

An Experimental Study on Distance-Based Graph Drawing (Extended Abstract)

Ulrik Brandes and Christian Pich

Department of Computer & Information Science, University of Konstanz
{Ulrik.Brandes,Christian.Pich}@uni-konstanz.de

Abstract. In numerous application areas, general undirected graphs need to be drawn, and force-directed layout appears to be the most frequent choice. We present an extensive experimental study showing that, if the goal is to represent the distances in a graph well, a combination of two simple algorithms based on variants of multidimensional scaling is to be preferred because of their efficiency, reliability, and even simplicity. We also hope that details in the design of our study help advance experimental methodology in algorithm engineering and graph drawing, independent of the case at hand.

1 Introduction

Graph drawing is concerned with the geometric representation of graphs. For general undirected graphs, force-directed and energy-based layout algorithms are commonly used, because they are often easy to implement and experience shows that they can result in undistorted and readable layouts which reveal structural features such as local clustering and symmetry [3].

Based on experimental evidence presented in this paper, we argue that approximate classical scaling with subsequent stress reduction should be used instead. The requirements leading to this argument are:

1. *quality*: pairwise distances between vertices are represented well,
2. *scalability*: the algorithm scales to very large graphs, and
3. *simplicity*: the algorithm is easy to understand and implement.

Note that the quality criterion is implicit in force-directed algorithms. Classical scaling and stress minimization are instances of the general concept of Multidimensional Scaling (MDS, see [1,8] for comprehensive references). MDS of graph-theoretic distances has been used early on for automatic layout of social networks [16], without explicit reference in the well-known algorithm of Kamada and Kawai [15], and in the wider context of data analysis (e.g., [5,10]), but the use of advanced MDS algorithms well-known in other fields has gained momentum only after Gansner, Koren, and North applied majorization to stress minimization in graph drawing [12]. Stress minimization is generally assumed to be the

method of choice for drawing general graphs, because of its intuitive and adaptable objective function and the visually pleasing layouts obtained. Yet, it is often found to be difficult to implement efficiently, and the presence of local minima is a serious concern.

Our study provides an assessment of layout quality and efficiency, and also yields a recommendation on how to implement the method to achieve reliability, efficiency, and simplicity at the same time. While a considerable number of experimental studies have been conducted to assess graph drawing criteria and algorithm performance, only two are closely related [2,13]. However, these compare implementations of suites of related algorithms which are treated as black boxes. The combination of our in-depth study with these more general comparisons provides additional support for our conclusion.

A methodological contribution of our study is the design of experiments along explicit hypotheses about the performance of algorithms. These guided our choice of experiments and structure argumentation.

The remainder of this paper is organized as follows: In Sect. 2, background on the relevant MDS variants and their application to graph drawing is given. The main hypotheses are stated in Sect. 3. The experimental setup is described in Sect. 4, and the actual experiments in Sect. 5. Section 6 discusses results with regard to our hypotheses. We conclude with a summary in Sect. 7.

2 Multidimensional Scaling

Let $V = \{1, \dots, n\}$ be the set of n objects and let $D \in \mathbb{R}^{n \times n}$ be a square matrix of dissimilarities d_{ij} for each pair of objects $i, j \in V$. MDS yields a matrix $X = [x_1, \dots, x_n]^T \in \mathbb{R}^{n \times d}$ of d -dimensional positions $x_1, \dots, x_n \in \mathbb{R}^d$ such that

$$\|x_i - x_j\| \approx d_{ij} \quad \text{for all } i, j \in V \quad (1)$$

is met as closely as possible; in our experiments, $d = 2$ throughout. We leave this somewhat informal for the moment and make it more precise in the following two subsections, where we describe the objective functions typically considered to assess compliance with (1). Straightforward implementations of these run in $\Theta(n^3)$ time, but we will discuss more efficient algorithms in Section 4.

Classical Scaling. The first approach to achieve (1) is based on linear algebra and is referred to as *classical* or *inner-product scaling*. Let $D \in \mathbb{R}^{n \times n}$ be defined as above, and let $D^{(2)}$ be matrix D with all entries squared. Classical scaling is based on a matrix $B \in \mathbb{R}^{n \times n}$ of *pseudo products* b_{ij} with

$$b_{ij} = -\frac{1}{2} \left(d_{ij}^2 - \frac{1}{n} \sum_{s=1}^n d_{is}^2 - \frac{1}{n} \sum_{r=1}^n d_{rj}^2 + \frac{1}{n^2} \sum_{r,s=1}^n d_{rs}^2 \right) \quad (2)$$

or equivalently, written in matrix form, by double-centering $D^{(2)}$ with $B = -\frac{1}{2} J_n D^{(2)} J_n$, where $J_n = I_n - \frac{1}{n} \cdot (1_n 1_n^T) \in \mathbb{R}^{n \times n}$, I_n being the identity matrix and $1_n \in \mathbb{R}^n$ the all-ones vector of length n .

