Explorative Analysis and Visualization of Large Information Spaces

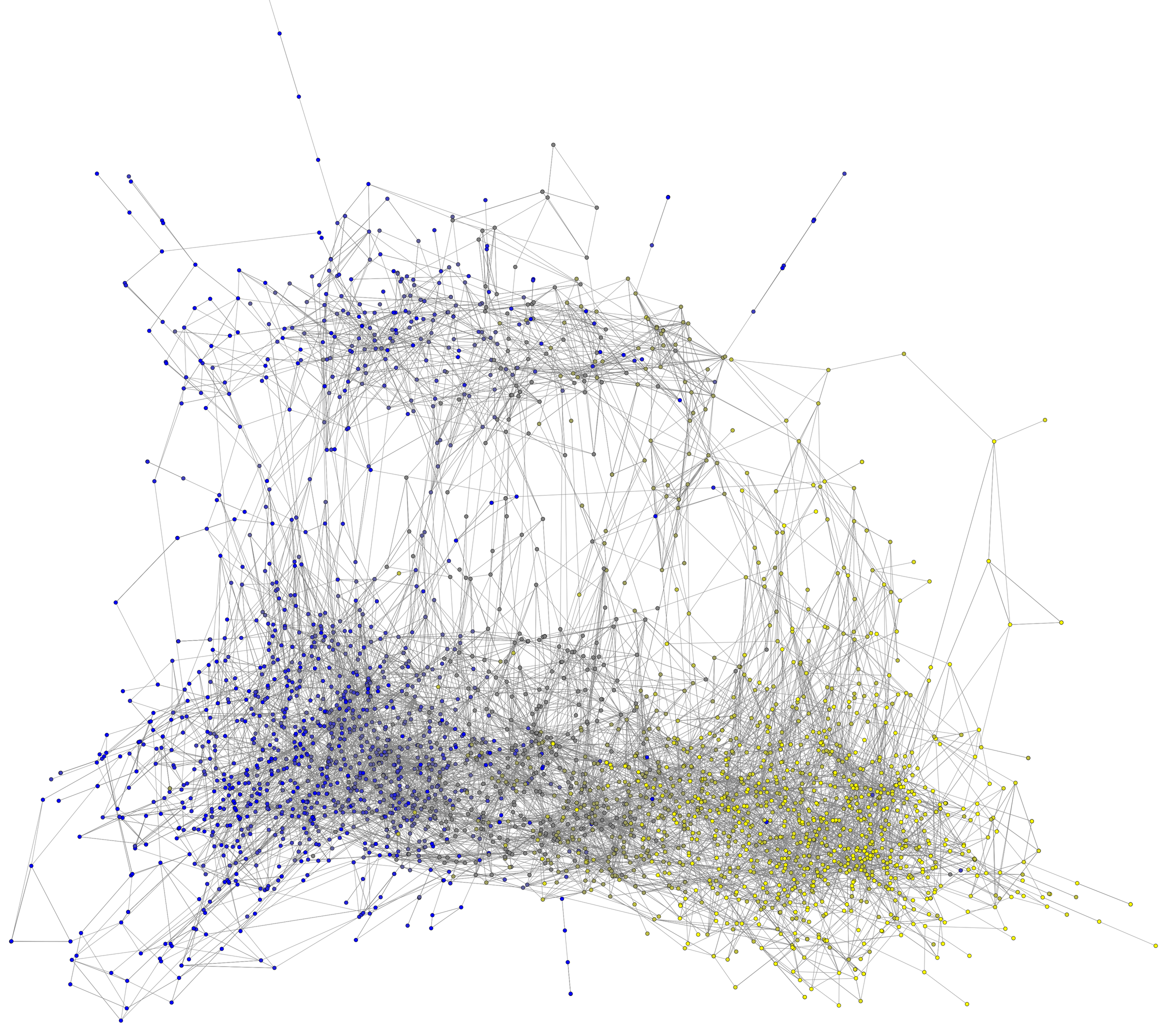
Applications of Multidimensional Scaling to Graph Drawing

