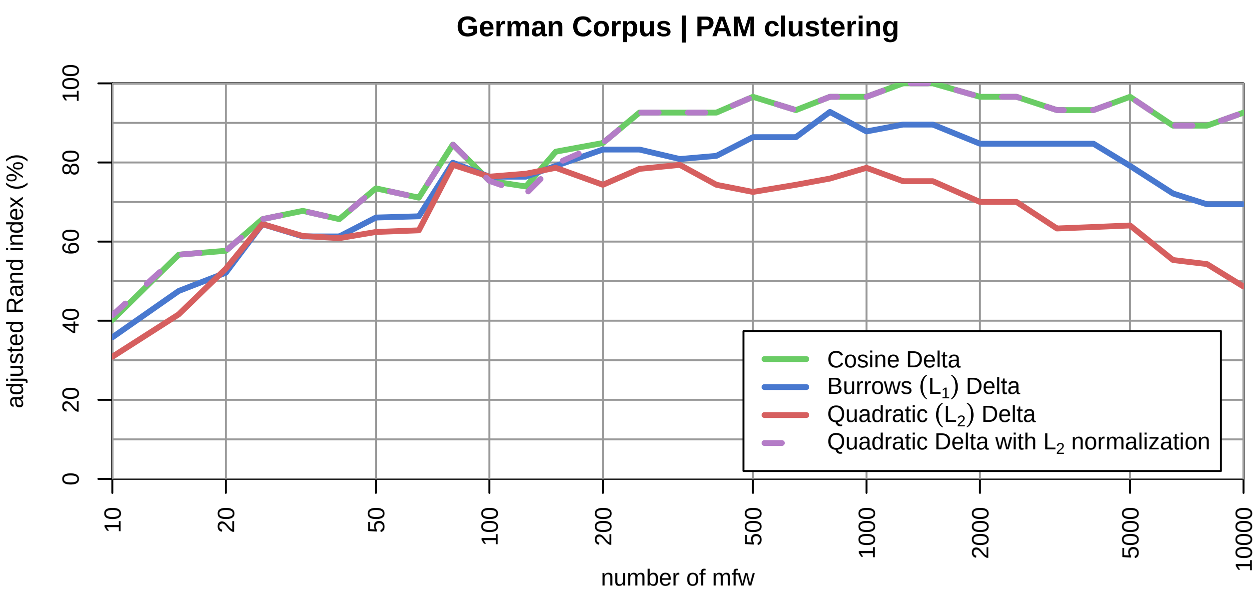
```
mfw.plot(zEN, goldEN, param=param.basic, main="English Corpus | PAM clustering")
```



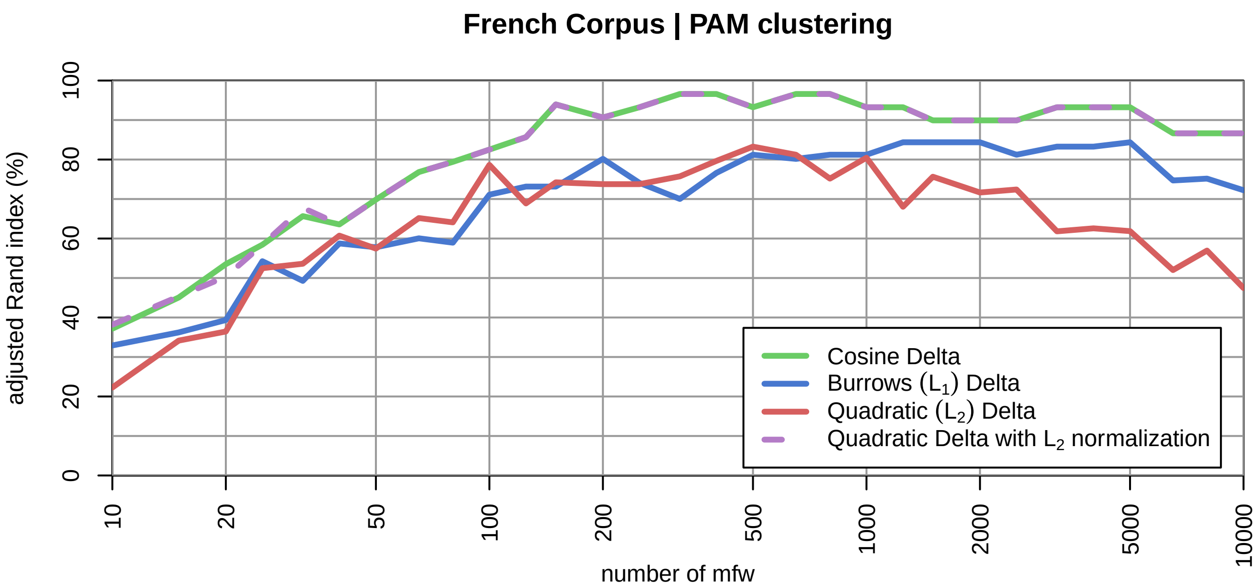

```
mfw.plot(zEN, goldEN, param=param.basic.bw, palette=grayscale.pal, main="English Corpus | PAM clusterin
```




```
mfw.plot(zDE, goldDE, param=param.basic, main="German Corpus | PAM clustering")
```

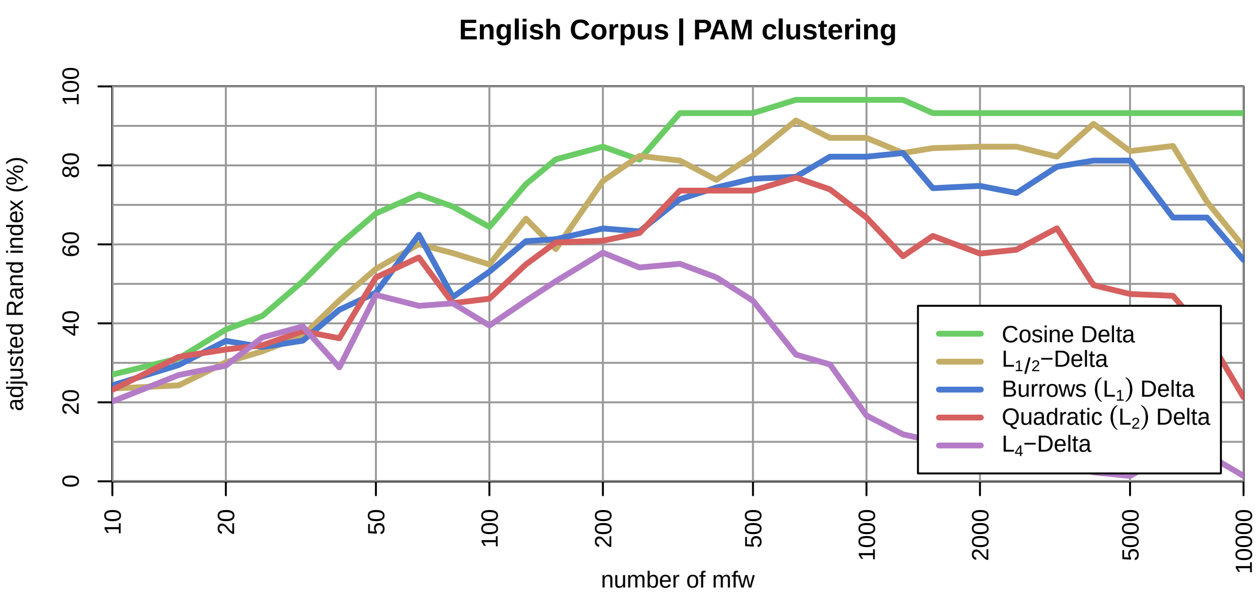



```
mfw.plot(zFR, goldFR, param=param.basic, main="French Corpus | PAM clustering")
```

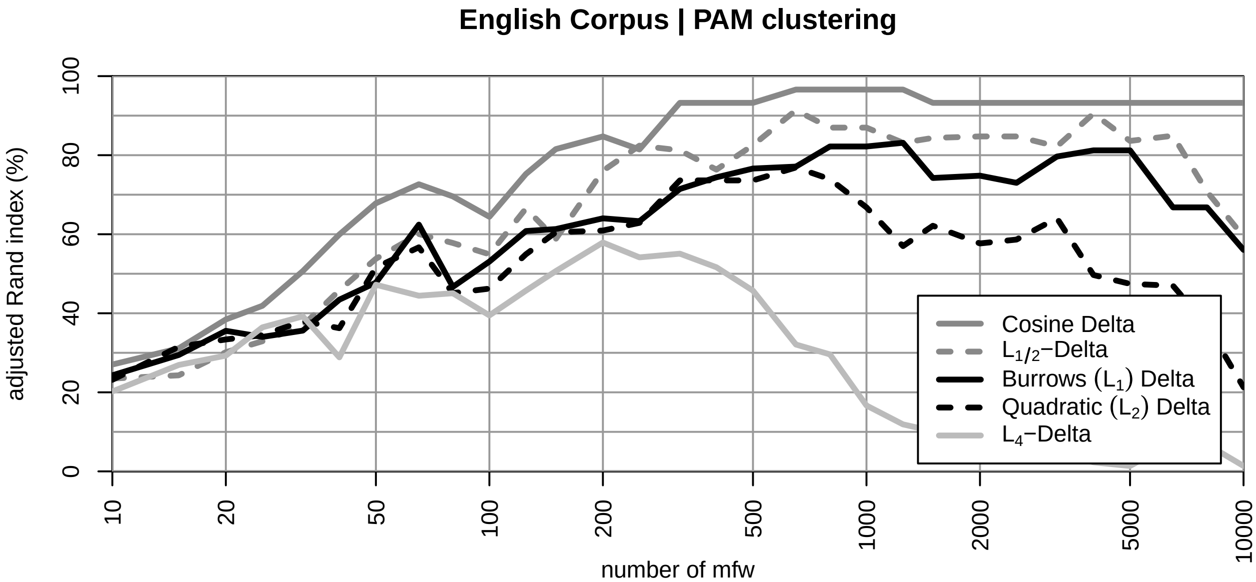



Evaluation of additional Minkowski $p$-distances with and without normalization, for all three languages. Parameters for the evaluation lines are in the default `param.list` structure.

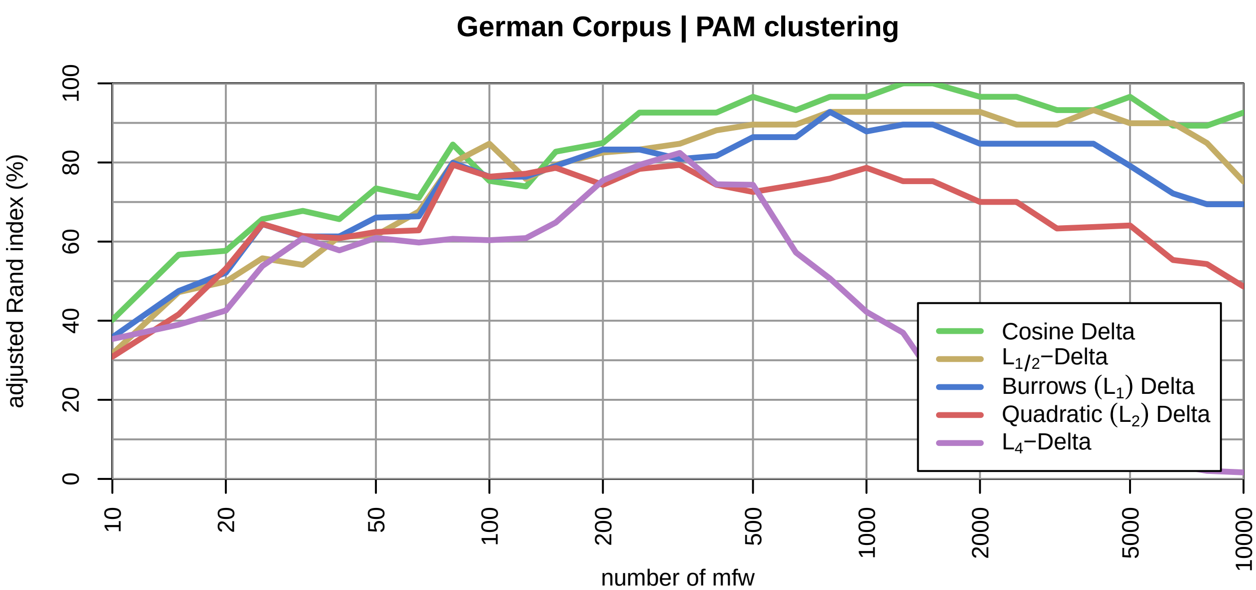
```
mfw.plot(zEN, goldEN, main="English Corpus | PAM clustering")
```



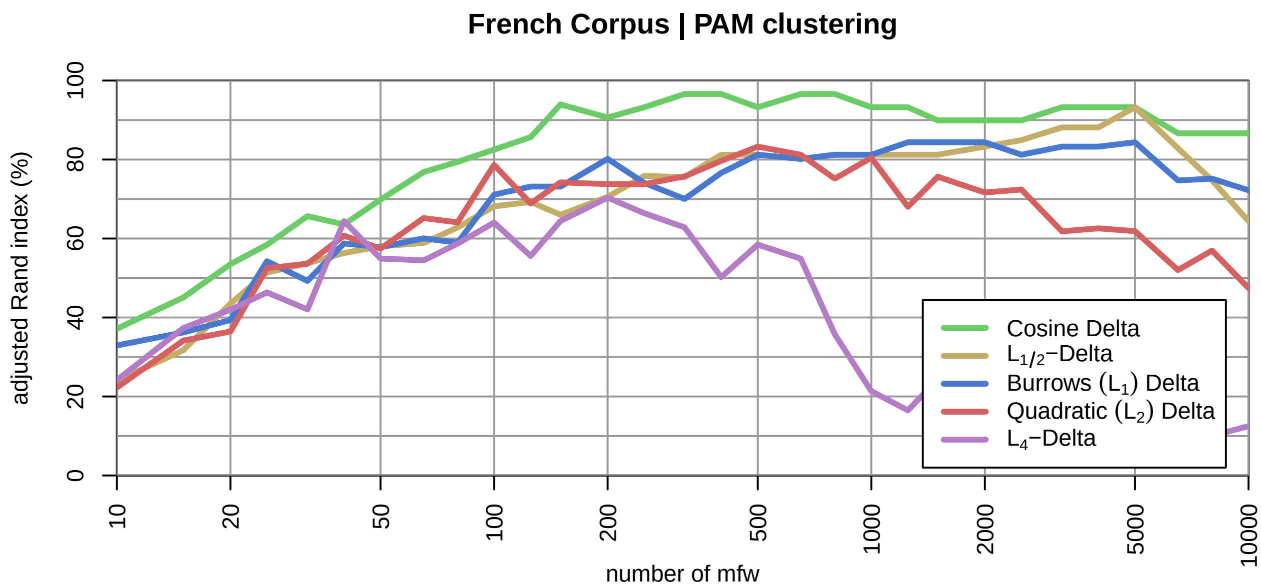

```
mfw.plot(zEN, goldEN, param=param.list.bw, palette=grayscale.pal, main="English Corpus | PAM clustering
```



