Transport Model for Feature Extraction*

Wojciech Czaja†, Dong Dong‡, Pierre-Emmanuel Jabin§, and Franck Olivier Ndjakou Njeunje¶

Abstract. We present a new feature extraction method for complex and large datasets, based on the concept of transport operators on graphs. The proposed approach generalizes and extends the many existing data representation methodologies built upon diffusion processes, to a new domain where dynamical systems play a key role. The main advantage of this approach comes from the ability to exploit different relationships than those arising in the context of e.g., graph Laplacians. Fundamental properties of the transport operators are proved. We demonstrate the flexibility of the method by introducing several diverse examples of transformations. We close the paper with a series of computational experiments and applications to the problem of image clustering and classification of hyperspectral data, to illustrate the practical implications of our algorithm and its ability to quantify new aspects of relationships within complicated datasets.

Key words. feature extraction, dimension reduction, machine learning, semi-supervised, transport operator, advection.

AMS subject classifications. 68Q25, 68R10, 68U05

1. Introduction.

Feature extraction has been at the core of many data science applications for more than a century. The goal of feature extraction is to derive new measurements (or features) from an initial set of measure data with the intention of retaining the core information while eliminating redundancies. A well-known feature extraction algorithm is principal components analysis (PCA) which can be traced back to the year 1901 [33]. However, due to the linear nature of PCA, the method falls short in capturing the intrinsic structure of the data when a non-linear relationship governs the underlying structure within the data. Since then, the complex, non-linear, and growing amount of data have led scientists to come up with new techniques. A few well-known techniques are: kernel PCA [37], isomap [42], locally linear embedding (LLE) [35], and Laplacian eigenmaps (LE) [2]. Today, the use of feature extraction techniques varies based on applications from the classification of hyperspectral images [6, 40, 41, 53] to the prediction of stock market prices [54].

The aforementioned non-linear feature extraction methods lead to applications of linear operators, e.g., the Laplacian. In the present study, we have developed a more general approach that constructs non-linear feature extraction algorithms based on non-linear operators, such as appropriately chosen transport by advection operators. A recent technique [23] sought

*Submitted to the editors DATE.

†CSCAMM and Department of Mathematics, University of Maryland, College Park, MD 20742 USA (wojtek@math.umd.edu). WC was partially supported by LTS grants DO 0048-0049-0050-0052 and D00030014.

‡Department of Mathematics, Northwestern University, Evanston, IL 60208 USA (dong@northwestern.edu). DD was partially supported by LTS grant DO 0052.

§ Department of Mathematics and Huck Institutes, Pennsylvania State University, State College, PA 16802 USA (pejabin@psu.edu). PEJ was partially supported by NSF DMS Grant 161453, 1908739, 2049020, NSF Grant RNMS (Ki-Net) 1107444 and by LTS grants DO 0048-0049-0050-0052 and D00030014

¶Department of Mathematics, University of Maryland, College Park, MD 20742 USA (fndjakou@math.umd.edu). FNN was partially supported by LTS grants DO 0048-0049-0050 and 0052.

to find the optimal transport method between two point sets based on an adaptive multiscale decomposition, which itself is derived from diffusion wavelets and diffusion maps. In our work, we focus on the transport operator directed by velocity fields [8, 29, 45], because of its well-studied properties as well as its partial similarity to the Schroedinger Eigenmaps method [19]. This transport model has not been used in the literature as a tool for building a feature extraction algorithm. Nevertheless, some related work can be found in the fields of water resource management and in bio-medical research [28], where feature extraction is used to construct simplified transport models for cardiovascular flow.

At its core, our work will focus on exploring and exploiting the differences and similarities of this novel approach to the state-of-the-art feature extraction algorithms used in clustering and classification tasks. After providing some background in Section 2, we introduce the model in Section 3 together with key properties and the algorithm of the model. We provide applications of the new algorithm for feature extraction and subsequent clustering and classification in Section 4 and Section 5. Some open problems are posed in the last section.

2. Background.

In many data science applications, high dimensional data tend to lie on low dimensional manifolds within the high dimensional space. To take advantage of this information, methods such as the Laplacian eigenmaps (LE) [2] and the Schroedinger eigenmaps (SE) [19], invoke the adjacency graph constructed from a set of initial points, $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ in $\mathbb{R}^d$, in order to extract the most important features from the bunch.

In LE, the first step is to construct a weighted graph based on the distances among given n points. A weight is assigned to each edge connecting two nearby points. Heat kernel is often used as the weight: $w_{ij} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{2\sigma^2}\right)$. To make sure close points stay close after mapping, the problem can be phrased as a minimization problem and then be reduced to solving the generalized eigenvector problem $L\,\mathbf{f} = \lambda\, D\,\mathbf{f}$, where $L = D - W$, viz., the Laplacian matrix, with W representing the (symmetric) weight matrix (w_{ij}) and D the diagonal matrix with entries $d_{ii} = \sum_j w_{ij}$. Let $\{\mathbf{f}_0, \mathbf{f}_1, \ldots, \mathbf{f}_{n-1}\}$ be the solution set written in ascending order according to their eigenvalues. The m-dimensional Euclidean space mapping is given by $\mathbf{x}_i \to [\mathbf{f}_1(i), \mathbf{f}_2(i), \ldots, \mathbf{f}_m(i)]$. See Section 3.4 for the detailed algorithm (just replace T with the Laplacian matrix L).

In SE, the m-dimensional Euclidean space mapping is given in a similar manner. As a generalization of the LE algorithm, SE uses partial knowledge about the data set X and fuses this information into the LE algorithm to obtain better representation or more desirable results. Additional work related to data fusion can be found in the following papers [5, 15, 21, 26]. The problem in SE is reduced to solving the following generalized eigenvector problem, $S\,\mathbf{f} = \lambda\, D\,\mathbf{f}$, where $S = L + \alpha V$, viz., the Schroedinger matrix, with V as the potential matrix encoding the partial information and α as a real parameter keeping the balance between the matrices L and V.

The algorithm we are developing in this article, transport eigenmaps (TE), has some similarities to SE in the sense that both algorithms use extra information about the data set to define a generalization of LE. Unlike supervised learning techniques which assume prior knowledge of the ground truth, SE and TE only assumes partial knowledge of said ground truth. This puts SE and TE in a class of machine learning techniques between supervised learning and unsupervised learning (no prior knowledge) called semi-supervised learning (see

[4, 20, 22, 46, 52] for more examples). While SE uses potentials to encode to additional information, TE may use advection (the active transportation of a distribution by a flow field) or measure/weight modifiers. In contrast to SE, TE could come from a non-linear operator which we will describe in section 3.

3. The transport model.

Transport operators have been used in modeling and analyzing data in a variety of fields [1, 9, 10, 25, 27, 38, 39]. We aim to bring this idea into the graph setting to help with data representation.

3.1. Notation and introduction.

We first briefly present the basic setting for studying transport model on graphs. Fix a weighted simple graph G with n nodes. Let v be a function defined on the edges of G. Such as function can be represented by an $n \times n$ matrix with nonzeros only where there are edges. We will further assume v to be anti-symmetric since it will be used to model a velocity field. We formally define the transport or advection operator in conservative form, acting on a vector $\mathbf{y}$ as

$$T\,\mathbf{y} = L\,\mathbf{y} - \operatorname{div}(v\mathbf{y}), \tag{3.1}$$

which corresponds to the continuous continuity equation.

In the continuous setting, transport or advection operators are easily defined, at least formally, on functions of $\mathbb{R}^d$. Given a velocity field $\mathbf{v} : \ x \in \mathbb{R}^d \to v(x) \in \mathbb{R}^d$, the pure transport is defined similarly as

$$T\,f(x) = \operatorname{div}(\mathbf{v}, f) = \sum_{i=1}^{d} \partial_{x_i}(v_i(x) f(x)).$$

for the conservative form. The solution to the transport equation

$$\partial_t f(t,x) + T\,f = 0$$

is directly connected to the transport along the characteristics or flow of the differential equation

$$\frac{d}{dt} X(t) = \mathbf{v}(X(t)), \quad X(t=0) = x_0.$$

In fact if the initial value x_0 is chosen randomly with the probability density $f_0(x)$ then the solution $f(t,x)$ to the transport equation with the initial condition $f(t=0,x) = f_0(x)$ is the probability density of $X(t)$.