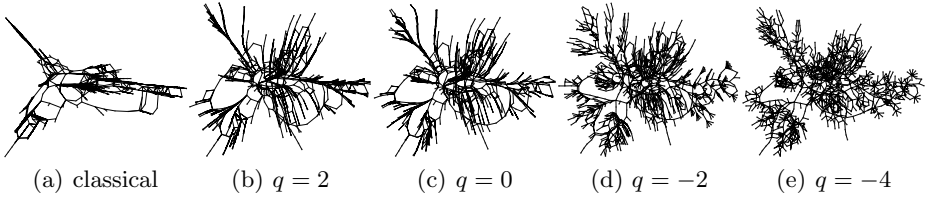


Fig. 1. Example drawings for the 1138bus graph. Drawing (a) is generated with classical scaling, drawings (b)–(e) with distance scaling and weights $w_{ij} = d_{ij}^q$.

Let $v_1 \dots, v_n \in \mathbb{R}^{n \times n}$ and $\lambda_1 \geq \dots \geq \lambda_n \in \mathbb{R}$ be the sequence of eigenvectors and corresponding eigenvalues of B . Two-dimensional coordinates are then obtained by setting the configuration matrix $X \in \mathbb{R}^{n \times 2}$ to be

$$X = \left[\sqrt{\lambda_1} v_1, \sqrt{\lambda_2} v_2 \right], \quad (3)$$

which is optimal [1] the mismatch between the pseudo inner-products derived from the d_{ij} 's in (2) and the inner products $x_i^T x_j$, namely

$$\text{strain}(X) = \|B - XX^T\|_2 = \sum_{i,j} (b_{ij} - x_i^T x_j)^2. \quad (4)$$

The advantage of this approach is that it gives analytic solutions which are essentially unique and optimal with respect to strain. A major drawback is the detour via inner products, sometimes leading to degenerate solutions.

Distance Scaling. Instead of achieving (1) by fitting inner products b_{ij} and $x_i^T x_j$, coordinates can be computed by directly fitting distances $\|x_i - x_j\|$ to dissimilarities d_{ij} . This leads to the objective function

$$\text{stress}(X) = \sum_{i,j} w_{ij} (d_{ij} - \|x_i - x_j\|)^2, \quad (5)$$

where $w_{ij} \geq 0$ weights the contribution of pair i, j ; frequently, $w_{ij} = d_{ij}^q$ for some $q \in \mathbb{R}$. Since there is no known method for directly computing a configuration X with minimal stress, the standard approach is iterative numerical optimization.

Graph Drawing and MDS. Most applications of MDS to graph drawing set the desired distances to be the shortest-path distances in the graph, which often spread nodes well over the drawing and display symmetries and clusterings.

While classical scaling was used for graph drawing [5] and made scalable to large graphs only recently [4,6], the distance scaling approach is pioneered much earlier [16]. Kamada and Kawai [15] used a layout energy equivalent to the objective function introduced independently by McGee [19] more than twenty years earlier (there termed *work*). In the framework of the more general weighted MDS, it corresponds to setting $w_{ij} = d_{ij}^{-2}$ in Eq. (5). Other weighting schemes and dissimilarities are discussed in [5,7]. Fig. 1 shows some example drawings.

3 Hypotheses

A combination of theoretical properties, previous experience, popular beliefs, and preliminary tests, led us to formulate and test the hypotheses below. These shall not be read as if they were results, but serve to focus attention and are formulated in such a way that they can be tested with algorithmic experiments. We therefore conducted a series of experiments described in the next section. See Section 6 for a discussion of the results.

The first hypothesis basically rules out force-directed methods.

Hypothesis 1. *For graph drawing representing graph-theoretic distances it is most appropriate to model this representation explicitly in the objective function.*

Given their objectives, both classical and distance scaling should represent graph-theoretic distances well in a geometric layout, and thus be useful for graph drawing. Because of the more direct influence on the objective function and a concave weighting of distance representation errors, it seems plausible that distance scaling would be the more suitable variant for graph drawing. While it is almost commonplace that classical scaling is better at representing global structure whereas distance scaling is better at representing fine details [5], we do not know of any systematic evaluation. We therefore provide experimental evidence for the following.

