Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.18576/amis/100112

Multidimensional Scaling in the Poincaré disk

Andrej Cvetkovski[1,∗] *and Mark Crovella*[2]

[1] School of Informatics, European University, Skopje, Macedonia
[2] Department of Computer Science, Boston University, Boston MA, USA

Received: 15 Jun. 2015, Revised: 13 Aug. 2015, Accepted: 14 Aug. 2015
Published online: 1 Jan. 2016

Abstract: Multidimensional scaling (MDS) is a class of projective algorithms traditionally used in *Euclidean space* to produce two- or three-dimensional visualizations of datasets of multidimensional points or point distances. More recently however, several authors have pointed out that for certain datasets, *hyperbolic target space* may provide a better fit than Euclidean space.
In this paper we develop PD-MDS, a metric MDS algorithm designed specifically for the Poincaré disk (PD) model of the hyperbolic plane. Emphasizing the importance of *proceeding from first principles* in spite of the availability of various black box optimizers, our construction is based on an elementary hyperbolic line search and reveals numerous particulars that need to be carefully addressed when implementing this as well as more sophisticated iterative optimization methods in a hyperbolic space model.

Keywords: Dimensionality reduction, hyperbolic multidimensional scaling, Poincaré disk, steepest descent, approximate line search, graph embedding

1 Introduction

Metric multidimensional scaling (MDS) [1,2] is a class of algorithms that take as input some or all of the inter-object distances (*pair dissimilarities*) for n objects and produce as output a *point configuration* of n points specified by their coordinates in a chosen d-dimensional *target space*.

The goal is to return the point configuration whose inter-point distances in the d-dimensional space match as closely as possible the original input distances. Usually, this goal is pursued by minimizing a scalar badness-of-fit *objective function* defined for an arbitrary n-point configuration in the target space; ideally, the output of an MDS algorithm should be the configuration that achieves the global minimum of the objective function.

If the target space dimension is 2 or 3, the output configuration can be graphically represented, which makes MDS a visualization tool seeking to preserve the input distances as faithfully as possible, thus clustering the objects in the target space by similarity. More generally, for a given dimension d, metric multidimensional scaling can be used to *embed* an input set of dissimilarities of the original objects into a d-dimensional metric space.

In order to apply MDS, several design decisions must be made. One first needs to choose a *target metric space* of appropriate dimension d and a corresponding distance function. An *objective function* should be chosen so that it provides a suitable measure of inaccuracy for a given embedding application. If the objective function is nonlinear but satisfies some mild general conditions (smoothness), a *numerical optimization* method can be chosen for the implementation.

1.1 Target space

The Euclidean plane is the most common choice of a target space for visualization and other applications due to its simplicity and intuitiveness. Spherical surface can be used, for example, to avoid the edge effect of a planar representation [3].

In general, MDS on curved subspaces of Euclidean space can be viewed as MDS in a higher dimensional Euclidean space constrained to a particular surface [4,5].

A multidimensional scaling algorithm for fitting distances to constant-curvature Riemannian spaces is given by [6]. This work uses the hyperboloid model of the hyperbolic space requiring an $n+1$-dimensional Euclidean space to represent an n-dimensional hyperbolic space, and is less suitable for visualization purposes. The reader is referred to [7] or [8] for a review of the history

∗ Corresponding author e-mail: acvetk@gmail.com of MDS on Riemannian manifolds of constant or nonconstant curvature.

The use of metric MDS in the hyperbolic plane in the context of interactive visualization is proposed by [9], inspired by the focus and context hyperbolic tree viewer of [10]. The study focuses on the task of embedding higher-dimensional point sets into 2-dimensional configurations for the purpose of interactive visualization. It is demonstrated that the PD has capacity to accommodate lower stress embedding than the Euclidean plane. Several important pointers to the difficulties one encounters in implementing such algorithms are given, but a definite specification or implementation is not provided.

The adequacy of the hyperbolic spaces for embedding of various data is also studied and confirmed in the contexts of network embedding for path cost estimation [11] and routing [12,13,14,15].

1.2 Objective function

A least squares formulation of MDS, to be used in conjunction with an iterative numerical method for unconstrained optimization is proposed by [16]. The objective function therein (the Sammon stress criterion) is defined as a normalized sum of the squared differences between the original dissimilarities and the embedded distances of the final point configuration. To minimize this function, Sammon proposes a descent method with step components calculated using the first two component derivatives of the objective function.

[9] adopts Sammon's badness-of-fit measure for hyperbolic MDS but observes that applying Sammon's iterative procedure in the Poincaré disk (PD) using exact derivatives is difficult due to the complicated symbolic expressions of the second derivative of the hyperbolic distance function in this model. Subsequently, the Levenberg-Marquardt least squares method is applied in [9], using only first-order derivatives for the optimization, but the details of applying this iterative method in the Poincaré disk are not elaborated.

The proposed method to convert the seemingly constrained optimization problem to an unconstrained one by [9] (Eq.12) ensures that the moving configuration would stay inside the model during the optimization. However, this transformation fails to follow the distance realizing (hyperbolic) lines, or even Euclidean lines. The problem is illustrated in Fig. 1. The possibility that the dissimilarity matrix has missing values is also not addressed in this work, as the dissimilarities are generated from higher-dimensional points. Input data, however, may also be sparse.

