Lab 3 - Parallelizing k-means
Stat 215A, Fall 2014

Russell Chen

1 Running time comparisons

The following was run on the scf cluster so that timing results are somewhat comparable:

```
set.seed(387)
testclust1 <- sample(1:6, size=45000, replace=TRUE)
testclust2 <- sample(1:6, size=45000, replace=TRUE)
microbenchmark(ComputeSim(testclust1, testclust2, 6), SimC(testclust1, testclust2), times=5)
```

Two clustering partitions are simulated, with 6 clusters in each. `ComputeSim()` is the function I wrote in R to compute the Fowlkes-Mallows index between the two partitions while `SimC()` is a function written in C++ to calculate the same index, although it is implemented with a different algorithm. The output of the last line calling `microbenchmark` is:

Unit: milliseconds

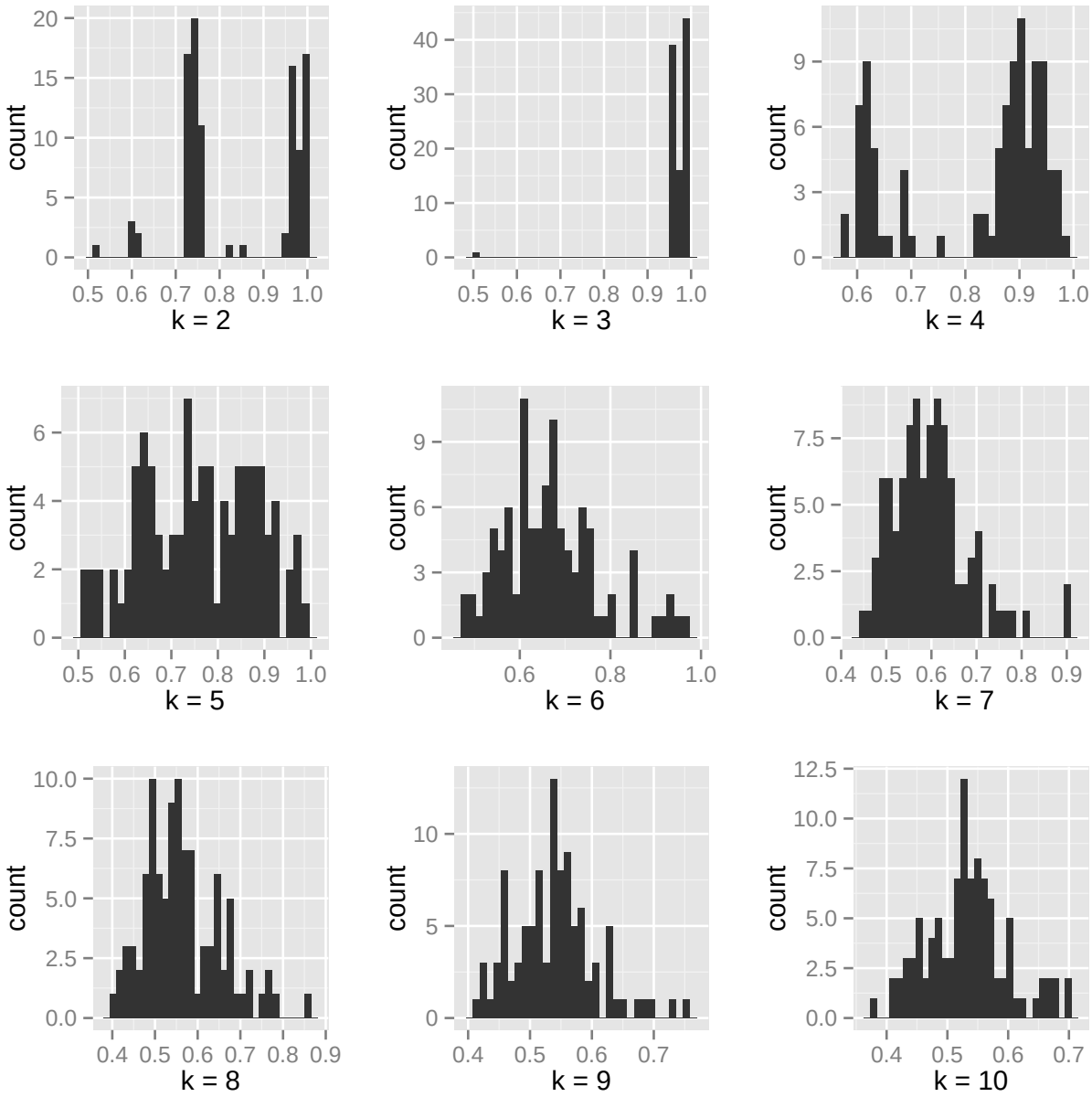
expr	min	lq	median	uq	max	neval
ComputeSim(testclust1, testclust2, 6)	44.71918	49.12471	58.62695	104.0415	104.3096	5
SimC(testclust1, testclust2)	6237.89078	6238.06708	6241.31555	6244.4172	6248.3520	5