Hypothesis 2. *Distance scaling compares favorably with classical scaling in terms of layout quality, because local details are represented better.*

In our experience, based on many conversations with implementors and users of graph drawing systems, a main reservation against distance scaling is its assumed non-scalability, due to a multitude of local minima and high computational demand. The next two hypotheses focus on how to ensure that the layouts produced by implementations of distance scaling are actually those supporting H1.

Hypothesis 3. *Distance scaling is susceptible to poor local minima, because it is highly dependent on the initial layout.*





















Hypothesis 4. *Classical scaling provides excellent initial layouts for distance scaling, because the better representation of large distances helps to avoid poor local minima.*

If H4 holds, we have complicated matters even more, because two demanding problems have to be solved rather than one. The final two hypotheses therefore regard the possibility of computing the initial and final layout efficiently.

Hypothesis 5. *Classical scaling layouts of very large graphs can be approximated efficiently using PivotMDS.*

Hypothesis 6. *Distance scaling is practical even on very large graphs.*

Table 1. Test set of graphs used in Experiments 1–3. n, m, D denote the number of nodes, the number of edges, and the diameter, respectively. The two rightmost columns contain plots for distance distributions and the 10 largest eigenvalues of B .

name	n	m	description	$D\{d_{ij}\}$	$\lambda_{1,\dots,10}$
516	516	729	finite element mesh describing adjacencies between faces in a triangulation	61 	
1138bus	1138	1458	network of high-voltage power distribution in the United States.	31 	
qh882	882	2856	matrix derived from Quebec hydroelectric power system's small signal model	31 	
plat1919	1919	15240	finite-difference model of shallow wave equations in Atlantic/Indian Ocean	43 	
esslingen1	2075	4769	social network in the city of Esslingen in the 19th century	15 	
sw0	500	1500	circle in which each node is adjacent to its 3 left and right neighbors	84 	
sw002	500	1500	graph sw0, each edge redirected randomly with probability 0.02	27 	
sw01	500	1500	graph sw0, each edge redirected randomly with probability 0.1	10 	
btrees	1023	1022	complete binary tree of height 10	18 	
prot1	3025	3629	largest component of protein interaction network	27 	

4 Experimental Design

Data. The experiments were run on a set of test graphs described in Table 1. The graphs were selected large enough to allow for extrapolation of the results to very large graphs, but also small enough to allow for, the exact computation of stress as given by (5) in a large number of experiments.

Note that the eigenvalues of the matrices B associated with each graph indicate the intrinsic dimensionality of the original distances d_{ij} . If, say, two dimensions suffice to reconstruct all the d_{ij} 's exactly, such that the strain criterion is zero, then $\lambda_1 \geq \lambda_2 > \lambda_3 = \dots = \lambda_n = 0$, and inversely, few large and many (near-) zero eigenvalues indicate the existence of a good low-dimensional layout.

Environment. We implemented all MDS algorithms and speed-up techniques ourselves to avoid bias due to coding, system, or timing. The algorithms were implemented in Java using Sun's SDK 1.6.0 and the yFiles 2.5.0.1 graph library (www.yworks.com). All experiments were run on a standard 1.4 GHz Compaq NX 7000 notebook with 512 MB of RAM, using Windows XP Service Pack 2.

Implementation. A simple and convenient way of implementing classical scaling is by constructing matrix B in (2) and computing its two extremal eigenvalues λ_1, λ_2 and eigenvectors v_1, v_2 by power iteration.

The problem of drawing graphs with fixed edge lengths is \mathcal{NP} -hard in general [9], and for distance scaling no analytic solution is known, so layouts have to

be computed iteratively. In Kruskal’s original proposal [17], stress is evaluated for the current positions, and new positions are computed by gradient descent; this is also done in [15,19,20] with gradient terms specific to the weights w_{ij} . These approaches were superseded by majorization [18], which generates a sequence of layouts with decreasing stress and can handle arbitrary weights $w_{ij} \geq 0$. In our experiments we use a “local iteration” with node-by-node updates [12].