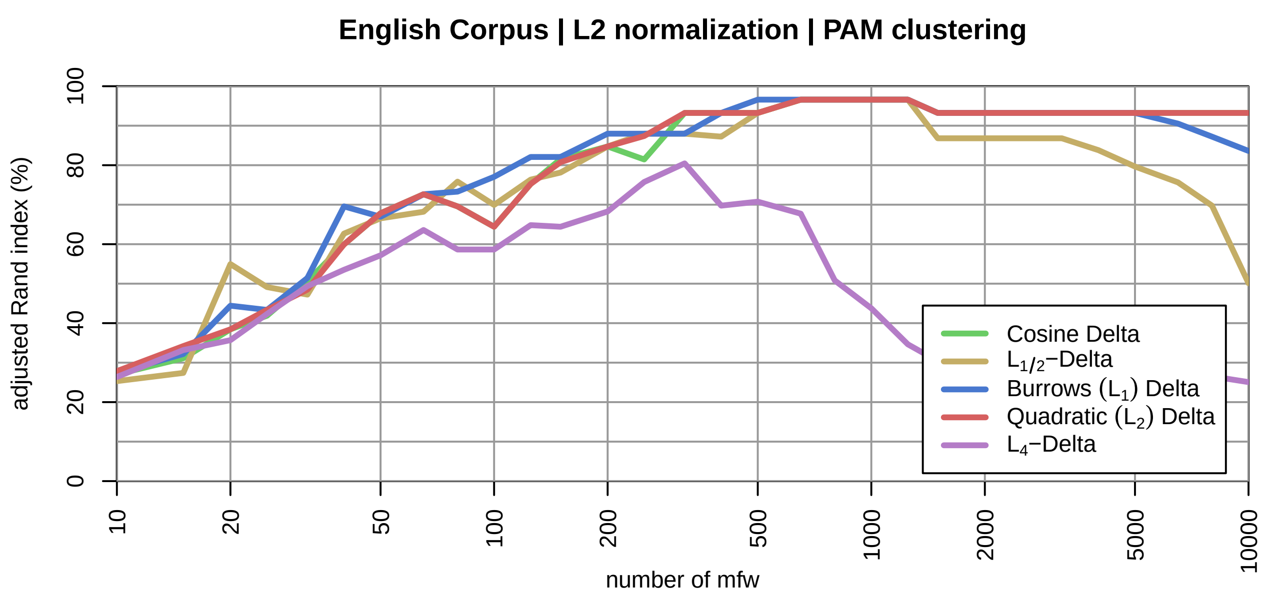

```r
mfw.plot(zDE, goldDE, main="German Corpus | PAM clustering")
```



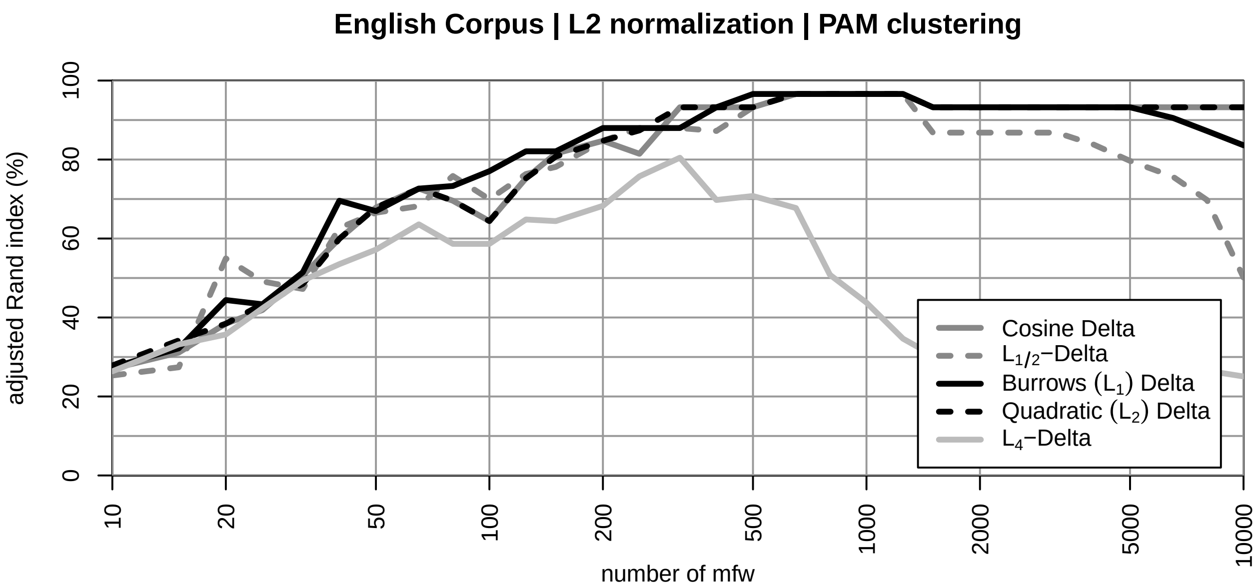

```r
mfw.plot(zFR, goldFR, main="French Corpus | PAM clustering")
```



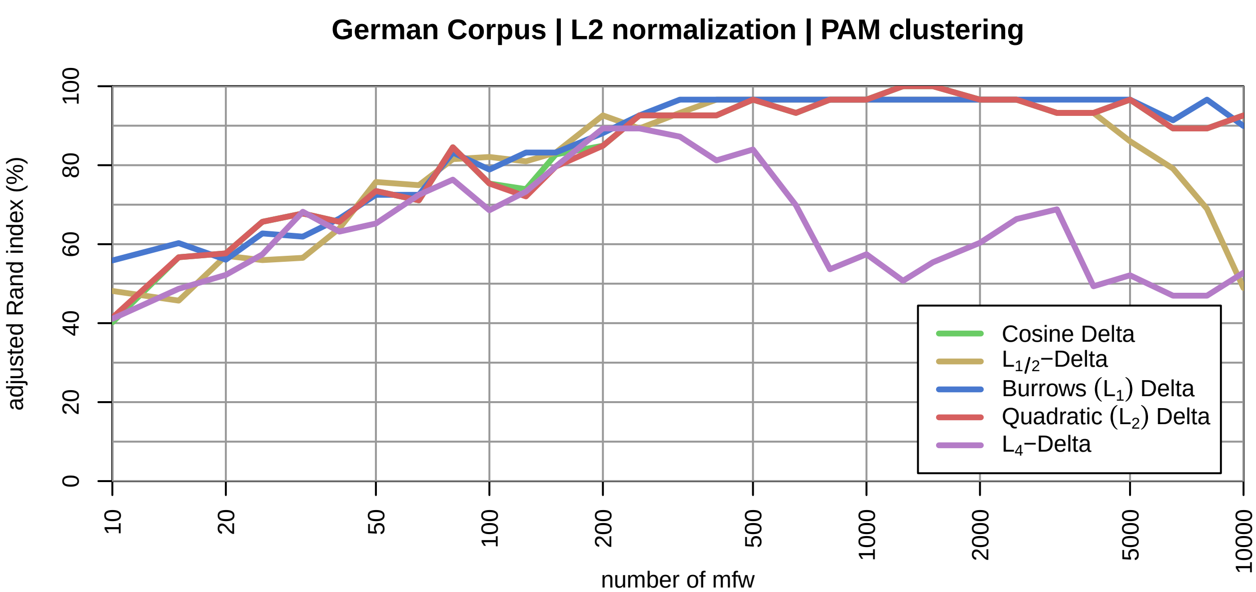

```
mfw.plot(zEN, goldEN, normalize="euclidean", main="English Corpus | L2 normalization | PAM clustering")
```



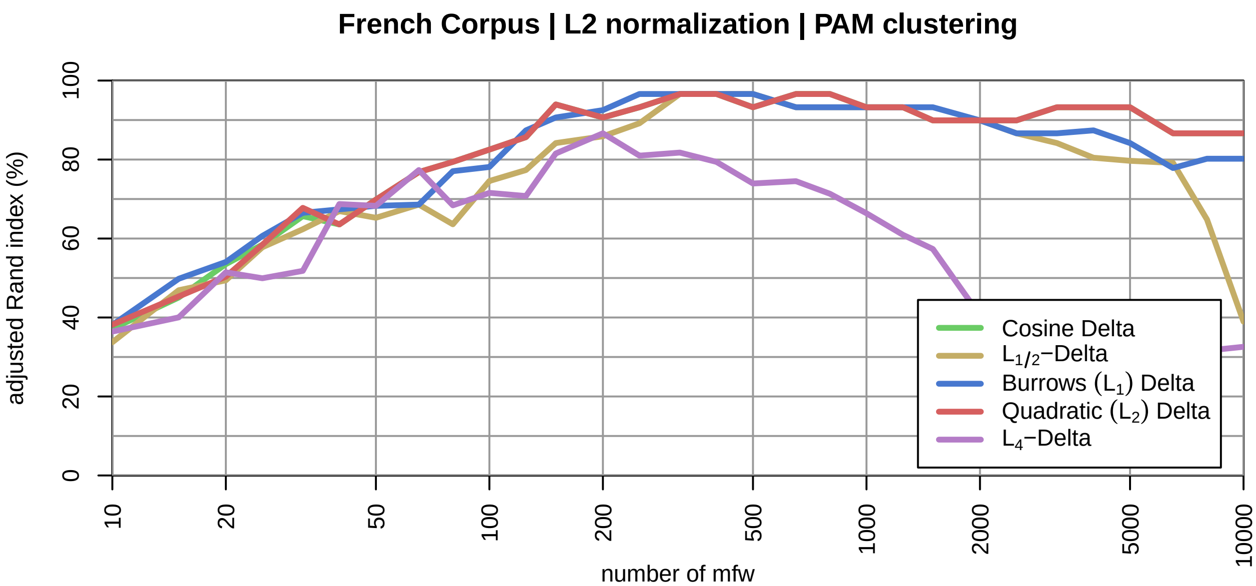

```
mfw.plot(zEN, goldEN, param=param.list.bw, palette=grayscale.pal, normalize="euclidean", main="English
```



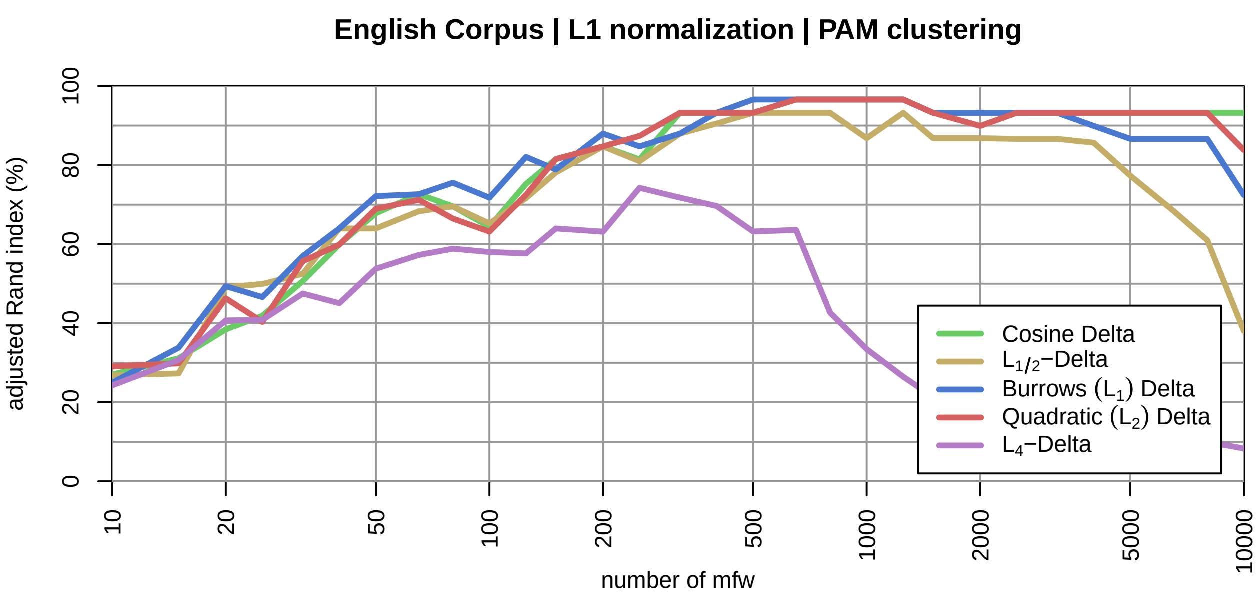

```
mfw.plot(zDE, goldDE, normalize="euclidean", main="German Corpus | L2 normalization | PAM clustering")
```



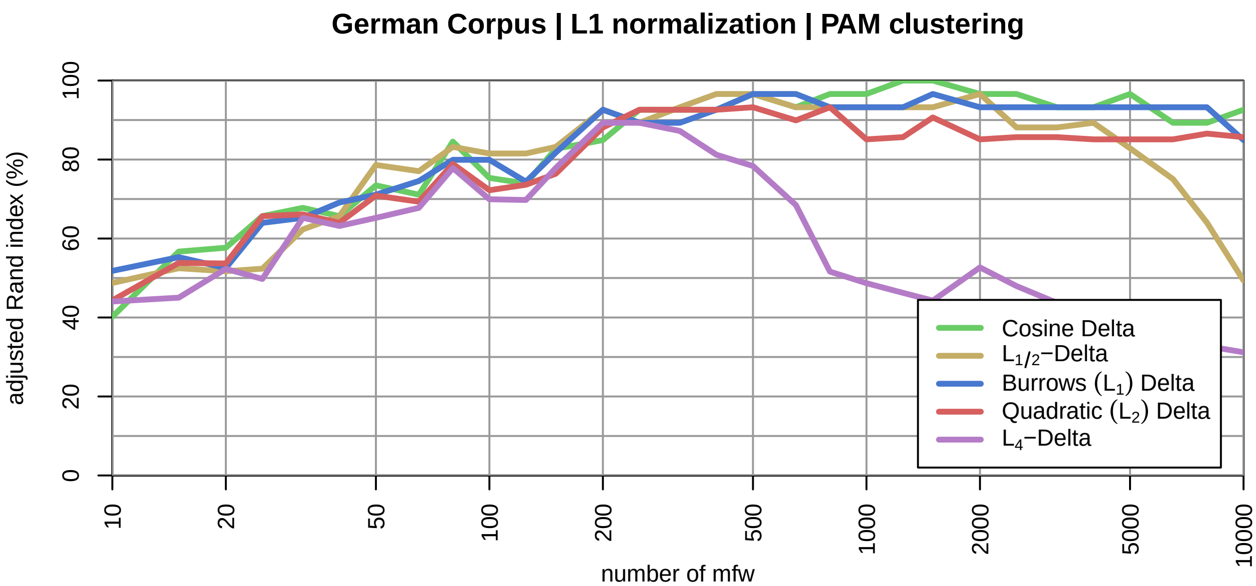

```
mfw.plot(zFR, goldFR, normalize="euclidean", main="French Corpus | L2 normalization | PAM clustering")
```