Instead of pure transport, it is also possible to consider advection-diffusion operators, which is what we are doing in the discrete setting below. In that case, one chooses

$$T\,f(x) = \operatorname{div}(\mathbf{v}, f) - w\,\Delta f = \sum_{i=1}^{d} \partial_{x_i}(v_i(x) f(x)) - w \sum_{i=1}^{d} \partial^2_{x_i,x_i} f,$$

where we use here a constant diffusion w. The advection-diffusion equation

$$\partial_t f(t,x) + T\,f = 0$$

is now connected to solutions to Stochastic Differential Equations.

Many key properties of T can be derived based on properties of its continuous analogue. For example, self-adjointness of both T and its continuous analogue requires v to be of the form $\frac{\nabla a}{a}$ (see Section 3.2 below and the supplements for detailed discussions). Therefore it is important to first setup the rules to translate between the continuous and discrete settings. For any matrix A, which is viewed as a function defined on the edges, the divergence of A is a function defined on nodes, i.e., is a vector:

$$\operatorname{div}(A)_i := \sum_j A_{ij}. \tag{3.2}$$

When A models a velocity field on the graph, $\operatorname{div}(A)_i$ is the net flow out of the node i (here j indexes outgoing edges).

For any function f defined on the nodes, its gradient, the dual operator of the divergence, is defined on the edges

$$(\nabla f)_{ij} := (f_j - f_i) w_{ij}. \tag{3.3}$$

A matrix A (e.g., a velocity field) can act on an $f \in \mathbb{R}^n$ (e.g., a probability distribution) in the following way

$$(Af)_{ij} = (fA)_{ij} := \frac{f_i + f_j}{2} A_{ij}.$$

This corresponds to the standard centered discretization of the transport operator (after taking the divergence).

The Laplacian of f, $\Delta f := \operatorname{div}(\nabla f)$, is defined on the nodes:

$$(\Delta f)_i = \sum_j (f_j - f_i) w_{ij}. \tag{3.4}$$

This agrees with the graph Laplacian L up to a sign (recall that the graph Laplacian is positive semi-definite whereas the continuous Laplacian operator is negative semi-definite). See [14] for a comprehensive introduction of the graph Laplacian.

Based on the above rules, we have $(v\mathbf{y})_{ij} = v_{ij} \frac{y_i + y_j}{2}$ and $\operatorname{div}(v\mathbf{y}) = \sum_j (v\mathbf{y})_{ij} = \frac{1}{2} \sum_j (y_i + y_j) v_{ij}$. Therefore, the definition of T (3.1) becomes

$$(T\,\mathbf{y})_i = \sum_j (y_i - y_j) w_{ij} - \sum_j (y_i + y_j) \frac{v_{ij}}{2}. \tag{3.5}$$

It is unclear from the expression (3.5) that a transport operator T would always produce real eigenvalues as the Laplacian and Schroedinger operators do. We will address this issue in the next subsection.

3.2. Self-adjointness. As the properties of the transport operator ultimately depend on v, an anti-symmetric matrix, we aim to find v's so that the corresponding transport operator T is self-adjoint, i.e., $\langle T\mathbf{y}, \mathbf{z} \rangle = \langle \mathbf{y}, T\mathbf{z} \rangle$ for some (possibly non-standard) inner product $\langle , \rangle$. In the supplement, we show that the continuous transport operator $F(y) = \Delta y + \operatorname{div}(\mathbf{v}y)$ is

self-adjoint with respect to the inner product $\langle f, g\rangle_a := \int f(x)\, g(x)\, a(x)\, dx$ associated with a certain function $a(x)$ whenever $\nabla_x a = a\,\mathbf{v}$ or for every coordinate i, $\partial_{x_i} a = a\, v_i$. Although it turns out that our choice of formalism make it that the condition reads the same in the discrete setting, the full similarity stops and solutions for $\mathbf{v}$ in both cases are different.

For any positive definite matrix A, we use $\langle , \rangle_A$ to denote the inner product

$$\langle \mathbf{y},\, \mathbf{z}\rangle_A := \mathbf{y}^t A \mathbf{z}.$$

When A is the identity matrix, this agrees with the standard inner-product.

Set $v_{ij} = 0$ if the nodes i and j are not connected. Let

$$\bar{v}_{ij} := \frac{v_{ij}}{2w_{ij}} \text{ if } i \text{ and } j \text{ are connected,}$$

and $\bar{v}_{ij} = 0$ otherwise. Then $\bar{v}$ is also anti-symmetric, $\frac{v_{ij}}{2} = \bar{v}_{ij} w_{ij}$, and

$$\begin{aligned}(T\,\mathbf{y})_i &= \sum_j (y_i - y_j)\, w_{ij} - \sum_j (y_i + y_j)\, \bar{v}_{ij}\, w_{ij} \\ &= \sum_j [(1 - \bar{v}_{ij})\, y_i - (1 + \bar{v}_{ij})\, y_j]\, w_{ij}\end{aligned}$$

Our goal is to find a suitable flow $\bar{v}$ so that T is self-adjoint with respect to certain inner product $\langle , \rangle_X$. A simple and natural anti-symmetric choice of $\bar{v}$ is of the form $\bar{v}_{ij} = a_j - a_i$, where a_i can be viewed as a positive potential on the node i. This $\bar{v}$ is invariant under translation of a_i. Another modified version is $\bar{v}_{ij} = \frac{a_j - a_i}{a_j + a_i}$, which is invariant under rescaling of a_i. This scaling-invariant property plays a key role in establishing the self-ajointness of T with arbitrary a_i (see the Supplementary Materials for a discussion and comparison of the two choices).

Theorem 3.1. *Let $W = (w_{ij})$ be a symmetric matrix. Assume $\bar{v}_{ij} = \frac{a_j - a_i}{a_j + a_i}$ for some positive a_i's. Then the operator $(T\,\boldsymbol{y})_i = \sum_j [y_i - y_j - \bar{v}_{ij}(y_i + y_j)] w_{ij}$ is self-adjoint with respect to the inner product $\langle , \rangle_X$, with $X = diag(ca_i)$ for some positive c.*

Proof. For the convenience of future discussion, denote $X = \mathrm{diag}(x_i)$ and we try to "solve" for x_i. In general, X could be non-diagonal. We need to verify that for any vectors $\mathbf{y}$ and $\mathbf{z}$

$$\sum_i (T\,\mathbf{y})_i\, z_i\, x_i = \sum_i y_i\, (T\,\mathbf{z})_i\, x_i \tag{3.6}$$

The left-hand-side (LHS) of (3.6) is

$$\begin{aligned}\sum_i (T\,\mathbf{y})_i\, z_i\, x_i &= \sum_{i,j} [(1 - \bar{v}_{ij})\, y_i - (1 + \bar{v}_{ij})\, y_j]\, z_i\, x_i\, w_{ij} \\ &= \sum_{ij} [(1 - \bar{v}_{ij})\, y_i\, z_i\, x_i - (1 - \bar{v}_{ij})\, y_i\, z_j\, x_j]\, w_{ij} \\ &= \sum_i y_i \sum_j [(1 - \bar{v}_{ij})\, z_i\, x_i - (1 - \bar{v}_{ij})\, z_j\, x_j]\, w_{ij}\end{aligned}$$

Compare this with the right-hand-side (RHS) of (3.6)

$$\sum_i y_i\,(T\mathbf{z})_i\,x_i = \sum_i y_i \sum_j [(1-\bar{v}_{ij})\,z_i\,x_i - (1+\bar{v}_{ij})\,z_j\,x_i]\,w_{ij},$$

and we see that in order to make (3.6) hold,

$$(1-\bar{v}_{ij})\,x_j = (1+\bar{v}_{ij})\,x_i \tag{3.7}$$

must be true for any pair of connected nodes i and j.

Now make use of the assumption $\bar{v}_{ij} = \frac{a_j - a_i}{a_j + a_i}$. In this case, the key condition (3.7) becomes

$$\left(1 - \frac{a_j - a_i}{a_j + a_i}\right) x_j = \left(1 + \frac{a_j - a_i}{a_j + a_i}\right) x_i,$$

which is

$$a_i\,x_j = a_j\,x_i.$$

This clearly holds as $x_i = ca_i$ by the assumption of the theorem. ∎

We can immediately extend this theorem to a more general model by introducing a symmetric matrix R. This new collection of parameters will allow us to implement the transport eigenmap method in various settings.