5 Experiments

The first experiments is to provides evidence for which method yields better layouts in principle (disregarding efficiency, ease of implementation, reliability, etc.), when graph-theoretic distances are to be represented by Euclidean distances. We use the following shorthand notation for the involved approaches:

- **random**: node coordinates drawn uniformly at random from $(0,1)$,
- **fm3**: fast multipole multilevel method [13],
- **grip**: multilevel force-directed layout method [11],
- **hde**: high dimensional embedder [14] (50 pivots),
- **cmds** classical scaling.

Experiment 1 (Layout approach). *All test graphs are laid out with cmds, distance scaling with unweighted and weighted stress, fm3, hde, and grip.*

For convenience, most implementations of iterative layout algorithms start from a random initial configuration. It is, however, widely known that smart initialization is preferable. We here compare different initialization strategies for distance scaling and evaluate the resulting stress. Before the iteration all initial solutions X are scaled such that $\sum_{i,j} \|x_i - x_j\| = \sum_{i,j} d_{ij}$.

Experiment 2 (Distance scaling and initialization). *All test graphs are laid out using each of the following layout algorithms: random, fm3, hde, grip, cmds, and then minimizing weighted stress using local iteration.*

Classical scaling has running time at least quadratic in the number of nodes n for constructing distance matrix $D \in \mathbb{R}^{n \times n}$ and decomposing the derived matrix $B \in \mathbb{R}^{n \times n}$. Quick estimates for the eigenvectors v_1, v_2 corresponding to λ_1, λ_2 are obtained by using only parts of D by selecting a subset $W \subset V$ of $k \ll n$ *pivot* or *landmark* nodes and taking only $k \cdot n$ rather than n^2 distances into account. Once W is constructed, two approaches for this are considered:

- Pivot MDS [4] uses the singular value decomposition of a rectangular matrix: Let $D_k \in \mathbb{R}^{n \times k}$ be the matrix of k columns of distances from nodes in W , e.g. in k breadth-first searches. Then the right singular vectors u_1, u_2 of $C = -\frac{1}{2}J_n D_k^{(2)} J_k$ are estimates for the eigenvectors v_1, v_2 of $B = -\frac{1}{2}J_n D^{(2)} J_n$.
- Landmark MDS [22] places nodes in W by classical MDS. The each node in $V \setminus W$ is placed based on its k distances to nodes in W .

The k pivots should be well-scattered over the graph; intuitively, this is to represent as much of the full distance information D as possible. Assuming that W contains $k - 1$ selected nodes, our strategies to determine the k -th pivot are

- **maxmin**: $\operatorname{argmax}_{i \in V \setminus W} \min_{j \in W} d_{ij}$, the node farthest from W ;
- **random**: with uniform probability, from W ;
- **mixed**: with **maxmin**, if k is even, with **random** otherwise;

combining them with the two estimation approaches above leads to six strategies.

Let $X, Y \in \mathbb{R}^{n \times 2}$ be the estimate and the actual solution, each centered at the origin. To find out how similar X is to Y we use the *Procrustes statistic*

$$R^2 = 1 - \left(\operatorname{tr}(X^T Y Y^T X)^{1/2} \right)^2 / \left(\operatorname{tr}(X^T X) \cdot \operatorname{tr}(Y^T Y) \right) \quad (6)$$

minimized by the *Procrustes rotation* $P \in \mathbb{R}^{2 \times 2}$ (see [21] for its formula) which, applied to each row in X , optimally dilates, scales, rotates, and reflects X to fit Y . It can be shown that $0 \leq R^2 \leq 1$; if $R^2 = 0$, X and Y can be perfectly matched, if $R^2 = 1$, they cannot be matched by any $P \in \mathbb{R}^{2 \times 2}$ at all.

Experiment 3 (Approximating classical scaling). *For each test graph, classical scaling is approximated using 6 strategies $\{\text{maxmin}, \text{random}, \text{mixed}\} \times \{\text{landmark}, \text{pivot}\}$, and compared to the exact solutions using the Procrustes statistic.*

Experiments 2 and 3 were repeated 25 times, and to control for biases due to the internal representation of graphs and matrices, we used as many instances of each graph, each with randomly permuted vertices and edges.

