IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 4, 2016, accepted November 28, 2016, date of publication December 15, 2016, date of current version February 25, 2017.

Digital Object Identifier 10.1109/ACCESS.2016.2639065

DOTmark – A Benchmark for Discrete Optimal Transport

JÖRN SCHRIEBER, DOMINIC SCHUHMACHER, AND CARSTEN GOTTSCHLICH
Institute for Mathematical Stochastics, University of Göttingen, 37077 Göttingen, Germany

Corresponding author: J. Schrieber (schrieber@math.uni-goettingen.de)

The work of J. Schrieber was supported by the DFG Research Training Group 2088 Discovering Structure in Complex Data: Statistics Meets Optimization and Inverse Problems.

ABSTRACT The Wasserstein metric or earth mover's distance is a useful tool in statistics, computer science and engineering with many applications to biological or medical imaging, among others. Especially in the light of increasingly complex data, the computation of these distances via optimal transport is often the limiting factor. Inspired by this challenge, a variety of new approaches to optimal transport has been proposed in recent years and along with these new methods comes the need for a meaningful comparison. In this paper, we introduce a benchmark for discrete optimal transport, called DOTmark, which is designed to serve as a neutral collection of problems, where discrete optimal transport methods can be tested, compared with one another, and brought to their limits on large-scale instances. It consists of a variety of grayscale images, in various resolutions and classes, such as several types of randomly generated images, classical test images and real data from microscopy. Along with the DOTmark we present a survey and a performance test for a cross section of established methods ranging from more traditional algorithms, such as the transportation simplex, to recently developed approaches, such as the shielding neighborhood method, and including also a comparison with commercial solvers.

INDEX TERMS Optimal transport, benchmark, Wasserstein metric, earth mover's distance.

I. INTRODUCTION

Despite being a classical problem, optimal transport appears in a plethora of modern applications, such as image retrieval [1], phishing web page detection [2], measuring plant color differences [3] or shape matching [4]. Motivated by the availability of ever larger and more complex data, which often can be cast into the form of one or several measures, and the corresponding development of analytical methods for such data, came the need for more efficient ways of computing optimal transportation plans and evaluating their costs. Accordingly, the last few years have seen the advent of many new methods for obtaining or approximating solutions to large transport problems.

While many of these methods undoubtedly mean substantial progress compared to what was available a decade ago, it is nearly impossible from the current literature to figure out how various of these methods compare to one another and which method is most suitable for a given task. This is mainly due to the fact that only for a few of the modern methods user-friendly code is publicly available. What is more, many articles that introduce new methods compare their computational performance only on a restricted set of self-generated ad-hoc examples and typically demonstrate improved performance only in comparison to some classical method or to a simplified version that does not use the novelty introduced.

The purpose of the present article is twofold. First, we propose a collection of real and simulated images, the DOTmark, that is designed to span a wide range of different mass distributions and serves as a benchmark for testing optimal transport algorithms. To the best of our knowledge this is the first initiative to establish a substantial benchmark of this type that is publicly available. The data can be downloaded at www.stochastik.math.uni-goettingen.de/DOTmark/. We invite other researchers to use this benchmark, report their results, and thus help building a more transparent picture of the suitability of different methods for various tasks.

The second purpose is to provide a survey and a time performance test based on the DOTmark for a cross section of established methods. Since not much code is freely available, we have used previous implementations of our own

2169-3536 © 2016 IEEE. Translations and content mining are permitted for academic research only.
Personal use is also permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

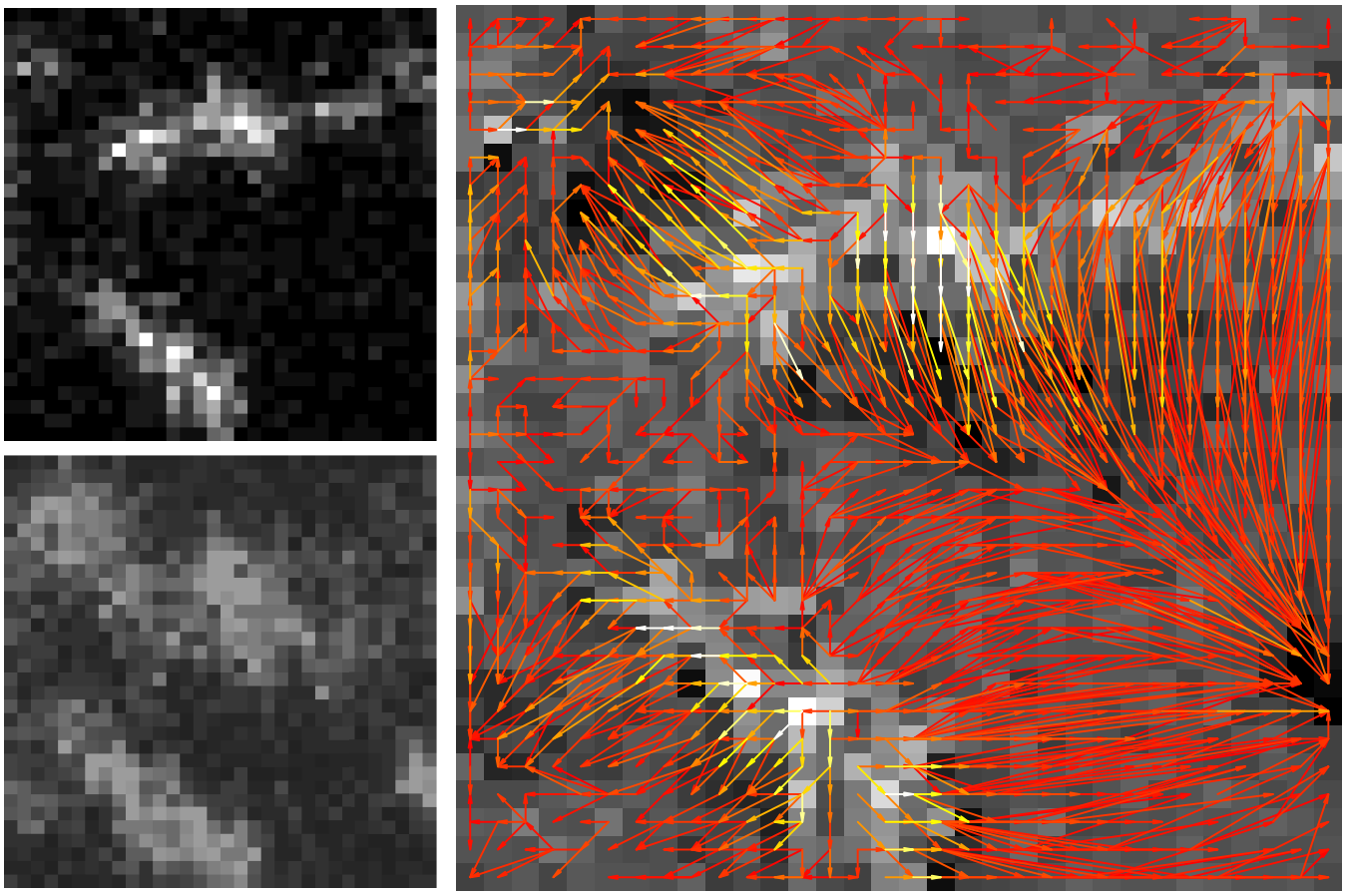

FIGURE 1. **Left panels: Two tiny clippings *A* (top) and *B* (bottom) from STED microscopy images of mitochondrial networks. Right panel: The difference *A* − *B* of the first two panels with an optimal transference plan for *p* = 2 superimposed. Arrows show from where to where mass is transported in the optimal transference plan. The colors indicate the amount of mass from dark red (small) to bright yellow (large). Since mass from individual sites is split (indicated by several arrows leaving the site), this transference plan cannot be represented by a transport map.**

(done to the best of our knowledge) of various methods, added an implementation of the recently proposed shielding neighborhood method [5] and let them compete against each other. This also allows us to draw conclusions about the behavior of different methods on different types of input data. In order to make this comparison as meaningful as possible, we restricted ourselves to using only singlescale methods and the squared Euclidean distance as a cost function. We hope this comparison will provide a first spark for a healthy competition of various methods in the public discussion.

