Supplementary materials to TITLE

Martin Papenberg

Standard algorithm for anticlustering

To assign samples to batches, we use anticlustering. Anticlustering is an optimization method that is characterized by (a) an objective function that quantifies the balance among batches, and (b) an algorithm that conducts the batch assignment in such a way that balance among batches is maximized via the objective function. Anticlustering owes its name to the fact that the objective functions it uses are the reversal of criteria used in cluster analysis. For example, Späth (1986) already recognized that by maximizing instead of minimizing the k-means criterion (the "variance"), he was able to create groups that are similar to each other, and presented it as an improvement over the more intuitive random assignment.

In our application, we optimize the (a) diversity objective using (b) the exchange algorithm by Papenberg and Klau (2021). This constellation of objective and algorithm is also the default method implemented in the R package `anticlust` (Papenberg 2024). The diversity is defined as the total sum the within-cluster sums of dissimilarities (Brusco, Cradit, and Steinley 2020). When assigning samples in such a way that the diversity is maximized, we simultaneously maximize similarity between batches. In Papenberg and Klau (2021), we referred to the maximization of the diversity as "anticluster editing" because the minimization of the diversity is also well-known from the area of cluster analysis—under the term "cluster editing" (Shamir, Sharan, and Tsur 2004; Böcker, Briesemeister, and Klau 2011).

The anticlustering exchange algorithm takes as input (1) a distance matrix, quantifying the pairwise dissimilarity among samples, and (2) the number of batches K. As measure of dissimilarity, we use the common Euclidean distance, which transfers the features of our samples to pairwise distances. It is defined as

$$d(x, y) = \sqrt{\sum_{i=1}^{M} (x_i - y_i)^2}$$

where M is the number of numeric features describing two samples $x = (x_1, \dots, x_M)$ and $y = (y_1, \dots, y_M)$. When samples are described by two features, the Euclidean distance corresponds to the geometric, "straightline" distance between two points in a two-dimensional space; more similar data points are closer to each other. Figure 1 illustrates the computation of the Euclidean distance and the diversity for the numeric features BMI and age that were used in our motivating application. For categorical variables, we use binary coding before including them in the computation of the Euclidean distance. Table 1 illustrates binary coding for the feature race in our data set, which had five unique values: (a) Asian, (b) White, (c) Native Hawaiian or Other Pacific Islander, (d) Black, (e) Native American. Such a categorical variable can be recoded into four binary variables to maintain the information included in the original variable.

Table 1: Illustrate the recoding of the variable Race using four binary variables.

Race	Black	Native American	Pacific Islander	White
Black	1	0	0	0
Pacific Islander	0	0	1	0
Native American	0	1	0	0
White	0	0	0	1

Race	Black	Native American	Pacific Islander	White
White	0	0	0	1
Pacific Islander	0	0	1	0
Pacific Islander	0	0	1	0
White	0	0	0	1
Asian	0	0	0	0
Native American	0	1	0	0
Native American	0	1	0	0
Asian	0	0	0	0

The exchange algorithm that maximizes the diversity consists of two steps: an initialization step and an optimization step. As initialization, it randomly assigns samples to K equal-sized batches. In principle, unequal-sized batches would also be possible with our algorithm, but equal-sized batches were required in the current application. After initialization, the algorithm selects the first sample and checks how the diversity would change if the sample were swapped with each sample that is currently assigned to a different batch. After simulating each exchange—that is $N - \frac{N}{K}$ exchanges—it realizes the one exchange that increases the diversity the most. It does not conduct an exchange if no improvement in diversity is possible. This procedure is repeated for each sample, leading to the evaluation of $N \cdot \frac{N}{K}$ exchanges in total; it terminates after the last sample was processed. The procedure might also restart at the first element and reiterate through all samples until no exchange leads to an improvement any more, i.e., until a local maximum is found. In `anticlust`, we also implemented this local maximum search, which corresponds to the algorithm LCW by Weitz and Lakshminarayanan (1998). For better results, it is also possible to restart the search algorithm multiple times using different (random) initializations.

