

A Sparse Stress Model

Mark Ortmann^(✉), Mirza Klimenta, and Ulrik Brandes

Computer and Information Science, University of Konstanz, Konstanz, Germany

`Mark.Ortmann@uni-konstanz.de`

Abstract. Force-directed layout methods constitute the most common approach to draw general graphs. Among them, stress minimization produces layouts of comparatively high quality but also imposes comparatively high computational demands. We propose a speed-up method based on the aggregation of terms in the objective function. It is akin to aggregate repulsion from far-away nodes during spring embedding but transfers the idea from the layout space into a preprocessing phase. An initial experimental study informs a method to select representatives, and subsequent more extensive experiments indicate that our method yields better approximations of minimum-stress layouts in less time than related methods.

1 Introduction

There are two main variants of force-directed layout methods, expressed either in terms of forces to balance or an energy function to minimize [3, 25]. For convenience, we refer to the former as spring embedders and to the latter as multi-dimensional scaling (MDS) methods.

Force-directed layout methods are in wide-spread use and of high practical significance, but their scalability is a recurring issue. Besides investigations into adaptation, robustness, and flexibility, much research has therefore been devoted to speed-up methods [20]. These efforts address, e.g., the speed of convergence [10, 11] or the time per iteration [1, 17]. Generally speaking, the most scalable methods are based on multi-level techniques [13, 18, 21, 35].

Experiments [5] suggest that minimization of the stress function [27]

$$s(x) = \sum_{i < j} w_{ij} (||x_i - x_j|| - d_{ij})^2 \quad (1)$$

is the primary candidate for high-quality force-directed layouts $x \in (\mathbb{R}^2)^V$ of a simple undirected graph $G = (V, E)$ with $V = \{1, \dots, n\}$ and $m = |E|$. The target distances d_{ij} are usually chosen to be the graph-theoretic distances, the weights set to $w_{ij} = 1/d_{ij}^2$, and the dominant method for minimization is majorization [16]. Several variant methods reduce the cost of evaluating the stress

We gratefully acknowledge financial support from Deutsche Forschungsgemeinschaft under grant Br 2158/11-1.

function by involving only a subset of node pairs over the course of the algorithm [6, 7, 13]. If long distances are represented well already, for instance because of initialization with a fast companion algorithm, it has been suggested that one restrict further attention to short-range influences from k -neighborhoods only [5].

We here propose to stabilize the sparse stress function restricted to 1-neighborhoods [5] with aggregated long-range influences inspired by the use of Barnes & Hut approximation [1] in spring embedders [33]. Extensive experiments suggest how to determine representatives for individually weak influences, and that the resulting method represents a favorable compromise between efficiency and quality.

Related work is discussed in more detail in the next section. Our approach is derived in Sect. 3, and evaluated in Sect. 4. We conclude in Sect. 5.

2 Related Work

While we are interested in approximating the full stress model of Eq. (1), there are other approaches capable of dealing with given target distances such as the strain model [4, 24, 32] or the Laplacian [19, 26].

An early attempt to make the full stress model scale to large graphs is GRIP [13]. Via a greedy maximal independent node set filtration, this multi-level approach constructs a hierarchy of more and more coarse graphs. While a sparse stress model calculates the layout of the coarsened levels, the finest level is drawn by a localized spring-embedder [11]. Given the coarsening hierarchy for graphs of bounded degree, GRIP requires $\mathcal{O}(nk^2)$ time and $\mathcal{O}(nk)$ space with $k = \log \max\{d_{ij} : i, j \in V\}$.

Another notable attempt has been made by Gansner et al. [15]. Like the spring embedder the maxent-model is split into two terms:

$$\sum_{\{i,j\} \in E} w_{ij} (||x_i - x_j|| - d_{ij})^2 - \alpha \sum_{\{i,j\} \notin E} \log ||x_i - x_j||$$

The first part is the 1-stress model [4, 13], while the second term tries to maximize the entropy. Applying Barnes & Hut approximation technique [1], the running time of the maxent-model can be reduced from $\mathcal{O}(n^2)$ per iteration to $\mathcal{O}(m + n \log n)$, e.g., using quad-trees [30, 34]. In order to make the maxent-model even more scalable Meyerhenke et al. [28] embed it into a multi-level framework, where the coarsening hierarchy is constructed using an adapted size-constrained label propagation algorithm.

Gansner et al. [14], inspired by the idea of decomposing the stress model into two parts, proposed COAST. The main difference between COAST and maxent is that it adds a square to the two terms in the 1-stress part and that the second term is quadratic instead of logarithmic. Transforming the energy system of COAST allows one to apply fast-convex optimization techniques making its running time comparable to the maxent model.

While all these approaches somewhat steer away from the stress model, MARS [23] tries to approximate the solution of the full stress model. Building

on a result of Drineas et al. [9], MARS requires only $k \ll n$ instead of n single-source shortest path computations. Reconstructing the distance matrix from two smaller matrices and by setting $w_{ij} = 1/d_{ij}$, MARS runs in $\mathcal{O}(kn + n \log n + m)$ per iteration with a preprocessing time in $\mathcal{O}(k^3 + k(m + n \log n + k^2 n))$, and a space requirement in $\mathcal{O}(nk)$.

3 Sparse Stress Model

The full stress model, Eq. (1), is in our opinion the best choice to draw general graphs, not least because of its very natural definition. However, its $\mathcal{O}(n^2)$ running time per iteration and space requirement, and expensive processing time of $\mathcal{O}(n(m + n \log n))$, hamper its way into practice.

The reason sparse stress models are still in early stages of development is that the adaption to large graphs requires not just a reduction in the running time per iteration, but also the preprocessing time and its associated space requirement. Where these problems originate from is best explained by rewriting Eq. (1) to the following form:

$$s(x) = \sum_{\{i,j\} \in E} w_{ij}(\|x_i - x_j\| - d_{ij})^2 + \sum_{\{i,j\} \in \binom{V}{2} \setminus E} w_{ij}(\|x_i - x_j\| - d_{ij})^2 \quad (2)$$

As minimizing the first term only requires $\mathcal{O}(m)$ computations and all d_{ij} are part of the input, solving this part of the stress model can be done efficiently. Yet, the second term requires an all-pairs shortest path computation (APSP), $\mathcal{O}(n^2)$ time per iteration, and in order to stay within this bound $\mathcal{O}(n^2)$ additional space. We note that the 1-stress approaches presented in Sect. 2 of Gajer et al. [13] and Brandes and Pich [4] ignore the second term, while Gansner et al. [14, 15] replace it. Discounting the problems arising from the APSP computation, we can see that the spring embedder suffered from exactly the same problem, namely the computation of the second term – there called repulsive forces. Barnes & Hut introduced a simple, yet ingenious and efficient solution, namely to approximate the second term by using only a subset of its addends.