II. BRIEF THEORETICAL BACKGROUND

For the present context it is sufficient to consider optimal transport on $\mathbb{R}^d$. Let X, Y be subsets of $\mathbb{R}^d$ and let μ and ν be probability measures on X and Y, respectively. In this paper it is always assumed that $X = Y$, but using different notation for domain and target space makes definitions easier to grasp.

A *transport map* T is any (measurable) map $T : X \to Y$ that transforms the measure μ into the measure ν. More precisely it satisfies $\mu(T^{-1}(B)) = \nu(B)$ for every measurable $B \subset Y$. A *transference plan* is a measure π on $X \times Y$ with marginals $\pi(\cdot \times Y) = \mu$ and $\pi(X \times \cdot) = \nu$. The set of transference plans from μ to ν is denoted by $\Pi(\mu, \nu)$. Any transport map T from μ to ν defines a transference plan π_T from μ to ν as the unique measure satisfying $\pi_T(A \times B) = \mu(A \cap T^{-1}(B))$ for all measurable $A \subset X$ and $B \subset Y$. Not every transference plan π can be represented in this way, because transference plans allow mass from one site $x \in X$ to be split between multiple destinations, which is not possible under a transport map. Figure 1 shows such an example.

Assuming that the cost of transporting a unit mass from $x \in X$ to $y \in Y$ is $c_p(x, y) = \|x - y\|^p$ for some $p \geq 1$, the minimum cost for transferring μ to ν is then given by

$$C_p(\mu, \nu) = \min_{\pi \in \Pi(\mu,\nu)} \int_{X \times Y} \|x - y\|^p \, d\pi(x, y). \qquad (1)$$

Taking the p-th root, we obtain the *Wasserstein metric* W_p. More precisely we have

$$W_p(\mu, \nu) = C_p(\mu, \nu)^{1/p}$$

for any measures μ and ν that satisfy $\int_X \|x\|^p \, d\mu(x) < \infty$ and $\int_Y \|y\|^p \, d\nu(y) < \infty$. In order to evaluate the Wasserstein metric, we need to find an optimal solution to (1), i.e., a minimizing transference plan π. This problem is often referred

to as the Kantorovich formulation of optimal transport. Note that by [6, Th. 4.1] a minimizing π always exists. However, it neither has to be unique nor representable in terms of an optimal transport map.

Often one would like to compare data sets that are available as images from a certain source, e.g. real photography, astronomical imagery, or microscopy data. We may think of such images as discrete measures on a grid. For example, the first two panels in Figure 1 show tiny clippings from STED microscopy images of mitochondrial networks. A question of interest might be whether both images stem from the same part of the network, which can in principle be answered by finding an optimal transference plan (third panel in Figure 1) and computing the Wasserstein distance. Note that this coarse resolution is not representative for a serious analysis, but was only chosen for illustrative purposes.

Even if the measures we would like to consider are more general probability measures, we can always approximate them (weakly) by a discrete probability measure, e.g. by considering the empirical distribution of a sample from the general measure or based on a more sophisticated quantization scheme. In most practical settings, e.g. if X and Y are bounded subsets of $\mathbb{R}^d$ as in the present article, both the optimal cost and the optimal tranference plans are then approximated by the corresponding objects based on the discretized measures. For general conditions and further details see [7, Lemma 8.3] for approximation of optimal costs and [6, Th. 5.20] and the subsequent discussion for approximation of optimal transference plans.

Assume now that we have discrete measures of the form $\mu = \sum_{i=1}^m \mu_i \delta_{x_i}$ and $\nu = \sum_{j=1}^n \nu_j \delta_{y_j}$, where δ_x is the Dirac mass at point x, and write $c_{ij} = \|x_i - y_j\|^p$. In what follows, we always have $m = n = r^2$, and $(x_i)_{1\le i\le m} = (y_j)_{1\le j\le n}$ form a regular square grid of resolution $r \times r$ in $\mathbb{R}^2$, but since it is more intuitive, we keep different notation for source locations and target locations. Let π_{ij} be the amount of mass transported from x_i to y_j. Then, the problem (1) can be rewritten as a linear program:

$$\begin{aligned} \text{OT} \qquad \min \quad & \sum_{i=1}^m \sum_{j=1}^n c_{ij}\pi_{ij} \\ \text{subject to} \quad & \sum_{j=1}^n \pi_{ij} = \mu_i \ \forall i = 1, \dots, m \\ & \sum_{i=1}^m \pi_{ij} = \nu_j \ \forall j = 1, \dots, n \\ & \pi_{ij} \ge 0. \end{aligned}$$

This is the classic transportation problem from linear programming. Efficient ways of solving this problem for small to medium sized (m and) n, up to a couple of hundred, have been known since the middle of the last century. However, in the context of modern optimal transport problems it has become necessary to solve such problems efficiently at a scale where (m and) n are many thousands or even tens of thousands and more. Currently, this cannot be done with the classical algorithms and requires to utilize the geometry of the problem in one way or the other.

III. BENCHMARK

Our philosophy in compiling this benchmark was to represent a wide range of theoretically different structures, while incorporating typical images that are used in praxis and/or have been used for previous performance tests in the literature. We refer to it as DOTmark, where DOT stands for discrete optimal transport.

The benchmark consists of 10 classes of 10 different images (in what follows sometimes called mass distributions or measures), each of which is available at the 5 different resolutions from 32×32 to 512×512 (in doubling steps per dimension). This allows for a total of 45 computations of Wasserstein distances between two images for any one class at any fixed resolution. Table 1 gives an overview of how the classes were created. Classes 1–7 are random simulations of scenarios based on various probability distributions. Images at different resolutions are generated independently from each other but according to the same laws. Classes 8–10 were obtained by ad-hoc choices of simple geometric shapes, classic test images and images of mitochondria acquired using STED super-resolution microscopy [8]–[10]. For geometric shapes and classic test images the various resolutions available are coarsenings of a single image. For the microscopy images different clippings of various sizes have been selected from larger images to obtain the various resolutions.

TABLE 1. The 10 classes in the DOTmark with details about their creation. GRF stands for Gaussian random field. For technical details and the meaning of the parameters see text.

#	Name	Description
1	WhiteNoise	i.i.d. uniformly distributed values in $[0, 1]$ at each pixel
2	GRFrough	GRF with $\sigma^2 = 1$, $\nu = 0.25$, $\gamma = 0.05$
3	GRFmoderate	GRF with $\sigma^2 = 1$, $\nu = 1$, $\gamma = 0.15$
4	GRFsmooth	GRF with $\sigma^2 = 1$, $\nu = 2.5$, $\gamma = 0.3$
5	LogGRF	exp-function of a GRF with $\sigma^2 = 1$, $\nu = 0.5$, $\gamma = 0.4$
6	LogitGRF	logistic function of a GRF with $\sigma^2 = 4$, $\nu = 4.5$, $\gamma = 0.1$
7	CauchyDensity	bivariate Cauchy density with random center and a varying scale ellipse
8	Shapes	an ad-hoc choice of simple geometric shapes
9	ClassicImages	standard grayscale test images used in image processing
10	Microscopy	clippings from STED microscopy images of mitochondria

We shifted and scaled the pixel values for all classes and randomly redistributed a small percentage of the mass in order to achieve non-negative integer values at each pixel with an average of 10^5. We chose integer values to make the benchmark (directly) accessible to a wide range of algorithms and to be able to verify correctness of the optimal transport

