Neural Networks 33 (2012) 136–147

Contents lists available at SciVerse ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Complex-valued autoencoders

Pierre Baldi [a,*], Zhiqin Lu [b]

[a] Department of Computer Science, UCI, Irvine, CA 92697-3435, United States
[b] Department of Mathematics, UCI, Irvine, CA 92697-3875, United States

ARTICLE INFO

Article history:
Received 18 August 2011
Received in revised form 20 April 2012
Accepted 22 April 2012

Keywords:
Autoencoders
Unsupervised learning
Complex numbers
Complex neural networks
Critical points
Linear networks
Principal component analysis
EM algorithm
Deep architectures
Differential geometry

ABSTRACT

Autoencoders are unsupervised machine learning circuits, with typically one hidden layer, whose learning goal is to minimize an average distortion measure between inputs and outputs. Linear autoencoders correspond to the special case where only linear transformations between visible and hidden variables are used. While linear autoencoders can be defined over any field, only real-valued linear autoencoders have been studied so far. Here we study complex-valued linear autoencoders where the components of the training vectors and adjustable matrices are defined over the complex field with the L_2 norm. We provide simpler and more general proofs that unify the real-valued and complex-valued cases, showing that in both cases the landscape of the error function is invariant under certain groups of transformations. The landscape has no local minima, a family of global minima associated with Principal Component Analysis, and many families of saddle points associated with orthogonal projections onto sub-space spanned by sub-optimal subsets of eigenvectors of the covariance matrix. The theory yields several iterative, convergent, learning algorithms, a clear understanding of the generalization properties of the trained autoencoders, and can equally be applied to the hetero-associative case when external targets are provided. Partial results on deep architecture as well as the differential geometry of autoencoders are also presented. The general framework described here is useful to classify autoencoders and identify general properties that ought to be investigated for each class, illuminating some of the connections between autoencoders, unsupervised learning, clustering, Hebbian learning, and information theory.

1. Introduction

Autoencoder circuits, which try to minimize a distortion measure between inputs and outputs, play a fundamental role in machine learning. They were introduced in the 1980s by the Parallel Distributed Processing (PDP) group (Rumelhart, Hinton, & Williams, 1986) as a way to address the problem of unsupervised learning, in contrast to supervised learning in backpropagation networks, by using the inputs as learning targets. More recently, autoencoders have been used extensively in the "deep architecture" approach (Bengio & LeCun, 2007; Erhan et al., 2010; Hinton, Osindero, & Teh, 2006; Hinton & Salakhutdinov, 2006), where autoencoders in the form of Restricted Boltzmann Machines (RBMs) are stacked and trained bottom up in unsupervised fashion to extract hidden features and efficient representations that can then be used to address supervised classification or regression tasks. In spite of the interest they have generated, and with a few exceptions (Roux & Bengio, 2010), little theoretical understanding of autoencoders and deep architectures has been obtained to date. One possible strategy for addressing these issues is to partition the autoencoder universe into different classes, for instance linear versus non-linear autoencoders, and identify classes that can be analyzed mathematically, with the hope that the precise understanding of several specific classes may lead to a clearer general picture. Within this background and strategy, the main purpose of this article is to provide a complete theory for a particular class of autoencoders, namely, linear autoencoders over the complex field.

In addition to trying to progressively derive a more complete theoretical understanding of autoencoders, there are several other reasons, primarily theoretical ones, for looking at linear complex-valued autoencoders. First, linear autoencoders over the real numbers were solved by Baldi and Hornik (1988) (see also Bourlard & Kamp, 1988). It is thus natural to ask whether linear autoencoders over the complex numbers share the same basic properties or not, and whether unified proofs can be derived to cover both the real- and complex-valued cases. More generally, linear autoencoders can be defined over any field and therefore one can raise similar questions for linear autoencoders over other fields, such as finite Galois fields (Lang, 1984).

Second, a specific class of non-linear autoencoders was recently introduced and analyzed mathematically (Baldi, in press). This is the class of Boolean autoencoders where all circuit operations are

* Corresponding author. Tel.: +1 9498245809; fax: +1 9498249813.
E-mail address: pfbaldi@ics.uci.edu (P. Baldi).

0893-6080/$ – see front matter © 2012 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2012.04.011

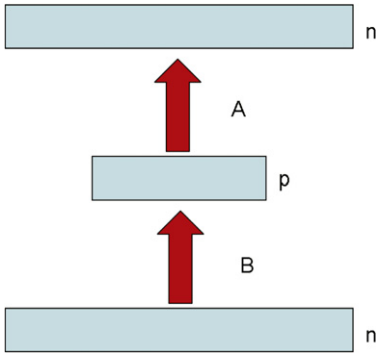

Fig. 1. An $n/p/n$ autoencoder architecture.

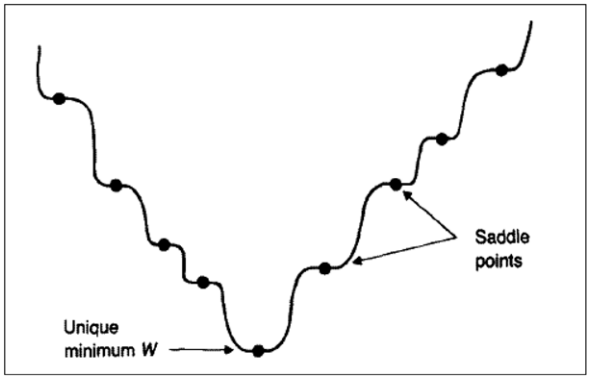

Fig. 2. Landscape of E.

Boolean functions. It can be shown that this class of autoencoders is intimately connected to clustering and so it is reasonable to both compare Boolean autoencoders to linear autoencoders, and to examine linear autoencoders from a clustering perspective.

Third, there has been a trend in recent years towards the use of linear networks and methods to address difficult tasks, such as building recommender systems (e.g. the Netflix prize challenge Bell & Koren, 2007; Takács, Pilászy, Németh, & Tikk, 2008) or modeling the development of sensory systems, in clever ways by introducing particular restrictions on the relevant matrices, such as sparsity or low-rank (Candès & Recht, 2009; Candes & Wakin, 2008). Autoencoders discussed in this paper can be viewed as linear, low-rank, approximations to the identity function and therefore fall within this general trend.

Finally, complex vector spaces and matrices have several areas of specific application, ranging from quantum mechanics, to fast Fourier transforms, to complex-valued neural networks (Hirose, 2003), and ought to be studied in their own right. Complex-valued linear autoencoders can be viewed as a particular class of complex-valued neural networks and may be used in applications involving complex-valued data.

With these motivations in mind, in order to provide a complete treatment of linear complex-valued autoencoders here we first introduce a general framework and notation, essential for a better understanding and classification of autoencoders, and for the identification of common properties that ought to be studied in any new specific autoencoder case. We then proceed to analytically solve the complex-valued linear autoencoder. While in the end the results obtained in the complex-valued case are similar to those previously obtained in the real-valued case (Baldi & Hornik, 1988) interchanging conjugate transposition with simple transposition, the approach adopted here allow us to derive simpler and more general proofs that unify both cases. In addition, we derive several new properties and results, addressing for instance learning algorithms and their convergence properties, and some of the connections to clustering, deep architectures, and other kinds of autoencoders. Finally, in the Appendix, we begin the study of real- and complex-valued autoencoders from a differential geometry perspective.

2. General autoencoder framework and preliminaries

2.1. General autoencoder framework

To derive a fairly general framework, an $n/p/n$ autoencoder (Fig. 1) is defined by a t-uple $\mathbb{F}$, $\mathbb{G}$, n, p, $\mathcal{A}$, $\mathcal{B}$, $\mathcal{X}$, Δ where:

1. $\mathbb{F}$ and $\mathbb{G}$ are sets.
2. n and p are positive integers. Here we consider primarily the case where $0 < p < n$.
3. $\mathcal{A}$ is a class of functions from $\mathbb{G}^p$ to $\mathbb{F}^n$.
4. $\mathcal{B}$ is a class of functions from $\mathbb{F}^n$ to $\mathbb{G}^p$.
5. $\mathcal{X} = \{x_1, \ldots, x_m\}$ is a set of m (training) vectors in $\mathbb{F}^n$. When external targets are present, we let $\mathcal{Y} = \{y_1, \ldots, y_m\}$ denote the corresponding set of target vectors in $\mathbb{F}^n$.
6. Δ is a dissimilarity or distortion function defined over $\mathbb{F}^n$.

For any $A \in \mathcal{A}$ and $B \in \mathcal{B}$, the autoencoder transforms an input vector $x \in \mathbb{F}^n$ into an output vector $A \circ B(x) \in \mathbb{F}^n$ (Fig. 2). The corresponding *autoencoder problem* is to find $A \in \mathcal{A}$ and $B \in \mathcal{B}$ that minimize the overall distortion (or error/energy) function:

$$\min E(A, B) = \min_{A,B} \sum_{t=1}^{m} E(x_t) = \min_{A,B} \sum_{t=1}^{m} \Delta\big(A \circ B(x_t), x_t\big). \tag{1}$$

In the non-auto-associative case, when external targets y_t are provided, the minimization problem becomes:

$$\min E(A, B) = \min_{A,B} \sum_{t=1}^{m} E(x_t) = \min_{A,B} \sum_{t=1}^{m} \Delta\big(A \circ B(x_t), y_t\big). \tag{2}$$

Note that $p < n$ corresponds to the regime where the autoencoder tries to implement some form of compression or feature extraction. The case $p > n$ is not treated here but can be interesting in situations which either (1) prevent the use of trivial solutions by enforcing additional constraints, such as sparsity, or (2) include noise in the hidden layer, corresponding to transmission over a noisy channel.