Distance scaling by stress minimization is mostly used for improving the representation of local details; setting $w_{ij} = d_{ij}^{-2}$ assigns large weight to the representation of small distances and vice versa. Initializing distance scaling with **cmds**, we hope that large distances are fitted well; the subsequent fitting of smaller distances and local details is achieved by discarding the large distances from the stress term to be minimized, which we dub *sparse stress*

$$\operatorname{stress}(X) = \sum_{\{i,j\} \in S} w_{ij} (d_{ij} - \|x_i - x_j\|)^2, \quad (7)$$

where $S \subseteq V \times V$ is a set of node pairs involved in the iteration, with $|S| \in \mathcal{O}(n)$. In our experiments we use *local neighborhoods* obtained by terminating the breadth-first searches after k neighbors have been found.

Experiment 4 (Sparse stress minimization). *For each of the test graphs the initial classical scaling configuration is subjected to sparse stress minimization using only local neighborhoods.*

We use another collection of larger graphs to examine the scalability of initialization and sparse stress minimization. Unlike the test graphs used earlier, their size prohibits methods using the full square matrices. The results are assessed visually with respect to the information known a priori.

Experiment 5 (Very large graphs). *Large graphs are laid out first using an approximation to classical scaling and then sparse stress minimization.*

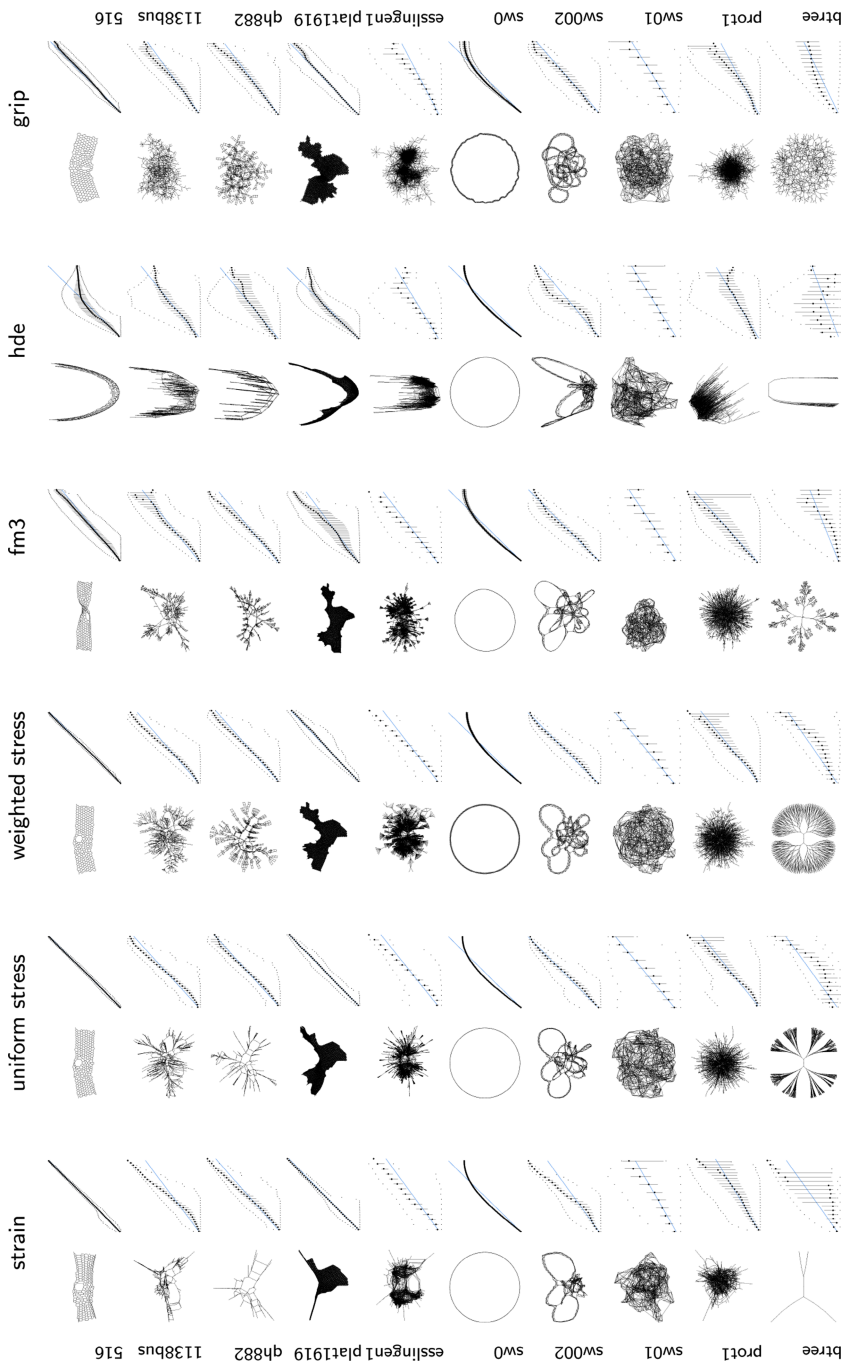


Fig. 2. Drawings the test graphs, and quartile plots of d_{ij} (abscissa) vs. $\|x_i - x_j\|$. Large dots indicate the median, small dots minimum and maximum, and black lines the range of the two middle quartiles (25–75 per cent). The thin blue line with slope 1 is a visual aid.