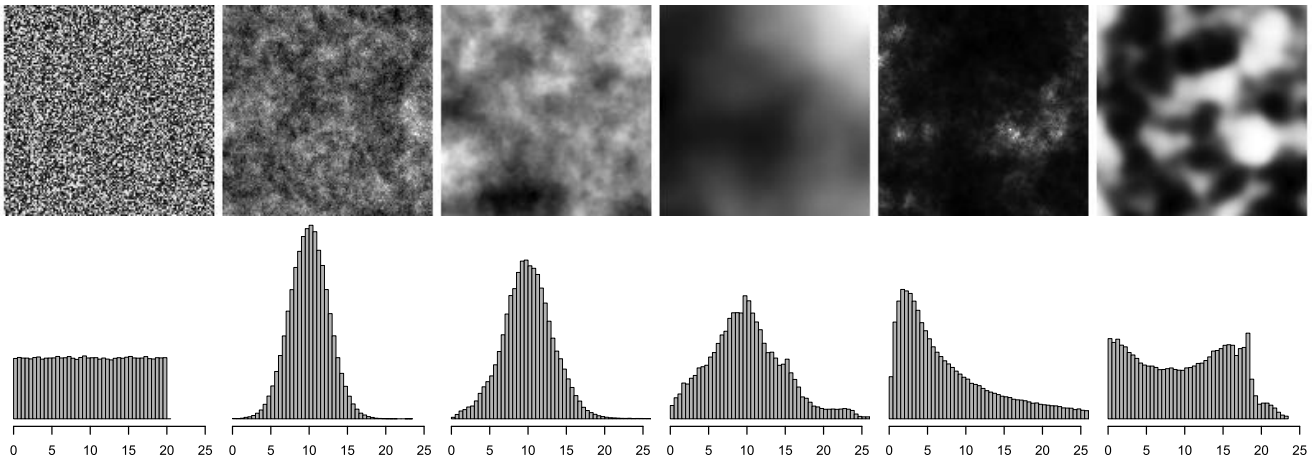

FIGURE 2. *Top row:* **One representative at resolution 128 × 128 for each of the completely randomly generated classes 1–6.** *Bottom row:* **Average histograms of all images at resolution 128 × 128 in classes 1–6. The *x*-axis gives the pixel values in multiples of 10^4, the lengths of the bars give the relative frequencies (total area of the bars is 1). All histograms are on the same scale, but the second from the right is truncated, having full domain of $[0, 198] \times 10^4$.**

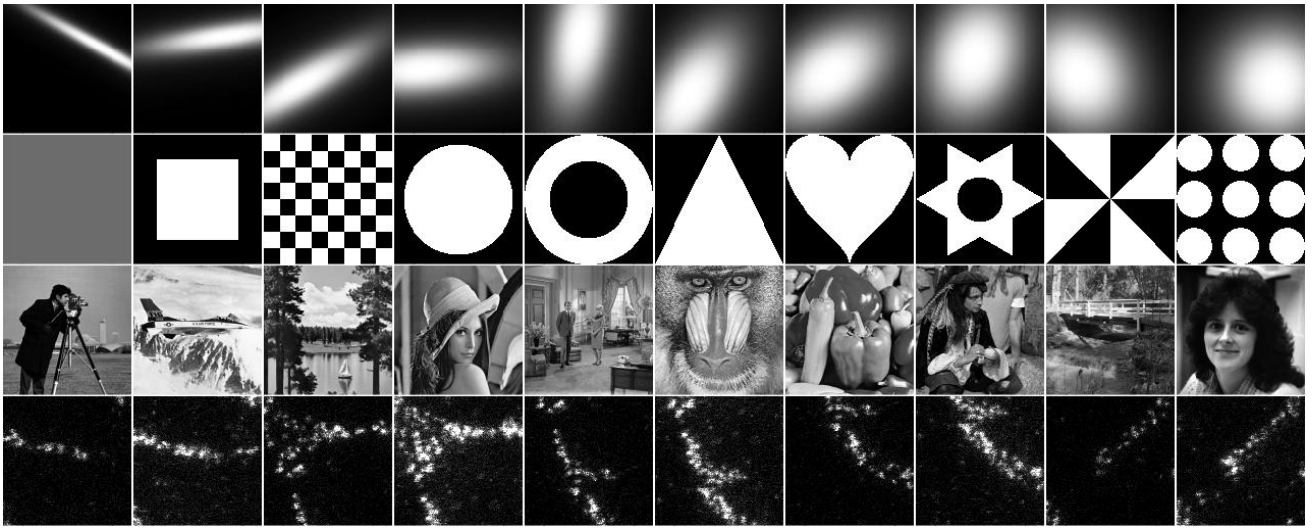

FIGURE 3. **The images in the classes 7–10 at resolution 128 × 128.**

cost precisely, at least in the case $p = 2$, where integer transportation costs between grid points may be assumed.

Figure 2 shows the first image of each of the classes 1–6 along with average histogram over all members of the class. Figure 3 shows the complete collection of images in classes 7–10.

We provide some more details on how classes 2–6 are generated. The images are simulated from stationary centered Gaussian random fields (GRF) on $[0, 1]^2$ with Matérn covariance function $k := k_{\sigma^2,\nu,\gamma} : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$,

$$k_{\sigma^2,\nu,\gamma}(x, y) = \sigma^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu}\frac{\|x-y\|}{\gamma}\right)^{\nu} K_\nu\left(\sqrt{2\nu}\|x-y\|/\gamma\right),$$

where K_ν is the modified Bessel function of the second kind of order ν. In brief this means that the pixel values are distributed according to a multivariate normal distribution with mean vector zero and covariance matrix $(k(x_i, x_j))_{1\le i,j\le m}$, where x_i, $1 \le i \le m$, is an enumeration of the pixel centers. The Matérn covariance function is a popular choice in spatial statistics. Its significance comes from the fact that in addition to having parameters $\sigma^2 > 0$ for the variance and $\gamma > 0$ for the range of the covariance, it also has a parameter $\nu > 0$ that allows to control the regularity of the random image created from very rough (ν small) to very smooth (ν large). Accordingly, classes 2–4 go from very rough with short range dependence to quite smooth with long range dependence. Class 5 is rough with long range dependence, which is hard to see from Figure 2 because of the exponential function applied to the pixel values. Class 6 is very smooth with medium range dependence and the logistic function $\psi : \mathbb{R} \to [0, 1]$, $\psi(x) = e^x/(1 + e^x)$ was applied to the pixel values. See [11] for more theoretical details about Gaussian random

fields. Our simulations were performed using the R package `RandomFields`; see [12], [13].

Histograms 4 and 6 in the second row of Figure 2 deviate quite a bit from the theoretical histograms expected due to the rescaling and redistribution of mass that we apply in order to obtain mass distributions that are non-negative, integer-valued, and have an average of 10^5. Also, due to long range dependence and smoothness, histogram 4 is based on a sample of much smaller effective size from the normal distribution than histograms 2 and 3.

On the whole we consider this a reasonable and versatile benchmark for many (planar and typically grid-based) optimal transport algorithms. It covers a wide selection of types of mass distributions whose comparisons are useful for theoretical or practical investigations. Similar types have been considered individually in the literature before. Gottschlich and Schuhmacher [14] have considered a sparse version of the WhiteNoise class. Schmitzer [5] considered sums of randomly scaled and positioned Gaussian densities (sometimes filtered by discontinous masks), which is a somewhat different type of random function generation along the lines of our classes 2–7. Further random or deterministic functions and geometric shapes were considered by Benamou, Carlier, Cuturi, Froese, Nenna, Oberman, Peyré, Ruan and others; see e.g. [15]–[17]. The use of real grayscale images as in class 9, but also color images, is abundant in the computer science literature (e.g. [1], [18]), where optimal transport is typically considered on some feature space. Mérigot [19] illustrated and tested his algorithm on grayscale images directly. Biological imagery has been used in [20] (fingerprints in feature space), [21] (brain MRIs) and elsewhere.