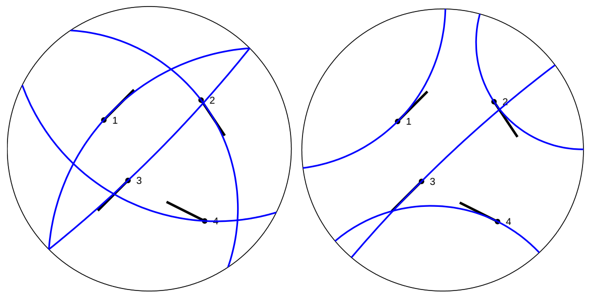

Fig. 1: Comparison of the point trajectories: H-MDS of [9] (left) vs. PD-MDS hyperbolic lines (right)

1.3 PD-MDS

In this paper we present PD-MDS, a metric multidimensional scaling algorithm using the Poincaré disc (PD) model. For didactic purposes, we complement our exhibition of PD-MDS with a simple steepest decent method with line search. We show the details of the steepest descent along hyperbolic lines in the PD and present a suitable approximate hyperbolic line search procedure. Based on this development, we show the particulars of a numerical implementation of PD-MDS.

PD-MDS is applicable in its own right; additionally, its construction also illustrates some of the specifics that need to be considered when transferring more sophisticated iterative optimization methods to the PD or to other hyperbolic models.

Our numerical experiments indicate that the performance of a steepest descent method for minimizing a least squares objective on large configurations in the PD is notably dependent on the line search method used, and that binary hyperbolic line search provides markedly better convergence and cost properties for PD-MDS compared to more sophisticated or precise methods.

The rest of this paper is organized as follows. Section 2 consolidates the notation and concepts from hyperbolic geometry that will be used throughout, and proceeds to develop two of the building blocks of PD-MDS – steepest descent in the PD and a corresponding hyperbolic line search. Section 3 considers particular objective functions and gradients and further discusses properties and applicability of multidimensional scaling in the PD. Section 4 provides results from the numerical evaluation of the proposed algorithm. Concluding remarks are given in Section 5.

2 A descent method for the Poincaré disk

In this section we introduce our notational conventions and establish some properties of the Poincaré disk that will be used in what follows. We then proceed to formally define a Poincaré-disk specific descent method and a binary hyperbolic line search, that together make a simple, yet efficient iterative minimization method for this model of the hyperbolic plane.

2.1 Preliminaries

The *Poincaré disk model* of the hyperbolic plane is convenient for our considerations since it has circular symmetry and a closed form of the inter-point distance formula exists [17].

We will be using complex rectangular coordinates to represent the points of the hyperbolic plane, making the PD model a subset of the complex plane $\mathbb{C}$:

$$\mathbb{D} = \{z \in \mathbb{C} \mid |z| < 1\}. \tag{1}$$

The *hyperbolic distance* between two points z_j and z_k in $\mathbb{D}$ is given by

$$d_{\mathbb{D}}(z_j, z_k) = 2\operatorname{atanh}\frac{|z_j - z_k|}{|1 - z_j\overline{z_k}|}, \tag{2}$$

where $\overline{z}$ denotes the complex conjugate.

Möbius transformations are a class of transformations of the complex plane that preserve generalized circles. The special Möbius transformations that take $\mathbb{D}$ to $\mathbb{D}$ and preserve the hyperbolic distance have the form

$$T(z) = \frac{az + b}{\overline{b}z + \overline{a}}, \quad a, b \in \mathbb{C}, \quad |a|^2 - |b|^2 \neq 0. \tag{3}$$

Given a point $z_0 \in \mathbb{D}$ and a direction $\gamma \in \mathbb{C}$ with $|\gamma| = 1$, we can travel a hyperbolic distance $s \geq 0$ along a hyperbolic line starting from z_0 in the direction γ, arriving at the point z_0'.

Lemma 1. For $z_0 \in \mathbb{D}$, $\gamma \in \mathbb{C}$ with $|\gamma| = 1$, and $s \geq 0$, the point

$$z_0' = \frac{\gamma \tanh\frac{s}{2} + z_0}{\overline{z_0}\gamma \tanh\frac{s}{2} + 1}$$

(i) belongs to the hyperbolic ray passing through z_0 and having direction γ at z_0, and
(ii) $d_{\mathbb{D}}(z_0, z_0') = s$.

Proof. Given a point $z_0 \in \mathbb{D}$ and a direction $\gamma \in \mathbb{C}$ with $|\gamma| = 1$, the hyperbolic ray in $\mathbb{D}$ passing through z_0 and having direction γ at z_0 can be parametrized by $r \in [0, 1)$ as

$$f(r) = \frac{r\gamma + z_0}{r\gamma\overline{z_0} + 1}. \tag{4}$$

Noting that (4), seen as a function of $z = r\gamma$:

$$T(z) = \frac{z + z_0}{z\overline{z_0} + 1}$$

is a Möbius transformation taking $\mathbb{D}$ to $\mathbb{D}$ and preserving hyperbolic distances, we see that

$$d_{\mathbb{D}}(f(r), z_0) = d_{\mathbb{D}}(0, r) = \ln\frac{1 + r}{1 - r}$$

whence it follows that moving z_0 along a hyperbolic line in the direction γ by a hyperbolic distance $s = \ln((1 + r)/(1 - r))$ we arrive at the point $z_0' = f\left(\tanh\frac{s}{2}\right)$. □

Next, we introduce some of the notation that will be used throughout.