Obviously, from this general framework, different kinds of autoencoders can be derived depending, for instance, on the choice of sets $\mathbb{F}$ and $\mathbb{G}$, transformation classes $\mathcal{A}$ and $\mathcal{B}$, distortion function Δ, as well as the presence of additional constraints. Linear autoencoders correspond to the case where $\mathbb{F}$ and $\mathbb{G}$ are fields and $\mathcal{A}$ and $\mathcal{B}$ are the classes of linear transformations, hence A and B are matrices of size $n \times p$ and $p \times n$ respectively. The linear real case where $\mathbb{F} = \mathbb{G} = \mathbb{R}$ and Δ is the squared Euclidean distance was addressed in Baldi and Hornik (1988) (see also Bourlard & Kamp, 1988).

2.2. Complex linear autoencoder

Here we consider the corresponding *complex* linear case where $\mathbb{F} = \mathbb{G} = \mathbb{C}$ and the goal is the minimization of the squared Euclidean distance

$$\begin{aligned} \min E(A, B) &= \min_{A,B} \sum_{t=1}^{m} ||x_t - AB(x_t)||^2 \\ &= \sum_{t=1}^{m} (x_t - AB(x_t))^*(x_t - AB(x_t)). \end{aligned} \tag{3}$$

Unless otherwise specified, all vectors are column vectors and we use x^* (resp. X^*) to denote the conjugate transpose of a vector x

(resp. of a matrix X). Note that the same notation works for both the complex and real case. As we shall see, in the linear complex case as in the linear real case, one can also address the case where external targets are available, in which case the goal is the minimization of the distance

$$\min E(A, B) = \min_{A,B} \sum_{t=1}^{m} ||y_t - AB(x_t)||^2$$
$$= \sum_{t=1}^{m} (y_t - AB(x_t))^*(y_t - AB(x_t)). \quad (4)$$

In practical applications, it is often preferable to work with centered data, after subtraction of the mean. The centered and non-centered versions of the problem are two different problems with in general two different solutions. The general equations to be derived apply equally to both cases.

In general, we define the covariance matrices as follows

$$\Sigma_{XY} = \sum_t x_t y_t^*. \quad (5)$$

Using this definition, Σ_{XX}, Σ_{YY} are Hermitian matrices $(\Sigma_{XX})^* = \Sigma_{XX}$ and $(\Sigma_{YY})^* = \Sigma_{YY}$, and $(\Sigma_{XY})^* = \Sigma_{YX}$. We let also

$$\Sigma = \Sigma_{YX} \Sigma_{XX}^{-1} \Sigma_{XY}. \quad (6)$$

Σ is also Hermitian. In the *auto-associative* case, $x_t = y_t$ for all t resulting in $\Sigma = \Sigma_{XX}$. Note that any Hermitian matrix admits a set of orthonormal eigenvectors and all its eigenvalues are real. Finally, we let I_m denote the $m \times m$ identity matrix.

For several results, we make the assumption that Σ is invertible. This is not a very restrictive assumption for several reasons. First, by adding a small amount of noise to the data, a non-invertible Σ could be converted to an invertible Σ, although this could potentially raise some numerical issues. More importantly, in most settings one can expect the training vectors to span the entire input space and thus Σ to be invertible. If the training vectors span a smaller subspace, then the original problem can be transformed to an equivalent problem defined on the smaller subspace.

2.3. Useful reminders

Standard linear regression. Consider the standard linear regression problem of minimizing $E(B) = \sum_t ||y_t - Bx_t||^2$, where B is a $p \times n$ matrix, corresponding to a linear neural network without any hidden layers. Then we can write

$$E(B) = \sum_t x_t^* B^* B x_t - 2\mathrm{Re}\,(y_t^* B x_t) + \|y_t\|^2. \quad (7)$$

Thus E is a convex function in B because the associated quadratic form is equal to

$$\sum_t x_t^* C^* C x_t = \sum_t \|Cx_t\|^2 \geq 0. \quad (8)$$

Let B be a critical point. Then by definition for any $p \times n$ matrix C we must have $\lim_{\epsilon \to 0}[E(B + \epsilon C) - E(B)]/\epsilon = 0$. Expanding and simplifying this expression gives

$$\sum_t x_t^* B^* C x_t - y_t^* B C x_t = 0 \quad (9)$$

for all $p \times n$ matrices C. Using the linearity of the trace operator and its invariance under circular permutation of its arguments,[1] this is equivalent to

$$\mathrm{Tr}\,((\Sigma_{XX} B^* - \Sigma_{XY})C) = 0 \quad (10)$$

for any C. Thus we have $\Sigma_{XX} B^* - \Sigma_{XY} = 0$ and therefore

$$B \Sigma_{XX} = \Sigma_{YX}. \quad (11)$$

If Σ_{XX} is invertible, then $Cx_t = 0$ for any t is equivalent to $C = 0$, and thus the function $E(B)$ is strictly convex in B. The unique critical point is the global minimum given by $B = \Sigma_{YX} \Sigma_{XX}^{-1}$. As we shall see, the solution to the standard linear regression problem, together with the general approach given here to solve it, is also key for solving the more general linear autoencoder problem. The solution will also involve projection matrices.

Projection matrices. For any $n \times k$ matrix A with $k \leq n$, let P_A denote the orthogonal projection onto the subspace generated by the columns of A. Then P_A is a Hermitian symmetric matrix and $P_A^2 = P_A$, $P_A A = A$ since the image of P_A is spanned by the columns of A and these are invariant under P_A. The kernel of P_A is the space $A^\perp$ orthogonal to the space spanned by the columns of A. Obviously, we have $P_A A^\perp = 0$ and $A^* P_A = A^*$. The projection onto the space orthogonal to the space spanned by the columns of A is given by $I_n - P_A$. In addition, if the columns of A are independent (i.e. A has full rank k), then the matrix of the orthogonal projection is given by $P_A = A(A^*A)^{-1}A^*$ (Meyer, 2000) and $P_A^* = P_A$. Note that all these relationships are true even when the columns of A are not orthonormal.

2.4. Some misconceptions

As we shall see, in the complex case as in the real case, the global minimum corresponds to Principal Component Analysis. While the global minimum solution of linear autoencoders over infinite fields can be expressed analytically, it is often not well appreciated that there is more to be understood about linear autoencoders and the landscape of E. In particular, if one is interested in learning algorithms that proceed through incremental and somewhat "blind" weight adjustments, then one must study the entire landscape of E, including all the critical points of E, and derive and compare different learning algorithms. A second misconception is to believe that the problem is a convex optimization problem, hence somewhat trivial, since after all the error function is quadratic and the transformation $W = AB$ is linear. The problem with this argument is that the small layer of size p forces W to be of rank p or less, and the set of matrices or rank at most p is *not* convex. Furthermore, the problem is not convex when finite fields are considered. What is true and crucial for solving the linear autoencoders over infinite fields is that the problem becomes convex when A or B is fixed. A third misconception, related to the illusion of convexity, is that the L_2 landscape of linear neural networks never has any local minima. In general this is not true, especially if there are additional constraints on the linear transformation, such as restricted connectivity between layers so that some of the matrix entries are constrained to assume fixed values.

3. Group invariances

For any autoencoder, it is important to investigate whether there are any group of transformations that leave its properties invariant.

Change of coordinates in the hidden layer. Note that for any invertible $p \times p$ complex matrix C, we have $W = AB = ACC^{-1}B$ and $E(A, B) = E(AC, C^{-1}B)$. Thus all the properties of the linear autoencoder are fundamentally invariant with respect to any change of coordinates in the hidden layer.

Change of coordinates in the input/output spaces. Consider an orthonormal change of coordinates in the output space defined by an orthogonal (or unitary) $n \times n$ matrix D, and any change of

[1] It is easy to show directly that for any matrices A and B of the proper size, $\mathrm{Tr}(AB) = \mathrm{Tr}(BA)$ (Lang, 1984). Therefore for any matrices A, B, and C of the proper size, we have $\mathrm{Tr}(ABC) = \mathrm{Tr}(CAB) = \mathrm{Tr}(BCA)$.

coordinates in the input space defined by an invertible $n \times n$ matrix C. This leads to a new autoencoder problem with input vectors $Cx_1, \ldots, Cx_m$ and target output vectors of the form $Dy_1, \ldots, Dy_m$ with reconstruction error of the form

$$E(A', B') = \sum_t ||Dy_t - A'B'Cx_t||^2. \tag{12}$$

If we use the one-to-one mapping between pairs of matrices (A, B) and (A', B') defined by $A' = DA$ and $B' = BC^{-1}$, we have

$$\begin{aligned} E(A', B') &= \sum_t ||Dy_t - A'B'Cx_t||^2 \\ &= \sum_t ||Dy_t - DABx_t||^2 = \sum_t ||y_t - ABx_t||^2 \end{aligned} \tag{13}$$

the last equality using the fact that D is an isometry which preserves distances. Thus, using the transformation $A' = DA$ and $B' = BC^{-1}$ the original problem and the transformed problem are equivalent and the function $E(A, B)$ and $E(A', B')$ have the same landscape. In particular, in the auto-associative case, we can take $C = D$ to be a unitary matrix. This leads to an equivalent autoencoder problems with input vectors Cx_t and covariance matrix $C\Sigma C^{-1}$. For the proper choice of C there is an equivalent problem where basis of the space is provided by the eigenvectors of the covariance matrix and the covariance matrix is a diagonal matrix with diagonal entries equal to the eigenvalues of the original covariance matrix Σ.