Including must-link constraints with anticlustering

To include must-link constraints with anticlustering, we adjusted both the initialization phase as well as the optimization phase of the exchange algorithm. For initialization, we recognize that we now may have to assign multiple samples simultaneously ("must-link groups") to a batch—not all samples can be considered independently of all other samples. Therefore, we first collect all samples that are have at least one other sample ("must-link partner") that has to be assigned to the same group.

- We obtain a list of all must-link groups together and store them with their cardinality.
- We must assign these must-link groups to the K batches.
- This corresponds to a bin packing problem!
    - We have n elements – n = numer of must-link groups – where each element has a size, corresponding to the number of elements in that must-link group.
    - Each bin corresponds to a batch and has target capacity of $\frac{N}{K}$
    - We use "randomized fit" heuristic to fill the batches: Select first bacthes that fits, and try out batches in random order (leads to rather even distribution in all batches). A "worst fit" heuristic might also be used, but some random element is good if we restart our optimization heuristic multiple times for better results.
- The remaining elements (that are not part of must-link groups) can be assigned randomly to batches (while respecting the cardinality constraints / equal-sized batches) – they are not part of the bin packing problem.

Optimization algorithm:

- "Merge" samples: A must-link group is treated as a single unit during optimization.
- New distance matrix has to be computed: New distances are given by summing up all distances between all elements that are part of must-link groups (this works for the diversity objective because it is a mere sum)
- Exchange heuristic is conducted on the merged elements. We only exchange must-link groups of the same size to ensure that the cardinality constraints are still respected in the end (equal-sized groups).