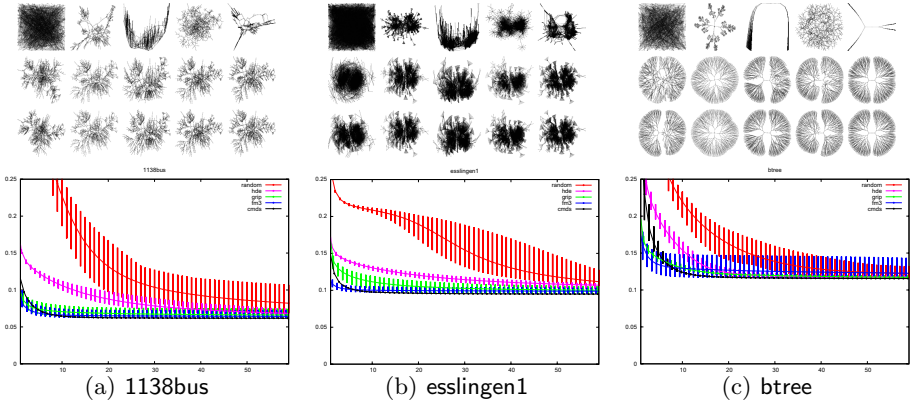


Fig. 3. Upper row: The majorization process with different initializations *random*, *fm3*, *hde*, *grip*, *cmds* after 0, 30, 60 iterations. Lower row: Number of iterations vs. stress. The bars indicate the range of values, the dots the median value, in 25 runs.

6 Results

Layout Quality. To assess layout quality both visually and quantitatively, aligned layouts and the distributions of layout distances are shown in Fig. 2 for each of the possible distance values between pairs of vertices, i.e. for values ranging from 1 to the diameter of the respective graph. The classical scaling layouts were generated with random initial positions and used as initial configurations for distance scaling. Initialization is further studied in Exp. 2.

The drawings for graphs *qh882*, *1138bus* seem to confirm *H1* and *H2*; using weights $w_{ij} = d_{ij}^{-2}$ helps to display local structures hidden by classical scaling or unweighted distance scaling. For regular structures *516*, *plat1919*, *sw0*, distance scaling does not improve the quality of local representation. In a few cases classical scaling represents the overall structure better, such as the known clustering of *esslingen1* into two densely connected parts.

In general, *H1* and *H2* can be accepted at least for graphs for which graph-theoretic distance is well representable in low dimensions. However, none of the MDS variants seems to be capable of representing both smaller and larger distances for small diameter graphs and other special types of graphs like *btree*. In such cases the MDS objective functions for distance representations is not always useful as an aesthetic criterion; see Section 7 for a discussion.

Initialization. For independence of graph size and distances we divide the stress by $\sum_{i,j} w_{ij} d_{ij}^2$, which allows for comparison between stress computations even for different graphs. We have carried out the iterative majorization process 25 times for each graph (with permuted edge list) and for each of the five initial placements.

The results of Exp. 2 are displayed in Fig. 3, which shows stress values over the majorization process for distance scaling, with weights $w_{ij} = d_{ij}^{-2}$. For almost

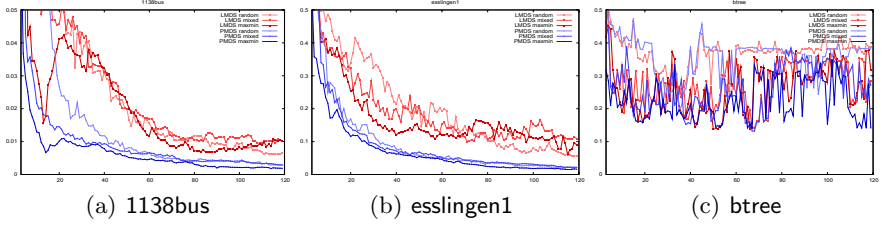


Fig. 4. Procrustes statistics measuring how well Pivot MDS (red) or Landmark MDS (blue) estimate the exact solution of classical scaling. Plotted are the median values of 25 runs with different node permutations, for $k \in \{3, \dots, 120\}$ pivots.

all graphs we have tested, basically the same ranking resulted, with *random* being worst, followed by *fm3*, *grip*, *hde*. Initially, *cmds* solutions tend to have higher values, but overtakes the other initializations after some iterations.