To approximate the repulsive forces operating on node i Barnes & Hut partition the graph. Associated with each of these $\mathcal{O}(\log n)$ partitions is an artificial representative, a so called super-node, used to approximate the repulsive forces of the nodes in its partition affecting i . However, as these super-nodes have only positions in the euclidean space, but no graph-theoretic distance to any node in the graph they cannot be processed in the stress model. Furthermore, deriving a distance for a super-node as a function of the graph-theoretic distance of the nodes it represents would require an APSP computation, which is too costly, and since the partitioning is computed in the layout space, probably not a good approximation. Choosing a node from the partition as a super-node would not solve the problems, not least because the partitioning changes over time.

Therefore, adapting this approach cannot be done in a straightforward manner. However, the model we are proposing sticks to its main ideas. In order to

reduce the complexity of the second term in Eq. (2), we restrict the stress computation of each $i \in V$ to a subset $\mathcal{P} \subseteq V$ of $k = |\mathcal{P}|$ representatives, from now on called pivots. The resulting sparse stress model, where $N(i)$ are the neighbors of i and w'_{ip} are adapted weights, has the following form:

$$s'(x) = \sum_{\{i,j\} \in E} w_{ij} (\|x_i - x_j\| - d_{ij})^2 + \sum_{i \in V} \sum_{p \in \mathcal{P} \setminus N(i)} w'_{ip} (\|x_i - x_p\| - d_{ip})^2 \quad (3)$$

Note that GLINT [22] uses a similar function, yet the pivots change in each iteration, no weights are involved, and it is assumed that d_{ip} is accessible in constant time.

Just like Barnes & Hut, we associate with each pivot $p \in \mathcal{P}$ a set of nodes $\mathcal{R}(p) \subseteq V$, where $p \in \mathcal{R}(p)$, $\bigcup_{p \in \mathcal{P}} \mathcal{R}(p) = V$, and $\mathcal{R}(p) \cap \mathcal{R}(p') = \emptyset$ for $p, p' \in \mathcal{P}$. However, we propose to use only one global partitioning of the graph that does not change over time. Still, just like the super-nodes, we want that the pivots are representative for their associated region. In terms of the localized stress minimization algorithm [16] this means we want that for each $i \in V$ and $p \in \mathcal{P}$

$$\frac{\sum_{j \in \mathcal{R}(p)} w_{ij} (x_j^\alpha + d_{ij} (x_i^\alpha - x_j^\alpha) / \|x_i - x_j\|)}{\sum_{j \in \mathcal{R}(p)} w_{ij}} \approx x_p^\alpha + \frac{d_{ip} (x_i^\alpha - x_p^\alpha)}{\|x_i - x_p\|},$$

where α is the dimension. As the left part is the weighted average of all positional votes of $j \in \mathcal{R}(p)$ for the new position of i , we require p to fulfill the following requirements in order to be a good representative:

- The graph-theoretic distances to i from all $j \in \mathcal{R}(p)$ should be similar to d_{ip}
- The positions of $j \in \mathcal{R}(p)$ in x should be well distributed in close proximity around p .

We propose to construct the partitioning induced by \mathcal{R} only based on the graph structure, not on the layout space, and associate each node $v \in V$ with $\mathcal{R}(p)$ of the closest pivot subject to their graph-theoretic distance. As our algorithm incrementally constructs \mathcal{R} , ties are broken by favoring the currently smallest partition. Given the case that \mathcal{P} has been chosen properly and since all nodes in $\mathcal{R}(p)$ are at least as close to p as to any other pivot, and consequently in the stress drawing, it is appropriate to assume that both conditions are met.

Even if the positional vote of each pivot is optimal w.r.t. $\mathcal{R}(p)$, it is still not enough to approximate the full stress model. In the full stress model the iterative algorithm to minimize the stress moves one node at a time while fixing the rest. By setting node i 's position in dimension α to

$$x_i^\alpha = \frac{\sum_{j \neq i} w_{ij} (x_j^\alpha + d_{ij} (x_i^\alpha - x_j^\alpha) / \|x_i - x_j\|)}{\sum_{j \neq i} w_{ij}},$$

it can be shown that the stress monotonically decreases [16]. However, in our model we move node i according to

$$x_i^\alpha = \frac{\sum_{j \in N(i)} w_{ij} \left(x_j^\alpha + \frac{d_{ij} (x_i^\alpha - x_j^\alpha)}{\|x_i - x_j\|} \right) + \sum_{p \in \mathcal{P} \setminus N(i)} w'_{ip} \left(x_p^\alpha + \frac{d_{ip} (x_i^\alpha - x_p^\alpha)}{\|x_i - x_p\|} \right)}{\sum_{j \in N(i)} w_{ij} + \sum_{p \in \mathcal{P} \setminus N(i)} w'_{ip}}. \quad (4)$$

Algorithm 1. Sparse Stress**Input:** Graph $G = (V, E)$ with $w : E \rightarrow \mathbb{R}_{>0}$, and k number of pivots.**Output:** α -dimensional layout $x \in (\mathbb{R}^\alpha)^V$

```

1 sample  $\mathcal{P}$  with  $|\mathcal{P}| = k$ 
2 calculate  $\mathcal{R}$ , all adapted weights  $w'_{ip}$ , and all  $d_{ip}$  via weighted MSSP
3  $x \leftarrow \text{PivotMDS}(G)$  [4]
4 rescale  $x$  such that  $\sum_{\{i,j\} \in E} \|x_i - x_j\| = \sum_{\{i,j\} \in E} w_{ij}$ 
5 while relative positional change  $> 10^{-4}$  do
6   foreach  $i \in V$  do
7     foreach dimension  $\alpha$  do
8        $t^\alpha \leftarrow \frac{\sum_{j \in N(i)} w_{ij} \left( x_j^\alpha + \frac{d_{ij}(x_i^\alpha - x_j^\alpha)}{\|x_i - x_j\|} \right) + \sum_{p \in \mathcal{P} \setminus N(i)} w'_{ip} \left( x_p^\alpha + \frac{d_{ip}(x_i^\alpha - x_p^\alpha)}{\|x_i - x_p\|} \right)}{\sum_{j \in N(i)} w_{ij} + \sum_{p \in \mathcal{P} \setminus N(i)} w'_{ip}}$ 
9      $x_i \leftarrow t$ 