Another example that is frequently used, in particular in the statistics and machine learning literature, are images from the MNIST handwritten digit database (http://yann.lecun.com/exdb/mnist/). Due to the low resolution of these images we do not consider them, but we might include other images of handwritten text in later revisions of the benchmark.

IV. TESTED METHODS

In what follows, we describe the methods that we have tested on the DOTmark. Due to the large number of suggested methods, which unfortunately is not well reflected in the number of user-friendly implementations available, some restrictions had to be made. We chose methods with a good track record, such as the AHA method [19], as well as some new and promising methods, such as the shielding method [5]. In order to make our comparison as meaningful as possible, we abstained from using methods of a purely approximative nature, such as Sinkhorn scaling [22].

For all of the tested methods there exist multiscale versions. Sometimes these are tailor-made, like for the AHA and the shielding methods (see [5], [19]). Sometimes these are just relatively simple but efficient procedures exploiting the fact that all mass distributions considered live on a square grid in $\mathbb{R}^2$. Such a simple strategy may be as follows: First solve the transport problem on coarsened images (e.g. obtained by adding up the pixel values in contiguous squares of four pixels); then refine the obtained transport plan in a suitable way so that a feasible transport plan for the finer images is obtained; finally solve the original (fine) problem using this transport plan as a starting value.

In our experience this simple strategy already results in an improvement by a factor of 2 to 5 in the transportation simplex at resolution 64×64. A more elaborate, efficient, but not entirely rigorous alternative was proposed in [17]. Since the precise variant and implementation of a multiscale method may distort competition and distract from the merit of an algorithm as such, we decided not to use *any* multiscale methods for this first comparison.

A. TRANSPORTATION SIMPLEX

One of the classical optimal transport methods we test in this paper is the transportation simplex, sometimes also referred to as the revised simplex. It is a specialized version of the network simplex and described in detail for example in [23]. Like other simplex variants, the transportation simplex has two phases: one phase to construct an initial basic feasible solution and another phase to improve this solution to optimality. Typically, the majority of time is spent in the second phase, as an initial solution to optimal transport is easily obtained.

There are quite a few different ways to construct an initial basic feasible solution. For a selection, see [14]. In the present test the modified row minimum rule is used, which has a universally solid performance both in runtime and quality of the solution constructed. We iterate over all source locations (rows) $x_i \in X$ that still have mass left and choose for each source the available transport π_{ij} with the least cost and include it with the maximal amount possible in our solution. This process is repeated until all sources are depleted. The resulting solution is automatically basic.

If the source and target locations are interpreted as nodes in a graph and arcs are inserted for every possible transport, this yields a complete bipartite graph. Every (non-degenerate) basic feasible solution can now be represented by a spanning tree in this graph by choosing all the arcs belonging to active transports in the solution. Given a basic feasible solution, a simplex step is performed as follows:

- A new variable (transport π_{ij}) is selected to enter the basis.
- This creates a cycle in the previous tree, which is then identified.
- The maximal amount of mass possible is shifted along this cycle, i.e., alternately added to and subtracted from consecutive transports.
- One variable that has become zero in the process is removed from the basis.

A row minimum strategy is employed when searching for a new basic variable: Based on the current solution the values of dual variables u_i for each source x_i and v_j for each target y_j are computed. Scanning the non-basic variables π_{ij} row

by row the reduced costs $r_{ij} = c_{ij} - u_i - v_j$ are examined. If a variable with negative reduced costs is encountered, the scanning of the current row is completed and the variable with the least reduced cost encountered thus far is chosen as a new basic variable. If no candidates are found among all rows, the current solution is optimal and the algorithm terminates.

The second phase of the transportation simplex is very similar to the network simplex. The only difference is that the network in the optimal transport problem is always a complete bipartite graph. In phase one, however, this structure is very beneficial and allows for easy construction of a feasible solution, whereas in the more general network case the introduction of artificial variables is often necessary. More details on the network simplex can also be found in [23].

B. SHORTLIST METHOD

At its core, the shortlist method is a variant of the transportation simplex. It comes with three parameters:

- A parameter s for the shortlist length.
- Parameters q and l that control how many variables are searched to find a new basic variable.

Before the optimization starts, a shortlist is created for every source, consisting of the s targets with least transport costs, ordered by cost. The basic feasible solution is again constructed by a modified row minimum rule, where the shortlists are prioritized. After that, the solution is improved by simplex steps similar to the transportation simplex, but the search is limited to the shortlists. The lists are scanned, until either l variables with negative reduced costs are found or q percent of the shortlists have been searched. Then the candidate with least reduced costs is chosen to enter the basis. If no improvement can be achieved within the shortlists, the last solution is improved to global optimality by the same simplex steps as in the transportation simplex. For further details, see [14].

We use the default parameters presented in [14]. They were chosen with regard to the problems considered in that paper — a version of the WhiteNoise class with the Euclidean distance as cost function ($p = 1$) and sparse source and target locations picked uniformly at random.

C. SHIELDING NEIGHBORHOOD METHOD

The shielding neighborhood method, or shortcut method, was introduced by Schmitzer [5]. Its main idea is to solve a sequence of sparse (i.e. restricted) optimal transport instances instead of the dense (full) problem. The algorithm is proposed as a multiscale method, but the singlescale variant basically works in the same way:

Starting with a basic feasible solution generated with the modified row minimum rule a *shielding neighborhood* for that solution is constructed as described in [5] for the squared Euclidean distance as cost function. This neighborhood is a small subset $N \subseteq X \times Y$ of the product space and imposes a restricted instance of the problem by only considering transport variables π_{ij} such that $(x_i, y_j) \in N$. Due to the significant reduction in the amount of variables, this instance is much faster to solve.

The idea behind the shielding neighborhood is the so-called *shielding condition*, which ensures that for $(x_i, y_j) \notin N$ a shortcut exists, i.e., a sequence of transports through the neighborhood whose combined cost is not higher than the cost of the direct transport. The algorithm alternates between optimizing the sparse instance of the problem and generating a new shielding neighborhood for the current solution. If a solution is optimal for two successive shielding neighborhoods the algorithm stops. In [5] it is proved that such a solution is always globally optimal.

D. AHA METHOD

The acronym AHA stands for Aurenhammer, Hoffmann and Aronov, who showed in their seminal paper [24] that the transport problem with squared Euclidean cost is equivalent to an unrestricted continuous minimization problem for a certain convex objective function Φ.

Our concrete implementation is largely based on [19], except for the multiscale scheme. The algorithm computes the optimal transport from an absolutely continuous measure μ on $\mathbb{R}^2$ to a discrete measure ν on $\mathbb{R}^2$, a problem sometimes referred to as *semidiscrete optimal transport*. The key observation utilized by the AHA method is that any power diagram (a.k.a. Laguerre tesselation) governed by the support points $y_1, \ldots, y_n$ of ν and arbitrary weights $w_1, \ldots, w_n \in \mathbb{R}$ characterizes an optimal transport plan from μ to *some* measure living on these support points. By minimizing the function Φ in the weights $w_1, \ldots, w_n$ we can find a power diagram that defines an optimal transport to *the correct* measure ν.

Evaluating Φ at a given weight vector $w \in \mathbb{R}^n$ involves computation of the power diagram for w and a rather simple integration procedure over each power cell. The gradient of Φ is accessible, and its i-th component is in fact just the difference between the ν-mass at the i-th support point and the μ-mass transported to this point under the current power diagram. A Hessian of Φ is not accessible. As in [19] the L-BFGS-B algorithm with Moré–Thuente type line search is used. Since this is a continuous optimization method, it typically cannot reach a weight vector w_* where the gradient of Φ is exactly zero (and hence the image measure of μ is exactly ν), but has to stop when its length is still slightly positive. The method thus commits a small controllable error, referred to as *precision error*.