4. Fixed-layer and convexity results

A key technique for studying any autoencoder, is to simplify the problem by fixing all its transformations but one. Thus in this section we study what happens to the complex-valued linear autoencoder problem when either A or B is fixed, essentially reducing the problem to standard linear regression. The same approach can be applied to an autoencoder with more than one hidden layer (see section on Deep Architectures).

Theorem 1 (*Fixed A*)**.** *For any fixed $n \times p$ matrix A, the function $E(A, B)$ is convex in the coefficients of B and attains its minimum for any B satisfying the equation*

$$A^*AB\Sigma_{XX} = A^*\Sigma_{YX}. \tag{14}$$

If Σ_{XX} is invertible and A is of full rank p, then E is strictly convex and has a unique minimum reached when

$$B = (A^*A)^{-1}A^*\Sigma_{YX}\Sigma_{XX}^{-1}. \tag{15}$$

In the auto-associative case, if Σ_{XX} is invertible and A is of full rank p, then the optimal B has full rank p and does not depend on the data. It is given by

$$B = (A^*A)^{-1}A^* \tag{16}$$

and in this case, $W = AB = A(A^*A)^{-1}A^* = P_A$ and $BA = I_p$.

Proof. We write

$$E(A, B) = \sum_t x_t^*B^*A^*ABx_t - 2\mathrm{Re}\,(y_t^*ABx_t) + \|y_t\|^2. \tag{17}$$

Then for fixed A, E is a convex function because the associated quadratic form is equal to

$$\sum_t x_t^*C^*A^*ACx_t = \sum_t \|ACx_t\|^2 \geq 0 \tag{18}$$

for any $p \times n$ matrix C. Let B be a critical point. Then by definition for any $p \times n$ matrix C we must have $\lim_{\epsilon \to 0}[E(A, B + \epsilon C) - E(A, B)]/\epsilon = 0$. Expanding and simplifying this expression gives

$$\sum_t x_t^*B^*A^*ACx_t - y_t^*ACx_t = 0 \tag{19}$$

for all $p \times n$ matrices C. Using the linearity of the trace operator and its invariance under circular permutation of its arguments, this is equivalent to

$$\mathrm{Tr}\,((\Sigma_{XX}B^*A^*A - \Sigma_{XY}A)C) = 0 \tag{20}$$

for any C. Thus we have $\Sigma_{XX}B^*A^*A - \Sigma_{XY}A = 0$ and therefore

$$A^*AB\Sigma_{XX} = A^*\Sigma_{YX}. \tag{21}$$

Finally, if Σ_{XX} is invertible and if A is of full rank, then $ACx_t = 0$ for any t is equivalent to $C = 0$, and thus the function $E(A, B)$ is strictly convex in B. Since A^*A is invertible, the unique critical point is obtained by solving Eq. (14). □

In similar fashion, we have the following theorem.

Theorem 2 (*Fixed B*)**.** *For any fixed $p \times n$ matrix B, the function $E(A, B)$ is convex in the coefficients of A and attains its minimum for any A satisfying the equation*

$$AB\Sigma_{XX}B^* = \Sigma_{YX}B^*. \tag{22}$$

If Σ_{XX} is invertible and B is of full rank, then E is strictly convex and has a unique minimum reached when

$$A = \Sigma_{YX}B^*(B\Sigma_{XX}B^*)^{-1}. \tag{23}$$

In the auto-associative case, if Σ_{XX} is invertible and B is of full rank, then the optimal A has full rank p and depends on the data. It is given by

$$A = \Sigma_{XX}B^*(B\Sigma_{XX}B^*)^{-1} \tag{24}$$

and $BA = I_p$.

Proof. From Eq. (17), the function $E(A, B)$ is a convex function in A. The condition for A to be a critical point is

$$\sum_t x_t^*B^*A^*CBx_t - y_t^*CBx_t = 0 \tag{25}$$

for any $p \times n$ matrix C, which is equivalent to

$$\mathrm{Tr}\,((B\Sigma_{XX}B^*A^* - B\Sigma_{XY})C) = 0 \tag{26}$$

for any matrix C. Thus $B\Sigma_{XX}B^*A^* - B\Sigma_{XY} = 0$ which implies Eq. (22). The other assertions of the theorem can easily be deduced. □

Remark 1. Note that from Theorems 1 and 2 and their proofs, we have that (A, B) is a critical point of $E(A, B)$ *if and only if* Eq. (14) and Eq. (22) are simultaneously satisfied, that is if and only if $A^*AB\Sigma_{XX} = A^*\Sigma_{YX}$ and $AB\Sigma_{XX}B^* = \Sigma_{YX}B^*$.

5. Critical points and the landscape of *E*

In this section we further study the landscape of E, its critical points, and the properties of $W = AB$ at those critical points.

Theorem 3 (*Critical Points*)**.** *Assume that Σ_{XX} is invertible. Then two matrices (A, B) define a critical point of E, if and only if the global map $W = AB$ is of the form*

$$W = P_A\Sigma_{YX}\Sigma_{XX}^{-1} \tag{27}$$

with A satisfying

$$P_A \Sigma = P_A \Sigma P_A = \Sigma P_A. \tag{28}$$

In the auto-associative case, the above becomes

$$W = AB = P_A \tag{29}$$

and

$$P_A \Sigma_{XX} = P_A \Sigma_{XX} P_A = \Sigma_{XX} P_A. \tag{30}$$

If A is of full rank, then the pair (A, B) defines a critical point of E if and only if A satisfies Eq. (28) and B satisfies Eq. (16). Hence B must also be of full rank.

Proof. If (A, B) is a critical point of E, then from Eq. (14), we must have

$$A^*(AB - \Sigma_{YX}\Sigma_{XX}^{-1}) = 0. \tag{31}$$

Let

$$S = AB - P_A \Sigma_{YX} \Sigma_{XX}^{-1}. \tag{32}$$

Then since $A^* P_A = A^*$, we have $A^* S = 0$. Thus the space spanned by the columns of S is a subset of the space orthogonal to the space spanned by the columns of A (i.e. $S \in A^{\perp}$). On the other hand, since

$$P_A S = S. \tag{33}$$

S is also in the space spanned by the columns of A (i.e. $S \in \mathrm{Span}(A)$). Taken together, these two facts imply that $S = 0$, resulting in $W = AB = P_A \Sigma_{YX} \Sigma_{XX}^{-1}$, which proves Eq. (27). Note that for this result, we need only B to be critical (i.e. optimized with respect to A). Using the definition of Σ, we have

$$P_A \Sigma P_A = P_A \Sigma_{YX} \Sigma_{XX}^{-1} \Sigma_{XX} \Sigma_{XX}^{-1} \Sigma_{XY} P_A. \tag{34}$$

Since $S = 0$, we have $AB = P_A \Sigma_{YX} \Sigma_{XX}^{-1}$ and thus

$$P_A \Sigma P_A = P_A \Sigma_{YX} \Sigma_{XX}^{-1} \Sigma_{XX} \Sigma_{XX}^{-1} \Sigma_{XY} P_A = AB \Sigma_{XX} B^* A^*. \tag{35}$$

Similarly, we have

$$P_A \Sigma = AB \Sigma_{XY} \tag{36}$$

and

$$\Sigma P_A = \Sigma_{YX} B^* A^*. \tag{37}$$

Then Eq. (28) results immediately by combining Eqs. (35)–(37) using Eq. (22). The rest of the theorem follows easily. □

Remark 2. The above proof unifies the cases when AB is of rank p and less than p and avoids the need for two separate proofs, as was done in earlier work (Baldi & Hornik, 1988) for the real-valued case.

Theorem 4 (*Critical Points of Full Rank*)**.** *Assume that Σ is of full rank with n distinct eigenvalues $\lambda_1 > \cdots > \lambda_n$ and let $u_1, \ldots, u_n$ denote a corresponding basis of orthonormal eigenvectors. If $\mathcal{I} = \{i_1, \ldots, i_p\} (1 \le i_1 < \cdots < i_p \le n)$ is any ordered set of indices of size p, let $U_{\mathcal{I}} = (u_{i_1}, \ldots, u_{i_p})$ denote the matrix formed using the corresponding column eigenvectors. Then two full rank matrices A, B define a critical point of E if and only if there exists an ordered p-index set $\mathcal{I}$ and an invertible $p \times p$ matrix C such that*

$$A = U_{\mathcal{I}} C \quad \text{and} \quad B = C^{-1} U_{\mathcal{I}}^* \Sigma_{YX} \Sigma_{XX}^{-1}. \tag{38}$$

For such critical point, we have

$$W = AB = P_{U_{\mathcal{I}}} \Sigma_{YX} \Sigma_{XX}^{-1} \tag{39}$$

and

$$E(A, B) = \mathrm{Tr}\, \Sigma_{YY} - \sum_{i \in \mathcal{I}} \lambda_i. \tag{40}$$

In the auto-associative case, these equations reduce to

$$A = U_{\mathcal{I}} C \quad \text{and} \quad B = C^{-1} U_{\mathcal{I}}^* \tag{41}$$

$$W = AB = P_{U_{\mathcal{I}}} \tag{42}$$

and

$$E(A, B) = \mathrm{Tr}\, \Sigma - \sum_{i \in \mathcal{I}} \lambda_i = \sum_{i \in \bar{\mathcal{I}}} \lambda_i \tag{43}$$

where $\bar{\mathcal{I}} = \{1, \ldots, n\} \setminus \mathcal{I}$ is the complement of $\mathcal{I}$.