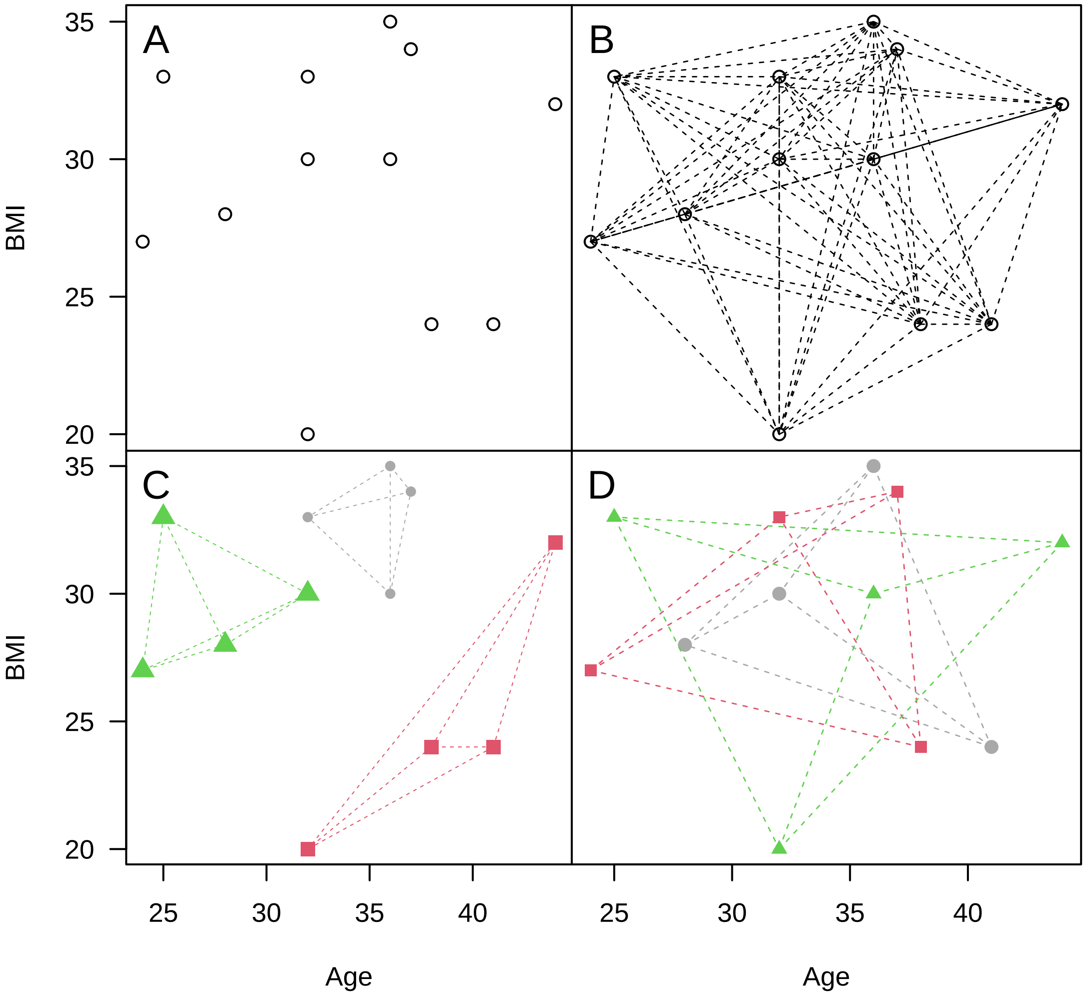

Figure 1: Illustrates the conversion from numeric features to Euclidean distance, and (anti)clustering assignments based on minimum and maximum diversity using the Euclidean distance. Panel A illustrates the BMI and age of twelve women in our synthetic data in a scatter plot. Panel B represents the Euclidean distances between features as a straight line in the two-dimensional space. The Euclidean distance is proportional to the length of the connecting lines in panel B. Panel C illustrates a clustering assignment of the 12 data points to $K = 3$ equal-sized groups via *minimum* diversity. Panel D illustrates an anticlustering assignment of the 12 data points to $K = 3$ equal-sized groups via *maximum* diversity. The diversity is computed as the sum of within-(anti)cluster distances, which are highlighted in Panel C and Panel D through connecting lines.

Anticlustering vs. OSAT

- Some more details on the simulation study
- Given the parameters of the simulation, anticlustering strongly outperforms the OSAT method

Optimal algorithm for anticlustering under cannot-link constraints

- Solving anticlustering optimally (i.e., finding a batch assignment with globally optimal value in diversity) under must-link constraints is possible using the ILP by Papenberg and Klau (2021)
- We have to adjust the input matrix: Samples that must be linked receive sufficiently large pairwise distance; in this case, a globally optimal solution must pair these samples in the same batch (if the constraints can be fulfilled at all)
- We investigate the feasibiliy of the optimal approach, which solves a computationally difficult (NP hard) problem (it seems to scale to about 40-50 samples; for larger data sets, a heuristic is required)
- We can use the optimal algorithm to validate the quality of our heuristic (how often does it find the globally optimal solution)

References

Böcker, Sebastian, Sebastian Briesemeister, and Gunnar W Klau. 2011. "Exact Algorithms for Cluster Editing: Evaluation and Experiments." *Algorithmica* 60: 316–34. https://doi.org/10.1007/s00453-009-9339-7.

Brusco, Michael J, J Dennis Cradit, and Douglas Steinley. 2020. "Combining Diversity and Dispersion Criteria for Anticlustering: A Bicriterion Approach." *British Journal of Mathematical and Statistical Psychology* 73 (3): 375–96. https://doi.org/10.1111/bmsp.12186.

Papenberg, Martin. 2024. "Anticlust: Subset Partitioning via Anticlustering." https://cran.r-project.org/package=anticlust.

Papenberg, Martin, and Gunnar W Klau. 2021. "Using Anticlustering to Partition Data Sets into Equivalent Parts." *Psychological Methods* 26 (2): 161–74. https://doi.org/10.1037/met0000301.

Shamir, Ron, Roded Sharan, and D Tsur. 2004. "Cluster Graph Modification Problems." *Discrete Applied Mathematics* 144: 173–82. https://doi.org/10.1016/j.dam.2004.01.007.

Späth, H. 1986. "Anticlustering: Maximizing the Variance Criterion." *Control and Cybernetics* 15 (2): 213–18.

Weitz, RR, and S Lakshminarayanan. 1998. "An Empirical Comparison of Heuristic Methods for Creating Maximally Diverse Groups." *Journal of the Operational Research Society* 49 (6): 635–46. https://doi.org/10.1057/palgrave.jors.2600510.