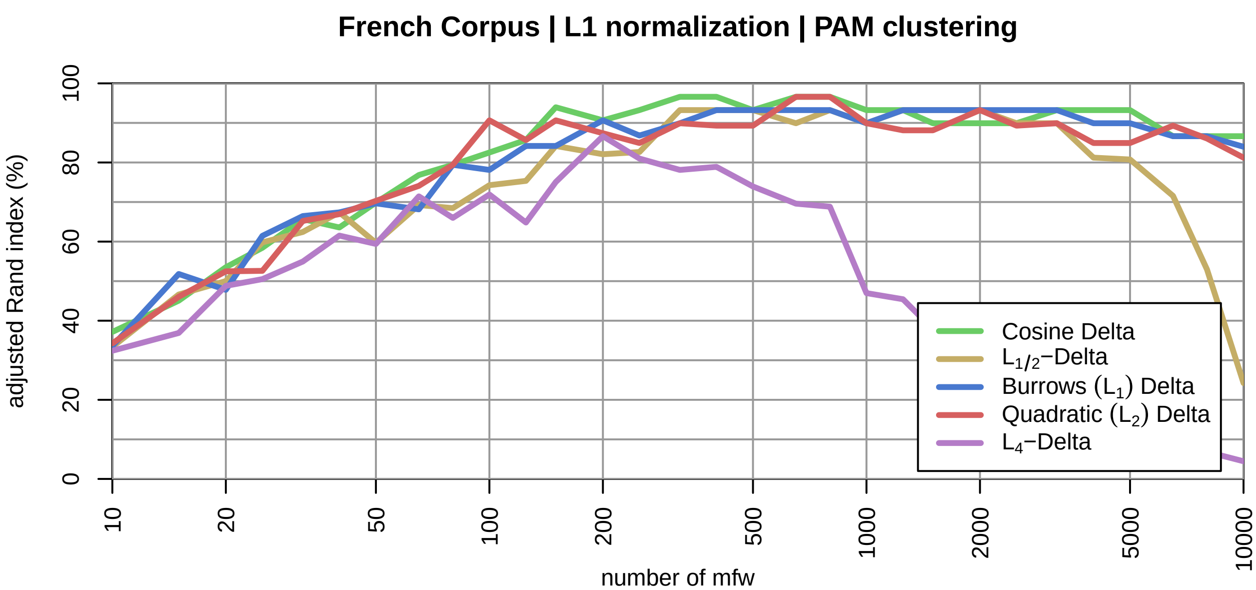

```
mfw.plot(zEN, goldEN, normalize="manhattan", main="English Corpus | L1 normalization | PAM clustering")
```




```
mfw.plot(zDE, goldDE, normalize="manhattan", main="German Corpus | L1 normalization | PAM clustering")
```

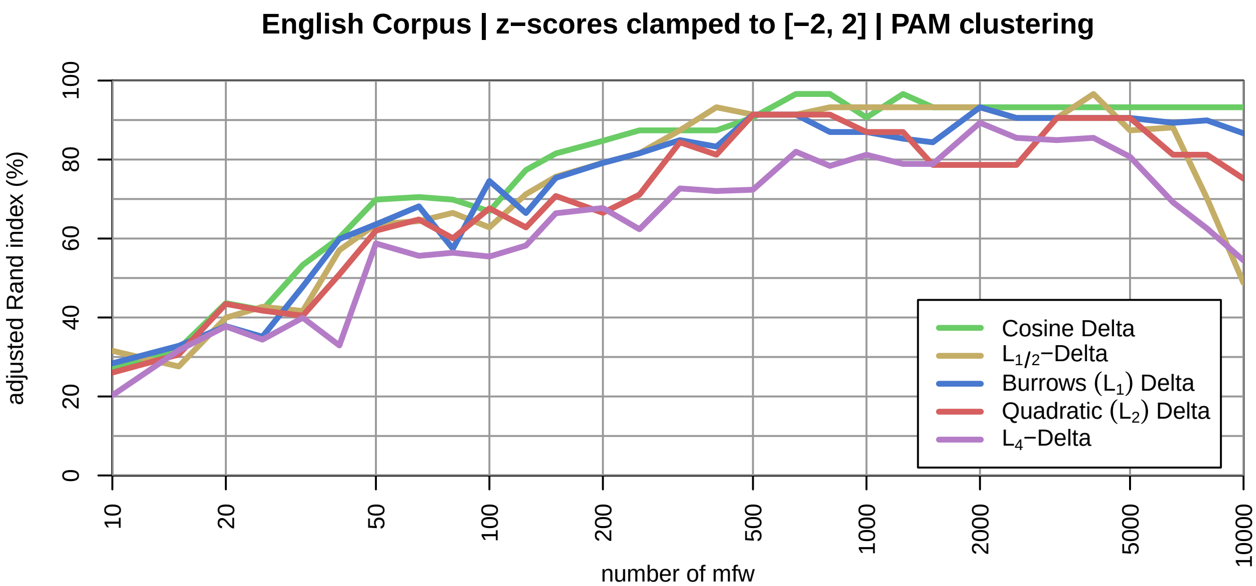



```
mfw.plot(zFR, goldFR, normalize="manhattan", main="French Corpus | L1 normalization | PAM clustering")
```

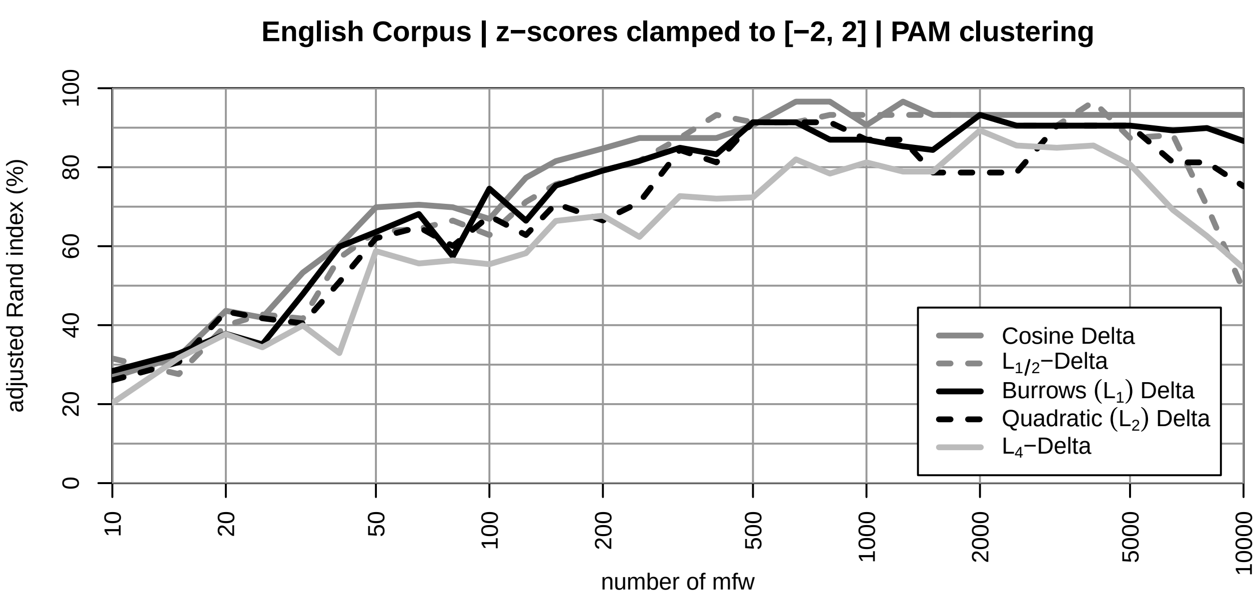



### 3.2 Truncating outliers and ternarization

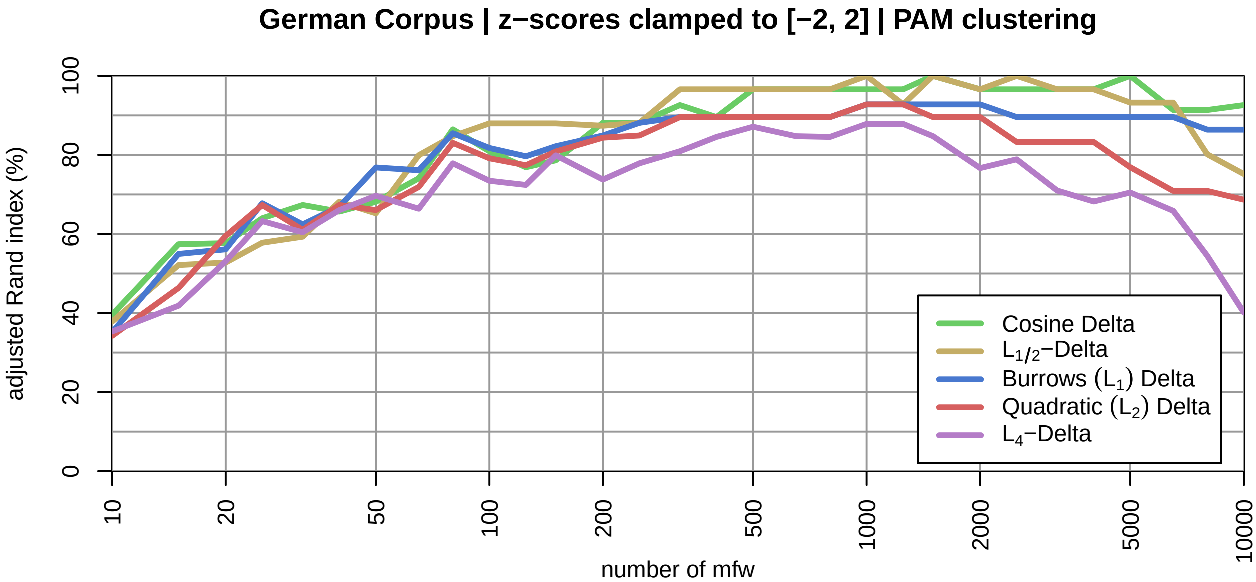
```
mfw.plot(zEN, goldEN, transform=clamp2, main="English Corpus | z-scores clamped to [-2, 2] | PAM cluste
```



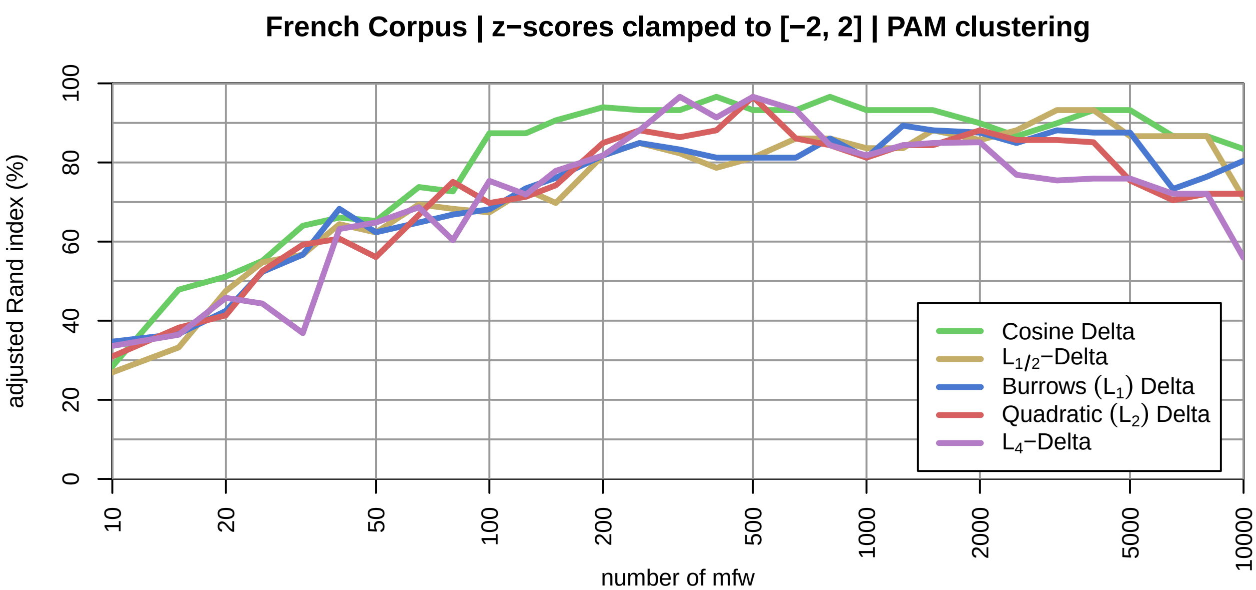

```
mfw.plot(zEN, goldEN, param=param.list.bw, palette=grayscale.pal, transform=clamp2, main="English Corpu
```



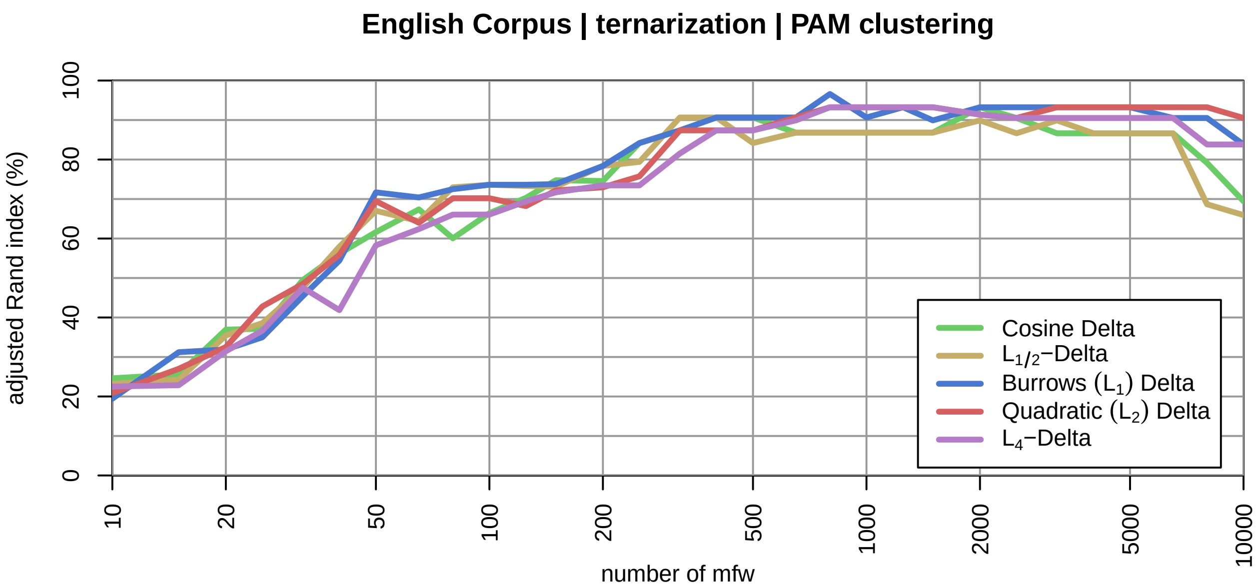

```
mfw.plot(zDE, goldDE, transform=clamp2, main="German Corpus | z-scores clamped to [-2, 2] | PAM cluster
```