In order to make the algorithm applicable to the fully discrete situation studied in the other algorithms, the first image is turned into an absolutely continuous measure μ by interpreting pixel values as masses uniformly distributed over the squared areas represented by the pixels, rather than centered at grid points. Compared to the other methods this leads to slightly different results, a discrepancy referred to as *blurring error*. Note that the term ''error'' is subjective. We might as well declare that we want to solve the semidiscrete problem, in which case all the other methods commit a ''concentration error''.

E. SOLVERS

As representatives of LP solvers, we used CPLEX (www.ibm.com/software/commerce/optimization/cplex-optimizer/) and Gurobi (www.gurobi.com/products/gurobi-optimizer). For both we modeled the optimal transport problem as an LP and used the default parameters. This is referred to below as CPLEX-Def and Gurobi-Def, respectively. Additionally, we tested the network simplex solver CPLEX provides (CPLEX-NWS). Gurobi does apparently not come with a network solver, but as the Gurobi documentation page for methods (www.gurobi.com/documentation/6.5/refman/method.html) recommends the dual simplex for memory intensive models, we included it in our tests (Gurobi-DS).

V. IMPLEMENTATION AND COMPUTATIONAL RESULTS

Most of the methods described above have been implemented in Java. As recommended in [5], the network simplex solver of CPLEX is used for the internal sparse instances of the shielding neighbourhood method with a warm start of the previously optimal basis. One call of the CPLEX solver for a single sparse instance is referred to as one iteration of the shielding method in the remainder of this paper. All models solved by CPLEX or Gurobi in the test were set up using the Java APIs of the solvers.

Schmitzer published his own code of the method on his website (wwwmath.uni-muenster.de/num/wirth/people/Schmitzer/). But since we already had our own well-working implementation of this method in place, we decided to use the latter in our test. As all the other methods it was implemented to the best of our knowledge. Since the majority of the runtime is occupied by the internal CPLEX solver, we expect our code to have a similar runtime as the code provided by Schmitzer.

Additionally, we tested the shielding method with an adapted implementation of the transportation simplex from Subsection IV-A as internal solver, using the same pivot strategy and simplex step routines as before. As in [5], a tiny mass is added to every pixel, which in some instances can slightly change the optimal transport cost.

For the AHA method, unlike for the other methods, we use an implementation in C with some minimal R overhead. This implementation is available in the R package `transport` [25]. There the construction of power diagrams is reimplemented in C based on ideas from the CGAL Regular_triangulation_2 package and other sources. For the L-BFGS-B algorithm the implementation in the R function `optim` is used.

Based on runtime comparisons between C and Java we performed for other optimal transport algorithms, we would not expect enormous differences between the two programming languages. Also all reasonable effort was made to optimize the two programs and run them under equal conditions and with the same resources. Nevertheless there is no good reason to assume that the runtimes obtained for the AHA method are precisely comparable to those obtained for the methods implemented in Java.

All of our tests were performed using a single core on a Linux server (AMD Opteron Processor 6140 from 2011 with 2.6 GHz). Note that much better absolute runtimes can be achieved when using modern CPU hardware. For many of the algorithms considerable further improvements are possible by multithreaded implementations that use multiple CPU cores simultaneously.

In our experiments we placed the main emphasis on ensuring that the commercial solvers and all of our own implementations were run under the same conditions. In particular, they were all restricted to use only one of 32 available cores, which was realized by the Linux kernel feature cgroups.

Pairing any two of the 10 images in each of the 10 classes gives 45 transport problems (''instances'') per class, yielding 450 instances in total. These were all solved at resolutions 32×32 and 64×64 by each of the described methods using the squared Euclidean metric as cost funtion. All optimal solutions returned were checked for correctness by evaluating and comparing their optimal objective values, except for the transportation simplex variant of the shielding method, which runs on slightly altered instances. The AHA method is the only other procedure, where we cannot expect precisely correct results due to the errors described in Subsection IV-D. These errors are reported in Subsection V-B. All other errors were zero.

A. RUNTIMES

The runtimes of the tests are listed in Table 2 for 32×32 and Table 3 for 64×64, respectively, averaged over all 45 instances in one class. The average over all classes can be found in the bottom row under 'Overall'. The fastest algorithm for each class is highlighted in bold. Additionally, boxplots for four selected methods are given in Figure 4.

As the numbers show, the shielding neighborhood method is clearly the fastest algorithm for 32×32 instances among the methods tested. It takes hardly more than half the time on average compared to the transportation simplex, the shortlist method and the AHA method. The default solvers of CPLEX and Gurobi have particularly long runtimes. It is remarkable, however, that the network simplex solver of CPLEX outperforms the default solvers and the Gurobi dual simplex by a huge margin. The performance is still somewhat worse than our implementations of the transportation simplex and the shortlist method.

At resolution 64×64 we see a similar picture with some exceptions. The shielding method (with CPLEX as internal solver) is even further ahead of most other algorithms, but at the same time the AHA method, which seems to be scaling much better than the linear programming approaches, has gained even more and in fact shows the best times now for many of the classes. However, keep in mind that the results of this method are not exactly correct and the timing varies according to the stopping criterion one applies (see the next subsection).

The CPLEX network solver is with the exception of the classes WhiteNoise and GRFrough quite a bit faster at

TABLE 2. Average runtimes on 32 × 32 instances in seconds. The columns represent the methods tested: transportation simplex (TPS), shortlist method (SHL), shielding neighborhood method with CPLEX (CPX) and TPS as internal solvers, AHA method, CPLEX with default (Def) parameters and the network simplex solver (NWS), as well as Gurobi with default parameters and the dual simplex solver (DS).

Instance Class	TPS	SHL	Shielding		AHA	CPLEX		Gurobi	
			CPX	TPS		Def	NWS	Def	DS
WhiteNoise	1.58	1.38	**0.67**	1.3	3.28	29.8	5.76	8.1	50.5
GRFrough	2.16	1.98	**1.08**	2.3	3.19	43.0	5.94	9.0	50.7
GRFmoderate	3.45	3.80	**1.86**	6.5	3.17	77.3	6.14	21.1	49.8
GRFsmooth	4.23	5.46	**2.66**	10.0	4.39	101.9	6.15	36.0	50.4
LogGRF	5.18	6.40	**3.00**	10.8	6.80	119.9	6.23	49.5	51.7
LogitGRF	4.50	5.26	**2.40**	7.9	8.49	98.4	6.33	31.0	51.7
CauchyDensity	4.46	6.06	**3.62**	12.7	3.76	140.4	6.09	54.1	49.3
Shapes	1.06	1.07	**0.92**	8.9	1.27	8.9	1.22	5.2	9.1
ClassicImages	3.33	3.26	**1.58**	5.2	2.01	68.5	6.05	18.0	49.6
Microscopy	2.34	3.02	**1.66**	9.4	3.14	35.2	2.74	20.7	22.1
Overall	3.23	3.77	**1.94**	7.5	3.95	72.3	5.26	25.3	43.5

TABLE 3. Average runtimes on 64 × 64 instances in seconds. See the caption of Table 2 for details.

Instance Class	TPS	SHL	Shielding		AHA	CPLEX		Gurobi	
			CPX	TPS		Def	NWS	Def	DS
WhiteNoise	74	56	**13**	65	36	2057	174	311	1657
GRFrough	153	110	24	153	**20**	3216	174	473	1659
GRFmoderate	261	306	51	469	**23**	4592	190	1971	1634
GRFsmooth	306	468	80	778	**57**	5621	195	3723	1590
LogGRF	439	531	79	756	**59**	7156	198	4628	1687
LogitGRF	333	362	**69**	552	77	6024	198	2294	1637
CauchyDensity	245	397	97	968	**30**	6336	195	4461	1435
Shapes	73	73	25	893	**12**	885	40	302	219
ClassicImages	218	246	41	418	**18**	6298	189	1551	1546
Microscopy	28	37	26	824	24	352	**18**	179	114
Overall	213	258	51	588	**36**	4254	157	1989	1318

resolution 64 × 64 than our implementations of the transportation simplex and the shortlist methods. In the two classes Shapes and Microscopy, which have many zeros in the images, it is even competitive with the shielding and AHA methods. Overall, it shows the most consistent performance both across the various benchmark classes (if effective size of the problem is taken into account) and within each class; see the last row in Figure 4.