```

This implies that in order to find the globally optimal position of i we furthermore have to find weights w'_{ip} , such that $\frac{w'_{ip}}{\sum_{j \in N(i)} w_{ij} + \sum_{p \in \mathcal{P} \setminus N(i)} w'_{ip}} \approx \frac{\sum_{j \in \mathcal{R}(p)} w_{ij}}{\sum_{i \neq j} w_{ij}}$. Since our goal is only to reconstruct the proportions, and our model only knows the shortest-path distance between all nodes $i \in V$ and $p \in \mathcal{P}$, we set $w'_{ip} = s/d_{ip}^2$ where $s \geq 1$. At the first glance setting $s = |\mathcal{R}(p)|$ seems appropriate, since p represents $|\mathcal{R}(p)|$ addends of the stress model. Nevertheless, this strongly overestimates the weight of close partitions. Therefore, we propose to set $s = |\{j \in \mathcal{R}(p) : d_{jp} \leq d_{ip}/2\}|$. This follows the idea that p is only a good representative for the nodes in $\mathcal{R}(p)$ that are at least as close to p as to i . Since the graph-theoretic distance between i and $j \in \mathcal{R}(p)$ is unknown, our best guess is that j lies on the shortest path from p to i . Consequently, if $d_{jp} \leq d_{ip}/2$ node j must be at least as close to p as to i . Note that $w'_{pp'}$ does not necessarily equal $w'_{p'p}$ for $p, p' \in \mathcal{P}$, and if $k = n$ our model reduces to the full stress model.

Asymptotic Running Time: To minimize Eq. (3) in each iteration we displace all nodes $i \in V$ according to Eq. (4). Since this requires $|N(i)| + k$ constant time operations, given that all graph-theoretic distances are known, the total time per iteration is in $\mathcal{O}(kn + m)$. Furthermore, only the distances between all $i \in V$ and $p \in \mathcal{P}$ have to be known, which can be done in $\mathcal{O}(k(m + n \log n))$ time and requires $\mathcal{O}(kn)$ additional space. If the graph-theoretic distances for all $p \in \mathcal{P}$ are computed with a multi-source shortest path algorithm (MSSP), it is possible to construct \mathcal{R} as well as calculate all w'_{ip} during its execution without increasing its asymptotic running time. The full algorithm to minimize our sparse stress model is presented in Algorithm 1.

Table 1. Dataset: n , m , $\delta(G)$, $\Delta(G)$, and $D(G)$ denote the number of nodes, edges, the min. and max. degree, and the diameter, respectively. Column $\{deg(i)\}$ shows the degree and $\{d_{ij}\}$ the distance distribution. Bipartite graphs are marked with * and weighted graphs with **

graph	n	m	$\delta(G)$	$\Delta(G)$	$D(G)$	$\{deg(i)\}$	$\{d_{ij}\}$	graph	n	m	$\delta(G)$	$\Delta(G)$	$D(G)$	$\{deg(i)\}$	$\{d_{ij}\}$
dwt1005	1005	3808	3	26	34			pesa	11738	33914	2	9	208		
1138bus	1138	1458	1	17	31			bodyy5	18589	55346	2	8	132		
plat1919	1919	15240	2	18	43			finance256	20657	71866	1	54	55		
3elt	4740	13722	3	9	65			btrees (binary tree)	1023*	1022	1	3	18		
USpowerGrid	4941	6594	1	19	46			qh882	1764*	3354	1	14	32		
commanche	7920	11880**	3	3	438.00			lpship04l	2526*	6380	1	84	13		
LeHavre	11730	15133**	1	7	33800.67										

4 Experimental Evaluation

We report on two sets of experiments. The first is concerned with the evaluation of the impact of different pivot sampling strategies. The second set is designed to assess how well the different sparse stress models approximate the full stress model, in both absolute terms and in relation to the speed-up achieved.

For the experiments we implemented the sparse stress model, Algorithm 1, as well as different sampling techniques in Java using Oracle SDK 1.8 and the yFiles 2.9 graph library (www.yworks.com). The tests were carried out on a single 64-bit machine with a 3.60GHz quad-core Intel Core i7-4790 CPU, 32GB RAM, running Ubuntu 14.10. Times were measured using the `System.currentTimeMillis()` command. The reported running times were averaged over 25 iterations. We note here that all drawing algorithms, except stated otherwise, were initialized with a 200 PivotMDS layout [4]. Furthermore, the maximum number of iterations for the full stress algorithm was set to 500. As stress is not resilient against scaling, see Eq. (1), we optimally rescaled each drawing such that it creates the lowest possible stress value [2].

Data: We conducted our experiments on a series of different graphs, see Tab. 1, most of them taken from the sparse matrix collection [8]. We selected these graphs as they differ in their structure and size, and are large enough to compare the results of different techniques. Two of the graphs, *LeHavre* and *commanche*, have predefined edge lengths that were derived from the node coordinates. We did not modify the graphs in any way, except for those that were disconnected. In this case we only kept the largest component.

4.1 Sampling Evaluation

In Sect. 3 we discussed how vital the proper selection of the pivots is for our model. In the optimal case we would sample pivots that are well distributed over the graph, creating regions of equal complexity, and are central in the drawing of their regions. In order to evaluate the impact of different sampling strategies

on the quality of our sparse stress model and recommend a proper sampling scheme, we compared a set of different strategies:

- random: nodes are selected uniformly at random
- MIS filtration: nodes are sampled according to the maximal independent set filtration algorithm by Gajer et al. [13]. Once $n \leq k$ the coarsening stops. If $n < k$, unsampled nodes from the previous level are randomly added
- max/min euclidean: starting with a uniformly randomly chosen node, \mathcal{P} is extended by adding $\arg \max_{i \in V \setminus \mathcal{P}} \min_{p \in \mathcal{P}} \|x_i - x_p\|$
- max/min sp: similar to max/min euclidean except that \mathcal{P} is extended according $\arg \max_{i \in V \setminus \mathcal{P}} \min_{p \in \mathcal{P}} d_{ip}$ [4]

Pretests showed that the max/min sp strategy initially favors sampling leaves, but nevertheless produces good results for large k . Thus, we also evaluated strategies building on this idea, yet try to overcome the problem of leaf node sampling.