Theorem 3.2. *Let $R = (r_{ij})$ and $W = (w_{ij})$ be symmetric matrices. Define T_v^R to be the operator such that*

$$(T_v^R\,\boldsymbol{y})_i = \sum_j [r_{ij}\,(y_i - y_j) - \bar{v}_{ij}\,(y_i + y_j)]\,w_{ij}. \tag{3.8}$$

Assume $\bar{v}_{ij} = \frac{a_j - a_i}{a_j + a_i}\,r_{ij}$ for some positive a_i's. Then T_v^R is self-adjoint with respect to the inner product $\langle , \rangle_X$, with $X = diag(ca_i)$ for some positive c.

Proof. Simply notice that the symmetric matrix R can be incorporated into the symmetric matrix W and thus the operator T_v^R has the same form as T in Theorem 3.1. ∎

When $\bar{v}_{ij} = \frac{a_j - a_i}{a_j + a_i}\,r_{ij}$, the general transport operator T_v^R can be rewritten as

$$(T_v^R\,\mathbf{y})_i = \sum_j \left(\frac{2a_i}{a_i + a_j}\,y_i - \frac{2a_j}{a_i + a_j}\,y_j\right) w_{ij} = \sum_j (a_i y_i - a_j y_j)\,w_{ij}\,\frac{2r_{ij}}{a_i + a_j}. \tag{3.9}$$

This expression also indicates that T_v^R is non-negative when $\bar{v}_{ij} = \frac{a_j - a_i}{a_j + a_i}\,r_{ij}$.

Theorem 3.3. *The operator defined by (3.9) is non-negative in ℓ^2_X, where $X = diag(ca_i)$ for some positive c. More precisely,*

$$\langle \boldsymbol{y},\, T_v^R\,\boldsymbol{y}\rangle_X = \frac{c}{2}\sum_{i,j} (\tilde{y}_i - \tilde{y}_j)^2\,\tilde{w}_{ij} \geq 0, \tag{3.10}$$

with $\tilde{w}_{ij} := w_{ij}\,\frac{2r_{ij}}{a_i + a_j}$ and $\tilde{y}_i := a_i\,y_i$. In particular, $T_v^R\,\boldsymbol{y} = 0$ iff the quantity $a_i y_i$ is constant on every connected component of the graph.

Proof. By a straightforward computation,

$$\langle \mathbf{y},\, T_v^R\,\mathbf{y}\rangle_X = c\sum_i y_i\, a_i\, (T_v^R\,\mathbf{y})_i = c\sum_{i,j} \tilde{y}_i\,(\tilde{y}_i - \tilde{y}_j)\,\tilde{w}_{ij} = \frac{c}{2}\sum_{i,j}(\tilde{y}_i - \tilde{y}_j)^2\,\tilde{w}_{ij} \geq 0.$$

When $T_v^R\mathbf{y} = 0$, the above expression is 0 and thus $\tilde{y}_i$ must the constant on any connected component. The converse is trivial by (3.9). ∎

The above theorem ensures that T_v^R is diagonalizable, with real-valued and negative eigenvalues. In applications, we will however look for the generalized eigenvectors of T_v^R (see Section 3.4 for the algorithm): eigenvectors that are normalized by the degree on the graph, *i.e.* vectors $\mathbf{u}$ s.t.

$$T_v^R\,\mathbf{u} = \lambda\, D\,\mathbf{u},$$

where D is the degree matrix as before: $d_{ii} = \sum_j w_{ij}$. Equivalently we are looking for the eigenvectors $\mathbf{y}$ of $D^{-1/2}\,T_v^R\,D^{-1/2}$ with $\mathbf{y} = D^{1/2}\,\mathbf{u}$ or $\mathbf{u} = D^{-1/2}\,\mathbf{y}$ and the same generalized eigenvalues. From Theorem 3.3, it is now straightforward to deduce that

Corollary 3.4. *Let T_v^R be given by (3.9) and let D be the degree matrix. Then the operator $D^{-1/2}\,T_v^R\,D^{-1/2}$ is self-adjoint in ℓ^2_X and non-negative, where $X = diag(ca_i)$ for some positive c. Furthermore, $D^{-1/2}\,T_\mu\,D^{-1/2}\,\boldsymbol{u} = 0$ iff $(D^{-1/2}\,\mathbf{u})_i a_i$ is constant on connected components of the graph.*

Proof. D is self-adjoint on ℓ^2_X, simply because D is diagonal and so is the metric provided by $\langle,\rangle_X$. It would be very different if we had to use non-diagonal metric (and we would have to study directly $D^{-1/2}\,T_v^R\,D^{-1/2}$ instead of T_v^R).

The operator $D^{-1/2}\,T_v^R\,D^{-1/2}$ is still non-negative with

$$\langle \mathbf{u},\, D^{-1/2}\,T_v^R\,D^{-1/2}\,\mathbf{u}\rangle_X = \langle D^{-1/2}\,\mathbf{u},\, T_v^R\,D^{-1/2}\,\mathbf{u}\rangle_X \geq 0,$$

and by Theorem 3.3, equality holds iff $(D^{-1/2}\,\mathbf{u})_i a_i$ is constant on connected components of the graph. ∎

Compared with the Laplacian operator $(L\mathbf{y})_i = \sum_j (y_i - y_j) w_{ij}$, we see that T_v^R generalizes L in the following ways:

- a_i modifies the measure/coordinate and thus makes the representation of i-th point closer to the origin if a_i is large or further away from the origin if a_i is small.
- r_{ij} can enlarge or reduce the weight w_{ij} between two nodes i and j, serving as a weight modifier.

We can then use these two sets of parameters to guide data representation given by LE.

3.3. Two examples of TE. We will use TE to denote the general transport operator (3.9). Although the matrix R can be used to fuse extra information, the implementation with R could be more time-consuming as the size of R is n^2. We will therefore first look at two examples (denoted by TA and TG respectively) using a_i only. As Section 5.4 will show, TA and TG are often good enough to handle classification tasks when one class is known. The general TE, however, is needed when more than one classes are known.

3.3.1. Transport by advection (TA). Advection is the active transportation of a distribution by a flow field. Let $\boldsymbol{\mu} = [\mu_1, \mu_2, \ldots, \mu_n]^t$ be a vector that will direct the clustering process. Let β be a real parameter which can be used to control the influence of $\boldsymbol{\mu}$ on the Laplacian. Set $a_i = 1 + \beta\mu_i$, $r_{ij} = (a_j + a_i)/2$, and $\bar{v}_{ij} = (a_j - a_i)/2$. Clearly $\bar{v}_{ij} = \frac{a_j - a_i}{a_j + a_i}\, r_{ij}$. By Theorem 3.2, the operator $T_{\boldsymbol{\mu}} := T_v^R$ with

$$(T_{\boldsymbol{\mu}}\, \mathbf{y})_i = \sum_j [(1 + \beta\mu_i)\, y_i - (1 + \beta\mu_j)\, y_j]\, w_{ij} \tag{3.11}$$

is self-adjoint and enjoys other desired properties.

The operator $T_{\boldsymbol{\mu}}$ can also be derived directly from the general operator T (3.5) by choosing the velocity field $v = \beta\nabla\, \mathbf{y}$, $\beta \in \mathbb{R}$. In this case, $v_{ij} = \beta\, (y_j - y_i)\, w_{ij}$ and T becomes

$$(T\, \mathbf{y})_i = \sum_j (y_i - y_j)\, w_{ij} - \frac{\beta}{2} \sum_j (y_j^2 - y_i^2)\, w_{ij}, \tag{3.12}$$

which is no longer linear. We can then linearize the second term in (3.12) in the direction of $\boldsymbol{\mu}$ and T will be exactly $T_{\boldsymbol{\mu}}$ (see [32] for details).

This choice of operator is inspired by the porous medium equation, for which we refer for example to [43] for a thorough discussion of this type of non-linear diffusion on $\mathbb{R}^d$. In the present context, the idea behind having $v(\mathbf{y}) = \beta\, \nabla \mathbf{y}$ is to use the distribution $\mathbf{y}$ itself to help with clustering. The velocity field $v(\mathbf{y})$ naturally points in the direction of the higher values of $\mathbf{y}$ if $\beta < 0$ or towards lower values if $\beta > 0$. Similarly solving the advection-diffusion equation

$$d_t\, \mathbf{y} + T\, \mathbf{y} = 0,$$

would naturally lead to concentration around higher values of $\mathbf{y}$ if $\beta < 0$ (limited by the dispersive effects of the graph Laplacian) or *a contrario* to faster dispersion if $\beta > 0$. The ability to control concentrations and hence clustering is of obvious interest for our purpose.