In contrast, the other methods show a much stronger sensitivity with respect to the class considered. Especially for classes 4–7 and to some extent also for class 3, we see higher average computation times and in particular a much wider spread of times including a number of outliers in Figure 4.

B. ERRORS OF THE AHA METHOD

As described in Subsection IV-D, the AHA method does not solve the problems we consider here with full accuracy, but makes a blurring error by interpreting pixel values of the source measure μ as uniformly distributed over small squares and a precision error by tackling a continuous minimization problem which for numerical reasons cannot be solved exactly.

In Table 4 we report the precision error (PE) in terms of the mass in the probability measure μ that is wrongly allocated, as well as the relative Wasserstein error (RWE) made with the AHA method, i.e.

$$\frac{W_2^{\mathrm{AHA}}(\mu,\tilde{\nu}) - W_2^{\mathrm{TSP}}(\mu,\nu)}{W_2^{\mathrm{TSP}}(\mu,\nu)},$$

where AHA and TSP denote the methods used and $\tilde{\nu}$ denotes the second marginal of the transference plan returned by AHA.

TABLE 4. Average precision error (PE) and relative Wasserstein error (RWE) over the ten classes. See text for details.

Class	PE 32	PE 64	RWE 32	RWE 64
1	0.3098e-05	0.3400e-05	0.031224	0.027232
2	0.4261e-05	0.6327e-05	0.018957	0.008058
3	0.9840e-05	1.8471e-05	0.003965	0.001075
4	2.4910e-05	4.0698e-05	0.001303	0.000349
5	4.6378e-05	5.6221e-05	0.000556	0.000230
6	1.6833e-05	3.7144e-05	0.002008	0.000682
7	3.9273e-05	8.2546e-05	0.000988	0.000289
8	1.3435e-05	3.3610e-05	0.003192	0.000994
9	0.7797e-05	1.8311e-05	0.005119	0.001512
10	2.7111e-05	6.1215e-05	0.001243	0.000412
Overall	1.9294e-05	3.5794e-05	0.006855	0.004083

We can see that the precision error is reasonably small. What is more, if we assume that we would have to reroute the wrongly allocated mass roughly by a distance that corresponds to the true Wasserstein distance between μ and ν (this is reasonable in the sense that it is roughly the same order of magnitude as relevant distances measured in

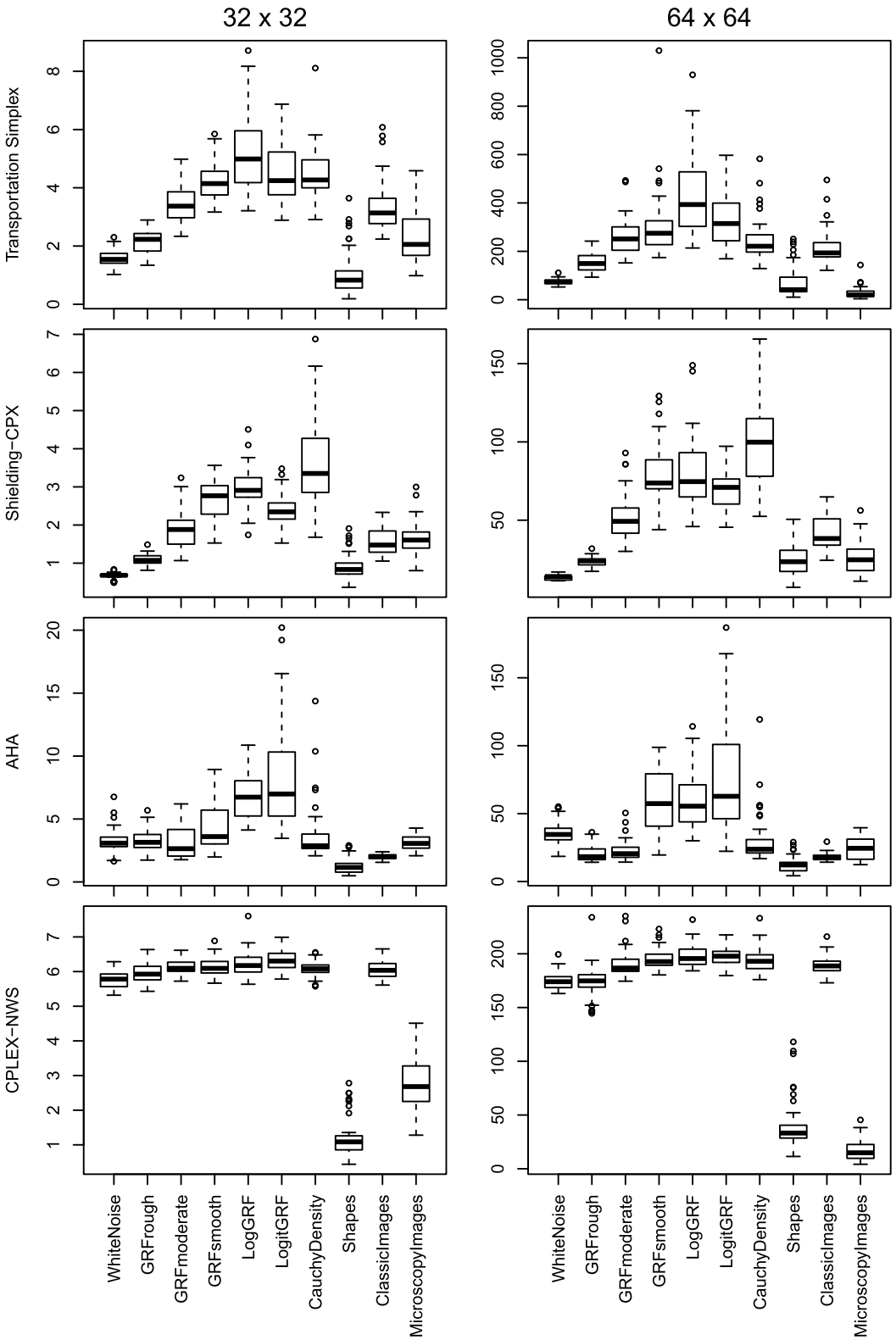

FIGURE 4. **Boxplots of the runtimes of selected methods in seconds. Every box represents 45 computed instances.** ***Left:*** **32 × 32.** ***Right*****: 64 × 64.**

the image), we can compare the precision errors to the relative Wasserstein errors and see that the former play only a minor role. Consequently, the RWE is mainly due to the blurring effect. This is corroborated further by the fact that the RWE for the 64 × 64 resolution is considerably smaller than for 32 × 32.

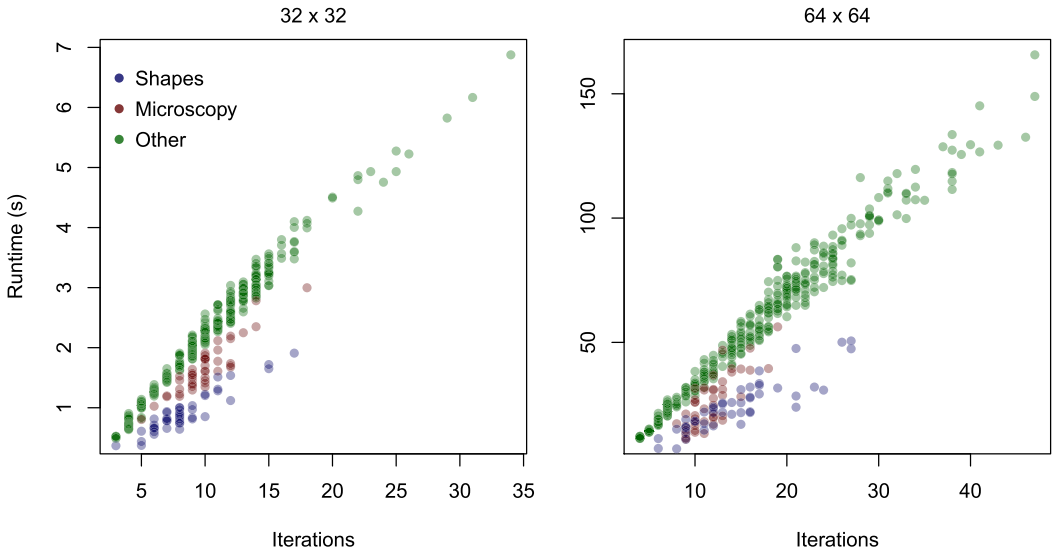

FIGURE 5. Scatterplots for Shielding-CPX showing the runtimes against the number of iterations for the classes Shapes (blue), Microscopy (red) and the other classes combined (green). Every data point represents one of the 450 computed instances. *Left:* 32 × 32. *Right:* 64 × 64.