- max/min random sp: similar to max/min sp, yet each node i is sampled with a probability proportional to $\min_{p \in \mathcal{P}} d_{ip}$
- k-means layout: the nodes are selected via a k-means algorithm, running at most 50 iterations, on the initial layout
- k-means sp: initially k nodes with max/min sp are sampled succeeded by k-means sampling using the shortest path entries of these pivots
- k-means + max/min sp: \mathcal{P} is initialized with $k/2$ pivots via k-means layout and the remaining nodes are sampled via max/min sp

To quantify how well suited each of the sampling techniques is for our model, we ran each combination on each graph with $k \in \{50, 51, \dots, 200\}$ pivots. For all tests the sparse stress algorithm terminated after 200 iterations. Since all techniques at some point rely on a random decision, we repeated each execution 20 times in order to ensure we do not rest our results upon outliers. To distinguish the applicability of the different techniques to our model, we used two measures. The first measure is the normalized stress, which is the stress value divided by $\binom{n}{2}$. While the normalized stress measures the quality of our drawing, we also calculated the Procrustes statistic, which measures how well the layout matches the full stress drawing [31]. The range of the Procrustes statistic is $[0, 1]$, where 0 is the optimal match.

The results of these experiments for some of the instances are presented in Figs. 1 and 2 (see the Appendix in [29] for the full set of data). In these plots each dot represents the median and each line starts at the 25%, 75% percentile and ends at the 5%, 95% percentile, respectively. For the sake of readability we binned each 25 consecutive sample sizes. Furthermore, the strategies were ordered according to their overall ranking w.r.t. the evaluated measure. For most of the graphs using k-means sp sampling yields the layouts with the lowest normalized stress value. There are only two graphs where this strategy performs worse than other tested strategies. The one graph where k-means sp is outclassed, yet only for large k by max/min sp, is *pesa*. The reason for this result is that k-means sp mainly samples pivots in the center of the left arm, see Table 4, creating

twists. Max/min sp for small k in contrast mostly samples nodes on the contour of the arm, yet once k reaches a certain threshold the resulting distribution of the pivots prevents twists, yielding a lower normalized stress value.

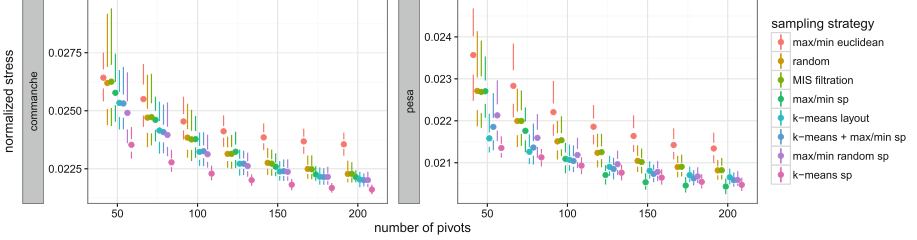


Fig. 1. Comparison of different sampling strategies and number of pivots w.r.t. the resulting normalized stress value

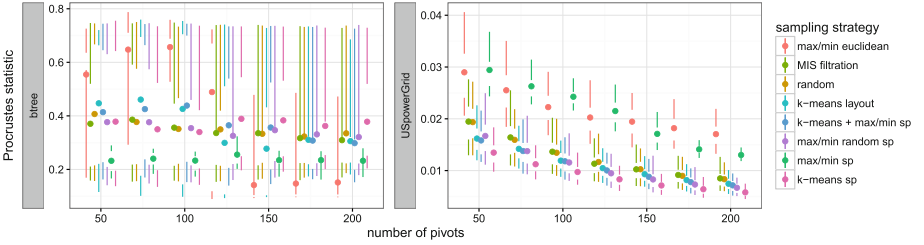


Fig. 2. Comparison of different sampling strategies and number of pivots w.r.t. the resulting Procrustes statistic

The explanation of the poor behavior for *lpship04l* is strongly related to its structure. The low diameter of 13 causes, after a few iterations, the max/min sp strategy to repeatedly sample nodes that are part of the same cluster, see Table 4, and consequently are structurally very similar. As k-means sp builds on max/min sp, it can only slightly improve the pivot distribution. The argument that the problem is related to the structure is reinforced by the outcome of the random strategy. Still, except for these two graphs k-means sp generates the best outcomes, and since this strategy is also strongly favorable over the others subject to the Procrustes statistics, see Fig. 2, our following evaluation always relies on this sampling strategy. However, we note that the Procrustes statistic for *btree* and *lpship04l* are by magnitudes larger than for any other tested graph. While for *lpship04l* this is mostly caused by the quality of the drawings, this is only partly true for *btree*. The other factor contributing to the high Procrustes statistic for *btree* is caused by the restricted set of operations provided by the Procrustes analysis. As dilation, translation, and rotation are used to find the best match between two layouts, the Procrustes analysis cannot

Table 2. Stress and Procrustes statistics: sparse model values are highlighted when no larger than minimum over previous methods