3.3.2. Transport by gradient flows (TG). Set $r_{ij} \equiv 1$ in (3.9). Then the general transport operator T_v^R becomes

$$(T_v\, \mathbf{y})_i = \sum_j (a_i y_i - a_j y_j)\, w_{ij}\, \frac{2}{a_i + a_j}. \tag{3.13}$$

Note that this is in fact the same operator appeared in Theorem 3.1, where v is an scaling-invariant gradient of $\mathbf{a} = [a_1, \ldots, a_n]^t$. Here a_i plays a similar role as $1 + \beta\mu_i$ in the first example of the transport by advection. One advantage of having the extra term $\frac{2}{a_i + a_j}$ is that even the weight modifier r is constant, the weight w_{ij} could still be changed. In applications, the default value for the measure modifier a_i is 1 and some of them may be greater than 1 if extra information is known. When $a_i \neq a_j$, which often indicates that the two points i and j belong to different clusters, the factor $\frac{2}{a_i + a_j} < 1$, weakening the original weight w_{ij}. Therefore, the formulation of the operator T_v achieves measure modification and weight modification simultaneously without using r.

3.4. The algorithm. We describe the implementation of our new TE (short for transport eigenmap) algorithm, including TA and TG as two important special cases.

The steps are identical to those of LE and SE. We only need to modify the matrix used in the generalized eigenvalue problem. Given a set of n points $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ in $\mathbb{R}^d$, the goal is to find a map

$$\Phi : \mathbb{R}^d \longrightarrow \mathbb{R}^m,$$

so that the n points $Y = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n\}$ in $\mathbb{R}^m$ given by $\mathbf{y}_i = \Phi(\mathbf{x}_i)$ represents $\mathbf{x}_i$ for all i from 1 to n.

The goal is typically to have a lower dimensional representation Y of the set of points X with $m \ll d$ while still keeping the main features of the original set X. For example if the points lie on a m-dimensional manifold where $m \ll d$, the hope would be to take as map Φ a good approximation of the projection on the manifold.

- **Step 1:** Construct the adjacency graph using the k-nearest neighbor (kNN) algorithm. This is done by putting an edge connecting nodes i and j given that $\mathbf{x}_i$ is among the k nearest neighbors of $\mathbf{x}_j$ according to the Euclidean metric. We choose k large enough so that the graph that we obtain is connected. This step can make the matrix W in the next step sparser.
- **Step 2:** Define the weight matrix, W, on the graph. The weights w_{ij} in W are chosen using the heat kernel with some parameter σ. If nodes i and j are connected,

$$w_{ij} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{2\sigma^2}\right);$$

  otherwise, $w_{ij} = 0$.
- **Step 3:** Construct the matrix representing the transport operator. Recall the general transport operator given in (3.9)

$$(T\,\mathbf{y})_i = \sum_j (a_i y_i - a_j y_j)\, w_{ij}\, \frac{2r_{ij}}{a_i + a_j}.$$

  Here, the vector $\mathbf{a} = [a_1, \ldots, a_n]^t$ and the matrix (r_{ij}) are the parameters to be chosen. Let W^r denote the matrix with entries $w^r_{ij} = w_{ij}\,\frac{2r_{ij}}{a_i + a_j}$. Then the matrix form of T is

$$T = \mathrm{diag}(a_i \sum_j w^r_{ij}) - W^r \mathrm{diag}(a_i). \tag{3.14}$$

  To get the matrix form of the special operator TA, we can either set $a_i = 1 + \beta\mu_i$ and $r_{ij} = (a_i + a_j)/2$ in (3.14), or use the operator form (3.11) to derive its matrix form directly

$$TA = L(I + \beta\,\mathrm{diag}(\mu_i)),$$

  where $L = D - W$ is the Laplacian matrix and I is the identity.

Similarly, for the operator TG, we can let $r_{ij} = 1$ in (3.14) or use the expression in Theorem 3.1 to get

$$TG = L - (D_v + Wv),$$

where $D_v = \mathrm{diag}(\sum_j w_{ij}v_{ij})$, $W_v = (w_{ij}v_{ij})$ and $v_{ij} = (a_j - a_i)/(a_j + a_i)$.

- **Step 4:** Find the m-dimensional transport mapping Φ_T by solving the generalized eigenvector problem,

$$T\,\mathbf{u} = \lambda\, D\,\mathbf{u}, \tag{3.15}$$

This can be done because of Corollary 3.4. Denote $\{u^0, u^1, \ldots, u^{n-1}\}$ be the solution set to (3.15) written in ascending order according to their eigenvalues. Since there is hence no additional information in u^0, we define the mapping Φ_T by

$$\mathbf{x}_i \longrightarrow \Phi_T(\mathbf{x}_i) = [u_i^1, u_i^2, \ldots, u_i^m].$$

4. The transport eigenmap for clustering.

4.1. Intuition of the parameters: a case study . We illustrate the behavior of LE, SE and TE (including TA and TG) with a toy example. The first picture in Figure 1 is a dataset with 500 points. The ground truth is that there are 5 clusters (labelled by different colors), each containing 100 points.

LE is an unsupervised method that preserves local distance. We chose $k = 50$ for KNN in Step 1 and $\sigma = 1$ in Step 2 for simplicity.

SE, which uses the matrix $S = L + \alpha V$, requires extra parameters: $\alpha \geq 0$ and the diagonal potential matrix V. Suppose experts suggests that the red points should be identified as one cluster (this is an extreme example of extra ground truth knowledge). Simply let $V_i = 1$ if the i-th point is red and $V_i = 0$ otherwise. Let $\alpha = \hat{\alpha} \cdot \mathrm{tr}\,(L)/\mathrm{tr}\,(V)$. This new parameter $\hat{\alpha}$ will allow us to balance the impact of the Laplacian matrix L and the potential V in the algorithm. We chose $\hat{\alpha} = 10$. As expected, points with non-zero potential (the red ones in this example) are pushed towards the origin. As L tries to preserve local distance, other points close to the red are dragged towards the origin as well.

For TA, we chose $\beta = 10$ and $\boldsymbol{\mu}$ in the same way as V: $\mu_i = 1$ for red and $\mu_i = 0$ for other points. The red go to the origin because of rescaling of the coordinates, but the surrounding points don't "see" any changes in distance. This explains the less dragging effect in TA compared with SE.

In TG, we set $a_i = 1$ by default and $a_i = 10$ for the red points. The red are even better separated from others. This is because the factor $\frac{2}{a_i + a_j}$ in (3.13) is less than 1 and thus weakens the original weight w_{ij} if i and j are not both red.

For the general TE, the matrix R needs to be determined. The default is $r_{ij} = 1$. Then it is natural to set

$$r_{ij} = \begin{cases} small(<1), & \text{if } i \text{ and } j \text{ belong to different clusters,} \\ big(>1), & \text{if } i \text{ and } j \text{ belong to the same cluster} \\ 1, & \text{if unknown} \end{cases} \tag{4.1}$$

The size of r_{ij} depends on how strong one believes i and j are in the same/different clusters. For example, we may set $r_{ij} = 10^{10}$ if one is very certain that i and j are alike. We set $small = 0.5$ and $big = 100$ in this toy experiment. This will further help gathering the red points.

If the pre-identified cluster is not near the center of the data points, e.g., the blue points, then we can set a_i to be less than 1 for the blue to push them away from the origin. The weight modifier R in TE is always helpful to gather these points to their natural location. See Figure 1 for the case $a_i = 0.5$ for the blue and 1 otherwise in TE (R remains to be in (4.1)).

The general TE can even handle the case when more than one cluster are known. Let $a_i = 10$ for red and $a_i = 0.5$ for blue. R is still given by (4.1). We can see in Figure 1 that both red and blue are well-separated from others.