C. ITERATIONS OF THE SHIELDING METHOD

The shielding method solves the optimal transport problem via a sequence of restricted instances. Here we have a look on how many iterations of these instances are necessary.

On the scale 32×32 the average number of iterations varies between 3.9 for WhiteNoise and 16.6 for CauchyDensity. The numbers for scale 64×64 are higher (4.6 through 28.5), but show similar behavior otherwise. Figure 5 presents scatterplots of the runtime against the number of iterations.

For most classes (green points) we observe a linear scaling of the runtime with the number of iterations. This means that the runtime in each iteration is roughly the same over these classes and we may conclude, since CPLEX has runtimes that scale consistently with model size, that the neighborhood sizes remain more or less constant as well. The only exceptions are the classes Shapes (blue) and Microscopy (red), where the runtimes are lower than expected from the number of iterations. This can be attributed to the internal solver, which benefits from the lower effective dimension that comes from the zero mass pixels occuring in these two classes. The difference is not as significant as for the global CPLEX network simplex runtimes, since the dimension has already been reduced by the construction of the shielding neighborhoods.

VI. DISCUSSION

Looking at the different classes, we note that the solvers perform much better on the classes where the number of pixels with mass zero is large (Microscopy and Shapes). This is because they seem to benefit particularly from the reduction of the effective dimension of the problem. The runtimes are very consistent across the other classes, which allows the conclusion that the solvers can only exploit the mathematical structure of the model, but are unable to use geometric features of the input data to their advantage.

Other linear programming methods benefit from the lower effective dimension as well, although the difference is not as significant. However, these methods seem to be comparatively faster on classes where most of the transports are rather short (rough structure, such as GRFrough or WhiteNoise), and slower on classes with longer transports (smooth structure, such as GRFsmooth or LogGRF). This can be explained by the initial solution routine, which is shared across many of the tested methods. The greedily selected initial transport plan, which favors short transports, is more likely to be close to optimal in short range classes than in long range classes. The shortlist method, which performs another greedy step in addition when searching for new basis variables within the shortlists, benefits particularly from short transports in the solutions, but not as much as in the sparse examples considered in [14] with the Euclidean metric as cost function.

The runtimes of the AHA method are relatively consistent. They are only considerably shorter for the class Shapes. This may be due to the fact that in these instances there are only a few different mass values in the images and the mass is uniformly distributed over large areas.

Interestingly, a comparison of the transportation simplex and the CPLEX network simplex reveals that the performance of the transportation simplex is better on 32×32 instances, while at resolution 64×64 the opposite is the case. This can be explained by looking at the two methods at hand. Although details of the CPLEX network simplex solver are not known, it is safe to assume that the simplex steps are implemented very efficiently. On the other hand, the transportation simplex has the advantage of an easily obtainable good initial solution, whereas in the network simplex a preceding simplex phase

is necessary. This makes the TPS perform better on smaller instances. On the higher resolution our results suggest that the advantage of a strong initial solution is not as influential as the efficient simplex steps.

Considering the small disparity between the transportation simplex and the CPLEX network simplex runtimes, the difference in performance between the two internal solvers for the shielding neighborhood method is surprisingly large. This is due to the fact that initial solutions to the interior models are available in the shielding method and thus the first phase of the network simplex is not necessary. This is why the initial advantage of the transportation simplex disappears and hence using CPLEX as the internal solver yields much lower runtimes.

Another observation worth mentioning is that the runtimes, and therefore numbers of iterations, of the shielding method for the randomly created classes 1–6 agree very well with the ranges of dependence within the data of the classes. The class with the lowest runtimes, WhiteNoise, has no dependence at all, whereas the classes with long range dependences, GRFsmooth and LogGRF, belong to the more difficult classes for the shielding method to solve. That seems to indicate that constructing small shielding neighborhoods prevents larger updates to the current solution per iteration, which might be required in these instances. This may also contribute to the comparatively long and inconsistent runtimes on the class CauchyDensity.

Based on this first comparison of singlescale methods, we give the following **recommendations** for large discrete transport problems (on grids):

1) If high precision is essential and the IBM CPLEX software is available, use the shielding method in combination with CPLEX.
2) If reasonable precision up to a small controllable error is sufficient, use the AHA method. This is especially advisable if a very high resolution for the mass distribution at the source is required, as this comes for the AHA method at virtually no extra cost.
3) Both methods are not widely available nor typically very efficient for costs other than the squared Euclidean metric. So for other costs direct use of a conventional simplex algorithm or the shortlist method may be preferable.
4) Whenever a solver is used directly, the choice of the software and the choice of the most appropriate function may be crucial. In particular, when using CPLEX, make sure to use the network simplex solver, setting up the input as a network structure. If possible, solve the model with a warm start.

We emphasize that the absolute runtimes given in Tables 2 and 3 should not be taken at face value and that actual computations on modern CPUs are typically much faster. While the relative comparison presented here is justified to the best of our knowledge, it allows only limited conclusions about the performance of multiscale variants and multithreaded implementations of the different methods. Also, some caution regarding the results of the AHA method is advisable due to its implementation in C, rather than Java.

VII. OUTLOOK

By providing this benchmark we hope to improve the comparability of different methods for solving discrete optimal transport problems. Contributions or suggestions for extending the benchmark are welcome. In particular we plan to include data sets concentrated on more general grid structures and especially with irregular support points if there is enough public interest.

The R package `transport` [25] offers user-friendly implementations that are mostly written in C of three of the methods presented here (transportation simplex, shortlist and the AHA method). It will be updated in the near future to include additional state-of-the-art methods.

Solving transport problems exactly for larger images (e.g. with 128, 256 or 512 pixels in each dimension) is still computationally very demanding, even for state-of-the-art methods. Efficient solutions of such large problems could pave the way for a new class of algorithms in image processing. In the area of computer vision and image processing, important applications include image enhancement, denoising, inpainting, feature extraction and compression. In one subdomain of image processing, these challenges are approached by decomposing images into two or three parts [26], e.g. a cartoon component, which contains piecewise constant or piecewise smooth parts, a texture component, which captures oscillating patterns, and a noise component, which contains small scale objects (corresponding to high frequency parts in the Fourier domain). After the decomposition step, the texture component can be utilized for applications such as fingerprint segmentation [27]. Image decompositions are obtained by formulating and solving minimization problems that impose suitable norms on the respective components. The total variation (TV) norm is commonly used for the cartoon component and the G-norm [28] for the texture component. Recent works by Brauer and Lorenz [29] and by Lellmann *et al.* [30] connect the G-norm to solutions of transport problems. Typically, the minimization problems described above are solved iteratively and are computationally expensive. It is conceivable to formulate transport norms for image decomposition, which would require to solve a large transport problem in each iteration. Thus, efficient algorithms for optimal transport are a prerequisite for future research in this direction.