– Let the *point configuration* at iteration $t = 1, 2, \ldots T$ consist of n points in the Poincaré disk $\mathbb{D}$

$$z_j(t), \quad j = 1 \ldots n$$

represented by their rectangular coordinates:

$$z_j(t) = y_{j,1}(t) + iy_{j,2}(t), \; i = \sqrt{-1}, \; y_{j,1}, y_{j,2} \in \mathbb{R}$$

with $|z_j(t)| < 1$.

– We also use vector notation to refer to the point configuration

$$\begin{aligned}\mathbf{z}(t) &= \left[z_1(t)\; z_2(t)\; \ldots\; z_n(t)\right]^T = \mathbf{y}_1 + i\mathbf{y}_2 = \\ &= \left[y_{1,1}(t)\; y_{2,1}(t)\; \ldots\; y_{n,1}(t)\right]^T + \\ &\quad i\left[y_{1,2}(t)\; y_{2,2}(t)\; \ldots\; y_{n,2}(t)\right]^T,\end{aligned}$$

where $[\cdot]^T$ in this work indicates the real matrix transpose (to be distinguished from the complex conjugate transpose.)

– The *distance matrix* for a given point configuration $\mathbf{z}$ is the real valued symmetric matrix $\mathbf{D}(\mathbf{z}) = [d_{jk}]_{n \times n}$ whose entry d_{jk} is the hyperbolic distance between points z_j and z_k in the configuration $\mathbf{z}$:

$$d_{jk} = d_{\mathbb{D}}(z_j, z_k).$$

– The *dissimilarity matrix* $\boldsymbol{\Delta} = [\delta_{jk}]_{n \times n}$ is a symmetric, real-valued matrix containing the desired inter-point distances of the final output configuration (the *dissimilarities*). The diagonal elements are $\delta_{jj} = 0$ and all other entries are positive real numbers: $\delta_{jk} = \delta_{kj} > 0$ for $j \neq k$.

– The *indicator matrix* $\mathbf{I} = [I_{jk}]_{n \times n}$ is a symmetric 0-1 matrix, used to allow for missing dissimilarity values. The entries of $\mathbf{I}$ corresponding to missing values in $\boldsymbol{\Delta}$ are set to 0. All other entries are set to 1.

– The *weight matrix* $\mathbf{W} = [w_{jk}]_{n \times n}$ is a symmetric, real-valued matrix introduced to enable weighting of the error terms for individual pairs of points in the objective function sum. For convenience, w_{jk} corresponding to missing dissimilarities are set to some finite value, e.g. 1.

– The *objective function* to be minimized is the *embedding error function* $E_t = E_t(\mathbf{z}, \boldsymbol{\Delta}, \mathbf{W}, \mathbf{I})$ that, given the sets of dissimilarities and weights, associates to a configuration $\mathbf{z}$ an embedding error E_t. An example of an error function is the sum of relative squared differences

$$E_t(\mathbf{z}, \boldsymbol{\Delta}, \mathbf{W}, \mathbf{I}) = \sum_{j=1}^{n}\sum_{k=j+1}^{n} w_{jk} I_{jk}\left(\frac{d_{jk}(t) - \delta_{jk}}{\delta_{jk}}\right)^2. \tag{5}$$

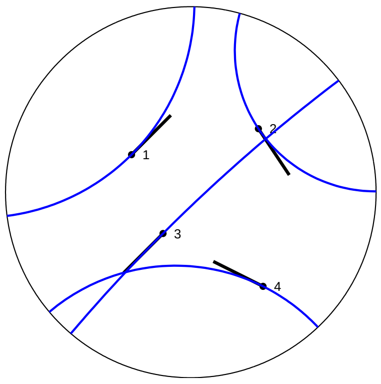

Fig. 2: An example of moving a 4-point configuration in a given (descent) direction along distance realizing paths of the Poincaré disk

The objective function can optionally be normalized per pair by dividing with the number of summands $(n^2 - n)/2$.

2.2 Descent in the Poincaré disk

Given a configuration of points $\mathbf{z}$, matrices $\boldsymbol{\Delta}$, $\mathbf{W}$, and $\mathbf{I}$, the distance function $d_{\mathbb{D}}(z_j, z_k)$, and an objective function $E(\mathbf{z}, \boldsymbol{\Delta}, \mathbf{W}, \mathbf{I})$, define

$$\mathbf{g} = \nabla E \overset{\text{def}}{=} \begin{bmatrix} \frac{\partial E}{\partial y_{1,1}} + i\frac{\partial E}{\partial y_{1,2}} \\ \frac{\partial E}{\partial y_{2,1}} + i\frac{\partial E}{\partial y_{2,2}} \\ \vdots \\ \frac{\partial E}{\partial y_{n,1}} + i\frac{\partial E}{\partial y_{n,2}} \end{bmatrix} = \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_n \end{bmatrix}. \qquad (6)$$

According to Lemma 1, moving the points $z_1, \ldots, z_n$ of the configuration $\mathbf{z}$ along distance realizing paths in the PD defined respectively by the directions $-g_1, \ldots, -g_n$ at $\mathbf{z}$ (Fig. 2) will result in configuration $\mathbf{z}'$ with points

$$z'_j = \frac{-rg_j + z_j}{-rg_j\overline{z_j} + 1} \qquad (7)$$

where $r \geq 0$ is the *step-size parameter* which determines the hyperbolic distances s_j traveled by z_j:

$$s_j = \ln \frac{1 + r\left|g_j\right|}{1 - r\left|g_j\right|}. \qquad (8)$$

The PD model (1) implies the constraints $\left|z_j\right| < 1$ for the point coordinates. Still, the optimization on the PD can be viewed as unconstrained by observing that the constraints $\left|z'_j\right| < 1$ will not be violated while moving a configuration $\mathbf{z}$ in $\mathbb{D}$ if the distances s_j traveled by each point are always kept finite, i.e.