graph	full stress	sparse 200	sparse 100	sparse 50	maxent	MARS 200	MARS 100	GRIP	1-stress	PivotMDS
stress										
dwt1005	10 729	10 940	11 081	11 329	21 623	17 660	20 134	52 517	12 495	14 459
1138bus	39 974	40 797	41 471	42 686	44 650	64 363	63 614	54 986	73 512	75 427
plat1919	18 572	18 840	19 072	19 719	23 850	53 246	64 166	113 765	75 973	82 865
3elt	422 940	426 564	430 200	437 051	585 967	503 600	754 134	934 206	555 934	634 401
USpowerGrid	702 055	720 642	731 187	749 464	1 021 457	766 535	783 888	1 495 373	1 111 216	1 123 698
commanche	654 094	677 220	699 890	749 609	1 507 634	2 761 605	3 145 480	1 539 767	2 085 818	2 157 943
LeHavre	439 188	433 030	441 986	454 785	1 231 283	12 012 307	12 570 692	8 658 371	1 255 474	1 305 577
pesa	1 373 514	1 417 449	1 452 975	1 495 512	10 423 779	3 563 772	8 281 116	2 957 738	3 486 176	3 325 889
bodyy5	3 547 659	3 566 636	3 585 087	3 630 380	5 248 755	6 385 559	4 072 905	10 389 846	4 245 006	4 715 728
finance256	6 175 210	6 415 761	6 474 787	6 582 890	8 151 335	7 267 598	8 643 239	19 817 355	12 257 268	11 380 089
btree	60 206	61 839	63 325	66 122	67 871	103 436	100 767	96 235	157 988	164 329
qh882	84 524	86 345	87 695	89 556	103 601	117 195	161 113	127 914	146 935	143 142
lpship041	250 599	297 547	316 674	343 694	329 255	558 923	542 667	771 284	775 813	793 238
Procrustes statistic										
dwt1005		0.001	0.005	0.003	0.027	0.008	0.018	0.263	0.004	0.008
1138bus		0.009	0.016	0.025	0.022	0.148	0.145	0.071	0.097	0.102
plat1919		0.000	0.000	0.001	0.015	0.026	0.031	0.236	0.045	0.051
3elt		0.001	0.001	0.002	0.026	0.009	0.029	0.199	0.017	0.023
USpowerGrid		0.006	0.008	0.012	0.068	0.014	0.018	0.256	0.051	0.051
commanche		0.001	0.002	0.005	0.039	0.026	0.167	0.092	0.066	0.066
LeHavre		0.001	0.001	0.001	0.012	0.163	0.173	0.256	0.010	0.010
pesa		0.009	0.010	0.010	0.095	0.025	0.070	0.017	0.021	0.021
bodyy5		0.000	0.000	0.000	0.012	0.011	0.003	0.100	0.004	0.007
finance256		0.009	0.006	0.005	0.013	0.007	0.018	0.206	0.042	0.041
btree		0.748	0.165	0.241	0.233	0.360	0.367	0.386	0.361	0.364
qh882		0.015	0.015	0.021	0.046	0.061	0.114	0.075	0.086	0.079
lpship041		0.176	0.112	0.148	0.160	0.246	0.587	0.463	0.393	0.401

Table 3. Runtime in seconds: fastest sparse model yielding lower stress than best previous method, c.f. Table 2, is highlighted. Marked implementations written in C/C++ with time measured via `clock()` command

graph	full stress	sparse 200	sparse 100	sparse 50	maxent*	MARS 200*	MARS 100*	GRIP*	1-stress	PivotMDS
dwt1005	1.26	0.22	0.14	0.10	0.47	1.02	2.36	0.06	0.08	0.06
1138bus	2.20	0.25	0.14	0.09	0.01	3.16	1.96	0.20	0.06	0.04
plat1919	9.70	0.74	0.51	0.42	0.15	6.80	4.79	0.19	0.31	0.20
3elt	31.82	0.94	0.59	0.46	2.26	16.31	8.43	0.71	0.37	0.23
USpowerGrid	36.48	0.81	0.48	0.35	2.53	13.54	7.62	1.67	0.28	0.21
commanche	340.10	4.89	2.56	1.47	3.60	22.72	12.43	2.29	0.47	0.35
LeHavre	475.05	6.53	3.48	2.37	6.31	27.57	19.50	10.18	0.81	0.54
pesa	373.23	4.25	2.47	1.53	5.96	50.10	42.68	3.56	0.95	0.60
bodyy5	463.47	3.84	2.63	1.78	9.97	46.63	9.27	10.43	1.64	1.04
finance256	1016.92	7.32	4.41	3.09	14.76	32.16	24.66	12.12	2.51	1.60
btree	7.79	0.38	0.15	0.10	0.63	2.70	1.48	0.06	0.06	0.03
qh882	6.61	0.45	0.30	0.23	0.97	8.45	5.79	0.15	0.17	0.14
lpship041	18.30	0.55	0.30	0.20	0.99	7.06	7.63	0.16	0.15	0.10

resolve reflections. Therefore, if in the one layout of *btree*, the subtree T_1 of v is drawn to the right of subtree T_2 of v and vice versa in the second drawing, although the two layouts are identical, the statistic will be high. This symmetry problem mainly explains the low performance w.r.t. *btree*.

4.2 Full Stress Layout Approximation

The next set of experiments is designed to assess how well our sparse stress model using k-means sp sampling, as well as related sparse stress techniques, resembles the full stress model. For this we compared the median stress layout over 25 repetitions on the same graph of our sparse stress model with $k \in \{50, 100, 200\}$, with MARS,¹ maxent,² PivotMDS, 1-stress, and the weighted version of GRIP.³

¹ <https://github.com/marckhoury/mars>.

² We are grateful to Yifan Hu for providing us with the code.

³ <http://www.cs.arizona.edu/~kobourov/GRIP/>.

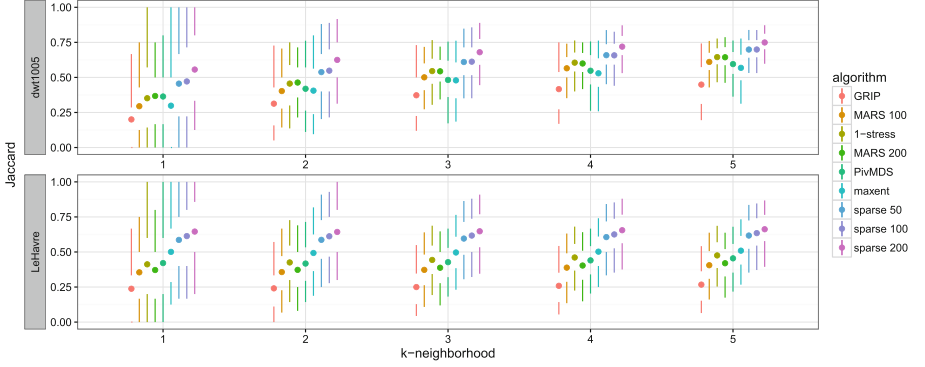


Fig. 3. The similarity of the Gabriel Graph of the full stress layout and the Gabriel Graph of the layout algorithms under consideration as a function of k . For each node of the graph the k -neighborhood in the Gabriel Graph of the full stress layout and the layout algorithm are compared by calculating the Jaccard coefficient. A higher value indicates that the nodes share a high percentage of common neighbors in the different Gabriel Graphs.