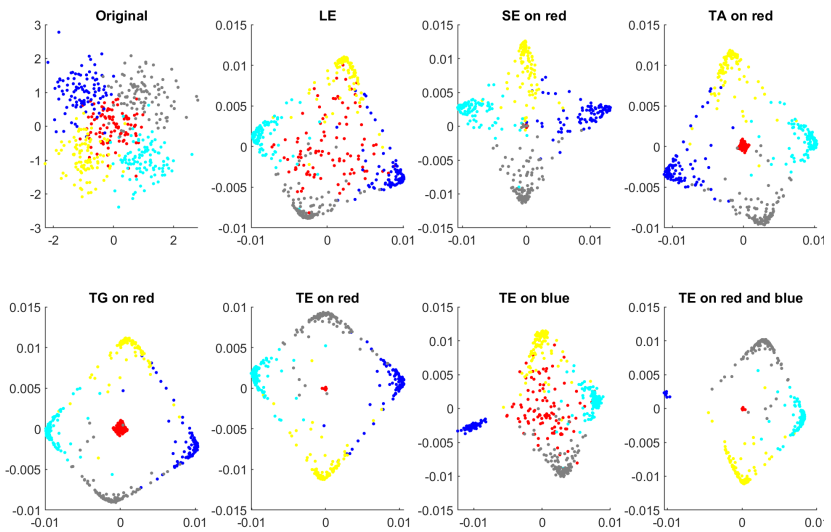

Figure 1. *The first plot presents the dataset,* 500 *points grouped in* 5 *clusters of* 100 *points each. The next plots show the results of various mappings.*

The above experiment shows how TE can be used to help with clustering. Next we will test our methods on a real image clustering task.

4.2. Image clustering tasks. Most of our numerical investigations have been performed on hyperspectral datasets. However as a complement and to further evaluate the quality of our feature extraction method and its ability in helping with clustering, we consider the challenging problem of clustering the CIFAR-100 dataset [31] and STL-10 dataset [16]. CIFAR-100 consists of 60000 images of size $32 \times 32 \times 3$ in 100 classes, which can be further grouped into 20 superclasses. This multi-tiered structure could be incorporated in TE ([18]), but for simplicity we use the 20 coarse labels in the test. The STL-10 dataset has 13000 images of size $96 \times 96 \times 3$ in 10 classes. We apply kmeans clustering after mapping by TE. We compare our method with many clustering methods listed in the following table. The metrics used here are normalized mutual information (NMI), accuracy (ACC) and the adjusted Rand index (ARI, [36]). The measurements of other methods are taken from the papers [13, 47].

Datasets	CIFAR-100			STL-10		
Method	NMI	ACC	ARI	NMI	ACC	ARI
K-means	0.084	0.130	0.028	0.125	0.192	0.061
SC [51]	0.090	0.136	0.022	0.098	0.159	0.048
AC [24]	0.098	0.138	0.034	0.239	0.332	0.140
NMF [12]	0.079	0.118	0.026	0.096	0.180	0.046
AE [7]	0.100	0.165	0.048	0.250	0.303	0.161
DAE [44]	0.111	0.151	0.046	0.224	0.302	0.152
GAN [34]	0.120	0.151	0.045	0.210	0.298	0.139
DeCNN [50]	0.092	0.133	0.038	0.227	0.299	0.162
VAE [30]	0.108	0.152	0.040	0.200	0.282	0.146
JULE [49]	0.103	0.137	0.033	0.182	0.277	0.164
DEC [48]	0.136	0.185	0.050	0.276	0.359	0.186
TE	0.157	0.167	0.052	0.352	0.360	0.199
DAC [13]	0.185	0.238	0.088	0.366	0.470	0.257
DCCM [47]	0.285	0.327	0.173	0.376	0.482	0.262

Table 1
Clustering results by various methods

On CIFAR-100, we assume the first class is pre-identified (The results do not change much if another class is used). That corresponds to only 5% of the ground truth. If we use 90% of points in the first class, the measurements will decrease slightly to NMI: 0.139, ACC: 0.159, ARI: 0.046. Due to the size of this dataset, the TA version of TE is used to speed up computations. On the smaller STL-10 dataset, we can just use the general TE, which is also suitable for handling more than one preidentified classes (e.g., Table 6 and Table 7 in Section 5.4). We randomly selected 90% of the points from two classes to supervise TE to get the results in the above table.

The table shows that TE is better than all but the recent methods DAC and DCCM. Although TE is semi-supervised, this performance is satisfied since TE only handles the feature extraction part: the clustering is done by kmeans. At present we emphasize that our investigations have necessarily been limited. Being a new method with a rich set of parameters, those partial results shows that TE has the potential to gain better performance given further understanding about the optimal choice of parameters.

Figure 2. *Classification performance measures for TA (red diamonds) as a function of the amount of information provided, from* 0% *to* 100% *with increments of* 5%.

5. The transport eigenmap for classification. We turn to test the feature extraction by TE with classification tasks, using hyperspectral images as examples.

5.1. The datasets. We consider two famous hyperspectral data sets: Indian Pines and Salinas. The Indian Pines dataset (cf. an example in Figure SM1 in the supplementary document) was gathered by AVIRIS (Airborne Visible/Infrared Imaging Spectrometer) sensor over the Indian Pines test site in North-western Indiana. The Indian Pines dataset consists of 145×145 pixels images that contain 224 spectral bands in the wavelength range 0.4×10^{-6} to 2.5×10^{-6} meters. The ground truth available is designated into sixteen classes (see Table SM1 in the supplement). The number of bands has been reduced to 200 by removing bands covering the region of water absorption.

The Salinas dataset was similarly gathered by AVIRIS sensor over Salinas Valley, California (see Figure SM2 in the supplementary document). With again a similar structure, Salinas images are 512×217 pixels with 224 spectral bands of approximately 3.7 meter high spatial resolution. The ground truth available is also clustered into sixteen classes (see Table SM2 in the supplement). We again reduce the number of bands to 204 by removing those bands covering the region of water absorption.

For easier testing purposes, we have also used a small sub-scene of the Salinas dataset, which we denote Salinas-B (shown in Figure SM3 in the supplement). Salinas-B consists of a $150 \times 100 \times 204$ data cube located within the same scene at [samples, lines]=[200:349, 40:139] and includes only eight classes (see Table SM3 in the supplement). The Salinas-B dataset was used to allow for a faster and more thorough exploration of the parameters' space.

After the various mappings, we employ Matlab's 1-nearest neighbor algorithm to classify the data sets. We use 10% of the data from each class to train the classifier and the remaining number of data points as the validation set. We took an average of ten runs to produce the confusion matrices, each using a disjoint set of data to train the classifier.

5.2. Choice of parameters. Following the description of the mapping algorithms for the various methods under consideration in subsection 3.4, we made the following choices to construct the graph over which all methods rely

- The adjacency graph is built using $k = 12$ nearest neighbors;
- The weight matrix was obtained by using $\sigma = 1$;
- We calculated $m = 50$ generalized eigenvectors for the Indian Pines dataset and $m = 25$ for the Salinas-B dataset. The final mappings were obtained from those generalized eigenvectors as described in Step 4 of subsection 3.4.

For SE, TA and TG, we also need to choose the potential V, the vector $\boldsymbol{\mu}$ and $\mathbf{a}$. In our testing, for example, we have assumed prior knowledge of either class 2-corn-notill or class 11-soybean-mintill in the Indian Pines dataset. This leads to the typical choice in the $11-soybean-mintill$ case

$$V_i,\ \mu_i = \begin{cases} 1, & \text{if } x_i \in \text{Class 11–soybean-mintill,} \\ 0, & \text{elsewhere.} \end{cases}$$

In TG, the default is $a_i = 1$ and we will set $a_i = \beta$ for the known points. It remains to chose the parameters α and β. For SE, recall that in Section 4.1 we introduced the parameter $\hat{\alpha}$ given by $\alpha = \hat{\alpha} \cdot \mathrm{tr}\,(\Delta)/\mathrm{tr}\,(V)$. To obtain the results listed in the next subsection, we used

- $\hat{\alpha} = 10^4$ for the Indian Pines data set and $\hat{\alpha} = 10^2$ for the Salinas-B data set for SE;
- $\beta = 20$ for both the Indian Pines and the Salinas-B data set for TA and TG.

The particular choices of parameters summarized here were obtained after a more thorough investigation and optimization among possible values. This parameter exploration is shown in Section SM3 in the supplementary material.

5.3. Measuring accuracy. We will compare the performance of several feature extraction methods in the next section. To obtain a more complete perspective, we consider several measurements of accuracy including the adjusted Rand index (ARI) [36], the overall accuracy (OA), the average or weighted accuracy (AA), the average F-score (FS) and Cohen's kappa coefficient (κ).

5.4. Results. We summarize the main results of our numerical experiments on the real hyperspectral images introduced in the previous section. More details are available in the supplementary document.

5.4.1. Overall performance. The following feature extraction algorithms are used in the experiment: principal components analysis [33] (PCA), Laplacian eigenmaps [3] (LE), diffusion maps [17] (DIF), isomap [42] (ISO), Schroedinger eigenmaps [11] (SE), transport eigenmaps (TE, including TA and TG). The classification maps for each of the results can be found in the supplement.