All experiments indicate that *H3 is valid for all types of graphs*. Since large distances and thus global structures are represented well, classical scaling gives excellent initial configurations for distance scaling.

The bandwidth of stress values we observed for *cmds*-initialized layouts was almost always negligible, whereas stress values vary largely for all other methods in the 25 runs. Classical scaling gives reproducible initial configurations throughout, which are also robust against permutation of the input. All these observations *support H4*. Interestingly, *btree* is the only graph for which classical scaling resulted in some variation; we attribute this to the multiple occurrence of equal eigenvalues of matrix B (see Table 1).

Scalability. We computed estimates for the solution to classical scaling for all graphs, again in 25 runs with random node permutations. In each run, three sets of pivots were grown from $k = 3$ to 120 (following *maxmin*, *random*, and *mixed*) and used for Pivot MDS and Landmark MDS. The plots for the median values of three selected graphs are shown in Fig. 4.

For regular graphs like *sw0*, *516*, the pivoting strategy is not crucial. In all other cases Pivot MDS is superior to Landmark MDS, regardless of the pivoting strategy. For Pivot MDS, the *maxmin* strategy performs better than *random* and slightly better than *mixed*. The corresponding plots seem to converge to zero faster and more smoothly than those for Landmark MDS. Once again, graph *btree* seems to be different from the others; estimating the full classical scaling solution appears to be unstable, no matter what pivoting strategy is used. Our observations indicate that *H5 is valid*.

We have conducted further experiments considering scalability, but omit them here due to space restrictions. One suite of experiments applies Pivot MDS to graphs with millions of nodes; we have observed that even those huge graphs, for which the full classical scaling is impractical, are laid out well with it, provided that two dimensions suffice, and, conversely, that increasing the number of pivots

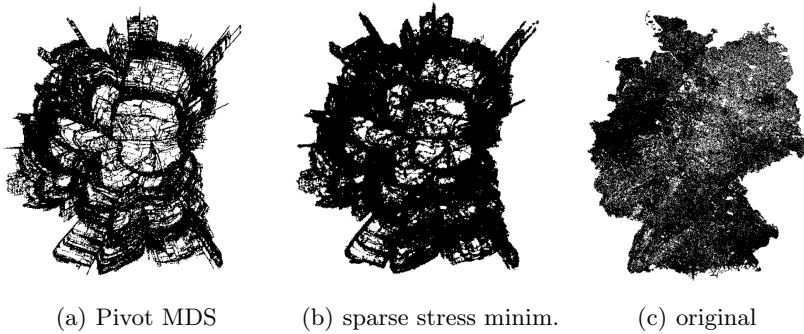


Fig. 5. Drawings for a large graph representing the street network in Germany (4 044 153 nodes, 9 564 235 edges, diameter 1 059)

does not improve layout quality if the graph is of higher intrinsic dimensionality; see also Sect. 4.

Another suite of experiments indicates that, technically, stress minimization scales even to very large graphs, but that *H6 is valid only with the limitation* that an appropriate sparsification scheme must be available.

7 Conclusion

We have studied different graph-layout approaches that aim at representing graph-theoretic distances by Euclidean distances. Our experiments suggest that minimization of weighted stress, an objective function that models the desired aesthetic properties explicitly, is to be preferred over force-directed placement. The recommended method for weighted stress minimization is to initialize with a fast approximation of classical scaling [4] and subsequent iterative improvement using localized stress reduction [12]. Both phases are easy to implement, but the second can be time-consuming. Approximation via sparse stress makes the algorithm scale to very large graphs, but further research on reliable sparsification schemes is needed.

The distance-based approach yields poor results on certain classes of graphs, which include small worlds and other graphs with many shortcuts or low diameter, and scale-free graphs with highly skewed degree distributions, large 1-shells, or other forms of structural imbalance. Some success has been obtained with stress weighting schemes based on graph invariants, but good characterizations of problematic graphs are missing and matching layout algorithms need to be developed further.