Proof. Since $P_A \Sigma = \Sigma P_A$, we have

$$P_A \Sigma A = \Sigma P_A A = \Sigma A. \tag{44}$$

Thus the columns of A form an invariant space of Σ. Thus A is of the form $U_{\mathcal{I}} C$. The conclusion for B follows from Eq. (27) and the rest is easily deduced, as in the real case. Eq. (43) can be derived easily by using the remarks in Section 3 and using the unitary change of coordinates under which Σ_{XX} becomes a diagonal matrix. In this system of coordinates, we have

$$E(A, B) = \sum_t \|y_t\|^2 + \sum_t \mathrm{Tr}\,(x_t^* (AB)^* AB x_t) - 2 \sum_t \mathrm{Tr}\,(y_t^* AB x_t).$$

Therefore, using the invariance property of the trace under permutation, we have

$$E(A, B) = \mathrm{Tr}\,(\Sigma) + \mathrm{Tr}\,((AB)^2 \Sigma) - 2\mathrm{Tr}\,(AB\Sigma).$$

Since AB is a projection operator, this yields Eq. (43). In the auto-associative case with these coordinates it is easy to see that $W(x_t)$ and $E(A, B) = \sum_t E(x_t)$ are easily computed from the values of $W(u_i)$. In particular, $E(A, B) = \sum_{i=1}^n \lambda_i (u_i - W(u_i))^2$. In addition, at the critical points, we have $W(u_i) = u_i$ if $i \in I$, and $W(u_i) = 0$ otherwise. □

Remark 3. All the previous theorems are true in the hetero-associative case with targets y_t. Thus they can readily be applied to address the linear denoising autoencoder (Vincent, 2011; Vincent, Larochelle, Bengio, & Manzagol, 2008) over $\mathbb{R}$ or $\mathbb{C}$. The linear denoising autoencoder is an autoencoder trained to remove noise by having to associate noisy versions of the inputs with the correct inputs. In other words, using the current notation, it is an autoencoder where the inputs x_t are replaced by $x_t + n_t$ where n_t is the noise vector and the target outputs y_t are of the form $y_t = x_t$. Thus the previous theorems can be applied using the following replacements: $\Sigma_{XX} = \Sigma_{XX} + \Sigma_{NN} + \Sigma_{NX} + \Sigma_{XN}$, $\Sigma_{XY} = \Sigma_{XX} + \Sigma_{NX}$, $\Sigma_{YX} = \Sigma_{XX} + \Sigma_{XN}$. Further simplifications can be obtained using particular assumptions on the noise, such as $\Sigma_{NX} = \Sigma_{XN} = 0$.

Theorem 5 (*Absence of Local Minima*)**.** *The global minimum of the complex linear autoencoder is achieved by full rank matrices A and B associated with the index set $1, \ldots, p$ of the p largest eigenvalues of Σ with $A = U_{\mathcal{I}} C$ and $B = C^{-1} U_{\mathcal{I}}^*$ (and where C is any invertible $p \times p$ matrix). When $C = I$, $A = B^*$. All other critical points are saddle points associated with corresponding projections onto non-optimal sets of eigenvectors of Σ of size p or less.*

Proof. The proof is by a perturbation argument, as in the real case, showing that critical points that are not associated with the global minimum there is always a direction of escape that can be derived using unused eigenvectors associated with higher eigenvalues in order to lower the error E (see Baldi & Hornik, 1988 for more details). The proof can be made very simple by using the group

invariance properties under transformation of the coordinates by a unitary matrix. With such a transformation, it is sufficient to study the landscape of E when Σ is a diagonal matrix and $A = B^* = U_1$. □

Remark 4. At the global minimum, if C is the $p \times p$ identity matrix ($C = I$), in the auto-associative case then the activities in the hidden layer are given by $u_1^* x, \ldots, u_p^* x$, corresponding to the coordinates of x along the first p eigenvectors of Σ_{XX}. These are the so called principal components of x and the autoencoder implements a form of Principal Component Analysis (PCA) also closely related to Singular Value Decomposition (SVD).

The theorem above shows that when Σ is full rank, there is a special class of critical points associated with $C = I$. In the auto-associative case, this class is characterized by the fact that A and B are conjugate transpose of each other ($A = B^*$) in the complex-valued case, or transpose of each other ($A = B^*$) in the real-valued case. This class of critical points is special for several reasons. For instance, in the related Restricted Boltzmann Machine Autoencoders the weights between visible and hidden units are require to be symmetric corresponding to $A = B^*$. More importantly, these critical points are closely connected to Hebbian learning (see also Oja, 1982, 1989, 1992). In particular, for linear real-valued autoencoders, if $A = B^*$ and $E = 0$ so that inputs are equal to outputs, any learning rule that is symmetric with respect to the pre- and post-synaptic activities – which is typically the case for Hebbian rules – will modify A and B but preserve the property that $A = B^*$. This remains roughly true even if E is not exactly zero. Thus for linear real-valued autoencoders, there is something special about transposition operating on A and B and more generally on can suspect a similar role is played by conjugate transposition in the case of linear complex-valued autoencoders. The next theorem and the following section on learning algorithm further clarify this point.

Theorem 6 (*Conjugate Transposition*)**.** *Assume Σ_{XX} is of full rank in the auto-associative case. Consider any point (A, B) where B has been optimized with respect to A, including all critical points. Then*

$$W = AB = B^*A^*AB = B^*A^* = W^* \quad \text{and} \quad E(A, B) = E(B^*, A^*). \tag{45}$$

Furthermore, when A is full rank

$$W = P_A = P_A^* = W^*. \tag{46}$$

Proof. By Theorem 1, in the auto-associate case, we have

$$A^*AB = A^*.$$

Thus, by taking the complex conjugate of each side, we have

$$B^*A^*A = A.$$

It follows that

$$B^*A^* = B^*A^*AB = AB$$

which proves Eq. (45). If in addition A is full rank, then by Theorem 1 $W = AB = P_A$ and the rest follows immediately. □

Remark 5. Note the following. Starting from a pair (A, B) with $W = AB$ and where B has been optimized with respect to A, let $A' = B^*$ and optimize B again so that $B' = (A'A'^*)^{-1}A'^*$. Then we also have

$$W' = A'B' = W^* = W = P_A \quad \text{and} \quad E(A, B) = E(A', B'). \tag{47}$$

6. Optimization or learning algorithms

Although mathematical formula for the global minimum solution of the linear autoencoder have been derived, the global solution may not be available immediately to a self-adjusting learning circuit capable of making only small adjustments at each learning steps. Small adjustments may also be preferable in a non-stationary environment where the set $\mathcal{X}$ of training vectors changes with time. Furthermore, the study of small adjustment algorithms in linear circuits may shed some light on similar incremental algorithms applied to non-linear circuits where the global optimum cannot be derived analytically. Thus, from a learning algorithm standpoint, it is still useful to consider incremental optimization algorithms, such as gradient descent or partial EM steps, even when such algorithms are slower or less accurate than direct global optimization. The previous theorems suggest two kinds of operations that could be used in various combinations to iteratively minimize E, taking full or partial steps: (1) Partial minimization: fix A (resp. B) and minimize for B (resp. A); (2) Conjugate Transposition: fix A (resp. B), and set $B = A^*$ (resp. $A = B^*$) (the latter being reserved for the auto-associative case, and particularly so if one is interested in converging to solutions where A and B are conjugate transpose of each other, i.e. where $C = I$).