Christian Pich

Universität Konstanz

Abstract

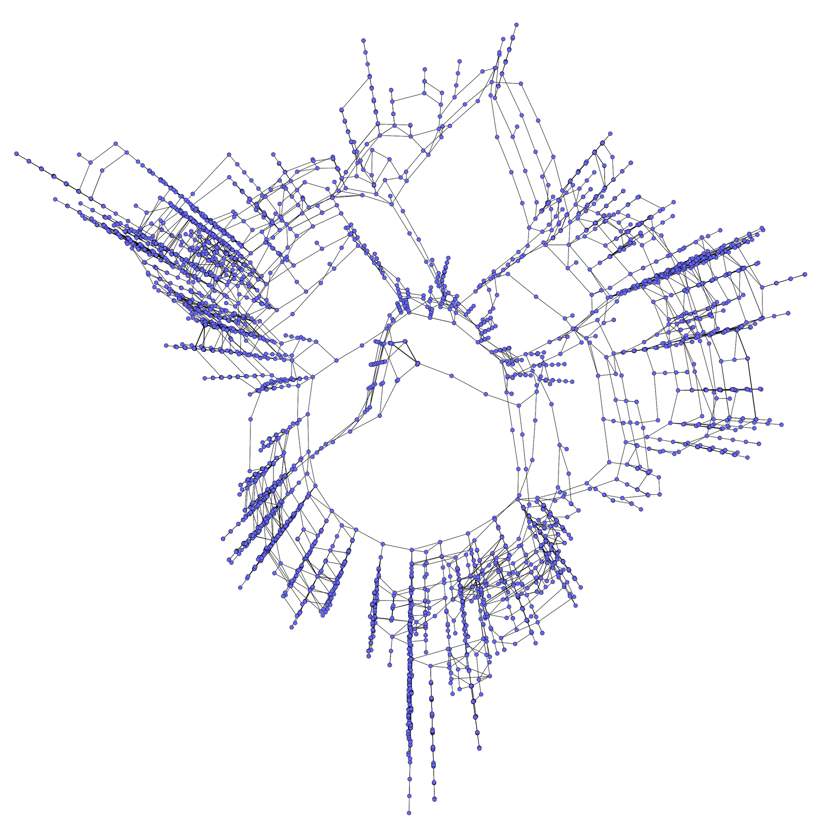
Multidimensional Scaling (MDS) is a family of techniques for analysis and visualization of complex data. Objects in a data set are represented as points in a geometric space; distance in this space represents proximity or similarity among objects. We study MDS in the context of network visualization and develop efficient algorithms that are scalable to very large instances.



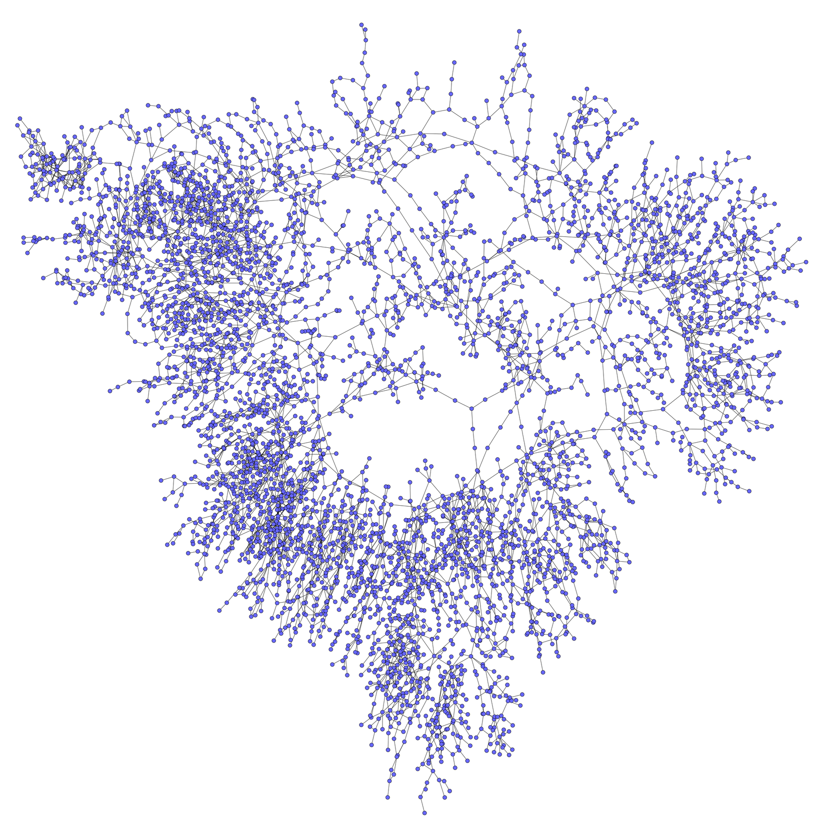
1 Foundations

There are two major branches that can be identified within the MDS family:

- *Classical* or *inner-product scaling* is the oldest approach and uses fundamental methods from linear algebra and Euclidean geometry, in particular results from spectral matrix analysis. The solutions are essentially unique and provably optimal with respect to certain criteria.
- *Distance scaling* is based on the iterative numerical optimization of an objective function associated with preliminary solutions. Thanks to its great flexibility and its wider range of potential applications, it early became the most frequently used MDS variant.