The number of iterations of our model as well as for MARS and 1-stress have been limited to 200. Furthermore, we tested MARS with 100 and 200 pivots and report the layout with the smallest stress from the drawings obtained by running `mars` with argument `-p` $\in \{1, 2\}$ combined with a PivotMDS or randomly initialized layout.

Besides comparing the resulting stress values and Procrustes statistics, we compared the distribution of pairwise euclidean distances subject to their graph-theoretic distances. Since the Procrustes statistic has problems with symmetries, as we pointed out in the previous subsection, we propose to evaluate the similarity of the sparse stress layouts with the full stress layout via Gabriel graphs [12]. The Gabriel graph of a given layout x contains an edge between a pair of points if and only if the disc associated with the diameter of the endpoints does not contain any other point. Since the treatment of identical positions is not defined for Gabriel Graphs, we resolve this by adding edges between each pair of identical positions. We assess the similarity between the Gabriel Graph of the full stress layout and the sparse stress layouts by comparing the k -neighborhoods of a node in the graphs using the Jaccard coefficient.

A further measure we introduce evaluates the visual error. More precisely we measure for a given node v the percentage of nodes that lie in the drawing area of the k -neighborhood, but are not part of it. We calculate this value by computing the convex hull induced by the k -neighborhood and then test for each other node if it belongs to the hull or not. This number is then divided by $n - |\{w \in V | d_{vw} \leq k\}|$. Therefore, a low value implies that there are only a few nodes lying in the region, while high values indicate we cannot distinguish non k -neighborhood and k -neighborhood nodes in the drawing. This measure is to a certain extend similar to the precision of neighborhood preservation [15].

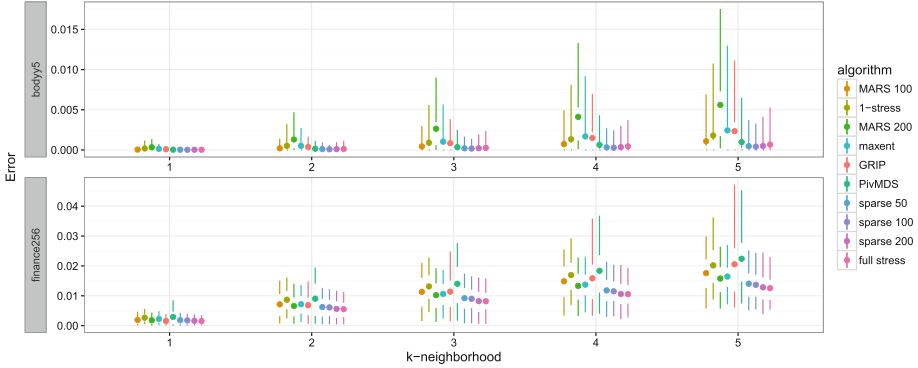
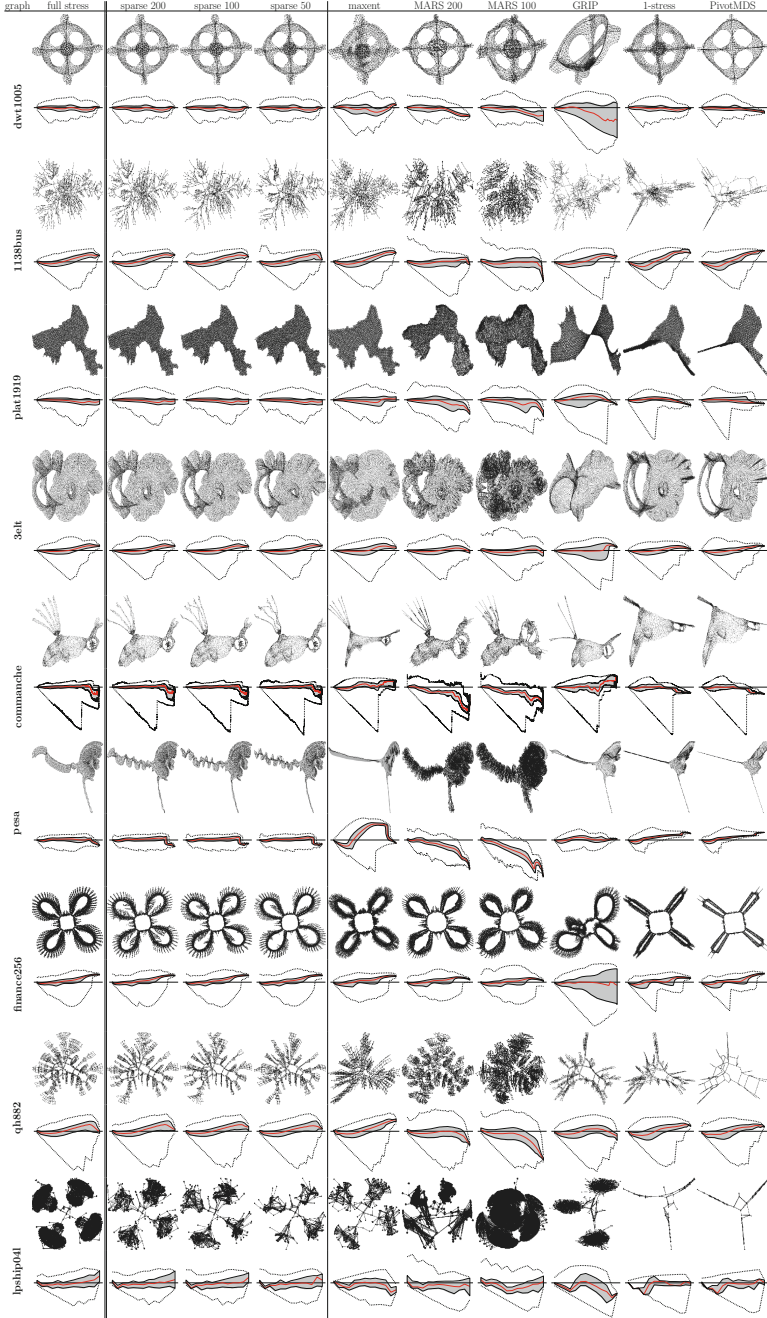


Fig. 4. Error charts as a function of k . For each node of the graph the convex hull w.r.t. the coordinates of the nodes in the k -neighborhood is computed. For each of the convex hulls the error is calculated by counting the number of non k -neighborhood nodes that lie inside or on the contour of this hull divided by their total number.