We especially focus on the Adjusted Rand Index, Overall Accuracy, and on the Cohen's kappa coefficient (emphasized in bold in the tables) as the main indicators for the performance of the algorithms.

Testing on two examples. We first test TA on the Salinas-B dataset (Table 2), assuming the class "lettuce" is known in SE and TA. Unsurprisingly, the semi-supervised algorithms, SE and TA, outperform the unsupervised algorithms, PCA, LE, DIF and ISO. The performance of the SE and TA is roughly similar, but with a small but consistent advantage to TA.

SB	PCA	LE	DIF	ISO	SE	TA
ARI	0.9429	0.9346	0.9164	0.9440	0.9439	**0.9463**
OA	0.9729	0.9685	0.9603	0.9733	0.9762	**0.9780**
AA	0.9690	0.9643	0.9564	0.9700	0.9777	**0.9802**
FS	0.9693	0.9638	0.9557	0.9696	0.9766	**0.9795**
κ	0.9682	0.9630	0.9534	0.9687	0.9720	**0.9742**

Table 2

Classification results for Salinas-B (SB): assume lettuce (class 14) is known

Classification algorithms frequently mis-classify samples of similar classes due to the similarities in their spectra information. For this reason, we tested the algorithms by grouping similar classes within the Indian Pines and Salinas-B data set to make new ground truths which we denote Indian Pines-G and Salinas-B-G (see Table SM4 and Table SM5 in the supplement).

It turns out SE and TA indeed perform better on grouped Salinas-B (Table 3) than on Salinas-B. TA remains to be the best method for the grouped dataset.

SBG	PCA	LE	DIF	ISO	SE	TA
ARI	0.9460	0.9421	0.9154	0.9480	0.9711	**0.9767**
OA	0.9791	0.9767	0.9677	0.9795	0.9858	**0.9880**
AA	0.9769	0.9750	0.9669	0.9784	0.9819	**0.9840**
FS	0.9797	0.9763	0.9697	0.9797	0.9829	**0.9850**
κ	0.9725	0.9694	0.9576	0.9731	0.9814	**0.9843**

Table 3

Classification results for Salinas-B-G (SBG): assume lettuce (class 11) is known

Using TA on the Salinas-B-G as an example, we also give the accuracy per class in the supplement, which shows that the improvement of accuracy comes from both preidentifed class and other classes. The results can be found in the supplements.

We then test TG on Indian Pines dataset and its grouped version, assuming the class "soybean" is known. In this difficult image, the gain of performance in using TG is significant. See Table 4 and Table 5 below.

IP	PCA	LE	DIF	ISO	SE	TG
ARI	0.4426	0.3745	0.4210	0.3930	0.6955	**0.7104**
OA	0.6761	0.6133	0.6557	0.6309	0.7354	**0.7431**
AA	**0.6403**	0.5782	0.6219	0.5979	0.6249	0.6248
FS	**0.6471**	0.5784	0.6212	0.5996	0.6255	0.6250
κ	0.6301	0.5592	0.6065	0.5785	0.6982	**0.7071**

Table 4

Classification results for Indian Pines (IP): assume soybean (class 11) is known.

IPG	PCA	LE	DIF	ISO	SE	TG
ARI	0.5330	0.4785	0.5102	0.4902	0.8929	**0.9264**
OA	0.7744	0.7307	0.7575	0.7418	0.9088	**0.9155**
AA	0.6987	0.6462	0.6883	0.6671	**0.7111**	0.7072
FS	0.7111	0.6479	0.6905	0.6739	**0.7157**	0.7087
κ	0.6996	0.6423	0.6770	0.6563	0.8788	**0.8877**

Table 5

Classification results for Indian Pines-G (IPG): assume soybean (class 10) is known

We remark that ideally the way to implement TE (e.g. TA or TG) should depend on physical interpretation of the data. The above tables show that TA and TG are good for "arbitrary" datasets.

Testing the general TE. Although being expensive in computation, the use of general TE is needed if information about more than one classes is known. Table 7 and Table 6 show that SE, TA and TG can often perform worse when two classes are known. However, TE gives significant improvements. Here we use r given by (4.1) with $small = 0.9$ and $big = 10^4$, and set $a_i = 10$ and $a_i = 20$ on the two known classes.

IP	SE	TG	TE	IPG	SE	TG	TE
ARI	0.5272	0.7693	**0.8169**	ARI	0.4351	0.8547	**0.9372**
OA	0.6855	0.8091	**0.8268**	OA	0.6858	0.8967	**0.9252**
AA	0.6221	0.6759	**0.6864**	AA	0.6431	0.7055	**0.7221**
FS	0.6229	0.6766	**0.6855**	FS	0.6467	0.7083	**0.7242**
κ	0.6409	0.7818	**0.8024**	κ	0.5821	0.8620	**0.9004**

Table 6

Classification results for Indian Pines (IP) and its grouped version (IPG): assume both corn and soybean are known.

SB	SE	TA	TE	SBG	SE	TA	TE
ARI	0.9381	0.9805	**0.9812**	ARI	0.7916	0.9773	**0.9823**
OA	0.9702	0.9909	**0.9914**	OA	0.9211	0.9902	**0.9921**
AA	0.9671	0.9903	**0.9908**	AA	0.9877	0.9877	**0.9900**
FS	0.9666	0.9902	**0.9909**	FS	0.9365	0.9889	**0.9906**
κ	0.9651	0.9894	**0.9899**	κ	0.8966	0.9871	**0.9896**

Table 7

Classification results for Salinas-B (SB) and its grouped version (SBG): assume both corn and lettuce are known

In SE, points with positive potential will always be mapped towards the origin. There is no mechanism to handle two different clusters. This explains that SE often perform worse in the above tests. In TA and TG, although the distance from the points to the origin can be modified in different ways by varying a_i, points from different classes could still collide after mapping because of their initial locations. The general TE has the power of minimizing the possibility of mixing two known classes since the matrix r provides internal force to group points in the same class.

5.4.2. Dependence on the amount of the information. We performed further experiments on Indian Pines-G and Salinas-G to see how the amount of information available from one particular class affects the performance measures for SE and transport methods TA and TG.

SE and transport methods have very close overall performance on the Indian Pines-G and Salinas-B-G datasets so the comparison may help to understand better the differences between them. As the amount of information increases, so do the performance measures. Figure 3 shows the change in performance of SE, TA and TG from using 0% to using 100% of the ground truth with increments of 5% from a particular class.

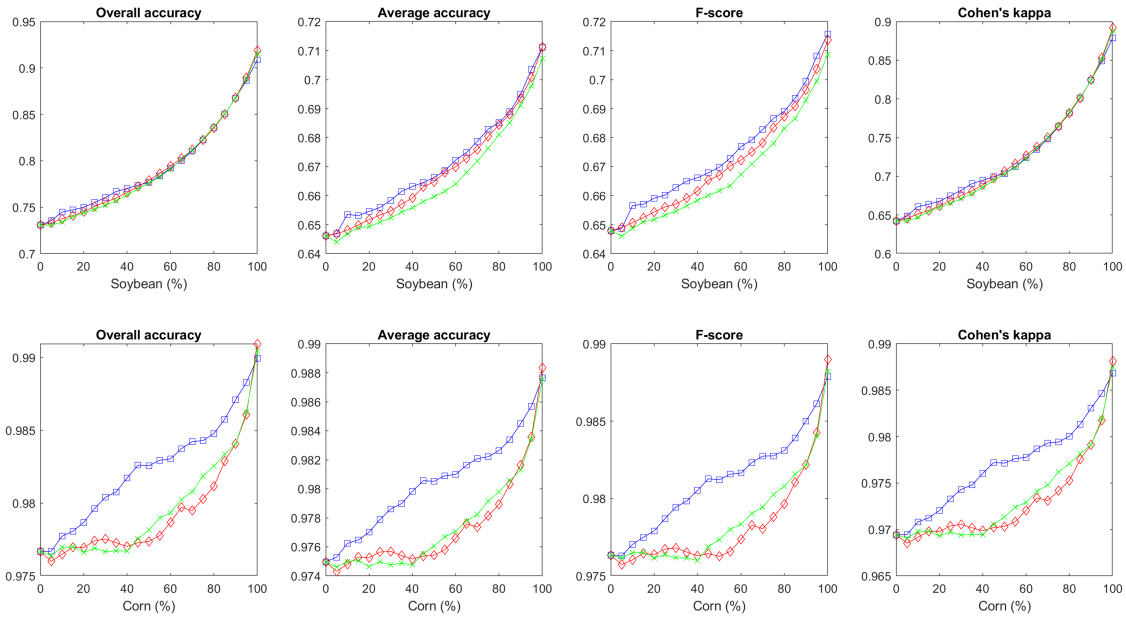

Figure 3. *Classification performance measures for SE (blue squares), TA (red diamonds) and TG (green x's) as a function of the amount of information provided, from* 0% *to* 100% *with increments of* 5%*. The Indian Pines-G data set (top row) is used with the advection and potential placed on class 10–soybean. The Salinas-B-G (bottom row) is used with the advection and potential placed on class 10–corn-senesced-green-weeds.*

Over most of the figure, the SE actually performs slightly better than the TA and TG, with TA and TG only surpassing SE when we have close to 100% of the information on the class. However *the difference between the two algorithms remains very small in those two simplified datasets*; this is especially striking on the Indian Pines-G.