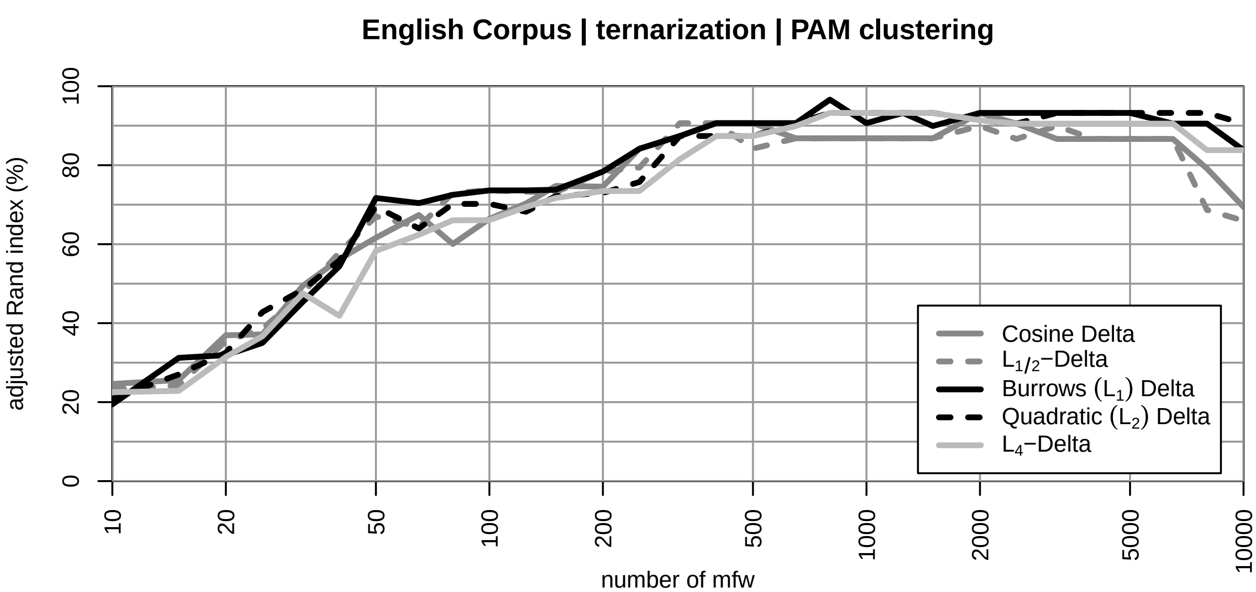

```
mfw.plot(zFR, goldFR, transform=clamp2, main="French Corpus | z-scores clamped to [-2, 2] | PAM cluster
```



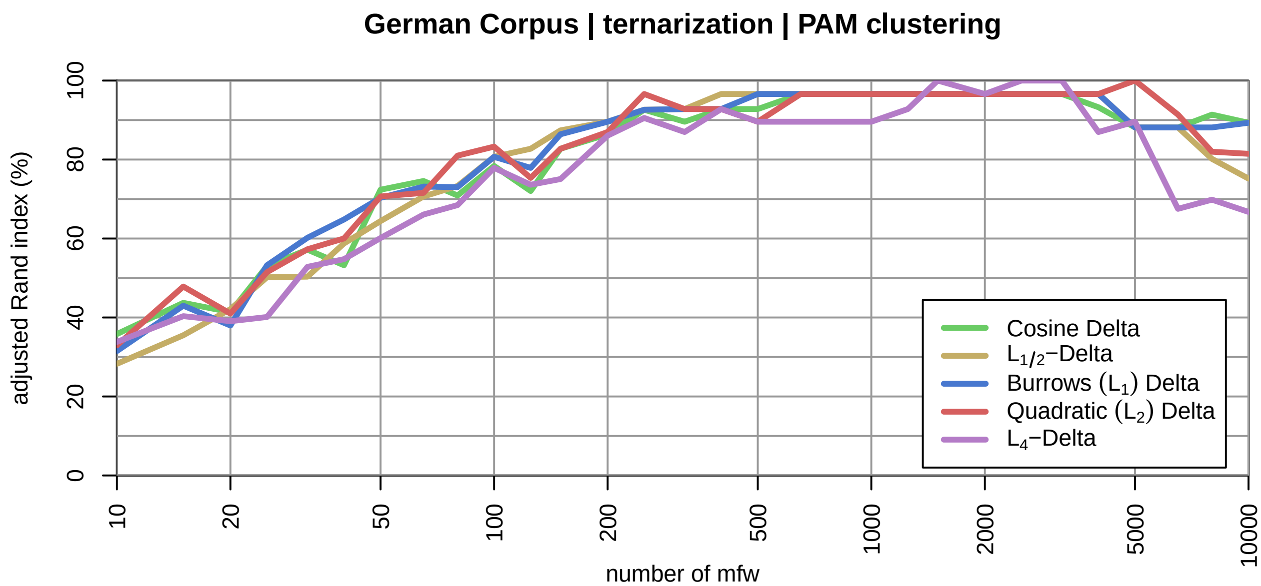

```
mfw.plot(zEN, goldEN, transform=ternarize3, main="English Corpus | ternarization | PAM clustering")
```



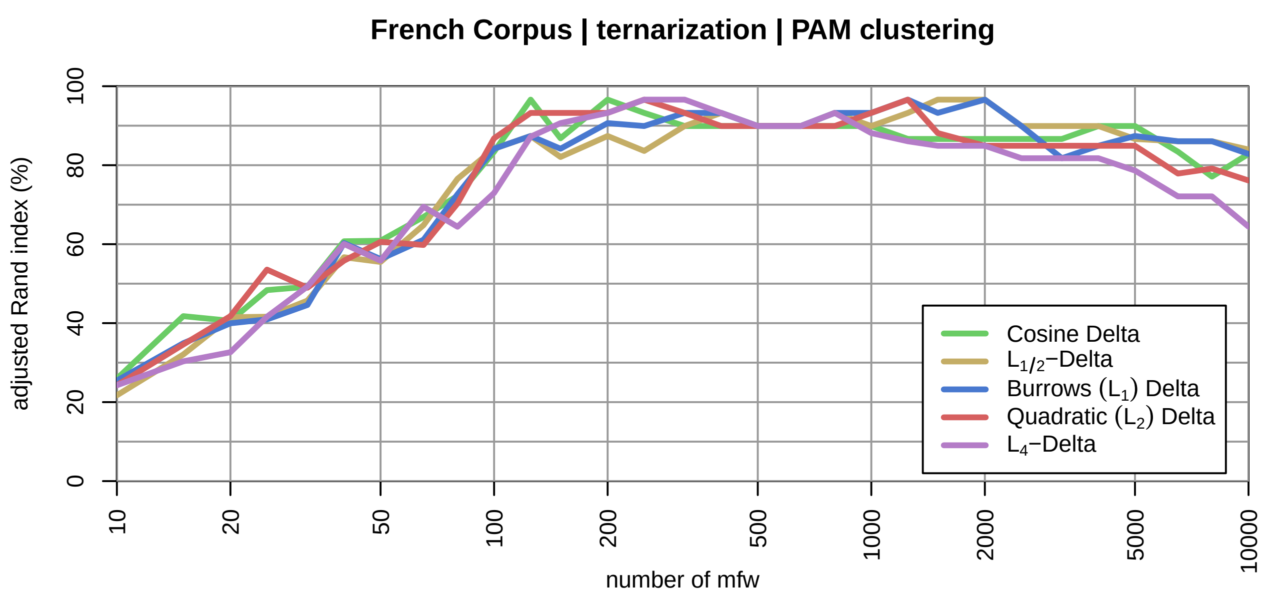

```
mfw.plot(zEN, goldEN, param=param.list.bw, palette=grayscale.pal, transform=ternarize3, main="English Co
```




```
mfw.plot(zDE, goldDE, transform=ternarize3, main="German Corpus | ternarization | PAM clustering")
```




```
mfw.plot(zFR, goldFR, transform=ternarize3, main="French Corpus | ternarization | PAM clustering")
```

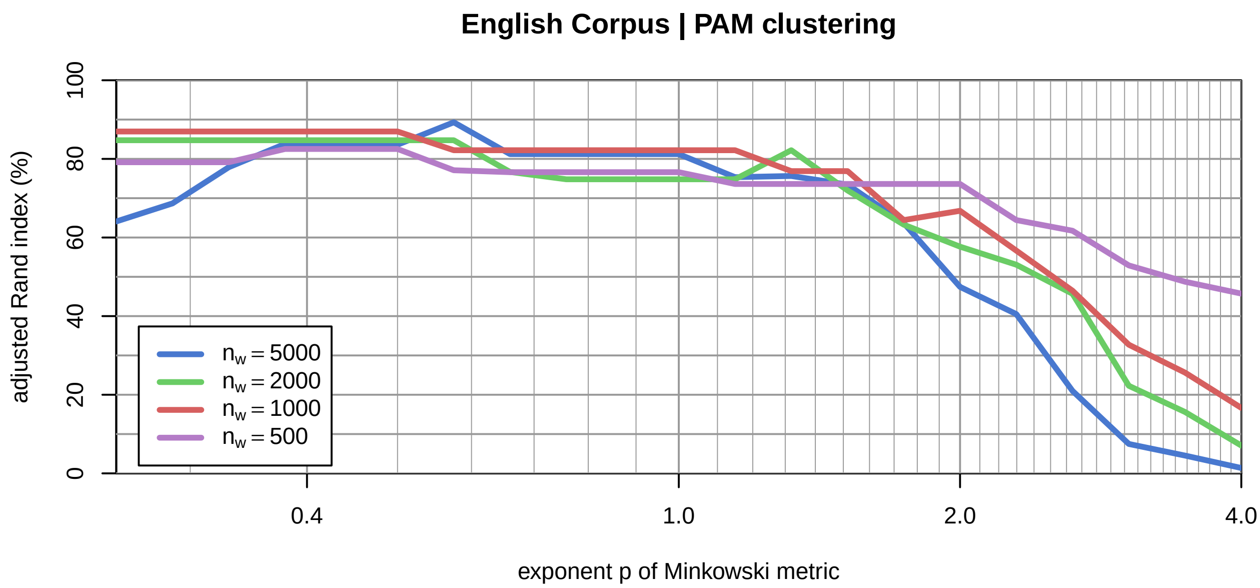



### 3.3 The effect of Minkowski $p$

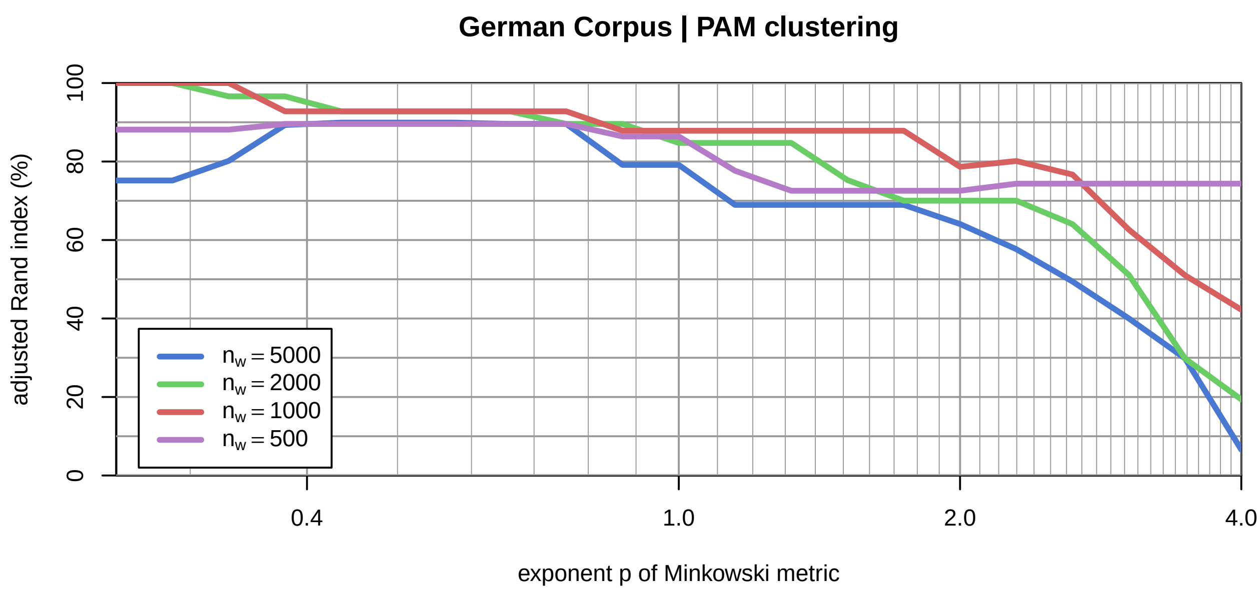
Plot clustering accuracy for different Minkowski $p$ metrics with fixed $n_w$. Since we want to combine evaluation settings in different ways, plots are created manually without a helper function.

```
p.vals <- 2 ^ seq(-2, 2, .2)
draw.grid.p <- function () {
  abline(h=seq(0, 100, 10), col="grey60")
  abline(v=seq(0.2, 4, .1), col="grey60", lwd=.5)
  abline(v=c(0.4, 1, 2), col="grey60")
}
```

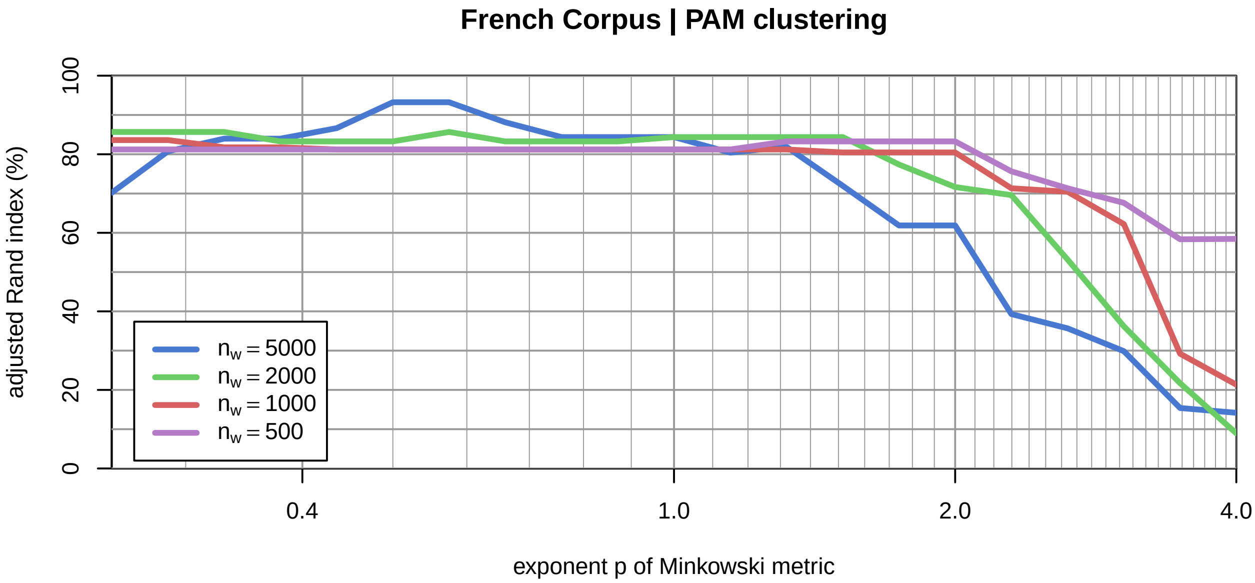
First determine how big the differences between the $p$-metrics are depending on the number $n_w$ of mfw used as features.