Theorem 7 (*Alternate Minimization*)**.** *Consider the algorithm where A and B are optimized in alternation (starting from A or B), holding the other one fixed. This algorithm will converge to a critical point of E. Furthermore, if the starting value of A or B is initialized randomly, then with probability one the algorithm will converge to a critical point where both A and B are full rank.*

Proof. A direct proof of convergence is given in Appendix B. Here we give an indirect, but perhaps more illuminating proof, by remarking that the alternate minimization algorithm is in fact an instance of the general EM algorithm (Dempster, Laird, & Rubin, 1977) combined with a hard decision, similar to the Viterbi learning algorithm for HMM or the k-means clustering algorithm with hard assignment. For this, consider that we have a probabilistic model over the data with parameters A and hidden variables B, or vice versa, with parameters B and hidden variables A. The conditional probability of the data and the hidden variables is given by:

$$P(\mathcal{X}, \mathcal{Y}, A|B) = \frac{1}{Z_1} e^{-E(A,B)} \tag{48}$$

or

$$P(\mathcal{X}, \mathcal{Y}, B|A) = \frac{1}{Z_2} e^{-E(A,B)} \tag{49}$$

where $Z1$ and Z_2 denote the proper normalizing constants (partition functions). During the E step, we find the most probable value of the hidden variables given the data and current value of the parameters. Since E is quadratic, the model in Eq. (49) is Gaussian and the mean and mode are identical. Thus the hard assignment of the hidden variables in the E step corresponds to optimizing A or B using Theorem 3 or Theorem 4. During the M step, the parameters are optimized given the value of the hidden variables. Thus the M step also corresponds to optimizing A or B using Theorem 3 or Theorem 4. As a result, convergence to a critical point of E is ensured by the general convergence theorem of the EM algorithm (Dempster et al., 1977). Since A and B are initialized randomly, they are full rank with probability one and, by Theorems 1 and 2 they retain their full rank after each optimization step. Note that the error E is always positive, strictly convex in A or B, decreases at each optimization step, and thus E must converge to a limit. By looking at every other step in the algorithm, it is easy to see that P_A must converge. From which one can see that A must converge, and so must B. □

Given the importance of conjugate transposition (Theorem 6) in the auto-associative case, one may also consider algorithms where

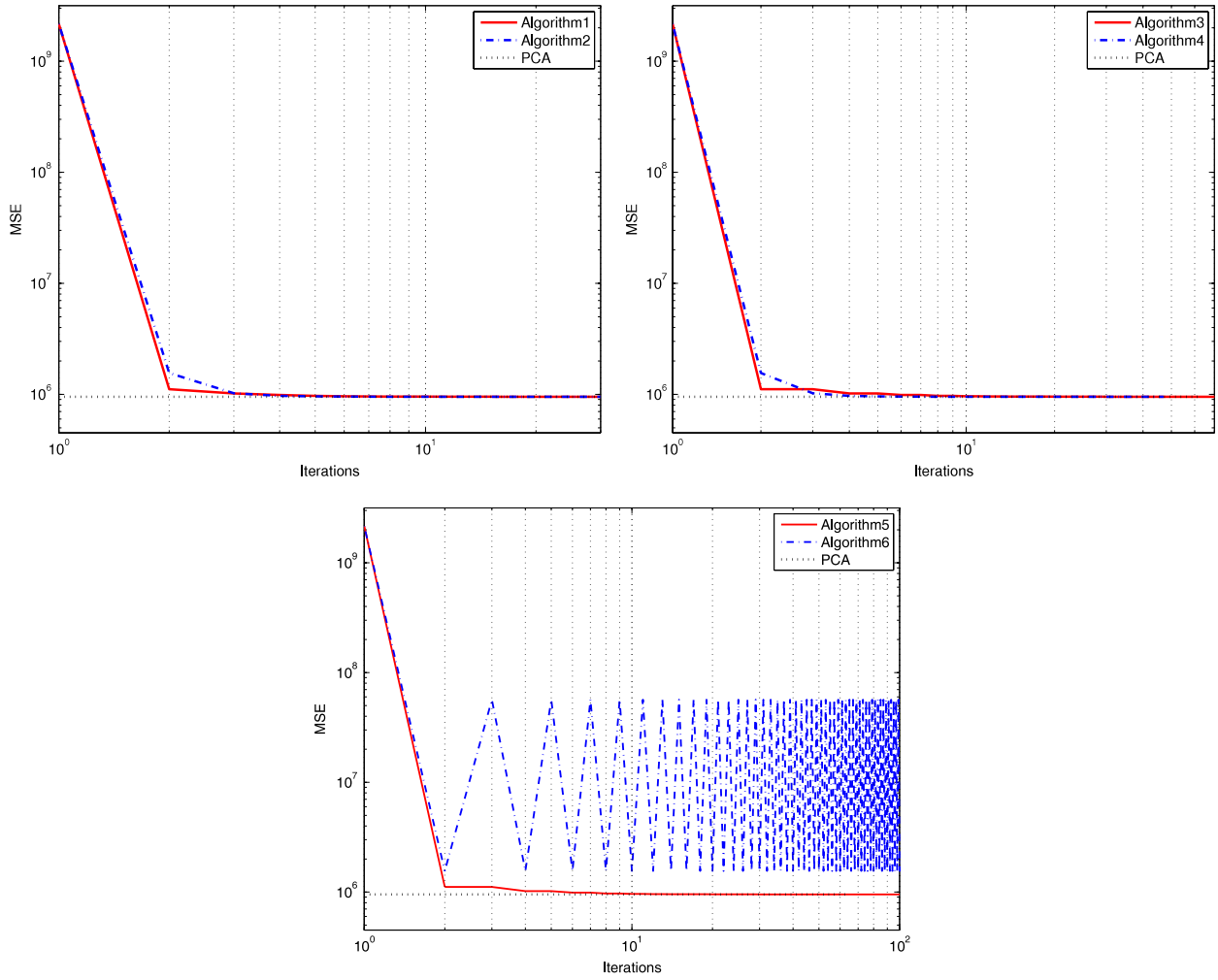

Fig. 3. Learning curves for Algorithm 1–6. The results are obtained using linear real-valued autoencoders of size 784-10-784 trained on images in the standard MNIST dataset for the digit "7" using 1000 samples. Each consecutive update of both A and B is considered as one iteration.

the operations of conjugate transposition and partial optimization of A and B are interleaved. This can be carried in many ways. Let $A \longrightarrow B$ denote that B is obtained from A by optimization (Eq. (16)) and $A \Longrightarrow B$ denote that B is obtained from A by conjugate transposition ($B = A^*$), and similarly for $B \longrightarrow A$ (Eq. (24)) and $B \Longrightarrow A$ ($A = B^*$). Let also $\Longleftrightarrow$ denote the operation where both A and B are obtained by simultaneous conjugate transposition from their current values. Then starting from (random) A and B, here are several possible algorithms:

- *Algorithm 1*: $B \longrightarrow A \longrightarrow B \longrightarrow A \longrightarrow B \ldots$.
- *Algorithm 2*: $A \longrightarrow B \longrightarrow A \longrightarrow B \longrightarrow A \ldots$.
- *Algorithm 3*: $B \longrightarrow A \longrightarrow B \Longrightarrow A \longrightarrow B \longrightarrow A \longrightarrow B \Longrightarrow A \ldots$.
- *Algorithm 4*: $A \longrightarrow B \longrightarrow A \Longrightarrow B \longrightarrow A \longrightarrow B \longrightarrow A \Longrightarrow B \ldots$.
- *Algorithm 5*: $B \longrightarrow A \longrightarrow B \Longleftrightarrow B \longrightarrow A \longrightarrow B \ldots$.
- *Algorithm 6*: $A \longrightarrow B \longrightarrow A \Longleftrightarrow A \longrightarrow B \longrightarrow A \Longleftrightarrow \cdots$.
- *Algorithm 7*: $A \longleftarrow B \Longleftrightarrow A \longleftarrow B \Longleftrightarrow \cdots$.

The theory presented so far allows us to understand their behavior easily (Fig. 3), considering a consecutive update of A and B as one iteration. Algorithms 1 and 2 converge with probability one to a critical point where A and B are full rank. Algorithm 1 may be slightly faster than Algorithm 2 at the beginning since in the first step Algorithm 1 takes into account the data (Eq. (24)), whereas Algorithm 2 ignores it. Algorithms 3, 4, and 5 converge and lead to a solution where $A = B^*$ (or, equivalently, $C = I$). Algorithms 3 and 5 take the same time and are faster than Algorithm 4. Algorithms 2 and 4 take the same time. Algorithm 3 requires almost twice the number of steps of Algorithm 1. But Algorithm 4 is faster than Algorithm 3. This is because in Algorithm 3, the steps $B \Longrightarrow A \longrightarrow B$ is basically like switching the matrices A and B, and the error after the step $B \longrightarrow A \longrightarrow B$ is the same as the error after the step $B \Longrightarrow A \longrightarrow B$. Algorithms 6 and 7 in general will not converge. Only optimization steps with respect to the B matrix are being carried and therefore the data is never considered.

7. Generalization properties

One of the most fundamental problems in machine learning is to understand the generalization properties of a learning system. Although in general this is not a simple problem, in the case of the autoencoder the generalization properties can easily be understood. After learning, A and B must be at a critical point. Assuming without much loss of generality that A is also full rank and Σ_{XX} is invertible, then from Theorem 1 we know in the auto-associative case that $W = P_A$. Thus we have the following result.

Theorem 8 (*Generalization Properties*)**.** *Assume in the auto-associative case that Σ_{XX} is invertible. For any learning algorithm that converges to a point where B is optimized with respect to A and A is full rank (including all full rank critical points), then for any vector x we have $Wx = ABx = P_A x$ and*

$$E(x) = ||x - ABx||^2 = ||x - P_A x||^2. \tag{50}$$

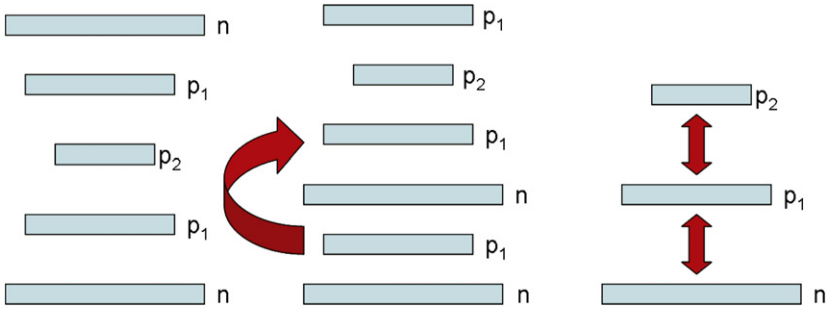

Fig. 4. Vertical composition of autoencoders.