5.4.3. Robustness of Transport eigenmaps. In a last set of experiments, we investigate the robustness of transport methods TA and TG and some of our other feature extraction algorithms such as PCA, LE, and SE. For this experiment, we have added Gaussian noise to individual data points in the data set before it is processed by the feature extraction algorithms. The added Gaussian noise has a mean of 0 and we selected 20 logarithmically spaced values for the standard deviation varying from 10^0 to 10^5 which covers the range for values taken by the individual data points in both set of data. For SE and transport methods, the ground truths for class 10–soybean (Indian Pines-G) and class 11–lettuce (Salinas-B-G) are added to the algorithms. The results are shown on Figure 4.

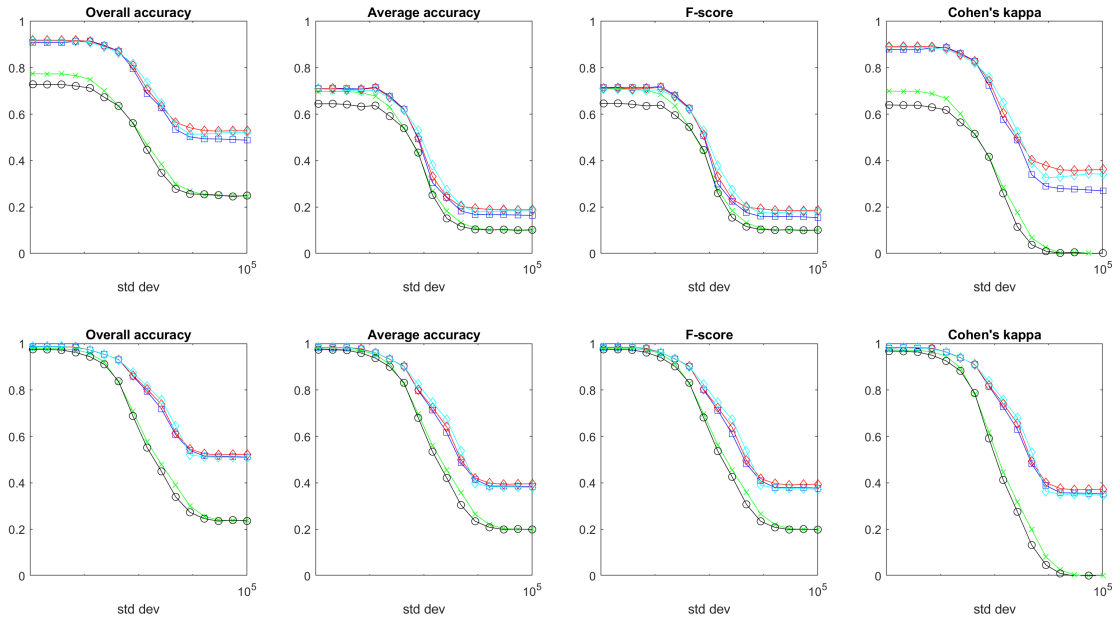

Figure 4. *Classification performance measures for TA (red diamonds), TG (cyan diamonds), SE (blue squares), PCA (green x's), and LE (black circles) as a function of noise. For Indian Pines-G (top row) the potential and advection are placed on class 10–soybean. For Salinas-B-G (bottom row) the potential and advection are placed on class 11–lettuce-romaine.*

We first gather from the experiments that SE and transport methods are more resilient to noise than PCA and LE. While the performance of all algorithms naturally decreases very significantly (and interestingly at almost the same mark), SE and transport methods resist better. On the Indian Pines-G, transport methods also end up being the best algorithm by a significant margin, performing $\sim 30\%$ better than SE for large noise whereas they are comparable for small noise. This again suggests that our new Transport algorithm is especially useful in difficult settings where previous methods do not perform well.

6. Conclusion. In this manuscript, we propose a novel approach to semi-supervised non-linear feature extraction extending the Laplacian eigenmaps. Similar in spirit to previous extension such as Schroedinger eigenmaps, our algorithm is derived from non-linear transport model. We first test this transport model on clustering some famous image datasets. Then we provide a set of experiments on hyperspectral data sets to compare the new method's performance to a variety of algorithms for reducing the dimension of the data provided to a standard classification algorithm. The experiments show intriguing possibilities for the new method, which has proved competitive with other algorithms in both clustering and classification tasks.

Our method performs the best when there are extra information about data points that are similar. In real-life applications, the extra information provided to the algorithms of transport methods usually does not come directly from the ground truth. Ideally, better and richer cluster information than the ground truth are produced using laboratory measurements. For example, in hyperspectral imaging, the laboratory measurements could include various

signals representing different materials in a wide range of conditions such as lighting and weather.

Our experiments demonstrate a strong potential for new methods using advection/gradient flow operators, with in particular the following open questions

- How to further generalize the transport operator? The choice of the velocity field v in Theorem 3.1 makes the transport operator self-adjoint with respect to an inner product associated with a diagonal matrix A. It is natural to investigate the case with a non-diagonal, positive definite A.
- Can we better relate the choice of an algorithm to the expected structure of the problem? A good example might be time-dependent data, where a clear direction of propagation of the signal would lead to conjecture a even better performance of advection-based eigenmaps.
- What is the best way to choose the parameters in the general transport method. The intuition provided in Section 4.1 is good only for low dimensional data. When the dimension is high or there are two or more clusters, the choice of r and a_i can be very complicated. We plan to use neural network to attack this problem.

REFERENCES

[1] J. Bartsch, K. Brander, M. Heath, P. Munk, K. Richardson, and E. Svendsen, *Modelling the advection of herring larvae in the north sea*, Nature, 340 (1989), p. 632.

[2] M. Belkin and P. Niyogi, *Laplacian eigenmaps and spectral techniques for embedding and clustering*, in Advances in Neural Information Processing Systems, 2002, pp. 585–591.

[3] M. Belkin and P. Niyogi, *Laplacian eigenmaps for dimensionality reduction and data representation*, Neural Computation, 15 (2003), pp. 1373–1396.

[4] M. Belkin and P. Niyogi, *Semi-supervised learning on Riemannian manifolds*, Machine Learning, 56 (2004), pp. 209–239.

[5] J. J. Benedetto, W. Czaja, J. Dobrosotskaya, T. Doster, K. Duke, and D. Gillis, *Semi-supervised learning of heterogeneous data in remote sensing imagery*, in Independent Component Analyses, Compressive Sampling, Wavelets, Neural Net, Biosystems, and Nanoengineering X, vol. 8401, International Society for Optics and Photonics, 2012, p. 840104.

[6] J. J. Benedetto, W. Czaja, J. C. Flake, and M. Hirn, *Frame based kernel methods for automatic classification in hyperspectral data*, in Geoscience and Remote Sensing Symposium, 2009 IEEE International, IGARSS 2009, vol. 4, IEEE, 2009, pp. IV–697.

[7] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, *Greedy layer-wise training of deep networks*, in Advances in neural information processing systems, 2007, pp. 153–160.

[8] T. Bennett, *Transport by advection and diffusion*, Wiley Global Education, 2012.

[9] H. Brogniez, R. Roca, and L. Picon, *A study of the free tropospheric humidity interannual variability using meteosat data and an advection–condensation transport model*, Journal of Climate, 22 (2009), pp. 6773–6787.

[10] W. Brutsaert and H. Stricker, *An advection-aridity approach to estimate actual regional evapotranspiration*, Water Resources Research, 15 (1979), pp. 443–450.