$$s_M = \max_j s_j < \infty. \qquad (9)$$

Since (9), according to (8), corresponds to $r\max_j \left|g_j\right| < 1$, we have the constraint on r

$$r < \frac{1}{\|\mathbf{g}\|_\infty}.$$

When implementing iterative descent minimization methods with line search in the Poincaré disk, it is important to specify a hyperbolic distance window s_M along the descent lines where the next configuration will be sought. In this case the corresponding value of the parameter r is

$$r_M = \frac{1}{\|\mathbf{g}\|_\infty} \cdot \tanh\frac{s_M}{2} < \frac{1}{\|\mathbf{g}\|_\infty}. \qquad (10)$$

Since the Poincaré disk model is conformal, following the direction $-\mathbf{g}$ (the opposite of (6)) corresponds to the *steepest descent* optimization method. Moving the point configuration along hyperbolic lines (distance realizing paths), on the other hand, ensures that the steepest descent direction is exhausted most efficiently given the current information about the objective function.

2.3 A steepest descent algorithm for the PD

Figure 3 shows a framework for PD-MDS.

The *input data* of PD-MDS consists of the initial configuration $\mathbf{z}(1)$, and the input metric: the dissimilarities $\boldsymbol{\Delta}$ with the associated weights $\mathbf{W}$ and the indicators of missing dissimilarities $\mathbf{I}$.

The *input parameters* are the objective error function $E(\mathbf{z}, \boldsymbol{\Delta}, \mathbf{W}, \mathbf{I})$ and the stopping tolerances ε_E, $\varepsilon_{\Delta E}$, $\varepsilon_{\mathbf{g}}$, ε_r, and T_M.

The *output* of PD-MDS consists of the final point configuration $\mathbf{z}(T)$ and its associated embedding error $E_T = E(\mathbf{z}(T), \boldsymbol{\Delta}, \mathbf{W}, \mathbf{I})$.

The *initialization* $\{3.1\}$ sets the maximum hyperbolic distance s_M that can be traveled by any point of the configuration, and the previous value of the embedding error E_{-1}.

Each iteration starts by determining the gradient of the error in the current configuration $\{3.2\}$ and the corresponding window r_M $\{3.3\}$ for the parameter r (Eq. (10)). A hyperbolic line search (described in Sec. 2.4) is performed $\{3.5\}$ in the direction of the steepest descent $-\mathbf{g}$ of the embedding error and the resulting step-size parameter r is used in $\{3.6\}$ to arrive at the next configuration as in (7).

Several *stopping criteria* are used (line $\{3.4\}$) to terminate the search. Ideally, the algorithm exits when the embedding error is close to 0 ($E < \varepsilon_E$). Termination also

```
Algorithm PD-MDS

Input data:
  an initial configuration $\mathbf{z}(1)$
  the dissimilarities $\boldsymbol{\Delta}$, weights $\mathbf{W}$, indicators $\mathbf{I}$
Input parameters:
  an objective function $E(\mathbf{z},\boldsymbol{\Delta},\mathbf{W},\mathbf{I})$
  the stopping tolerances $\varepsilon_E$, $\varepsilon_{\Delta E}$, $\varepsilon_{\mathbf{g}}$, $\varepsilon_r$, $T_M$
Output:
  a final point configuration $\mathbf{z}(T)$
  a final embedding error $E_T$

Initialize:
  $t \leftarrow 1;\ s_M \leftarrow 10;\ E_{-1} \leftarrow \infty;\ \mathbf{z} \leftarrow \mathbf{z}(1);$ .......... {3.1}
Loop:
    $E \leftarrow E(\mathbf{z},\boldsymbol{\Delta},\mathbf{W},\mathbf{I});\ \mathbf{g} \leftarrow \nabla E(\mathbf{z},\boldsymbol{\Delta},\mathbf{W},\mathbf{I});$ ....... {3.2}
    $r_M \leftarrow \frac{1}{\|\mathbf{g}\|_\infty} \cdot \tanh\frac{s_M}{2};$ ......................... {3.3}
    Break if
            $E < \varepsilon_E$
        or  $E_{-1} - E < \varepsilon_{\Delta E}$
        or  $\|\mathbf{g}\|_\infty < \varepsilon_{\mathbf{g}}$       } ..................... {3.4}
        or  $r_M < \varepsilon_r$
        or  $t > T_M$;
    $E_{-1} \leftarrow E;$
    $r \leftarrow \mathrm{HypLineSearch}(E(\mathbf{z},\boldsymbol{\Delta},\mathbf{W},\mathbf{I}), -\mathbf{g}, 0, r_M);$ ... {3.5}
    $\forall j \in \{1..n\},\ z_j \leftarrow \frac{-rg_j+z_j}{-rg_j\overline{z_j}+1};$ ................... {3.6}
    $t \leftarrow t+1;$
Return $\mathbf{z}(T) \leftarrow \mathbf{z}$ and $E_T \leftarrow E(\mathbf{z},\boldsymbol{\Delta},\mathbf{W},\mathbf{I})$.
```

Fig. 3: PD-MDS

occurs in the cases when the error decreases too slowly ($E_{-1} - E < \varepsilon_{\Delta E}$), or when the gradient or the stepping parameter become too small ($\|\mathbf{g}\|_\infty < \varepsilon_{\mathbf{g}}$, $r_M < \varepsilon_r$). Finally, T_M, the maximum allowed number of iterations, is used as a guard against infinite looping.