The results of all these experiments, see Tables 2 and 4, Figs. 3 and 4, and the Appendix in [29], reveal that our model is more adequate in resembling the full stress drawing than any other of the tested algorithm, while showing comparable running times that scale nicely with k , cf. Table 3. The error plots in Table 4 expose the strength of our approximation scheme. We can see that, while all approaches work very well in representing short distances, our approach is more precise in approximating middle and especially long distances, explaining our good results. As the evaluation clearly shows that our approach yields better approximations of the full stress model, we rather want to discuss the low performance of our model for *lpship04l* and thereby expose one weakness of our approach.

Looking at the sparse 50 drawing of *lpship04l* in Table 4, we can see that a large portion of nodes share a similar or even the same position. This is because *lpship04l* has a lot of nodes that share very similar graph-theoretic distance vectors, exhibit highly overlapping neighborhoods, and are drawn in close proximity in the initial PivotMDS layout. While our model would rely on small variations of the graph-theoretic distances to create a good drawing we diminish these differences even further by restricting our model to \mathcal{P} . Consequently, the positional vote for two similar non-pivot nodes i and j that lie in the same partition will only slightly differ, mainly caused by their distinct neighbors. However, as these neighbors are also in close proximity in the initial drawing of *lpship04l* the distance between i and j will not increase. Therefore, if the graph has a lot of structurally very similar nodes and the initial layout has poor quality, our approach will inevitably create drawings where nodes are placed very close to one another.

Table 4. Layouts and error charts of the algorithms. Each chart shows the zero y coordinate (black horizontal line), the median (red line), the 25 and 75 percentiles (black/gray ribbon) and the min/max error (outer black dashed line). The error (y-axis) is the difference between the euclidean distance and the graph-theoretic distance (x-axis). 1000 bins have been used for weighted graphs



5 Conclusion

In this paper we proposed a sparse stress model that requires $\mathcal{O}(kn + m)$ space and time per iteration, and a preprocessing time of $\mathcal{O}(k(m + n \log n))$. While Barnes & Hut derive their representatives from a given partitioning, we argued that for our model it is more appropriate to first select the pivots and then to partition the graph only relying on its structure. Since the approximation quality heavily depends on the proper selection of these pivots, we evaluated different sampling techniques, showing that k-means sp works very well in practice.

Furthermore, we compared a variety of sparse stress models w.r.t. their performance in approximating the full stress model. We therefore proposed two new measures to assemble the similarity between two layouts of the same graph. For the tested graphs, all our experiments clearly showed that our proposed sparse stress model exceeds related approaches in approximating the full stress layout without compromising the computation time.

References

1. Barnes, J., Hut, P.: A hierarchical $\mathcal{O}(n \log n)$ force-calculation algorithm. *Nature* **324**(6096), 446–449 (1986). <http://dx.doi.org/10.1038/324446a0>
2. Borg, I., Groenen, P.J.: *Modern Multidimensional Scaling: Theory and Applications*. Springer, New York (2005)
3. Brandes, U.: Drawing on physical analogies. In: Kaufmann, M., Wagner, D. (eds.) *Drawing Graphs. LNCS*, vol. 2025, pp. 71–86. Springer, Heidelberg (2001). doi:[10.1007/3-540-44969-8_4](https://doi.org/10.1007/3-540-44969-8_4)
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). doi:[10.1007/978-3-540-70904-6_6](https://doi.org/10.1007/978-3-540-70904-6_6)
5. Brandes, U., Pich, C.: An experimental study on distance-based graph drawing. In: Tollis, I.G., Patrignani, M. (eds.) *GD 2008. LNCS*, vol. 5417, pp. 218–229. Springer, Heidelberg (2009). doi:[10.1007/978-3-642-00219-9_21](https://doi.org/10.1007/978-3-642-00219-9_21)
6. Brandes, U., Schulz, F., Wagner, D., Willhalm, T.: Travel planning with self-made maps. In: Buchsbaum, A.L., Snoeyink, J. (eds.) *ALLENEX 2001. LNCS*, vol. 2153, pp. 132–144. Springer, Heidelberg (2001). doi:[10.1007/3-540-44808-X_10](https://doi.org/10.1007/3-540-44808-X_10)
7. Cohen, J.D.: Drawing graphs to convey proximity: an incremental arrangement method. *ACM Trans. Comput. Hum. Interact.* **4**(3), 197–229 (1997). <http://doi.acm.org/10.1145/264645.264657>
8. Davis, T.A., Hu, Y.: The university of Florida sparse matrix collection. *ACM Trans. Math. Softw.* **38**(1), 1: 1–1: 25 (2011). <http://www.cise.ufl.edu/research/sparse/matrices>
9. Drineas, P., Frieze, A.M., Kannan, R., Vempala, S., Vinay, V.: Clustering large graphs via the singular value decomposition. *Mach. Learn.* **56**(1–3), 9–33 (2004). <http://dx.doi.org/10.1023/B:MACH.0000033113.59016.96>
10. Frick, A., Ludwig, A., Mehldau, H.: A fast adaptive layout algorithm for undirected graphs (extended abstract and system demonstration). In: Tamassia, R., Tollis, I.G. (eds.) *GD 1994. LNCS*, vol. 894, pp. 388–403. Springer, Heidelberg (1995). doi:[10.1007/3-540-58950-3_393](https://doi.org/10.1007/3-540-58950-3_393)