English Corpus | PAM clustering
adjusted Rand index (%)
0
20
40
60
80
100
0.4
1.0
2.0
4.0
exponent p of Minkowski metric
$n_w = 5000$
$n_w = 2000$
$n_w = 1000$
$n_w = 500$

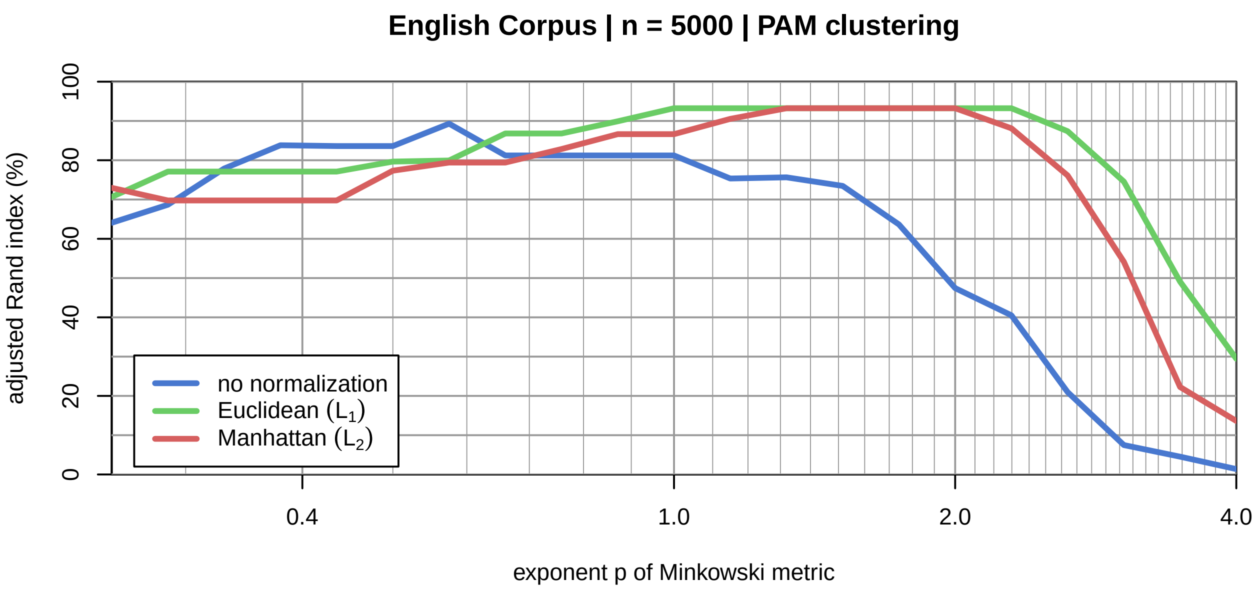



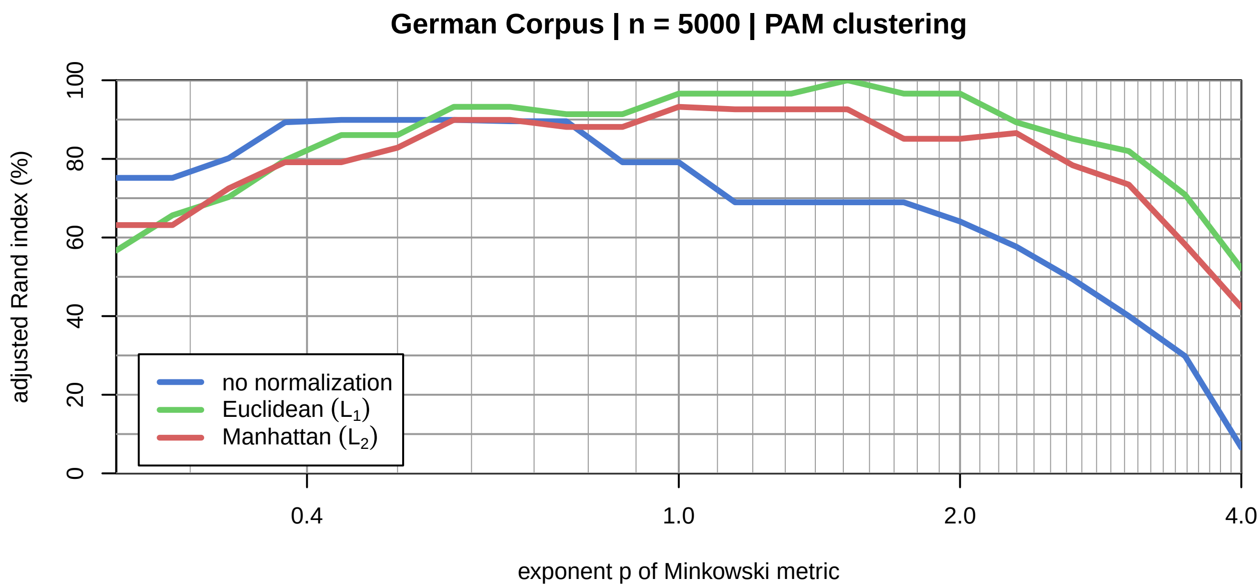
German Corpus | PAM clustering
adjusted Rand index (%)
0
20
40
60
80
100
0.4
1.0
2.0
4.0
exponent p of Minkowski metric
$n_w = 5000$
$n_w = 2000$
$n_w = 1000$
$n_w = 500$

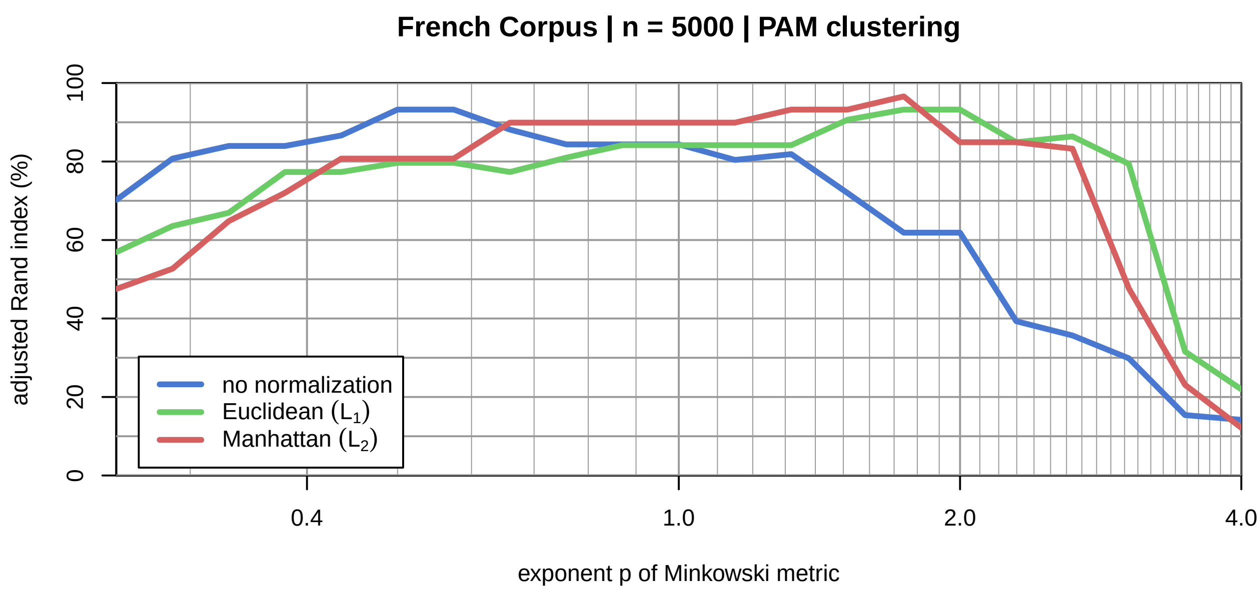



For the following experiments we set $n_w = 5000$ where robustness becomes an issue and there are substantial differences between the $p$-metrics. In particular, performance degrades for large *and* for very small $p$. However, with the right $p$, it can still outperform smaller $n_w$ except for the German corpus.

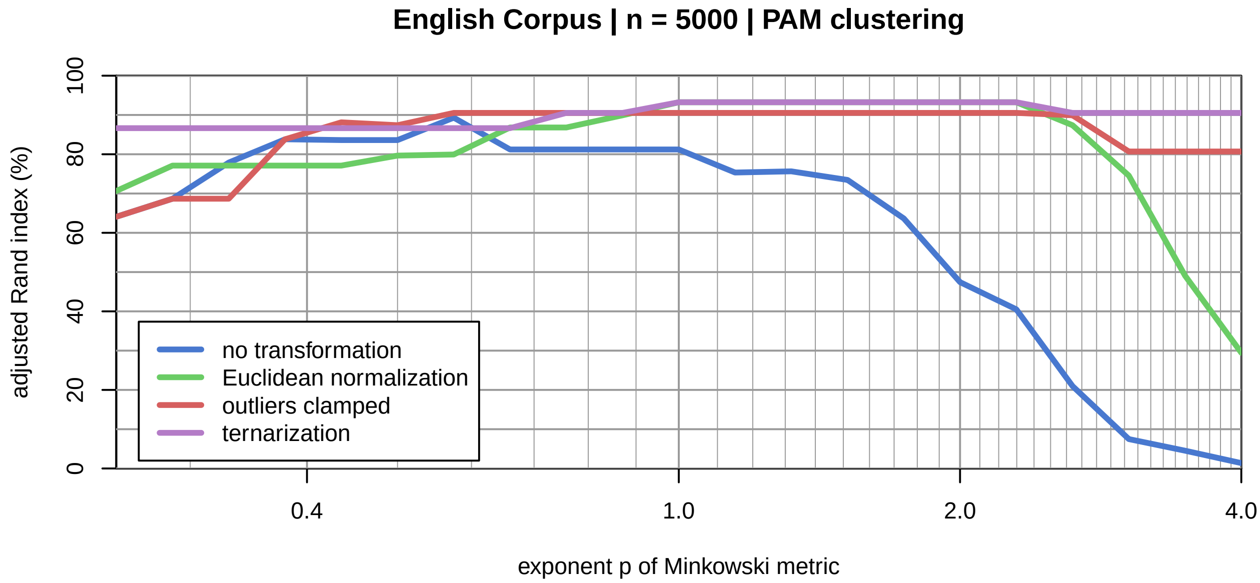
In the next step, we compare the effect of different vector normalizations.