The *line search* subprogram used in {3.5} is described next.

2.4 Approximate hyperbolic line search

An *exact line search* could be used in line {3.5} (Fig. 3) to determine a value for the step size r such that the corresponding new configuration {3.6} achieves a local minimum of the embedding error along the search path with tight tolerance:

$$r \approx \operatorname{argmin}_{r\in[0,r_M]} q(r), \tag{11}$$

where $q(r)$ is the embedding error as a function of r.

However, increasing the precision of this computation is not essential to the convergence performance since the steepest descent search direction is only locally optimal. Further, exact line search can fail to converge to a local minimum even for a second degree polynomial due to finite machine precision [18].

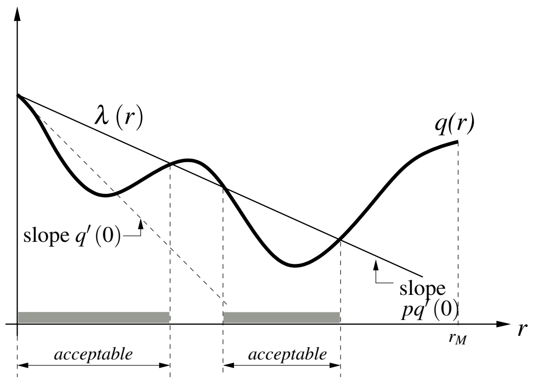

Fig. 4: Acceptable step lengths for inexact line search obtained from the sufficient decrease condition.

On the other hand, *approximate line search* generally provides convergence rates comparable to the exact line search while significantly reducing the computational cost per line search. In fact, the step calculation used in [16] is a "zero-iteration" approximate line search, where the step size is simply guessed based on the first two derivatives of the error. Conceivably, the simplest inexact step calculation would guess the step size based only on the directional gradient at the current configuration.

Approximate line search procedures aim to reduce the computational cost of determining the step parameter by posing weaker conditions on the found solution: Rather than searching for a local or global minimizer of $q(r)$ on $(0, r_M]$, a value is returned by the line search function as satisfactory if it provides sufficient decrease of the objective function and sufficient progress toward the solution configuration. A common approach to defining sufficient decrease is to define the "roof" function

$$\lambda(r) = q(0) + p\cdot q'(0)\cdot r, \quad 0 < p < 1 \tag{12}$$

which is a line passing through $(0, q(0))$ and having a slope which is a fraction of the slope of $q(r)$ at $r = 0$. With this function, we define that sufficient decrease is provided by all values of r such that

$$q(r) < \lambda(r),\ r \in (0, r_M] \tag{13}$$

Fig. 4 shows an example of acceptable step length segments obtained from the sufficient decrease condition (13).

To ensure sufficient progress, we adopt a binary search algorithm motivated by the simple backtracking approach (e.g. [19]). The details are given in Fig. 5.

We start the line search with an initial guess r_0 for the step size parameter, and in the expansion phase {5.1} we double it until it violates the window r_M or the sufficient

```
Procedure HypLineSearch
Input data:
  an initial guess of the step parameter $r_0$
  the maximum step value $r_M$
  the function $q(r)$
Input parameters:
  the slope parameter $p$ for the roof function $\lambda(r)$;
Output:
  an acceptable step parameter $r$

Initialize:
  $r \leftarrow r_0$;
While $r < r_M$ and $q(r) < \lambda(r)$,
    $r \leftarrow 2 \cdot r$; .................... {5.1}
While $r < r_M$ or $q(r) > \lambda(r)$,
    $r \leftarrow r/2$; .................... {5.2}
Return $r$.
```

Fig. 5: Line search procedure for PD-MDS

decrease condition. In the reduction phase $\{5.2\}$, we halve r until it finally satisfies both the window requirement $r < r_M$ and the decrease criterion $q(r) < \lambda(r)$.

We observe that, when started at a point with nonzero gradient, the line search will always return a nonzero value for r. Since the returned acceptable step r is such that the step $2 \cdot r$ is not acceptable, there will be a maximum acceptable point r_m from the same acceptable segment as r, such that $r \leq r_m < 2 \cdot r$, whence $r > r_m/2$. In other words, the returned value is always in the upper half of the interval $[0, r_m]$ and we accept this as sufficient progress toward the solution, thus eliminating some more computationally demanding progress criteria that would require calculation of $q'(r)$ at points other than $r = 0$ or cannot always return a nonzero r [19, 18].