[11] N. D. Cahill, W. Czaja, and D. W. Messinger, *Schroedinger eigenmaps with nondiagonal potentials for spatial-spectral clustering of hyperspectral imagery*, in Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XX, vol. 9088, International Society for Optics and Photonics, 2014, p. 908804.

[12] D. Cai, X. He, X. Wang, H. Bao, and J. Han, *Locality preserving nonnegative matrix factorization.*, in IJCAI, vol. 9, 2009, pp. 1010–1015.

[13] J. Chang, L. Wang, G. Meng, S. Xiang, and C. Pan, *Deep adaptive image clustering*, in Proceedings

of the IEEE international conference on computer vision, 2017, pp. 5879–5887.
[14] F. R. Chung and F. C. Graham, *Spectral graph theory*, no. 92, American Mathematical Soc., 1997.
[15] A. Cloninger, W. Czaja, and T. Doster, *The pre-image problem for Laplacian eigenmaps utilizing L^1 regularization with applications to data fusion*, Inverse Problems, 33 (2017), p. 074006.
[16] A. Coates, A. Ng, and H. Lee, *An analysis of single-layer networks in unsupervised feature learning*, in Proceedings of the fourteenth international conference on artificial intelligence and statistics, 2011, pp. 215–223.
[17] R. R. Coifman and S. Lafon, *Diffusion maps*, Applied and Computational Harmonic Analysis, 21 (2006), pp. 5–30.
[18] W. Czaja, D. Dong, P. Jabin, and F. Njeunje, *Analysis of hyperspectral data by means of transport models and machine learning*, IEEE International Geoscience and Remote Sensing Symposium, to appear, (2020).
[19] W. Czaja and M. Ehler, *Schroedinger eigenmaps for the analysis of biomedical data*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 35 (2013), pp. 1274–1280.
[20] I. Davidson, *Knowledge driven dimension reduction*, in Twenty-First International Joint Conference on Artificial Intelligence, IJCAI-09, 2009, pp. 1034–1039.
[21] T. J. Doster, *Harmonic analysis inspired data fusion for applications in remote sensing*, PhD thesis, University of Maryland, College Park, 2014.
[22] H. Fang, M. Cheng, and C. Hsieh, *A hyperplane-based algorithm for semi-supervised dimension reduction*, in IEEE International Conference on Data Mining, IEEE, 2017.
[23] S. Gerber and M. Maggioni, *Multiscale strategies for computing optimal transport*, J. Mach. Learn. Res., 18 (2017), pp. 2440–2471, http://dl.acm.org/citation.cfm?id=3122009.3176816.
[24] K. C. Gowda and G. Krishna, *Agglomerative clustering using the concept of mutual nearest neighbourhood*, Pattern recognition, 10 (1978), pp. 105–112.
[25] H. Guo, J. Zhang, R. Liu, L. Liu, X. Yuan, J. Huang, X. Meng, and J. Pan, *Advection-based sparse data management for visualizing unsteady flow*, IEEE Transactions on Visualization and Computer Graphics, 20 (2014), pp. 2555–2564.
[26] A. Halevy, *Extensions of Laplacian eigenmaps for manifold learning*, PhD thesis, University of Maryland, College Park, 2011.
[27] Han-Wei Shen, C. R. Johnson, and Kwan-Liu Ma, *Visualizing vector fields using line integral convolution and dye advection*, in Proceedings of 1996 Symposium on Volume Visualization, 1996, pp. 63–70.
[28] K. B. Hansen and S. C. Shadden, *A reduced-dimensional model for near-wall transport in cardiovascular flows*, Biomechanics and Modeling in Mechanobiology, 15 (2016), pp. 713–722.
[29] W. Hundsdorfer and J. G. Verwer, *Numerical solution of time-dependent advection-diffusion-reaction equations*, vol. 33, Springer Science & Business Media, 2013.
[30] D. P. Kingma and M. Welling, *Auto-encoding variational bayes*, arXiv preprint arXiv:1312.6114, (2013).
[31] A. Krizhevsky, G. Hinton, et al., *Learning multiple layers of features from tiny images*, (2009).
[32] F. Njeunje, *Computational methods in machine learning: transport model, Haar wavelet, DNA classification, and MRI*, PhD thesis, University of Maryland, College Park, 2018.
[33] K. Pearson, *On lines and planes of closest fit to systems of point in space*, Philosophical Magazine, 2 (1901), pp. 559–572.
[34] A. Radford, L. Metz, and S. Chintala, *Unsupervised representation learning with deep convolutional generative adversarial networks*, arXiv preprint arXiv:1511.06434, (2015).
[35] S. T. Roweis and L. K. Saul, *Nonlinear dimensionality reduction by locally linear embedding*, Science, 290 (2000), pp. 2323–2326.
[36] J. M. Santos and M. Embrechts, *On the use of the adjusted Rand index as a metric for evaluating supervised classification*, in International Conference on Artificial Neural Networks, Springer, 2009, pp. 175–184.
[37] B. Schölkopf, A. Smola, and K.-R. Müller, *Kernel principal component analysis*, in International Conference on Artificial Neural Networks, Springer, 1997, pp. 583–588.
[38] J. R. Sibert, J. Hampton, D. A. Fournier, and P. J. Bills, *An advection–diffusion–reaction model for the estimation of fish movement parameters from tagging data, with application to skipjack tuna (katsuwonus pelamis)*, Canadian Journal of Fisheries and Aquatic Sciences, 56 (1999), pp. 925–938.

[39] H. Song, M. R. Bringer, J. D. Tice, C. J. Gerdts, and R. F. Ismagilov, *Experimental test of scaling of mixing by chaotic advection in droplets moving through microfluidic channels*, Applied Physics Letters, 83 (2003), pp. 4664–4666.
[40] W. Sun, A. Halevy, J. J. Benedetto, W. Czaja, W. Li, C. Liu, B. Shi, and R. Wang, *Nonlinear dimensionality reduction via the ENH-LTSA method for hyperspectral image classification*, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 7 (2014), pp. 375–388.
[41] W. Sun, A. Halevy, J. J. Benedetto, W. Czaja, C. Liu, H. Wu, B. Shi, and W. Li, *UL-Isomap based nonlinear dimensionality reduction for hyperspectral imagery classification*, ISPRS Journal of Photogrammetry and Remote Sensing, 89 (2014), pp. 25–36.
[42] J. B. Tenenbaum, V. De Silva, and J. C. Langford, *A global geometric framework for nonlinear dimensionality reduction*, Science, 290 (2000), pp. 2319–2323.
[43] J. L. Vázquez, *The porous medium equation*, Oxford Mathematical Monographs, The Clarendon Press, Oxford University Press, Oxford, 2007. Mathematical theory.
[44] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, and L. Bottou, *Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion.*, Journal of machine learning research, 11 (2010).
[45] C. Vreugdenhil and B. Koren, *Numerical methods for advection- diffusion problems*, Notes on Numerical Fluid Mechanics, (1993).
[46] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl, *Constrained k-means clustering with background knowledge*, in ICML, vol. 1, 2001, pp. 577–584.
[47] J. Wu, K. Long, F. Wang, C. Qian, C. Li, Z. Lin, and H. Zha, *Deep comprehensive correlation mining for image clustering*, in Proceedings of the IEEE International Conference on Computer Vision, 2019, pp. 8150–8159.
[48] J. Xie, R. Girshick, and A. Farhadi, *Unsupervised deep embedding for clustering analysis*, in International conference on machine learning, 2016, pp. 478–487.
[49] J. Yang, D. Parikh, and D. Batra, *Joint unsupervised learning of deep representations and image clusters*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 5147–5156.
[50] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, *Deconvolutional networks*, in 2010 IEEE Computer Society Conference on computer vision and pattern recognition, IEEE, 2010, pp. 2528–2535.
[51] L. Zelnik-Manor and P. Perona, *Self-tuning spectral clustering*, in Advances in neural information processing systems, 2005, pp. 1601–1608.
[52] D. Zhang, Z. Zhou, and S. Chen, *Semi-supervised dimensionality reduction*, in Proceedings of the 2007 SIAM International Conference on Data Mining, 2007, pp. 629–634.
[53] W. Zhao and S. Du, *Spectral–spatial feature extraction for hyperspectral image classification: a dimension reduction and deep learning approach*, IEEE Transactions on Geoscience and Remote Sensing, 54 (2016), pp. 4544–4554.
[54] X. Zhong and D. Enke, *Forecasting daily stock market return using dimensionality reduction*, Expert Systems with Applications, 67 (2017), pp. 126–139.