Remark 6. Thus the reconstruction error of any vector is equal to the square of its distance to the subspace spanned by the columns of A, or the square of the norm of its projection onto the orthogonal subspace. The general hetero-associative case can also be treated using Theorem 1. In this case, under the same assumptions, we have: $W = P_A \Sigma_{YX} \Sigma_{XX}^{-1}$.

8. Recycling or iteration properties

Likewise, for the linear auto-associative case, one can also easily understand what happens when the outputs of the network are recycled into the inputs after learning. In the RBMs case, this is similar to alternatively sampling from the input and hidden layer. Interestingly, this provides also an alternative characterization of the critical points. At a critical points where W is a projection, we must have $W^2 = W$. Thus, after learning, the iterates $W^m x$ are easy to understand and converge after a single cycle and all points become stable after a single cycle. If x is in the space spanned by the columns of A we have $W^m(x) = x$ for any $m \geq 1$. If x is not in the space spanned by the columns of A, then $W^m x = y$ for $m \geq 2$, where y is the projection of x onto the space spanned by the columns of A ($Wx = P_A x = y$).

Theorem 9 (*Generalization Properties*)**.** *Assume in the auto-associative case that Σ_{XX} is invertible. For any learning algorithm that converges to a point where B is optimized with respect to A and A is full rank (including all full rank critical points), then for any vector x and any integer $m > 1$, we have*

$$W^m(x) = P_A^m(x) = P_A(x). \tag{51}$$

Remark 7. There is a partial converse to this result, in the following sense. Assume that W is a projection ($W^2 = W$) and therefore $ABAB = AB$. If A is of full rank, then $BAB = B$. Furthermore, if B is of full rank, then $BA = I_p$ (note that $BA = I_p$ immediately implies that $W^2 = W$). Multiplying this relation by A^*A on the left and A on the right, yields $A^*AB = A^*$ after simplification, and therefore $B = (A^*A)^{-1}A^*$. Thus according to Theorem 1 B is critical and $W = P_A$. Note that under the sole assumption that W is a projection, there is no reason for A to be critical, since there is no reason for A to depend on the data and on Σ_{XX}.

9. Deep architectures

Autoencoders can be composed vertically (Fig. 4), as in the deep architecture approach described in Hinton et al. (2006); Hinton and Salakhutdinov (2006), where a stack of RBMs is trained in an unsupervised way, in bottom up fashion, by using the activity in the hidden layer of a RBM in the stack as the input for the next RBM in the stack. Similar architectures and algorithms can be applied to linear networks. Consider for instance training a 10/5/10 autoencoder and then using the activities in the hidden layer to train a 5/3/5 autoencoder. This architecture can be contrasted with a 10/5/3/5/10 architecture, or a 10/3/10 architecture. In all cases, the overall transformation W is linear and constrained in rank by the size of the smallest layer in the architecture. Thus all three architectures have the same optimal solution associated with Principal Component Analysis using the top 3 eigenvalues. However the landscapes of the error functions and the learning trajectories may be different and other considerations may play a role in the choice of an architecture.

In any case, the theory developed here can be adapted to multi-layer real-valued or complex-valued linear networks. Overall, such networks implement a linear transformation with a rank restriction associated with the smallest hidden layer. As in the single hidden layer case, the overall distortion is convex in any single matrix while all the other matrices are held fixed. Any algorithm that successively, or randomly, optimizes each matrix with respect to all the others will converge to a critical point, which will be full rank with probability one if the matrices are initialized randomly. For instance, to be more precise, consider a network with five stages associated with the five matrices A, B, C, D and F of the proper sizes and the error function $E(A, B, C, D, F) = \sum_t ||(y_t - ABCDFx_t)||^2$.

Theorem 10. *For any fix set of matrices A, B, D and F, the function $E(A, B, C, D, F)$ is convex in the coefficients of C and attains its minimum for any C satisfying the equation*

$$B^*A^*ABCDF\,\Sigma_{XX}F^*D^* = B^*A^*\,\Sigma_{YX}F^*D^*. \tag{52}$$

If Σ_{XX} is invertible and AB and DF are of full rank, then E is strictly convex and has a unique minimum reached when

$$C = (B^*A^*AB)^{-1}B^*A^*\,\Sigma_{YX}F^*D^*(DF\,\Sigma_{XX}F^*D^*)^{-1}. \tag{53}$$

Proof. We write

$$E(A, B) = \sum_t x_t^*F^*D^*C^*B^*A^*ABCDFx_t - 2\mathrm{Re}\,(y_t^*ABCDFx_t) + \|y_t\|^2. \tag{54}$$

Then for fixed A, B, D, F, E is a convex function because the associated quadratic form is equal to

$$\sum_t x_t^*F^*D^*L^*B^*A^*ABLDFx_t = \sum_t \|ABLDFx_t\|^2 \geq 0 \tag{55}$$

for any matrix L of the proper size. Let C be a critical point. Then by definition for any matrix L of the proper size, we must have $\lim_{\epsilon \to 0}[E(A, B, C + \epsilon L, D, F) - E(A, B, C, D, F)]/\epsilon = 0$. Expanding and simplifying this expression gives

$$\sum_t x_t^*F^*D^*C^*B^*A^*ABLDFx_t - y_t^*ABLDFx_t = 0 \tag{56}$$

for all matrices C of the proper size. Using the linearity of the trace operator and its invariance under circular permutation of its arguments, this is equivalent to

$$\mathrm{Tr}\left((DF\Sigma_{XX}F^*D^*C^*B^*A^*AB - DF\Sigma_{XY}AB)L\right) = 0 \tag{57}$$

for any L. Thus we have $DF\Sigma_{XX}F^*D^*C^*B^*A^*AB - DF\Sigma_{XY}AB = 0$ and therefore

$$B^*A^*ABCDF\Sigma_{XX}F^*D^* = B^*A^*\Sigma_{YX}F^*D^*. \tag{58}$$

Finally, if Σ_{XX} is invertible and AB and DF are of full rank, then $ABLDFCx_t = 0$ for any t is equivalent to $L = 0$, and thus the function $E(A, B, C, D, F)$ is strictly convex in C. Thus in this case we can solve Eq. (58) for C to get Eq. (53). □

10. Conclusion

We have provided a fairly complete and general analysis of complex-valued linear autoencoders. The analysis can readily be applied to special cases, for instance when the vectors are real-valued and the matrices are complex-valued, or the vectors are complex-valued and the matrices are real-valued. More importantly, the analysis provides a unified view of real-valued and complex-valued linear autoencoders. In the Appendix A, we further extend the treatment of linear autoencoders over infinite fields by looking at their properties from a differential geometry perspective.

More broadly, the framework used here identifies key questions and strategies that ought to be studied for any class of autoencoders, whether linear or non-linear. For instance:

1. What are the relevant group actions and invariances for the problem?
2. Can one of the transformations (A or B) be solved while the other is held fixed? Are there useful convex relaxations or restrictions?
3. Are there any critical points, and how can they be characterized?
4. Is there a notion of symmetry or transposition between the transformations A and B around critical points?
5. Is there an overall analytical solution? Is the problem NP-hard? What is the landscape of E?
6. What are the learning algorithms and their properties?
7. What are the generalization properties?
8. What happens if the outputs are recycled?
9. What happens if autoencoders are stacked vertically?

All these questions can be raised anew for other linear autoencoders, for instance over $\mathbb{R}$ or $\mathbb{C}$ with the L_p norm ($p \neq 2$), or over other fields, in particular over finite fields with the Hamming distance. While results for finite fields will be published elsewhere, it is clear that these questions have different answers in the finite field case. For instance, the notion of using convexity to analytically solve for A or B, while holding the other one fixed, breaks down in the finite field case.

These questions can also be applied to non-linear autoencoders. While in general non-linear autoencoders are difficult to treat analytically, the case of Boolean autoencoders was recently solved using this framework (Baldi, in press). Boolean autoencoders implement a form of clustering when $p < n$ and, in retrospect, all linear autoencoders implement also a form of clustering when $p < n$. In the linear case, for any vector x and any $W = AB$, we have $W(x + \mathrm{Ker}\, W) = W(x)$. $\mathrm{Ker}\, W$ is the kernel of W which contains the kernel of B, and is equal to it when A is of full-rank. Thus, in general, linear autoencoders implement clustering "by hyperplane" associated with the kernel of B. Taken together, these facts point to the more general unity connecting unsupervised learning, clustering, Hebbian learning, and autoencoders.

Finally, there is the case of autoencoders,linear or non-linear, with $p \geq n$ which has not been addressed here. Clearly, additional restrictions or conditions must be imposed in this case, such as sparse encoding in the hidden layer or sparse matrices using L1 regularization, to avoid trivial solutions associated with the identity function. Although beyond the scope of this paper, these autoencoders are also of interest. For instance, the linear case over finite fields with noise added to the hidden layer, subsumes the theory of linear codes in coding theory (McEliece, 1977). Thus, in short, one can expect autoencoders to continue to play an important role in machine learning and provide fertile connections to other areas, from clustering to information and coding theory.