It remains to show how to calculate the slope of $\lambda(r)$, that is $pq'(0)$ (Eq. 12). Given a configuration $\mathbf{z}$ and a direction $-\mathbf{g} = -\nabla E(\mathbf{z}, \boldsymbol{\Delta}, \mathbf{W}, \mathbf{I})$, the configuration $\mathbf{z}'$ as a function of r (7) can be conveniently represented as a column-vector function

$$\mathbf{M}(-r\mathbf{g}, \mathbf{z}) \tag{14}$$

whose j-th entry is the Möbius transform

$$M_j(r) = \frac{-rg_j + z_j}{-rg_j\overline{z_j} + 1}.$$

The associated embedding error as a function of r is then

$$q(r) = E(\mathbf{M}(-r\mathbf{g}, \mathbf{z}), \boldsymbol{\Delta}, \mathbf{W}, \mathbf{I}), \tag{15}$$

and it can be easily shown that its slope is given by

$$\begin{aligned} q'(r) &= \frac{d}{dr} q(r) = \\ &= \left(Re\,\mathbf{M}'(-r\mathbf{g}, \mathbf{z})\right)^T Re\,\nabla E(\mathbf{M}(-r\mathbf{g}, \mathbf{z}), \boldsymbol{\Delta}, \mathbf{W}, \mathbf{I}) \\ &+ \left(Im\,\mathbf{M}'(-r\mathbf{g}, \mathbf{z})\right)^T Im\,\nabla E(\mathbf{M}(-r\mathbf{g}, \mathbf{z}), \boldsymbol{\Delta}, \mathbf{W}, \mathbf{I}) \end{aligned}$$

where the entries of $\mathbf{M}'(-r\mathbf{g}, \mathbf{z})$ are given by

$$M_j'(r) = \frac{d}{dr} M_j(r) = g_j \frac{|z_j|^2 - 1}{(1 - rg_j\overline{z_j})^2}.$$

We thus have a general explicit formula for calculating $q'(r)$ given a configuration $\mathbf{z}$ and the corresponding gradient $\mathbf{g}$ of E at $\mathbf{z}$. In particular, this formula can be used to calculate $pq'(0)$, the slope of $\lambda(r)$.

3 Multidimensional scaling in the PD

3.1 Objective functions and gradients

The iterative minimization method presented in Sec. 2 requires a choice of an embedding error function with continuous first derivatives. In this work we consider the least squares error function

$$E = c \sum_{j=1}^{n} \sum_{k=j+1}^{n} c_{jk} \left(d_{jk} - a\delta_{jk}\right)^2. \tag{16}$$

We note that (16) is a general form from which several special embedding error functions can be obtained by substituting appropriate values of the constants c, c_{jk}, and a. Examples include:

–Absolute Differences Squared (ADS)

$$E = \sum_{j=1}^{n} \sum_{k=j+1}^{n} w_{jk} \left(I_{jk}\left(d_{jk} - a\delta_{jk}\right)\right)^2 \tag{17}$$

–Relative Differences Squared (RDS)

$$E = \sum_{j=1}^{n} \sum_{k=j+1}^{n} w_{jk} \left(I_{jk} \frac{d_{jk} - a\delta_{jk}}{a\delta_{jk}}\right)^2 \tag{18}$$

–Sammon Stress Criterion (SAM)

$$E = \frac{1}{a \sum_{j=1}^{n} \sum_{k=j+1}^{n} I_{jk}\delta_{jk}} \cdot \sum_{j=1}^{n} \sum_{k=j+1}^{n} w_{jk} \frac{\left(I_{jk}\left(d_{jk} - a\delta_{jk}\right)\right)^2}{a\delta_{jk}} \tag{19}$$

As the most general case of (16), individual importance dependent on the input dissimilarities can be assigned to the pairwise error terms using the weights terms w_{jk}.

PD-MDS also requires calculation of the gradient of the error function. For a general error function, closed form symbolic derivatives may or may not exist. In any case, one can resort to approximating the gradient using finite difference calculations. Numerical approximation may also have lower computational and implementation costs than the formal derivatives. However, the use of numerical derivatives can introduce additional convergence problems due to limited machine precision.

For the sum (16), a symbolic derivation of the gradient of (16), including both the Euclidean and hyperbolic cases, can be easily carried out and is omitted here for brevity. From the obtained result, symbolic derivatives of (17)–(19), as well as any other special cases derivable from (16) can be obtained by substituting appropriate constants.

3.2 Local vs. global minima

PD-MDS, being a steepest descent method that terminates at near-zero progress, can find a *stationary point* of the objective function. In the least squares case, if the value at the returned solution is close to zero (that is, $E < \varepsilon_E$), then the final configuration can be considered a global minimizer that embeds the input metric with no error. In all other cases, a single run of PD-MDS cannot distinguish between local and global points of minimum or between a minimizer and a stationary point. A common way of getting closer to the global minimum in MDS is to run the minimization multiple times with different starting configurations. Expectedly, there will be accumulation of the results at several values, and the more values are accumulated at the lowest accumulation point, the better the confidence that the minimal value represents a global minimum i.e. the least achievable embedding error.

Numerous methods that are more likely to find a lower minimum than the simplest repeated descent methods in a single run have been contemplated in the numerical optimization literature. However, to guarantee in general that the global minimizer is found is difficult with any such method. It may be necessary to resort to running the sophisticated methods several times as well in order to gain confidence in the final result. Since these methods are usually computationally more complex or incorporate a larger number of heuristic parameters, the incurred computational and implementational costs often offset the benefits of their sophistication.

3.3 Dissimilarity scaling

The objective functions used in metric *Euclidean* MDS are typically constructed to be *scale-invariant* in the sense that scaling the input dissimilarities and the coordinates of the output configuration with the same constant factor a does not change the embedding error. This is possible for Euclidean space since the Euclidean distance function scales by the same constant factor as the point coordinates:

$$\left(\sum_{s=1}^{L}(a\cdot y_{js}-a\cdot y_{ks})^2\right)^{1/2}=a\cdot d_{jk}.$$

Thus, for example, if d_{jk} is the Euclidean distance, then the sums (18) and (19) are scale-invariant, whereas (17) is not.