ACKNOWLEDGMENT

The authors would like to thank Björn Bähre for implementing the AHA algorithm as part of a student project. We are also grateful to Stefan Stoldt and Stefan Jakobs (https://jakobs.mpibpc.mpg.de/) for sharing their microscopy images and for their permission to include these in the benchmark. We owe further thanks to two anonymous referees, whose pertinent comments have substantially improved the presentation. The authors gratefully acknowledge support

by the German Research Foundation (DFG) and the Open Access Publication Funds of the University of Göttingen.

REFERENCES

[1] Y. Rubner, C. Tomasi, and L. J. Guibas, "The earth mover's distance as a metric for image retrieval," *Int. J. Comput. Vis.*, vol. 40, no. 2, pp. 99–121, Nov. 2000.
[2] A. Y. Fu, L. Wenyin, and X. Deng, "Detecting phishing Web pages with visual similarity assessment based on earth mover's distance (EMD)," *IEEE Trans. Depend. Sec. Comput.*, vol. 3, no. 4, pp. 301–311, Oct./Dec. 2006.
[3] D. Kendal, C. E. Hauser, G. E. Garrard, S. Jellinek, K. M. Giljohann, and J. L. Moore, "Quantifying plant colour and colour difference as perceived by humans using digital images," *PLoS ONE*, vol. 8, no. 8, p. e72296, Aug. 2013.
[4] K. Grauman and T. Darrell, "Fast contour matching using approximate earth mover's distance," in *Proc. CVPR*, Washington, DC, USA, Jun. 2004, pp. 220–227.
[5] B. Schmitzer, "A sparse multiscale algorithm for dense optimal transport," *J. Math. Imag. Vis.*, vol. 56, no. 2, pp. 238–259, 2016.
[6] C. Villani, *Optimal Transport: Old and New* (Grundlehren der Mathematischen Wissenschaften), vol. 338. Berlin, Germany: Springer-Verlag, 2009.
[7] P. J. Bickel and D. A. Freedman, "Some asymptotic theory for the bootstrap," *Ann. Statist.*, vol. 9, no. 6, pp. 1196–1217, 1981.
[8] P. Ilgen *et al.*, "STED super-resolution microscopy of clinical paraffin-embedded human rectal cancer tissue," *PLoS ONE*, vol. 9, no. 7, p. e101563, Jul. 2014.
[9] D. C. Jans *et al.*, "STED super-resolution microscopy reveals an array of MINOS clusters along human mitochondria," *Proc. Nat. Acad. Sci. USA*, vol. 110, no. 22, pp. 8936–8941, May 2013.
[10] C. A. Wurm *et al.*, "Nanoscale distribution of mitochondrial import receptor Tom20 is adjusted to cellular conditions and exhibits an inner-cellular gradient," *Proc. Nat. Acad. Sci. USA*, vol. 108, no. 33, pp. 13546–13551, Aug. 2011.
[11] T. Gneiting and P. Guttorp, "Continuous parameter spatio-temporal processes," in *Handbook of Spatial Statistics*. Boca Raton, FL, USA: CRC Press, 2010, pp. 427–436.
[12] M. Schlather *et al.* (2016). *RandomFields: Simulation and Analysis of Random Fields, R Package Version 3.1.12*. [Online]. Available: https://cran.r-project.org/web/packages/RandomFields/
[13] R Core Team. (2016). R: A language environment for statistical computing, version 3.3.0. R Foundation for Statistical Computing, Vienna, Austria. [Online]. Available: https://www.R-project.org/
[14] C. Gottschlich and D. Schuhmacher, "The shortlist method for fast computation of the earth mover's distance and finding optimal solutions to transportation problems," *PLoS ONE*, vol. 9, no. 10, p. e110214, Oct. 2014.
[15] J.-D. Benamou, B. D. Froese, and A. M. Oberman, "Numerical solution of the optimal transportation problem using the Monge–Ampère equation," *J. Comput. Phys.*, vol. 260, pp. 107–126, Mar. 2014.
[16] J.-D. Benamou, G. Carlier, M. Cuturi, L. Nenna, and G. Peyré, "Iterative Bregman projections for regularized transportation problems," *SIAM J. Sci. Comput.*, vol. 37, no. 2, pp. A1111–A1138, 2015.
[17] A. Oberman and Y. Ruan. "An efficient linear programming method for optimal transportation." [Online]. Available: http://arxiv.org/abs/1509.03668
[18] O. Pele and M. Werman, "Fast and robust earth mover's distances," in *Proc. ICCV*, 2009, pp. 460–467.
[19] Q. Mérigot, "A multiscale approach to optimal transport," *Comput. Graph. Forum*, vol. 30, no. 5, pp. 1583–1592, 2011.
[20] C. Gottschlich and S. Huckemann, "Separating the real from the synthetic: Minutiae histograms as fingerprints of fingerprints," *IET Biometrics*, vol. 3, no. 4, pp. 291–301, Dec. 2014.
[21] S. Gerber and M. Maggioni. "Multiscale strategies for computing optimal transport." [Online]. Available: http://pages.uoregon.edu/sgerber/research/mop.htm
[22] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," in *Proc. NIPS*, 2013, pp. 2292–2300.
[23] D. G. Luenberger and Y. Ye, *Linear and Nonlinear Programming* (International Series in Operations Research & Management Science), vol. 116, 3rd ed. New York, NY, USA: Springer, 2008.
[24] F. Aurenhammer, F. Hoffmann, and B. Aronov, "Minkowski-type theorems and least-squares clustering," *Algorithmica*, vol. 20, no. 1, pp. 61–76, 1998.
[25] D. Schuhmacher, B. Bähre, and C. Gottschlich. (2016). *Transport: Optimal Transport in Various Forms, R Package Version 0.7-4*. [Online]. Available: https://cran.r-project.org/web/packages/transport/
[26] D. Thai and C. Gottschlich, "Directional global three-part image decomposition," *EURASIP J. Image Video Process.*, vol. 2016, no. 12, pp. 1–20, Mar. 2016.
[27] D. Thai and C. Gottschlich, "Global variational method for fingerprint segmentation by three-part decomposition," *IET Biometrics*, vol. 5, no. 2, pp. 120–130, Jun. 2016.
[28] Y. Meyer, *Oscillating Patterns in Image Processing and Nonlinear Evolution Equations*. Boston, MA, USA: AMS, 2001.
[29] C. Brauer and D. Lorenz, "Cartoon-texture-noise decomposition with transport norms," in *Proc. SSVM*, Lège-Cap-Ferret, France, May 2015, pp. 142–153.
[30] J. Lellmann, D. A. Lorenz, C.-B. Schönlieb, and T. Valkonen, "Imaging with Kantorovich–Rubinstein discrepancy," *SIAM J. Imag. Sci.*, vol. 7, no. 4, pp. 2833–2859, Dec. 2014.

JÖRN SCHRIEBER recieved the B.Sc. and M.Sc. degrees in mathematics from the University of Göttingen, in 2013 and 2015, respectively, where he is currently pursuing the Ph.D. degree with the DFG Research Training Group 2088 "Discovering Structure in Complex Data: Statistics Meets Optimization and Inverse Problems."

His research interests include computational and algorithmic aspects of optimal transport and its applications, and optimization, in particular linear programming and robust optimization.

DOMINIC SCHUHMACHER received the Diploma and Ph.D. degrees in mathematics from the University of Zürich, Switzerland, in 2000 and 2005, respectively, and the Habilitation degree in stochastics from the University of Bern, Switzerland, in 2013. He is currently a Full Professor of Stochastics with the University of Göttingen, Germany.

His research interests include spatial statistics, probability metrics, Stein's method for distributional approximation, and computational methods for spatial problems, including optimal transport.

CARSTEN GOTTSCHLICH received the M.Sc. and Ph.D. degrees in computer science from the University of Göttingen, Germany, in 2007 and 2010, respectively.

His research interests include machine learning, pattern recognition, image processing, computer vision, and biometrics, particularly fingerprint recognition.