Using a hypotheses-based experimental design, we hope to foster clarity and reproducibility of our results, and to contribute to experimental evaluation of graph drawing algorithms in general.

References

1. Borg, I., Groenen, P.J.F.: *Modern Multidimensional Scaling*, 2nd edn. Springer, Heidelberg (2005)
2. Brandenburg, F.-J., Himsolt, M., Rohrer, C.: An experimental comparison of force-directed and randomized graph drawing algorithms. In: Brandenburg, F.J. (ed.) *GD 1995. LNCS*, vol. 1027, pp. 76–87. Springer, Heidelberg (1996)
3. Brandes, U.: Drawing on physical analogies. In: Kaufmann, M., Wagner, D. (eds.) *Drawing Graphs. LNCS*, vol. 2025, pp. 71–86. Springer, Heidelberg (2001)
4. Brandes, U., Pich, C.: Eigensolver methods for progressive multidimensional scaling of large data. In: Kaufmann, M., Wagner, D. (eds.) *GD 2006. LNCS*, vol. 4372, pp. 42–53. Springer, Heidelberg (2007)
5. Buja, A., Swayne, D.F.: Visualization methodology for multidimensional scaling. *Journal of Classification* 19, 7–43 (2002)
6. Civril, A., Magdon-Ismaïl, M., Bocek-Rivele, E.: SSDE: Fast graph drawing using sampled spectral distance embedding. In: Kaufmann, M., Wagner, D. (eds.) *GD 2006. LNCS*, vol. 4372, pp. 30–41. Springer, Heidelberg (2007)
7. Cohen, J.D.: Drawing graphs to convey proximity. *ACM Transactions on Computer-Human Interaction* 4(3), 197–229 (1997)
8. Cox, T.F., Cox, M.A.A.: *Multidimensional Scaling*, 2nd edn. CRC/Chapman and Hall, Boca Raton (2001)
9. Eades, P., Wormald, N.C.: Fixed edge-length graph drawing is NP-hard. *Discrete Applied Mathematics* 28(2), 111–134 (1990)
10. Freeman, L.C.: Graph layout techniques and multidimensional analysis. *Journal of Social Structure* 1 (2000)
11. Gajer, P., Kobourov, S.: GRIP – Graph drawing with intelligent placement. In: Marks, J. (ed.) *GD 2000. LNCS*, vol. 1984, pp. 222–228. Springer, Heidelberg (2001)
12. Gansner, E.R., Koren, Y., North, S.C.: Graph drawing by stress majorization. In: Pach, J. (ed.) *GD 2004. LNCS*, vol. 3383, pp. 239–250. Springer, Heidelberg (2005)
13. Hachul, S., Jünger, M.: An experimental comparison of fast algorithms for drawing general large graphs. In: Healy, P., Nikolov, N.S. (eds.) *GD 2005. LNCS*, vol. 3843, pp. 235–250. Springer, Heidelberg (2006)
14. Harel, D., Koren, Y.: Graph drawing by high-dimensional embedding. In: Goodrich, M.T., Kobourov, S.G. (eds.) *GD 2002. LNCS*, vol. 2528, pp. 207–219. Springer, Heidelberg (2002)
15. Kamada, T., Kawai, S.: An algorithm for drawing general undirected graphs. *Information Processing Letters* 31, 7–15 (1989)
16. Kruskal, J.B., Seery, J.B.: Designing network diagrams. In: *Proc. First General Conference on Social Graphics*, pp. 22–50 (1980)
17. Kruskal, J.B.: Multidimensional scaling by optimizing goodness of fit to a non-metric hypothesis. *Psychometrika* 29(1), 1–27 (1964)
18. de Leeuw, J.: Applications of convex analysis to multidimensional scaling. In: Barra, J.R., Brodeau, F., Romier, G., van Cutsem, B. (eds.) *Recent Developments in Statistics*, pp. 133–145. North-Holland, Amsterdam (1977)
19. McGee, V.E.: The multidimensional scaling of “elastic” distances. *Br. J. Math. Stat. Psychol.* 19, 181–196 (1966)
20. Sammon, J.W.: A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers* 18(5), 401–409 (1969)
21. Sibson, R.: Studies in the robustness of multidimensional scaling: Procrustes statistics. *J. R. Stat. Soc.* 40(2), 234–238 (1978)
22. de Silva, V., Tenenbaum, J.B.: Sparse multidimensional scaling using landmark points. Tech. rep., Stanford University (2004)