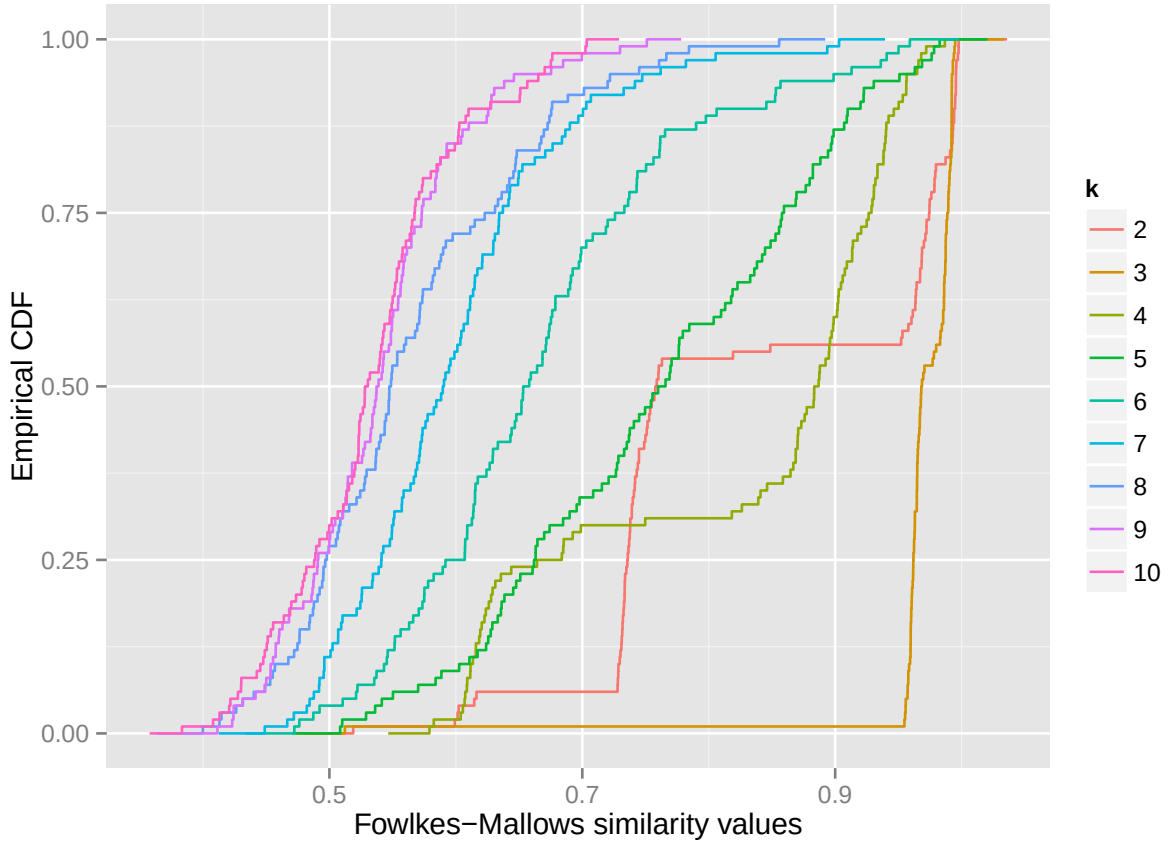
Unexpectedly, the function written in R is much faster than the one written in C++ despite having a rather large variance in running time. This is probably due to the fact that `ComputeSim()` only loops through k (length 6 in this case) in each of the nested for-loops while `SimC()` loops through an entire vector of clusterings (of length 45000 in this case) in each nested for-loop.

I do not know why the variance in running times for `ComputeSim()` is so large, nor how this can be reduced. Unfortunately, I could not implement `ComputeSim()` in C++ to see if a substantial speed-up could be achieved. My guess is that this would be difficult since it already makes extensive use of vectorizing.

2 Choice of k

On the next couple pages is a plot of the histograms of the Fowlkes-Mallows similarity indices for each value of k from 2 to 10 together with the empirical c.d.f.s overlaid in one plot, as Figures 3, 6 and 7 in Ben-Hur et al. similarly depict for various other datasets. The authors recommend that in these plots, we should look for a "transition from a stable clustering to an unstable one." For the binary-coded linguistic data provided, it seems that the only stable clustering is for $k = 3$. By comparison, the histograms for all other values of k show unstable partitions. Going by the procedure outlined in the paper, I would say there are 3 clusters in the data.