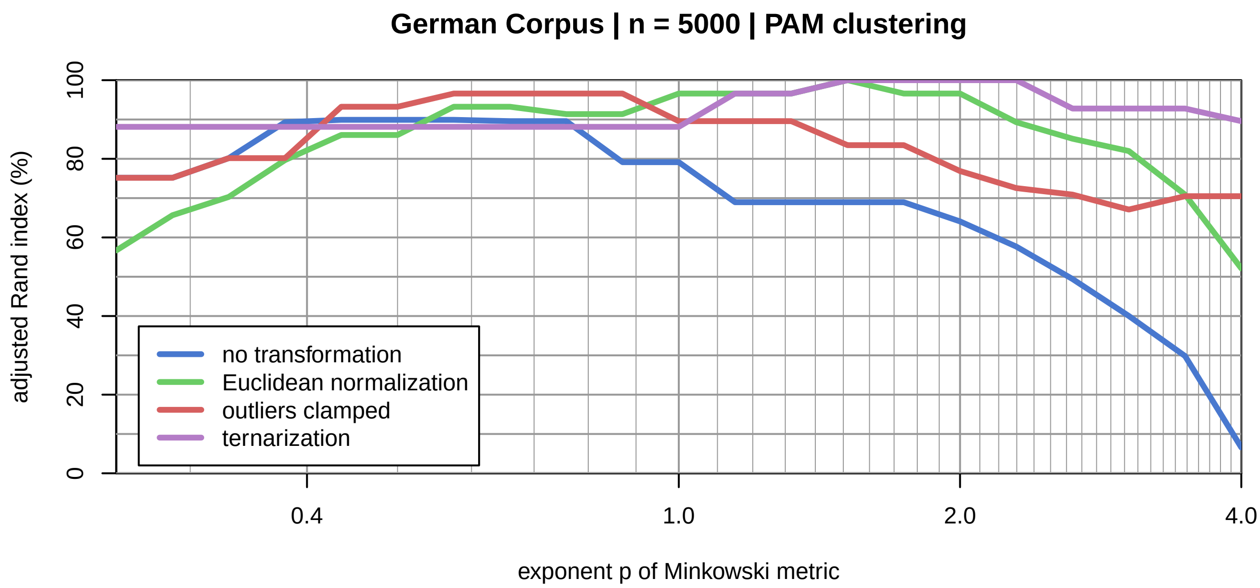


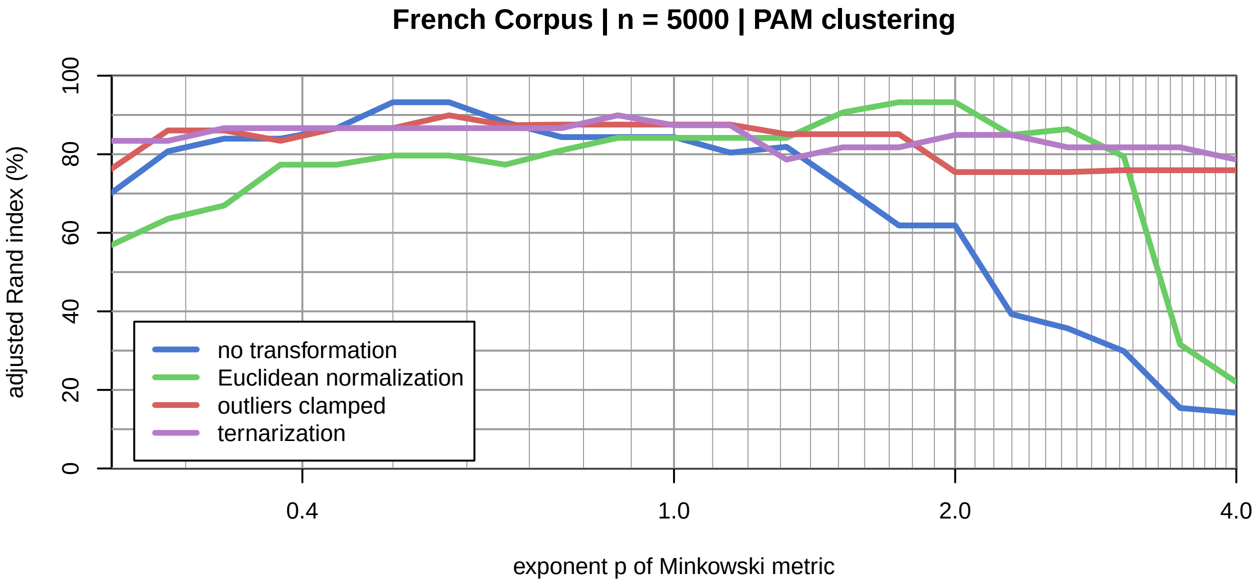


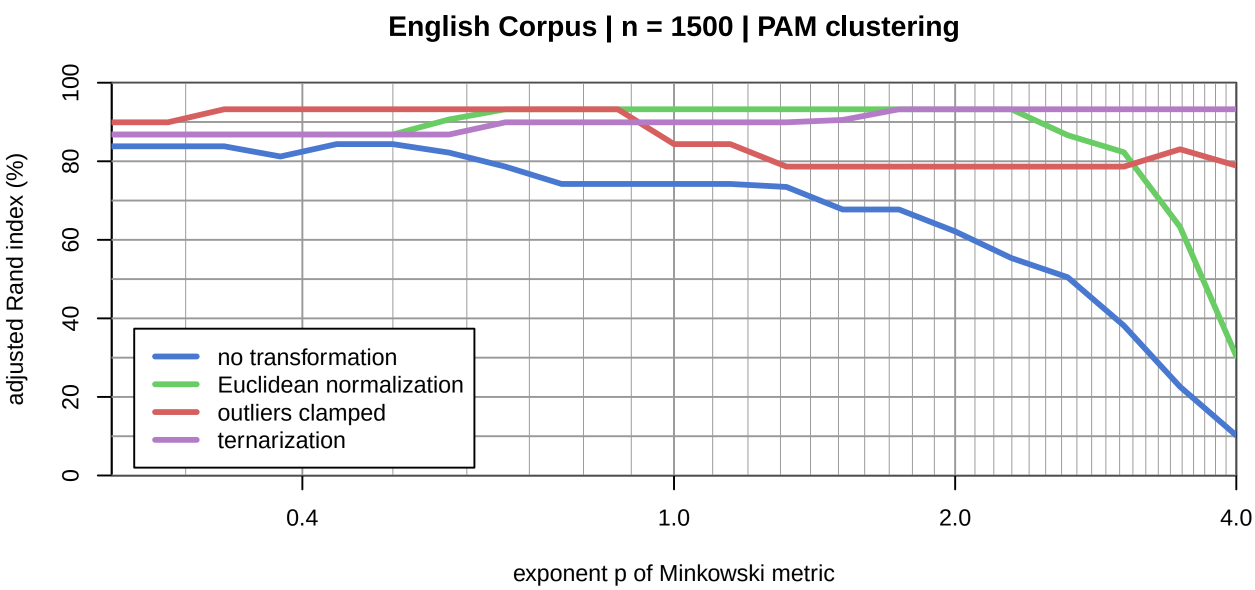

Finally, compare Euclidean normalization with clamping of outliers and ternarization of the vectors.

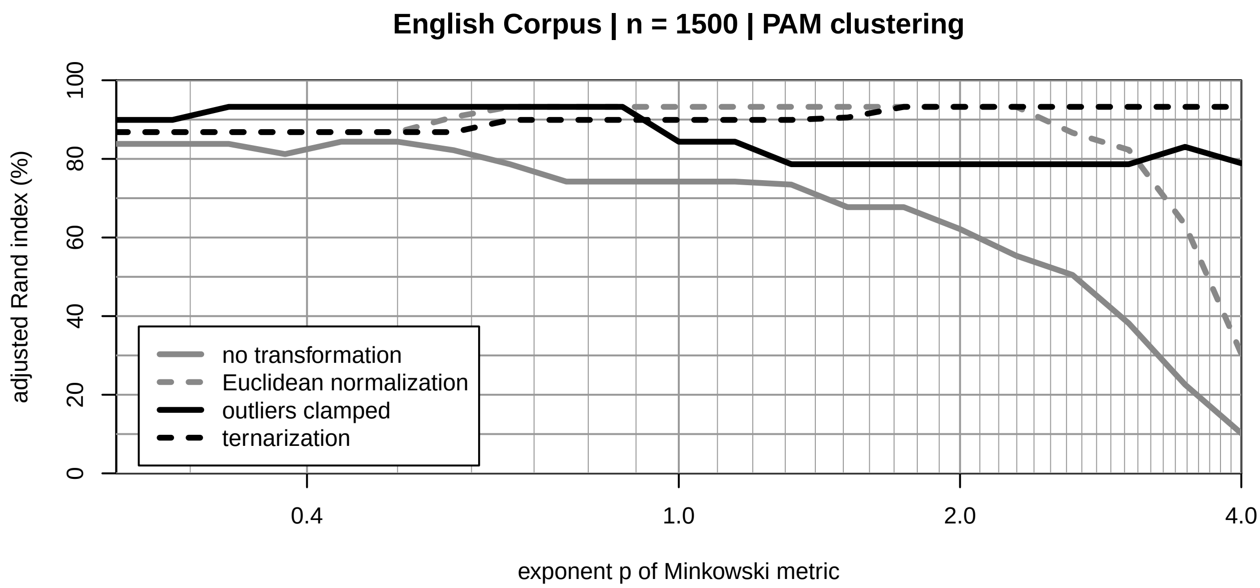

English Corpus | n = 5000 | PAM clustering
adjusted Rand index (%)
0
20
40
60
80
100
0.4
1.0
2.0
4.0
exponent p of Minkowski metric
no transformation
Euclidean normalization
outliers clamped
ternarization

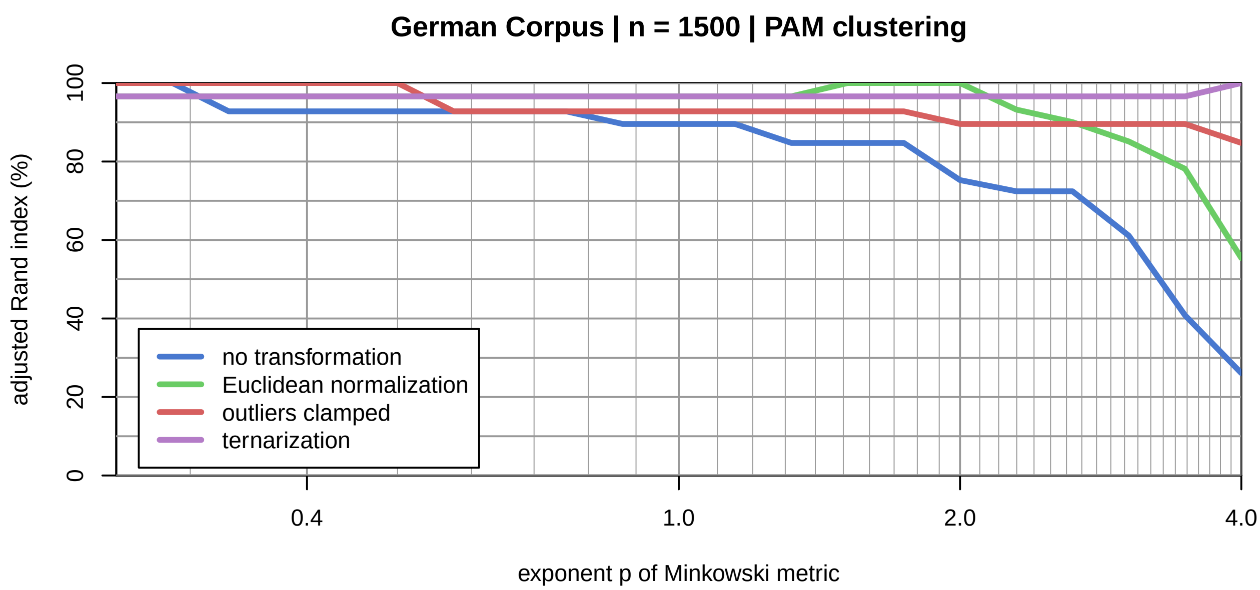

German Corpus | n = 5000 | PAM clustering
adjusted Rand index (%)
0
20
40
60
80
100
0.4
1.0
2.0
4.0
exponent p of Minkowski metric
no transformation
Euclidean normalization
outliers clamped
ternarization

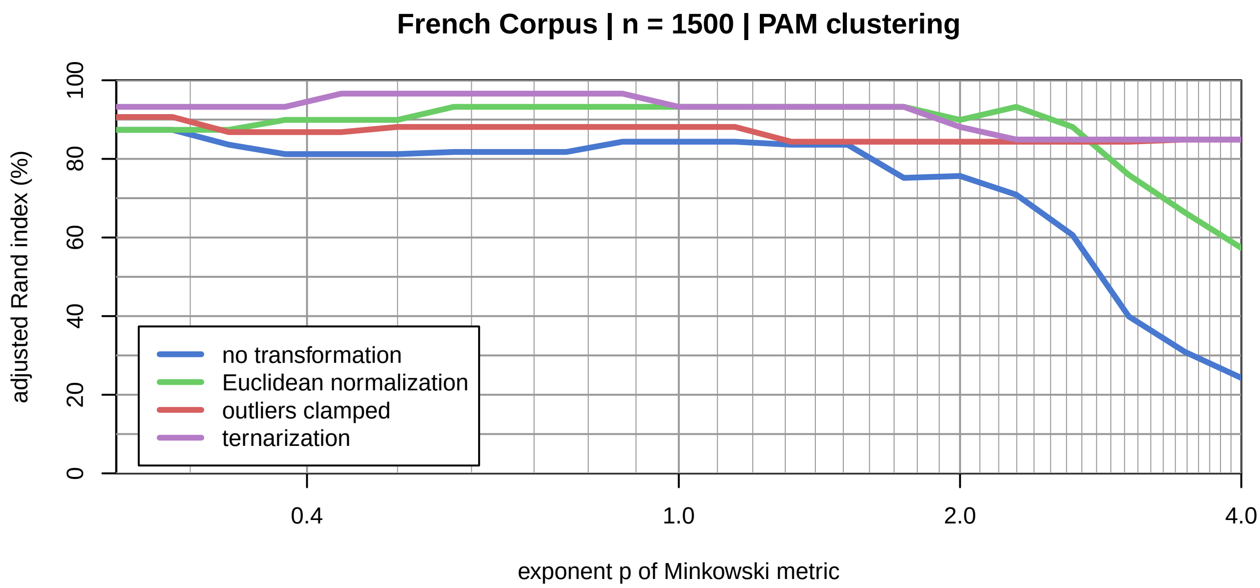



Clamping outliers and – even more so – ternarization lead to very robust and good performance. Except for the French corpus, they are as good as cosine / Euclidean normalization and aren't sensitive to the choice of $p$. The unusually high dimensionality $n_w = 5000$ may have some influence as well, though, so reproduce the analysis for $n_w = 1500$ (as a compromise between the commonly used values $n_w = 1000$ and $n_w = 2000$).




English Corpus | n = 1500 | PAM clustering
adjusted Rand index (%)
0
20
40
60
80
100
0.4
1.0
2.0
4.0
exponent p of Minkowski metric
no transformation
Euclidean normalization
outliers clamped
ternarization


German Corpus | n = 1500 | PAM clustering
adjusted Rand index (%)
0
20
40
60
80
100
0.4
1.0
2.0
4.0
exponent p of Minkowski metric
no transformation
Euclidean normalization
outliers clamped
ternarization




# 4 Analyzing the distance distributions

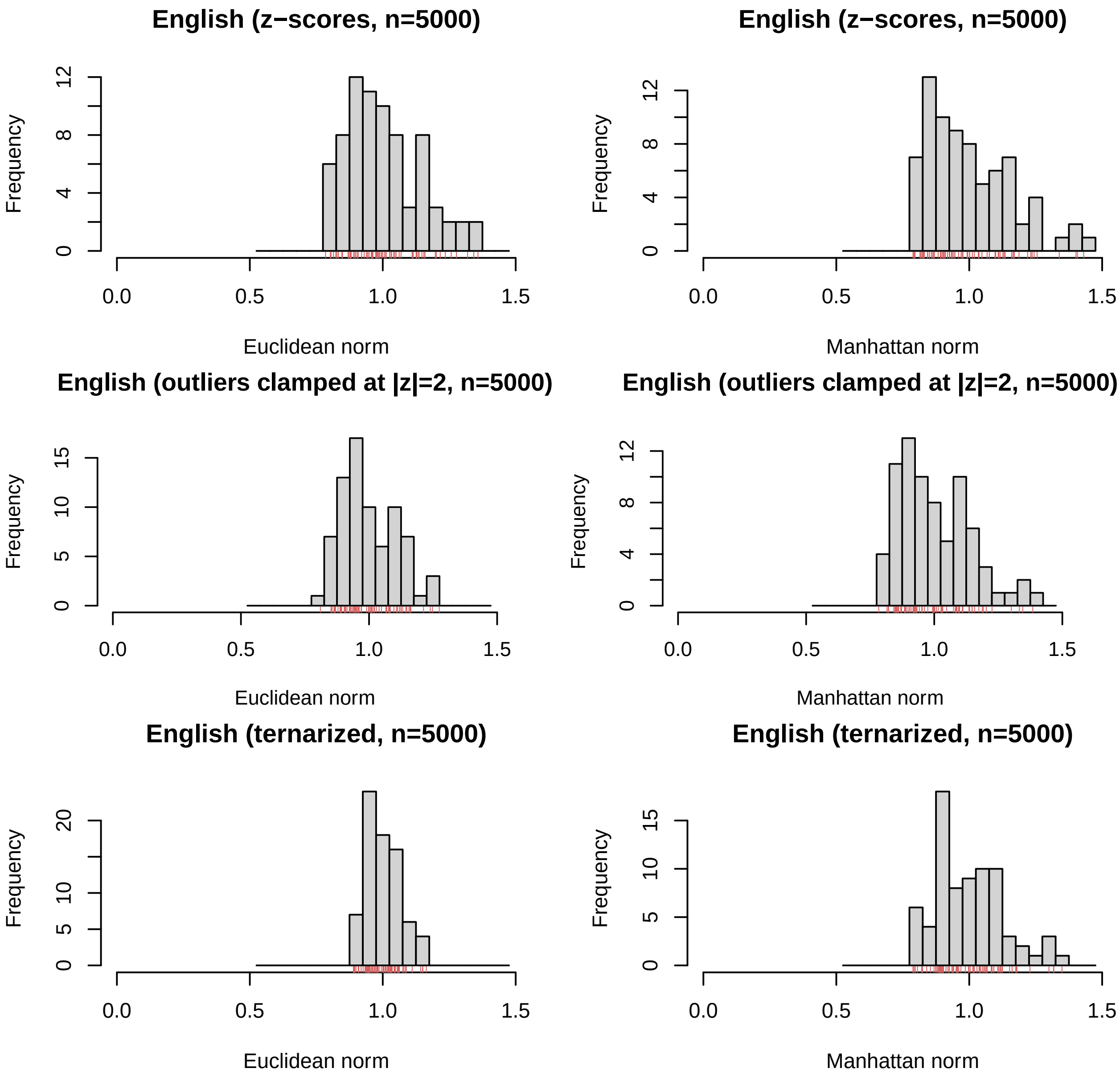
In order to gain a better understanding of how normalization and other vector transformations affect the distances between points, we compare the distributions of distances between texts from the same author and texts from different authors. Since transformations (esp. normalization) can drastically change the scale of distances, we rescale vectors so that they have an *average* length of 1 (which is a no-op for normalized vectors).