However, when d_{jk} is the *hyperbolic* distance function (2), none of the (17)–(19) are scale-invariant. Therefore, the simplest ADS error function (17) may be a preferable choice for reducing the computational cost in the hyperbolic case.

The lack of scale-invariance of the hyperbolic distance formula (2) implies an additional degree of freedom in the optimization of the embedding error – the *dissimilarity scaling factor*. In Eqs. (16)–(19) this extra degree of freedom is captured via the parameter a that scales the original entries of the dissimilarity matrix.

4 Numerical results

4.1 A synthetic example

To illustrate the functioning of PD-MDS, we provide an example random configuration consisting of seven points in the Poincaré disk.

To carry out this experiment, we populate the input dissimilarity matrix with the hyperbolic inter-point distances and start PD-MDS from another randomly-generated seven point initial configuration in the PD. Fig. 6 shows the trajectories traveled by the points during the minimization. The clear points denote the initial configuration, whereas the solid ones represent the final point configuration.

The operation of the PD-MDS algorithm as it iterates over the provided example configuration is examined in detail in Fig. 7. The figure shows the PD-MDS internal parameters vs. the iteration number: In Fig. 7a, the embedding error E monotonically decreases with every iteration; the iterations terminate at the fulfillment of $E < \varepsilon_E = 10^{-6}$, which means that likely the output configuration represents the global minimum and the final inter-point distances match the input dissimilarities very closely. The step-size parameter r is initialized with a value of 1 and assumes only values of the form 2^k, for integral k (Fig. 7b).

The exponential character of the change of r in accord with $\{5.1\}$ and $\{5.2\}$ (Fig. 5) ensures the low computational cost of the line search subprogram.

The refining of the step size as the current configuration approaches a local minimum of the error function, on the other hand, is achieved by the decrease of

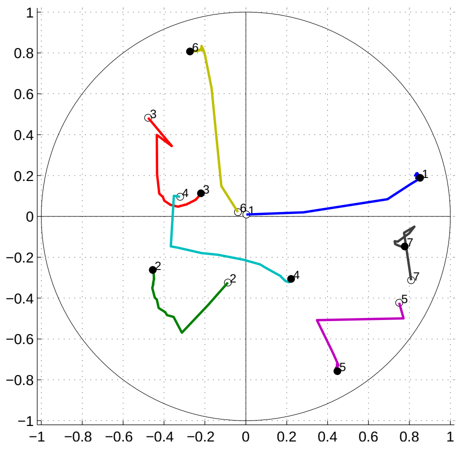

Fig. 6: The minimization trajectory for a seven point configuration using PD-MDS. The clear and the solid points are respectively the initial and the final point configuration.

the gradient norm. This is further illustrated in Figs. 7c and 7d.

In our pool of numerical experiments, we produced graphs similar to those shown in Fig. 7 while using two other line search strategies: (i) exact search and (ii) line search using an adaptive approximate step-size parameter. Both of these strategies showed slower convergence compared to the binary hyperbolic line search, and were of higher computational cost.

4.2 Scaling of the Iris dataset in the PD

As a first experiment on real-world data, we apply PD-MDS to the Iris dataset [20]. This classical dataset consists of 150 4-dimensional points from which we extract the Euclidean inter-point distances and use them as input dissimilarities. The embedding error as a function of the scaling factor a is shown in Fig. 8. Each value in the diagram is obtained as a minimum embedding error in a series of 100 replicates starting from randomly chosen initial configurations.

Minimal embedding error overall is achieved for $a \approx 4$. The improvement with respect to the 2-dimensional Euclidean case is 10%. Thus, the Iris dataset is an example of dimensionality reduction of an original higher-dimensional dataset that can be done more successfully using the PD model.

5 Conclusion

In this paper, we elaborated the details of PD-MDS, an iterative minimization method for metric multidimensional scaling of dissimilarity data in the Poincaré disk model of the hyperbolic plane. While our exposition concentrated on a simple steepest descent minimization with approximate binary hyperbolic line search, we believe that elements of the presented material will also be useful as a general recipe for transferring

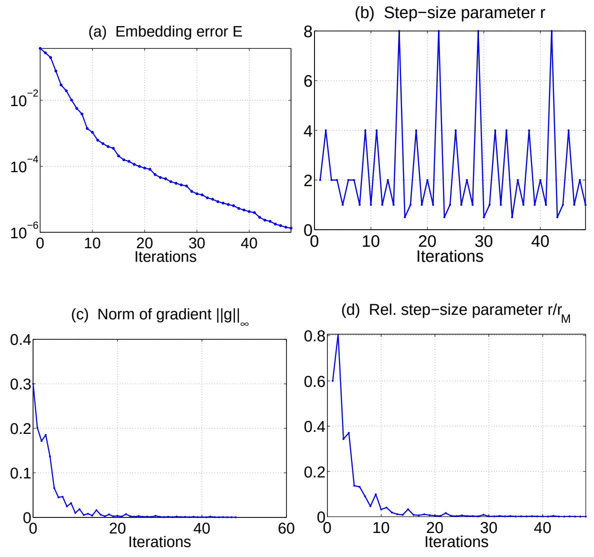

Fig. 7: The PD-MDS internal parameters vs. the iteration number for the seven point example of Fig. 6: (a) the embedding error E, (b) the step-size parameter r, (c) the norm of the gradient $\|\mathbf{g}\|_\infty$, and (d) the step-size parameter relative to the maximum allowed value r/r_M.