11. Fruchterman, T.M.J., Reingold, E.M.: Graph drawing by force-directed placement. *Softw. Pract. Exp.* **21**(11), 1129–1164 (1991). <http://dx.doi.org/10.1002/spe.4380211102>
12. Gabriel, R.K., Sokal, R.R.: A new statistical approach to geographic variation analysis. *Syst. Zool.* **18**(3), 259–278 (1969)
13. Gajer, P., Goodrich, M.T., Kobourov, S.G.: A multi-dimensional approach to force-directed layouts of large graphs. In: Marks, J. (ed.) *GD 2000*. LNCS, vol. 1984, pp. 211–221. Springer, Heidelberg (2001). doi:[10.1007/3-540-44541-2_20](https://doi.org/10.1007/3-540-44541-2_20)
14. Gansner, E.R., Hu, Y., Krishnan, S.: COAST: a convex optimization approach to stress-based embedding. In: Wismath, S., Wolff, A. (eds.) *GD 2013*. LNCS, vol. 8242, pp. 268–279. Springer, Heidelberg (2013). doi:[10.1007/978-3-319-03841-4_24](https://doi.org/10.1007/978-3-319-03841-4_24)
15. Gansner, E.R., Hu, Y., North, S.C.: A maxent-stress model for graph layout. *IEEE Trans. Vis. Comput. Graph.* **19**(6), 927–940 (2013)
16. Gansner, E.R., Koren, Y., North, S.: Graph drawing by stress majorization. In: Pach, J. (ed.) *GD 2004*. LNCS, vol. 3383, pp. 239–250. Springer, Heidelberg (2005). doi:[10.1007/978-3-540-31843-9_25](https://doi.org/10.1007/978-3-540-31843-9_25)
17. Greengard, L.: The Rapid evaluation of potential fields in particle systems. *ACM distinguished dissertations*, MIT Press, Cambridge (1988). <http://opac.inria.fr/record=b1086802>
18. Hachul, S., Jünger, M.: Drawing large graphs with a potential-field-based multilevel algorithm. In: Pach, J. (ed.) *GD 2004*. LNCS, vol. 3383, pp. 285–295. Springer, Heidelberg (2005). doi:[10.1007/978-3-540-31843-9_29](https://doi.org/10.1007/978-3-540-31843-9_29)
19. Hall, K.M.: An r -dimensional quadratic placement algorithm. *Manag. Sci.* **17**(3), 219–229 (1970). <http://dx.doi.org/10.1287/mnsc.17.3.219>
20. Hu, Y., Shi, L.: Visualizing large graphs. *Wiley Interdiscip. Rev. Comput. Stat.* **7**(2), 115–136 (2015). <http://dx.doi.org/10.1002/wics.1343>
21. Hu, Y.F.: Efficient and high quality force-directed graph drawing. *Mathematica J.* **10**, 37–71 (2005). <http://www.mathematica-journal.com/issue/v10i1/contents/graph.draw/graph.draw.pdf>
22. Ingram, S., Munzner, T.: Glint: An MDS framework for costly distance functions. In: Kerren, A., Seipel, S. (eds.) *Proceedings of SIGRAD 2012, Interactive Visual Analysis of Data*. Linköping Electronic Conference Proceedings, vol. 81, pp. 29–38. Linköping University Electronic Press (2012). http://www.ep.liu.se/ecp_article/index.en.aspx?issue=081;article=005
23. Khoury, M., Hu, Y., Krishnan, S., Scheidegger, C.E.: Drawing large graphs by low-rank stress majorization. *Comput. Graph. Forum* **31**(3), 975–984 (2012). <http://dx.doi.org/10.1111/j.1467-8659.2012.03090.x>
24. Klimenta, M., Brandes, U.: Graph drawing by classical multidimensional scaling: new perspectives. In: Didimo, W., Patrignani, M. (eds.) *GD 2012*. LNCS, vol. 704, pp. 55–66. Springer, Heidelberg (2013). doi:[10.1007/978-3-642-36763-2_6](https://doi.org/10.1007/978-3-642-36763-2_6)
25. Kobourov, S.G.: Force-directed drawing algorithms. In: Tamassia, R. (ed.) *Handbook of Graph Drawing and Visualization*, pp. 383–408. CRC Press, Boca Raton (2013)
26. Koren, Y., Carmel, L., Harel, D.: ACE: A fast multiscale eigenvectors computation for drawing huge graphs. In: Wong, P.C., Andrews, K. (eds.) *InfoVis 2002*, pp. 137–144. IEEE Computer Society (2002). <http://dx.doi.org/10.1109/INFVIS.2002.1173159>
27. McGee, V.E.: The multidimensional analysis of “elastic” distances. *Br. J. Math. Stat. Psychol.* **19**(2), 181–196 (1966). <http://dx.doi.org/10.1111/j.2044-8317.1966.tb00367.x>

28. Meyerhenke, H., Nöllenburg, M., Schulz, C.: Drawing large graphs by multilevel maxent-stress optimization. In: Di Giacomo, E., Lubiw, A. (eds.) GD 2015. LNCS, vol. 9411, pp. 30–43. Springer, Heidelberg (2015). doi:[10.1007/978-3-319-27261-0_3](https://doi.org/10.1007/978-3-319-27261-0_3)
29. Ortmann, M., Klimenta, M., Brandes, U.: A sparse stress model. CoRR abs/1608.08909 (2016). <http://arxiv.org/abs/1608.08909>
30. Quigley, A.J.: Large scale relational information visualization, clustering, and abstraction. Ph.D. thesis, University of Newcastle (2000)
31. Sibson, R.: Studies in the robustness of multidimensional scaling: procrustes statistics. *J. Roy. Stat. Soc. Ser. B (Methodol.)* **40**(2), 234–238 (1978). <http://www.jstor.org/stable/2984761>
32. de Silva, V., Tenenbaum, J.B.: Global versus local methods in nonlinear dimensionality reduction. In: Becker, S., Thrun, S., Obermayer, K. (eds.) NIPS 2002, pp. 705–712. MIT Press, Cambridge (2002). <http://papers.nips.cc/paper/2141-global-versus-local-methods-in-nonlinear-dimensionality-reduction>
33. Tunkelang, D.: JIGGLE: java interactive graph layout environment. In: Whitesides, S.H. (ed.) GD 1998. LNCS, vol. 1547, pp. 413–422. Springer, Heidelberg (1998). doi:[10.1007/3-540-37623-2_33](https://doi.org/10.1007/3-540-37623-2_33)
34. Tunkelang, D.: A numerical optimization approach to general graph drawing. Ph.D. thesis, Carnegie Mellon University (1999)
35. Walshaw, C.: A multilevel algorithm for force-directed graph-drawing. *J. Graph Algorithms Appl.* **7**(3), 253–285 (2003). <http://www.cs.brown.edu/publications/jgaa/accepted/2003/Walshaw2003.7.3.pdf>

Graph Drawing and Network Visualization
24th International Symposium, GD 2016, Athens,
Greece, September 19-21, 2016, Revised Selected
Papers

Hu, Y.; Nöllenburg, M. (Eds.)

2016, XX, 642 p. 201 illus., Softcover

ISBN: 978-3-319-50105-5