Helper functions for rescaling and for obtaining distances of same-author and different-author text pairs. Both functions pass on additional arguments to `rowNorms` and `dist.matrix`, respecticely, so different norms and metrics can be selected.

```
avg.normalize <- function (M, ...) {
  scaleMargins(M, rows = 1 / mean(rowNorms(M, ... )))
}
dist.text.pairs <- function (M, gold, ...) {
  DM <- dist.matrix(M, ...)
  is.same <- outer(gold, gold, `==`) # marks same-author pairs
  res1 <- data.frame(d=DM[upper.tri(DM) & is.same], same="same author")
  res2 <- data.frame(d=DM[upper.tri(DM) & !is.same], same="different authors")
  rbind(res1, res2)
}
```

## 4.1 The effect of vector transformations on vector length

The following histograms show how vector lengths are affected by the different feature transformations. Keep in mind that $L_2$-normalized vectors have the same Euclidean length of 1. We plot the distributions in the English corpus for $n_w = 5000$, where $\Delta_Q$ already shows a substantial decline in clustering quality.

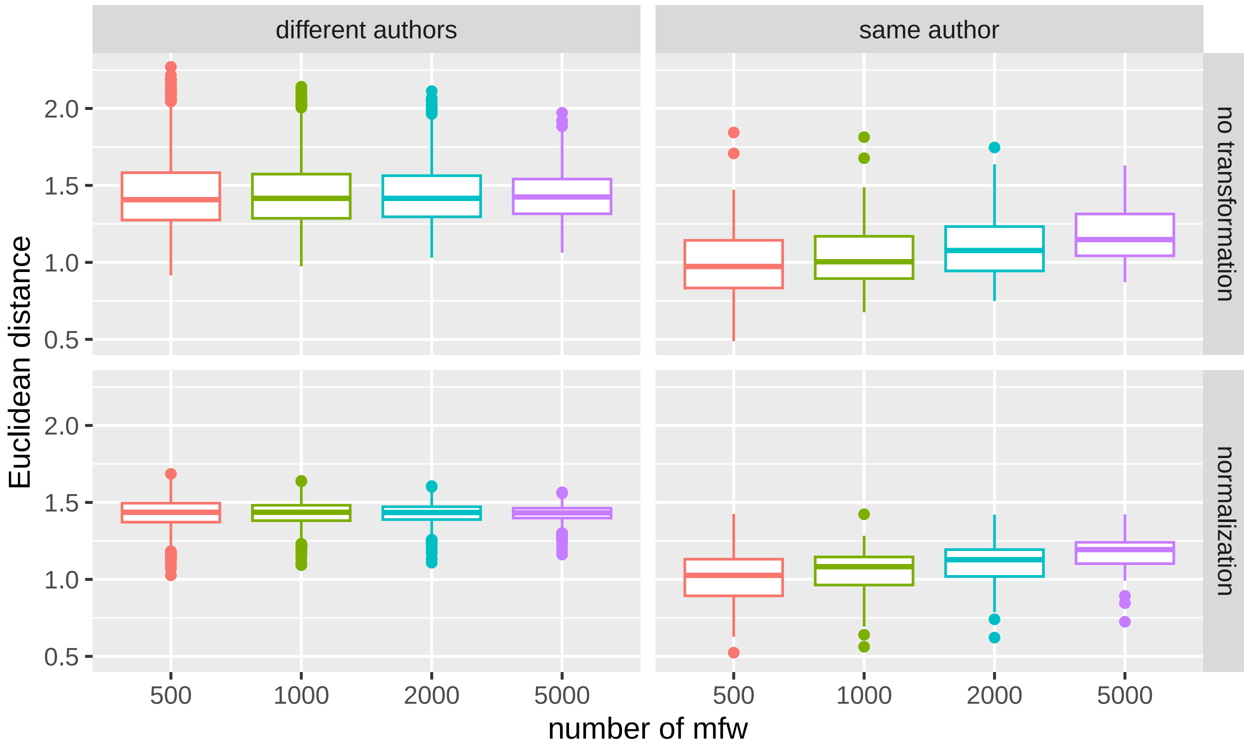



## 4.2 The effect of vector transformation on distances

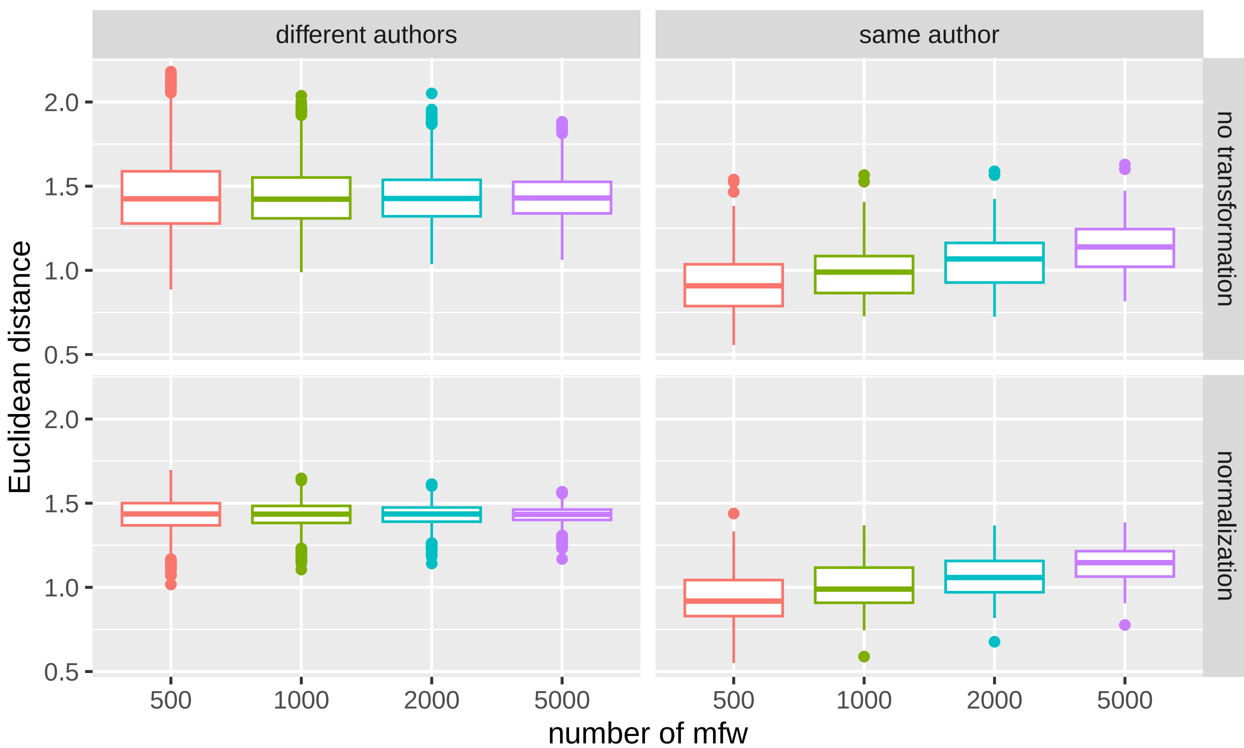
In a first step, look at how the length $n_w$ of the feature vectors affects the distance distributions with and without normalization. Obviously the main effect of normalization is not to bring texts from the same author closer to each other (left panels), but rather to reduce variability of distances both within the same-author text pairs and within the different-author text pairs. This indicates that vector length – i.e. the amplitude of deviations $\mathbf{z}(D)$ from the mean – is indeed a "noise" factor affecting all text pairs. With the reduced variability, there is a fairly good separation between the groups, explaining the excellent clustering results.

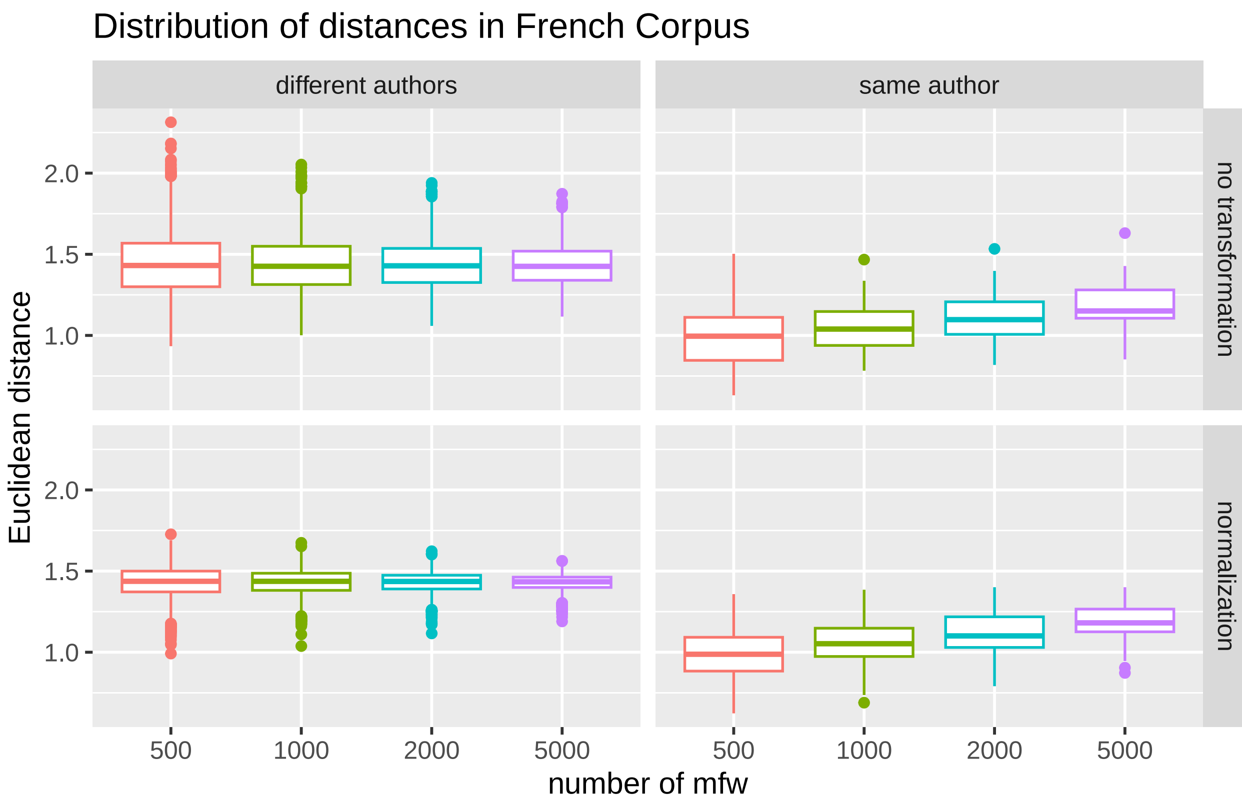
Two other observations are also noteworthy for the normalized vectors: (i) different-author pairs are almost precisely orthogonal (an angle of 90° corresponds to a Euclidean distance of $\sqrt{2} = 1.414\ldots$ between normalized vectors); (ii) the variability in both groups becomes smaller with an increasing number $n_w$ of mfw. This suggests a possible explanation in terms of the "curse of dimensionality" (which states that random vectors in a high-dimensional space are nearly orthogonal to each other with high probability), which should be explored further in subsequent experiments.