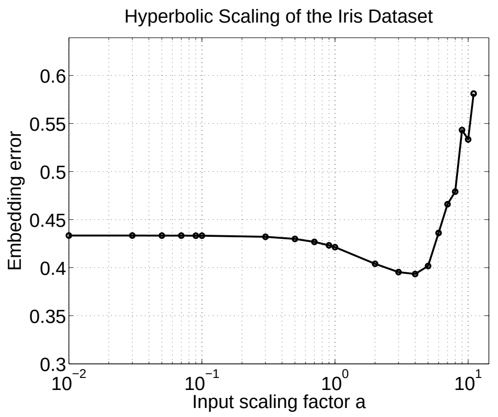

Fig. 8: The effect of scaling of the dissimilarities on the embedding error for the Iris Dataset [20]. The input dissimilarities are the Euclidean distances between pairs of original points. This PD-MDS result reveals that the Iris dataset is better suited for embedding to the hyperbolic plane that to the Euclidean plane.

other, more sophisticated iterative methods of unconstrained optimization to various models of the hyperbolic space.

Acknowledgement

The authors acknowledge the National Science Foundation for supporting this work under Grant no. CNS-1018266.

References

[1] T. F. Cox and M. A. A. Cox. *Multidimensional Scaling (Monographs on Statistics and Applied Probability)*. Chapman & Hall/CRC, 2nd edition, 2000.
[2] I. Borg and P. J. F. Groenen. *Modern Multidimensional Scaling: Theory and Applications (Springer Series in Statistics)*. Springer, Berlin, 2nd edition, 2005.
[3] T. F. Cox and M. A. A. Cox. Multidimensional scaling on a sphere. *Communications in Statistics*, 20:2943–2953, 1991.
[4] P. M. Bentler and D. G. Weeks. Restricted multidimensional scaling models. *J. Math. Psychol.*, 17:138–151, 1978.
[5] B. Bloxom. Constrained multidimensional scaling in N-spaces. *Psychometrika*, 43:397–408, 1978.
[6] H. Lindman and T. Caelli. Constant curvature Riemannian scaling. *J. Math. Psychol.*, 2:89–109, 1978.
[7] J. D. Carroll and P. Arabie. Multidimensional scaling. *Ann. Rev. Psychol.*, 31:607–649, 1980.
[8] J. De Leeuw and W. Heiser. Theory of multidimensional scaling. In P. R. Krishnaiah and L. N. Kanal, editors, *Handbook of statistics*, volume 2. North-Holland, 1982.
[9] J. A. Walter. H-MDS: a new approach for interactive visualization with multidimensional scaling in the hyperbolic space. *Information Systems*, 29(4):273 – 292, 2004.
[10] J. Lamping and R. Rao. Laying out and visualizing large trees using a hyperbolic space. In *UIST '94: Proceedings of the 7th annual ACM symposium on User interface software and technology*, pages 13–14, New York, NY, USA, 1994. ACM.
[11] Y. Shavitt and T. Tankel. Hyperbolic embedding of Internet graph for distance estimation and overlay construction. *IEEE/ACM Trans. Netw.*, 16(1):25–36, Feb. 2008.
[12] R. Kleinberg. Geographic routing using hyperbolic space. In *Proceedings of IEEE Infocom 2007*, pages 1902–1909, May 2007.
[13] D. Krioukov, F. Papadopoulos, M. Boguñá, and A. Vahdat. Greedy forwarding in scale-free networks embedded in hyperbolic metric spaces. *SIGMETRICS Perform. Eval. Rev.*, 37:15–17, October 2009.
[14] A. Cvetkovski and M. Crovella. Hyperbolic embedding and routing for dynamic graphs. In *Proceedings of IEEE Infocom 2009*, pages 1647–1655, Apr 2009.
[15] F. Papadopoulos, D. Krioukov, M. Boguñá, and A. Vahdat. Greedy forwarding in dynamic scale-free networks embedded in hyperbolic metric spaces. In *INFOCOM, 2010 Proceedings IEEE*, March 2010.
[16] J. W. Sammon. A nonlinear mapping for data structure analysis. *IEEE Trans. Comput.*, C-18(5):401–409, May 1969.
[17] J. W. Anderson. *Hyperbolic Geometry*. Springer, 2nd edition, 2007.
[18] P. E. Frandsen, K. Jonasson, H. B. Nielsen, and O. Tingleff. *Unconstrained Optimization*. IMM, DTU, 3rd edition, 2004.
[19] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 1999.
[20] E. Anderson. The irises of the Gaspé peninsula. *Bulletin of the American Iris Society*, 59:2–5, 1935.

Andrej Cvetkovski is Assistant Professor at the School of Informatics, European University, Republic of Macedonia. He holds a Dipl. Ing. degree from Ss. Cyril and Methodius University, an MSEE degree from the Polytechnic Institute of New York University, and a PhD degreee in Computer Science from Boston University. His current research interests are in the area of algorithms applicable to communication and networked systems.

Mark Crovella is Professor and Chair of the Department of Computer Science at Boston University. He holds a BS degree from Cornell University, an MS degree from the State University of New York at Buffalo, and a PhD degreee in Computer Science from the University of Rochester. His research interests are in parallel and networked computer systems, mainly through the application of data mining, statistics, and performance evaluation.