classical scaling

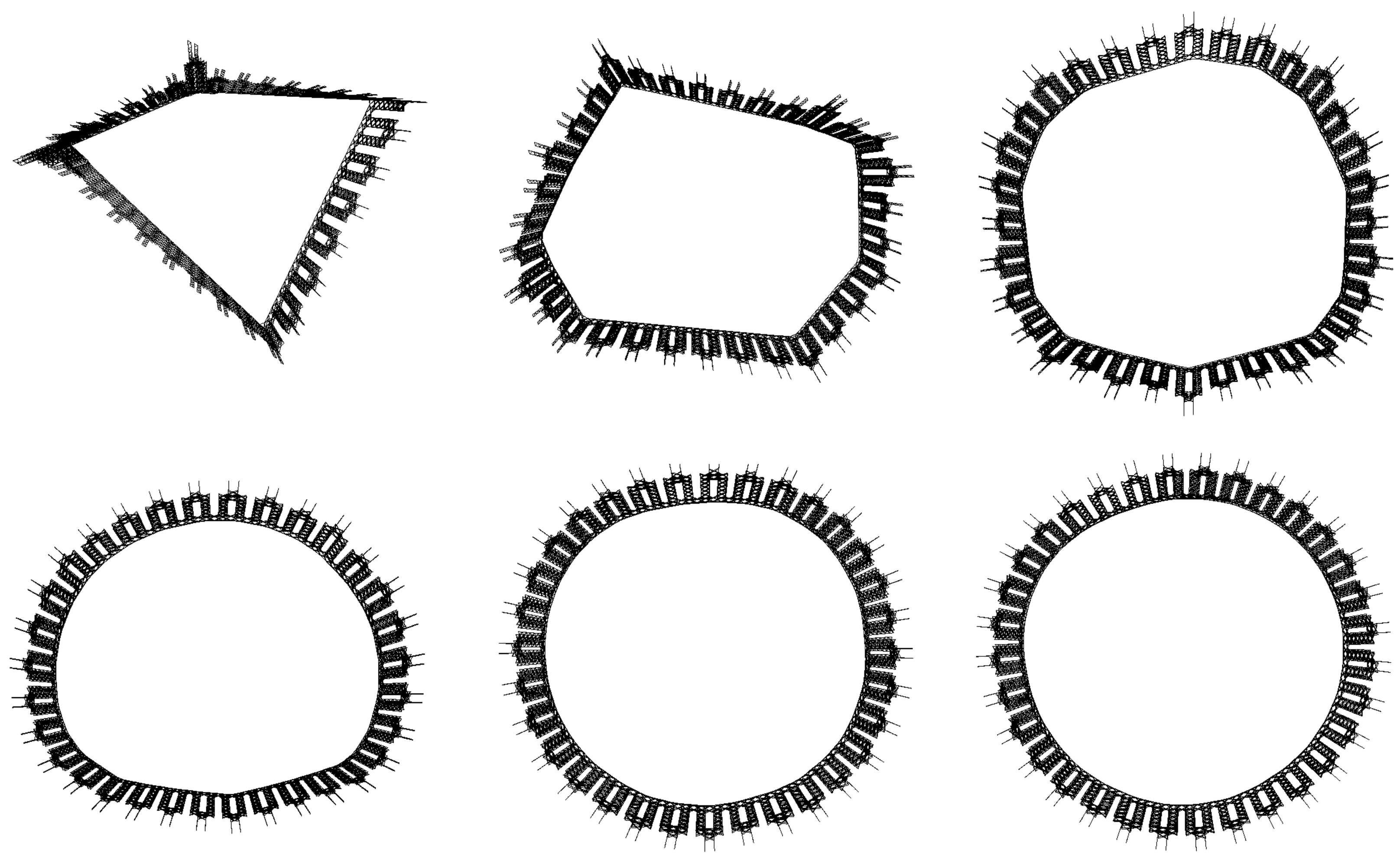


distance scaling

While the latter approach is well-established in the graph drawing literature explicitly or implicitly as force-directed drawing, the former approach has re-attracted attention in this field of research only recently.

2 Methods and Algorithms

In its basic form, MDS requires as input a set of dissimilarities between the objects to be placed. In the context of graphs, this is typically the matrix of shortest-path distances. Since time and space complexity are at least quadratic in the size of the input graph, application of MDS is limited to instances with at most a few thousands of nodes.
This restriction can be overcome by collecting and using only a selected small part of the necessary distance information. This can be achieved by sampling a set of *pivot* nodes or by approximation using the solution for a suitable subgraph. Using efficient implementation, computational effort and storage requirements can thus be brought down essentially to linearity.



Successively extending the information incorporated in the computation gives rise to *progressive* methods trading off time and quality: Preview drawings are generated quickly and can be incrementally improved and refined at the expense of running time.

3 Applications

We describe various variants and algorithmic aspects of different MDS approaches, input information, and acceleration strategies; we show that combinations can yield a powerful set of methods to efficiently generate good drawings. The power and limitations of the proposed techniques are illustrated with some experiments.
The usefulness of these methods is illustrated by various applications, such as social networks, bibliographic and citation networks, or software dependencies [1, 2, 3].

References

[1] Krists Boitmanis, Ulrik Brandes, and Christian Pich. Visualizing internet evolution on the autonomous systems level. In *Proc. Graph Drawing*, pages 265–276, 2007.

[2] Ulrik Brandes and Christian Pich. Eigensolver methods for progressive multidimensional scaling of large data. In *Proc. Graph Drawing*, pages 42–53, 2006.

[3] Christian Pich, Lev Nachmanson, and George Robertson. Visual analysis of importance and grouping in software dependency graphs. Submitted for publication.