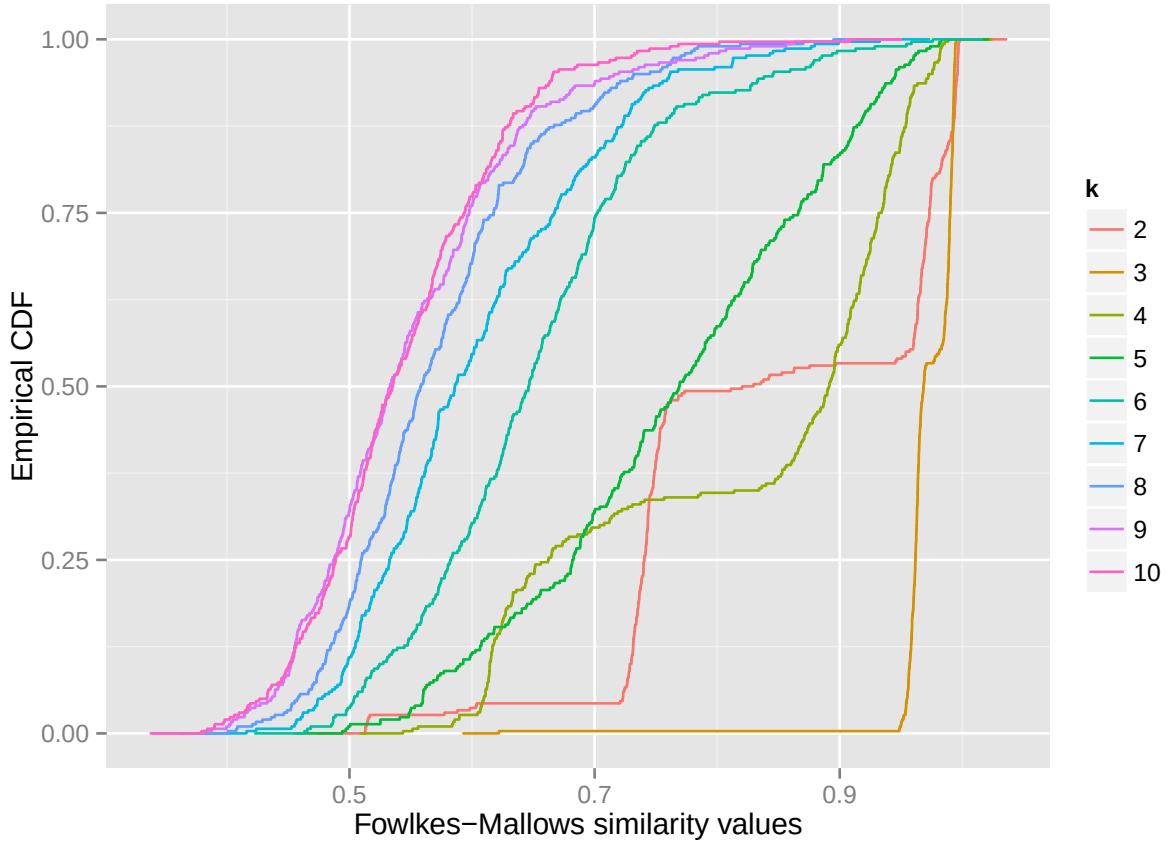

count
k = 2
k = 3
k = 4
k = 5
k = 6
k = 7
k = 8
k = 9
k = 10



3 Comments on method of assessing stability

The method described in Ben-Hur et al. of assessing the stability of a clustering result is interesting. In particular, I agree with the authors that this method can be used to show evidence of a lack of stable clusters in a dataset, with a caveat described below. However, as demonstrated with the linguistic data, ambiguous cases may still arise where it is not clear what the optimal number of clusters is. Looking at the empirical c.d.f. plot above again, one might argue that between $k = 5$ and $k = 6$, the clustering transitions from stable to unstable and we should pick 5 as the optimal number of clusters. The authors acknowledge that there will be situations where a judgement call is necessary, leading them to conclude that this procedure should be viewed "as a general exploratory tool, and not just as a way of selecting an optimal number of clusters." I think it is prudent to take note of this caution that the procedure is not a one-size-fits-all solution for selecting the optimal number of clusters in any given dataset.

One concern I have that was not addressed by the authors is: how do we know that the inner loop is long enough? For the binary-coded linguistic data, the inner loop was chosen to be 100 iterations long. Is it possible that this was not long enough, thereby producing a lot of variation in results? As a brief experiment, I ran the algorithm with a longer inner loop of 300. The resulting empirical c.d.f. plot is shown on the next page. It seems that the transition from unstable to more unstable clustering between $k = 5$ and $k = 6$ is ever so slightly clearer. This suggests that the results obtained from this procedure are indeed slightly sensitive to the length of the inner loop. Of course, if we had endless computing power at our disposal, we could run the inner loop 50000 times. Even then, if there is no evidence of a transition, we may still suspect that the inner loop was not long enough to see such a transition clearly. Especially for large datasets (for which computing power is a significant limitation), this undercuts the authors' claim that seeing no transition in the ecdf plot is evidence of absence of natural clusters in the data.


1.00
0.75
0.50
0.25
0.00
Empirical CDF
0.5
0.7
0.9
Fowlkes−Mallows similarity values
k
2
3
4
5
6
7
8
9
10