Acknowledgments

Work in part supported by grants NSF IIS-0513376, NIH LM010235, and NIH-NLM T15 LM07443 to PB, and NSF DMS-09-04653 to ZL. We wish to acknowledge Sholeh Forouzan for running the simulation for Fig. 3.

Appendix A. Differential geometry of autoencoders

Methods from differential geometry has been applied effectively to statistical machine learning in previous studies by (Amari, 1990; Amari & Nagaoka, 2007) and others. Here however we introduce a novel approach for looking at the manifolds of relevant parameters for linear autoencoders over the real or complex fields. While the basic results in this section are not difficult, they do assume some understanding of the most basic concepts of differential geometry (Spivak, 1999).

Let R_p be the set of $n \times n$ complex matrices of rank at most equal to p. Obviously, $AB \in R_p$. In general, R_p is a singular variety (a Brill–Noether variety). We let also $R_p \setminus R_{p-1}$ be the set of $n \times n$ matrices of rank exactly p. As we shall see, $R_p \setminus R_{p-1}$ is a complex manifold.

Definition 1. We let

$$F_p(W) = \sum_{t=1}^{m} \|y_t - Wx_t\|^2 \tag{59}$$

where $W \in R_p$.

Let $M^{p\times q}$ be the set of all $p \times q$ complex matrices. Define the mapping

$$\iota : M^{n\times p} \times M^{p\times n} \to R_p \quad \text{with } \iota(A, B) = A \tag{60}$$

by taking the product of the corresponding matrices. Then we have $F \circ \iota = E$. We are going to show that ι is surjective and the differential of ι is of full rank at any point.

Lemma 1. *$R_p \setminus R_{p-1}$ is a complex manifold of dimension $2np - p^2$.*

Proof. Let $W \in R_p \setminus R_{p-1}$. To construct a set of local coordinates of $R_p \setminus R_{p-1}$ near W, we write W

$$W = (w_1, \ldots, w_n) \tag{61}$$

where $w_1, \ldots, w_n$ are column vectors. Without any loss of generality, we assume that $w_1, \ldots, w_p$ are linearly independent. Thus we must have

$$w_j = \sum_{i=1}^{p} \xi_{ij} w_i \tag{62}$$

for $j > p$, with complex coefficients ξ_{ij}. The local coordinates of $R_p \setminus R_{p-1}$ are $(\xi_{ij})_{1\leq i\leq p, p<j\leq n}$ and $(w_{ik})_{1\leq i\leq p, 1\leq k\leq n}$. Thus

$$\dim(R_p \setminus R_{p-1}) = p(n-p) + pn = 2pn - p^2. \quad \square \tag{63}$$

Next, we consider the tangent space T_W of $R_p \setminus R_{p-1}$ at W. By definition, a basis of $T_C(R_p \setminus R_{p-1})$ is given by

$$\frac{\partial}{\partial w_{ik}} \quad 1 \le i \le p, 1 \le k \le n; \tag{64}$$

$$\frac{\partial}{\partial \xi_{ij}} \quad 1 \le i \le p, p < j \le n. \tag{65}$$

Let $(e_1, \ldots, e_n)$ be the standard basis of $\mathbb{C}^n$. Then the corresponding matrices of the tangent vectors are

$$\frac{\partial}{\partial c_{ik}} \longrightarrow \big(0, \ldots, \underset{i\text{-th place}}{e_k}, \ldots, 0, \xi_{i,p+1}e_k, \ldots, \xi_{in}e_k\big);$$

$$\frac{\partial}{\partial \xi_{ij}} \longrightarrow \big(0, \ldots, 0, 0, \ldots, \underset{j\text{-th place}}{w_i}, \ldots, 0\big).$$

Lemma 2. *Let $W = AB$, where A, B are full-rank $n \times p$ and $p \times n$ matrices, respectively. Let A_1, B_1 be $n \times p$ and $p \times n$ matrices such that*

$$AB_1 + A_1B = 0. \tag{66}$$

Then there is an invertible $p \times p$ matrix V such that

$$A_1 = AV, \qquad B_1 = -VB. \tag{67}$$

Proof. By multiplying on the left by A^*, we have

$$A^*AB_1 + A^*A_1B = 0. \tag{68}$$

Since A is full rank, A^*A is an invertible $p \times p$ matrix. Thus

$$B_1 = -(A^*A)^{-1}A^*A_1B. \tag{69}$$

Substituting the above into Eq. (66) yields

$$-A(A^*A)^{-1}A^*A_1B + A_1B = 0. \tag{70}$$

Since B is of full rank, we get

$$-A(A^*A)^{-1}A^*A_1 + A_1 = (1 - P_A)A_1 = 0 \tag{71}$$

which implies that the columns of A_1 span the same linear space as the image of P_A, i.e. the same space spanned by the columns of A. Hence $A_1 = AV$ for some $p \times p$ matrix V. □

Lemma 3. *The tangent space $T_W(R_p \setminus R_{p-1})$ is spanned by the matrices of the form*

$$AB_1 + A_1B \tag{72}$$

where A and B are fixed, $AB = W$, and A_1, B_1 are $n \times p$ and $p \times n$ matrices, respectively.

Proof. Define a linear map

$$\sigma : M^{n\times p} \times M^{p\times n} \to M^{n\times n} \quad \text{with } \sigma(A_1, B_1) = AB_1 + A_1B. \tag{73}$$

We have

$$\dim \mathrm{Im}(\sigma) = 2np - \dim \mathrm{Ker}(\sigma). \tag{74}$$

By the above lemma, $\dim \mathrm{Ker}(\sigma) = p^2$. Thus the image of σ has the same dimension as the manifold $R_p \setminus R_{p-1}$. Hence all the tangent vectors must be of the form $AB_1 + A_1B$. □

Corollary. *The map ι is of full rank at any point (A, B) where A and B are of rank p.*

Proof. The space spanned by all pairs (A_1, B_1) has dimension $2np$. By Lemma 2, the space of all matrices of the form $AB_1 + A_1B$ has dimension $2np - p^2$, which is the dimension of R_p by Lemma 1. Since the dimension of the image of the Jacobian of ι at (A, B) is equal to the dimension of the manifold R_p, ι is of full rank. □

We need to prove that ι is of full rank because we want to measure how far away the function is from being convex. The Hessian of the function E is the sum of two terms, the first of which is positive definite (see Remark 8). If ι were of lesser rank, this first term would contribute less to the total Hessian. In particular, if ι had rank zero, then the first term of the Hessian would be equal to zero and, as a result, a point where the Hessian is positive would not necessarily exist preventing the existence of a global minimum.

Lemma 4. *For any $W \in R_p$, there exist an $n \times p$ matrix A and a $p \times n$ matrix B such that $W = AB$. In other words, ι is a surjective map.*

Proof. We use the following singular decomposition of matrices

$$W = U_1 \Lambda U_2, \tag{75}$$

where U_1, U_2 are unitary matrices and Λ is a diagonal matrix. Since W is of rank no more than p, we can write Λ as

$$\Lambda = \begin{pmatrix} \lambda_1 & & & & & \\ & \ddots & & & & \\ & & \lambda_p & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{pmatrix}. \tag{76}$$

Let $(U_1)_p$ represent the first p columns of U_1 and let $(U_2)^p$ be the first p rows of U_2. Let Λ_1 be the first $p \times p$ minor of Λ. Then

$$W = (U_1)_p \Lambda_1 (U_2)^p. \tag{77}$$

Thus the theorem is proved by letting $A = (U_1)_p\Lambda_1$ and $B = (U_2)^p$. □

In general, R_p is *not* a manifold. One of the resolution $\tilde{R}_p$ of R_p is defined as follows

$$\tilde{R}_p = \{(A, V) \mid A \in R_p, V \subset \ker A^*, \dim V = n - p\}.$$

In this case, $\tilde{R}_p$ is a manifold and we can extend the function F to $\tilde{R}_p$ in a natural way: for $(A, V) \in \tilde{R}_p$, we let $\tilde{F}_p(A, V) = F_p(A)$.

By the convexity of the quadratic function $\sum \|y_t - Ax_t\|^2$, we get the following conclusion

Theorem 11. *Both $F_p, \tilde{F}_p$ are convex functions on $R_p \setminus R_{p-1}$. In particular, all critical points of the functions are global minima.*

Remark 8. By the relation $E = F \circ \iota$, we have

$$D^2E = D^2F(\nabla\iota, \nabla\iota) + \nabla F \circ D^2\iota.$$

The first term on the right-hand side is always nonnegative by the convexity of F_p. However the second term can be positive or negative, which partly explains why E is not convex and has many critical points that are saddle points.

Theorems 11 and 5 are related but one does not imply the other. Theorem 11 shows that for complex-valued autoencoders, the error E has a global minimum. However Theorem 11 does not provide further information about the global minimum, not it implies that all other critical points are saddle points. We end this section with the following result.