Distribution of distances in English Corpus



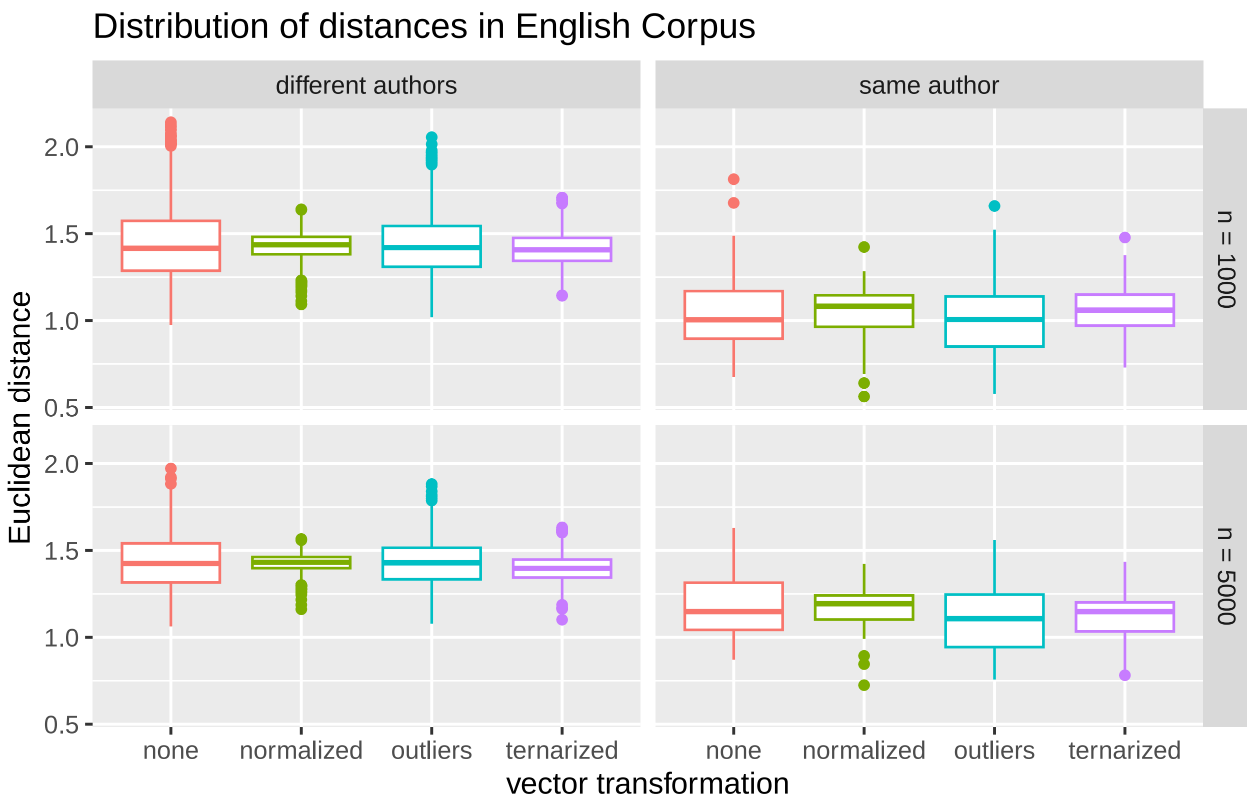

Distribution of distances in German Corpus

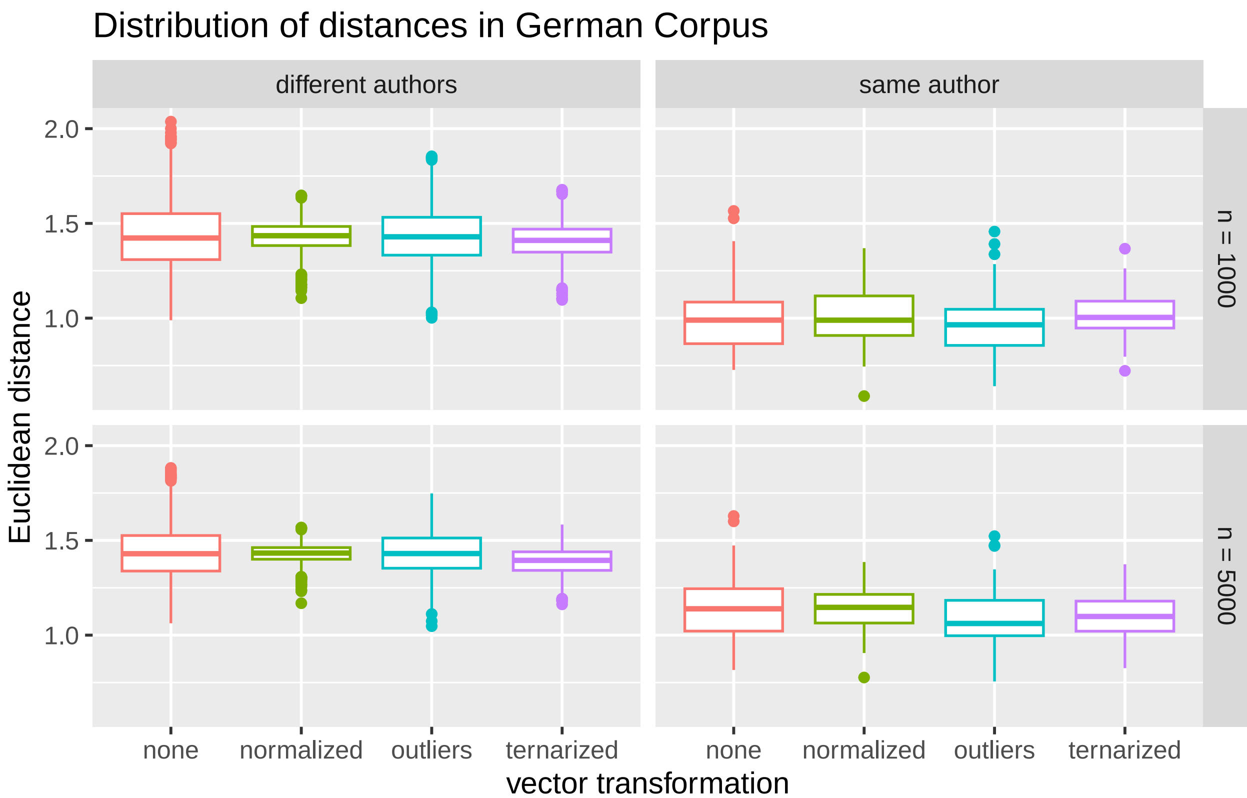


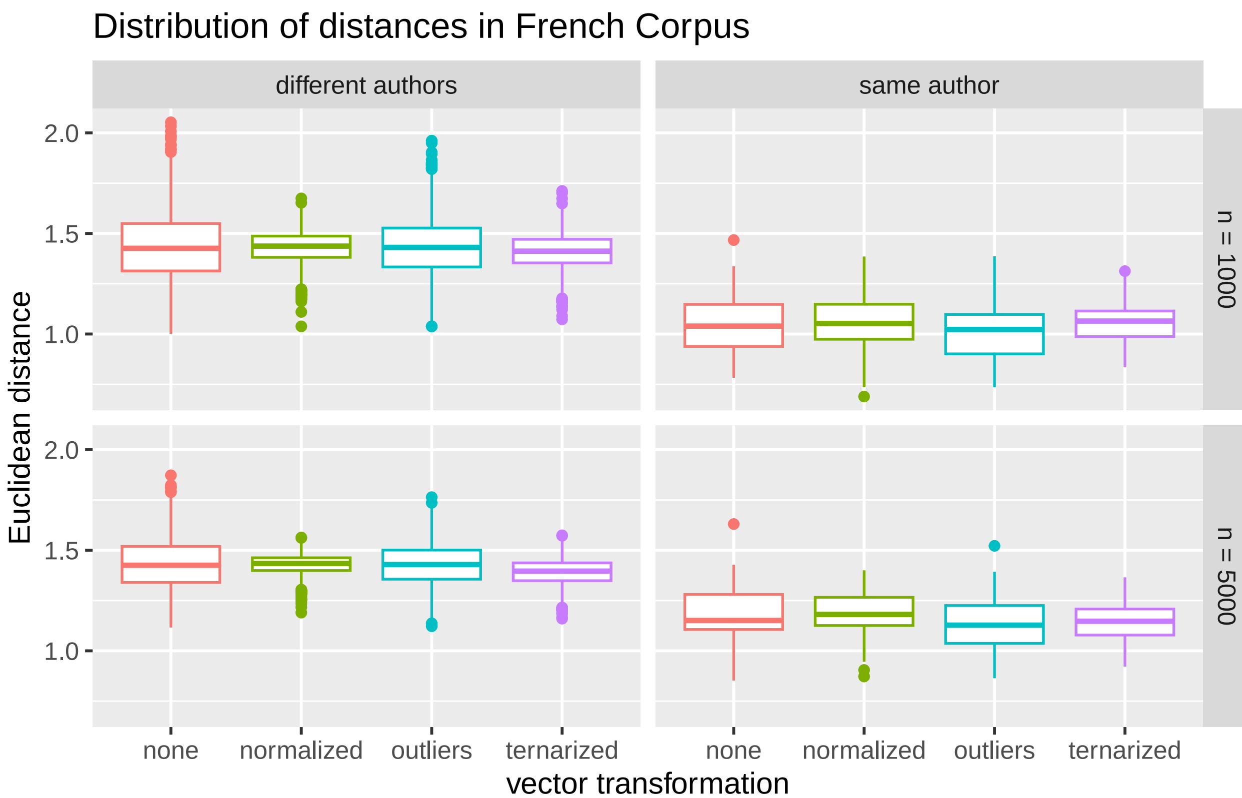



We can now compare the distance distribution under normalization with outlier clamping and ternarization. Again, the plots suggest a straightforward interpretation. Clamping outliers (with $|z| > 2$) has little effect on the distance distributions: variability is reduced slightly and distances between same-author texts are reduced by a very small amount (compare the medians and lower middle quartiles). Ternarization, on the other hand, has almost the same effect as normalization, providing very clear support for the key profile hypothesis (H2).

Further experiments should explore the connection between H2 and the "curse of dimensionality". Perhaps our "key profiles" provide an intuitive explanation for the latter.


Distribution of distances in English Corpus
different authors
same author
n = 1000
n = 5000
Euclidean distance
2.0
1.5
1.0
0.5
none
normalized
outliers
ternarized
vector transformation


Distribution of distances in German Corpus
different authors
same author
n = 1000
n = 5000
Euclidean distance
2.0
1.5
1.0
none
normalized
outliers
ternarized
vector transformation




## 5 Addendum: N-Gram Tracing

Grieve et al. (submitted) propose an **n-gram tracing** technique for authorship attribution of short texts, which is based on the proportion of word or character n-gram *types* that are known from a relatively large amount of training data of the respective candidate author. They demonstrate high accuracy in distinguishing between texts from Abraham Lincoln and John Hay and use the method to attribute the *Bixby letter* to Hay.

- for word unigrams, this is an extension of our ternarization approach and supports the same intuitive hypothesis: that it is patterns of word use rather than numeric frequency differences which determine the characteristic fingerprint of an author
- test whether n-gram tracing is likely to work on our data, using a binarized matrix of word unigrams and cosine distance (which gives us a rough normalization for text size, but may not be as good as the original technique); note that Manhattan distance with L1 normalization doesn't make sense because we need to compute distances between binary vectors (and normalize post-hoc)

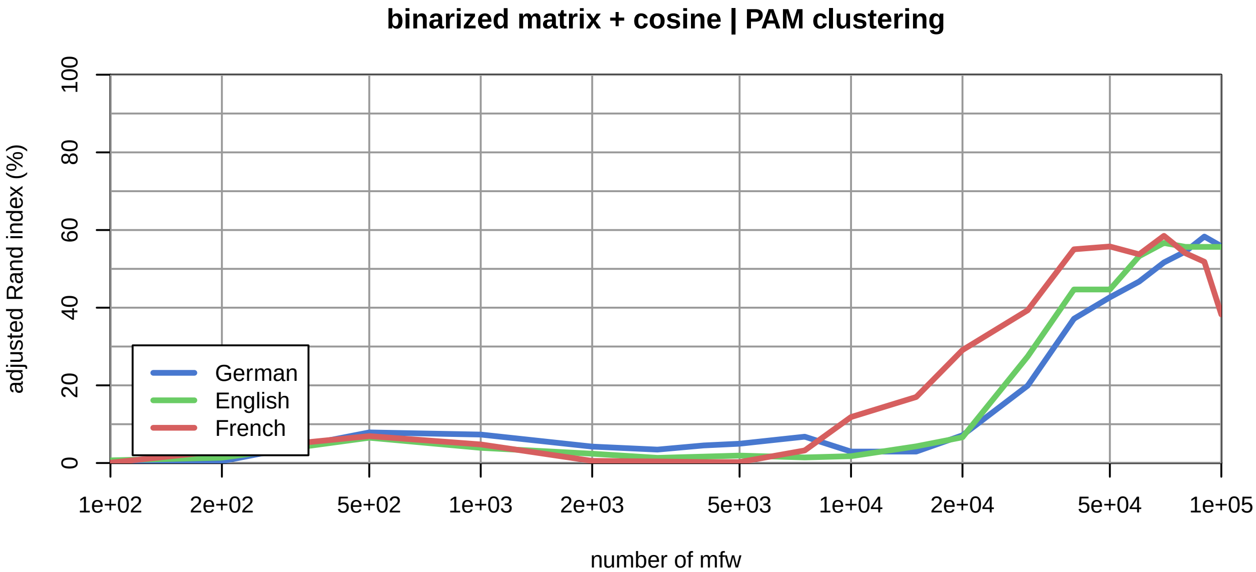
Preparation: binarize matrices (and perhasps convert to sparse format for better efficiency)

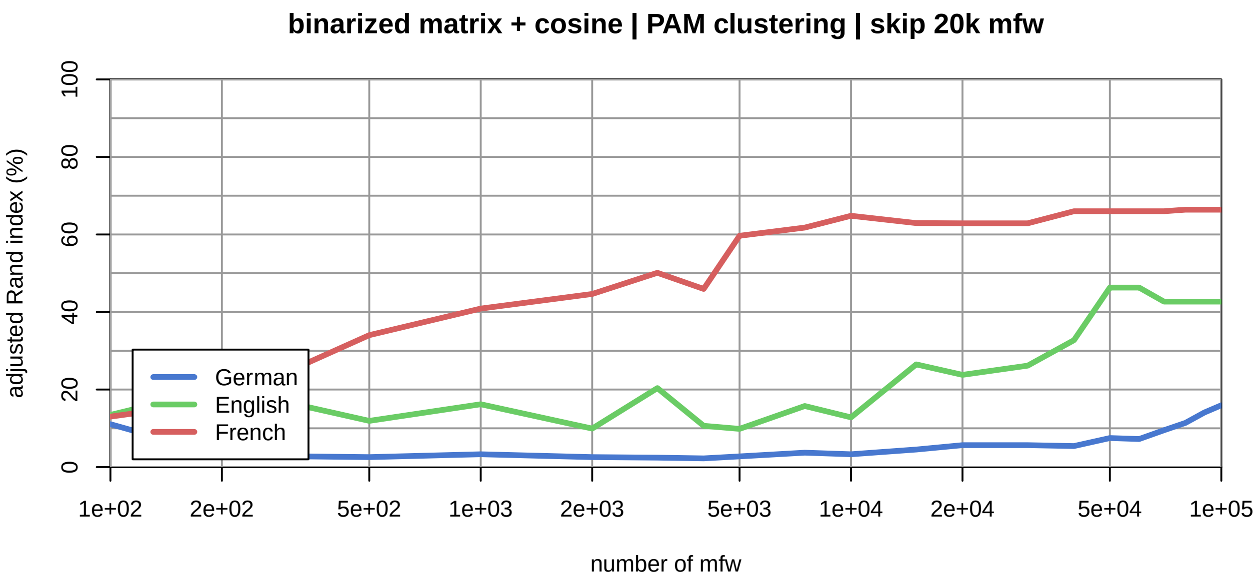
```
BinDE <- sign(FreqDE$M)
BinEN <- sign(FreqEN$M)
BinFR <- sign(FreqFR$M)
```

Set up the evaluation plots

```
n.vals <- c(100,200,500,1000,2000,3000,4000,5000,7500,10000,15000,20000,
            30000,40000,50000,60000,70000,80000,90000,100000)
draw.grid <- function () {
  abline(h=seq(0, 100, 10), col="grey60")
  abline(v=c(100,200,500,1000,2000,5000,10000,20000,50000,100000), col="grey60")
}
```

And go:


binarized matrix + cosine | PAM clustering
adjusted Rand index (%)
0
20
40
60
80
100
1e+02
2e+02
5e+02
1e+03
2e+03
5e+03
1e+04
2e+04
5e+04
1e+05
number of mfw
German
English
French


binarized matrix + cosine | PAM clustering | skip 20k mfw
adjusted Rand index (%)
0
20
40
60
80
100
1e+02
2e+02
5e+02
1e+03
2e+03
5e+03
1e+04
2e+04
5e+04
1e+05
number of mfw
German
English
French