Theorem 12. *Let*

$$E(A_1, \ldots, A_k) = \sum \|y_t - A_1 \cdots A_k x_t\|^2,$$

where A_i are (μ_i, δ_i) matrices. Let

$$\sigma = \min(\mu_i, \delta_i).$$

Then

$$E(A_1, \ldots, A_k) = F_\sigma(A_1 \cdots A_k).$$

Proof. The only non-trivial point is that any rank σ matrix can be decomposed into a product of the form $A_1 \cdots A_k$, where A_j is a $\mu_j \times \delta_j$ matrix. For $k = 2$, this is just Lemma 4. For $k > 2$, the statement can be proved using mathematical induction. □

Appendix B. Direct proof of convergence for the alternate minimization algorithm

It is expected that starting from any full rank initial matrices (A_1, B_1), if we inductively define

$$A_{k+1} = \Sigma_{YX} B_k^* (B_k \Sigma_{XX} B_k^*)^{-1}$$
$$B_{k+1} = (A_{k+1}^* A_{k+1})^{-1} A_{k+1}^* \Sigma_{YX} \Sigma_{XX}^{-1},$$

then (A_k, B_k) should converge to a critical point of E. In this section, we prove the following

Theorem 13. *In the auto-associative case, assume that*

$$\sum_{j \in \mathcal{I}} \lambda_j \tag{78}$$

are different for different set $\mathcal{I}$, where $\mathcal{I}$ is defined in Theorem 4. Then (A_k, B_k) converges to a critical point of $E(A, B)$.

Remark 9. The assumption in Theorem 13 is a technical assumption to separate the distortion values of non-equivalent critical point. In fact, using Theorem 4, it is equivalent to assuming that each equivalence class of critical points is associated with a different distortion level which characterizes the corresponding critical points.

Proof. In what follows, we use the Hilbert–Schmidt norm of a matrix:

$$\|A\| = \sqrt{Tr(A^* A)}.$$

In the auto-associative case, the algorithm becomes

$$A_{k+1} = \Sigma B_k^* (B_k \Sigma B_k^*)^{-1}$$
$$B_{k+1} = (A_{k+1}^* A_{k+1})^{-1} A_{k+1}^*.$$

The proof of the theorem is in three steps. First, we prove that the sequences $\|A_k\|$ and $\|B_k\|$ are bounded so they both have limiting points; second, we prove that the limiting points must be nonsingular matrices if the initial A_1 and B_1 are nonsingular; finally, we prove that the sequences A_k and B_k are actually convergent under the assumption of the theorem.

Step 1. Substituting A_{k+1} in the definition of B_{k+1}, we obtain

$$B_{k+1} = (B_k \Sigma B_k^*)(B_k \Sigma^2 B_k^*)^{-1} B_k \Sigma.$$

Substituting the expression of B_k into the definition of A_{k+1}, we obtain

$$A_{k+1} = \Sigma A_k (A_k^* \Sigma A_k)^{-1} (A_k^* A_k).$$

Since $\Sigma B_k^* (B_k \Sigma^2 B_k^*)^{-1} B_k \Sigma$ is a projection operator, we have

$$\|B_{k+1}\| \leq \|B_k\|. \tag{79}$$

Similarly, if we write $\Sigma = \Sigma_1^2$ for a positive symmetric matrix Σ_1, we obtain

$$\Sigma_1^{-1} A_{k+1} = \Sigma_1 A_k (A_k^* \Sigma A_k)^{-1} A_k^* \Sigma_1^* \Sigma_1^{-1} A_k$$

and hence $\|\Sigma_1^{-1} A_{k+1}\| \leq \|\Sigma_1^{-1} A_k\|$. Thus we conclude that both A_k and B_k are bounded sequences.

Step 2. We have $B_k A_{k+1} = I_p$, where I_p is the $p \times p$ identity matrix. Thus by continuity any limiting point of B_k or A_k must be nonsingular.

Step 3. To prove that the set of limiting points of B_k contains only one point, we observe that the sequence $E(A_k, B_k)$ is decreasing. If B and B' are two limiting points of the sequence B_k, we must have

$$E(A, B) = E(A', B')$$

where $A = \Sigma B^* (B \Sigma B^*)^{-1}$ and $A' = B' \Sigma (B')^* ((B') \Sigma (B')^*)^{-1}$. By the assumption of the theorem (or Eq. (43)), if $B \neq B'$ we must have $E(A, B) \neq E(A', B')$, which yields a contradiction. Thus B_k and hence A_k must be convergent.

Since the limit (A, B) satisfies the equations

$$A = \Sigma B^* (B \Sigma B^*)^{-1} \quad \text{and} \quad B = (A^* A)^{-1} A^*$$

by Theorems 1 and 2, (A, B) must be a critical point of E.

Finally, note that if Σ is singular or close to singular, or if there are critical points with the same distortion levels, then the algorithm above could run into numerical issues. □

Examples. The convergence can be better seen when $p = 1$. Let

$$\Sigma = \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix} \tag{80}$$

with $\lambda_1 > \cdots > \lambda_n$. If $p = 1$, then there is a sequence c_k of real numbers such that

$$B_{k+1} = c_k B_1 \Sigma^k. \tag{81}$$

Let $B_1 = (b_1, \ldots, b_n)$ and let i be the smallest index such that $b_i \neq 0$. Then

$$B_k = c_k (\lambda_1^k b_1, \ldots, \lambda_n^k b_n).$$

Since $\|B_{k+1}\| \leq \|B_k\|$ (Eq. (79)), the sequence $c_k \lambda_i^k$ is bounded for $k \to \infty$. It follows that for any $j > i$, $c_k \lambda_j^k b_j \to 0$ as $k \to \infty$. Therefore, using Eq. (79) again, we have $b_k \to c e_i$ for some constant c, with $e_1, \ldots, e_n$ denoting the standard basis of $\mathbb{C}^n$. Moreover, $c = b_i$ by a straightforward computation.

The case of arbitrary p values can be addressed using the above example: let $j < i$ and $i - j + 1 = p$. Let

$$B_1 = \begin{pmatrix} 0 & \cdots & 0 & b_i & \cdots & b_n \\ & & & e_j & & \\ & & & \vdots & & \\ & & & e_{i-1} & & \end{pmatrix} \tag{82}$$

be a matrix of rank p with the same matrix Σ as above. Then

$$B_k \to \begin{pmatrix} e_i \\ e_j \\ \vdots \\ e_{i-1} \end{pmatrix}. \tag{83}$$

In conclusion, for any saddle point, one can construct a sequence that converges to it.

References

Amari, S. (1990). *Differential-geometrical methods in statistics*. Berlin, New York: Springer-Verlag.

Amari, S., & Nagaoka, H. (2007). *Methods of information geometry*. Amer. Mathematical Society.

Baldi, P. (2012). Autoencoders, unsupervised learning, and deep architectures. *Journal of Machine Learning Research* (in press).

Baldi, P., & Hornik, K. (1988). Neural networks and principal component analysis: learning from examples without local minima. *Neural Networks*, *2*(1), 53–58.

Bell, R., & Koren, Y. (2007). Lessons from the Netflix prize challenge. *ACM SIGKDD Explorations Newsletter*, *9*(2), 75–79.

Bengio, Y., & LeCun, Y. (2007). Scaling learning algorithms towards AI. In L. Bottou, O. Chapelle, D. DeCoste, & J. Weston (Eds.), *Large-scale kernel machines*. MIT Press.

Bourlard, H., & Kamp, Y. (1988). Auto-association by multilayer perceptrons and singular value decomposition. *Biological Cybernetics*, *59*(4), 291–294.

Candès, E., & Recht, B. (2009). Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, *9*(6), 717–772.
Candes, E., & Wakin, M. (2008). People hearing without listening: an introduction to compressive sampling. *IEEE Signal Processing Magazine*, *25*(2), 21–30.
Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B*, *39*, 1–22.
Erhan, D., Bengio, Y., Courville, A., Manzagol, P.-A., Vincent, P., & Bengio, S. (2010). Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, *11*, 625–660.
Hinton, G., Osindero, S., & Teh, Y. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, *18*(7), 1527–1554.
Hinton, G., & Salakhutdinov, R. (2006). Reducing the dimensionality of data with neural networks. *Science*, *313*(5786), 504.
Hirose, A. (2003). *Complex-valued neural networks: theories and applications*: *Vol. 5*. World Scientific Pub. Co. Inc.
Lang, S. (1984). *Algebra*. Reading, MA: Addison-Wesley.
McEliece, R. J. (1977). *The theory of information and coding*. Reading, MA: Addison-Wesley Publishing Company.
Meyer, C. (2000). *Matrix analysis and applied linear algebra*. Society for Industrial and Applied Mathematics.
Oja, E. (1982). Simplified neuron model as a principal component analyzer. *Journal of Mathematical Biology*, *15*(3), 267–273.
Oja, E. (1989). Neural networks, principal components, and subspaces. *International Journal of Neural Systems*, *1*(1), 61–68.
Oja, E. (1992). Principal components, minor components, and linear neural networks. *Neural Networks*, *5*(6), 927–935.
Roux, N. L., & Bengio, Y. (2010). Deep belief networks are compact universal approximators. *Neural Computation*, *22*(8), 2192–2207.
Rumelhart, D., Hinton, G., & Williams, R. (1986). Learning internal representations by error propagation. In *Parallel distributed processing. Vol 1: foundations*. Cambridge, MA: MIT Press.
Spivak, M. (1999). *A comprehensive introduction to differential geometry, volume 1* (3rd ed.) Houston, TX: Publish or Perish.
Takács, G., Pilászy, I., Németh, B., & Tikk, D. (2008). Matrix factorization and neighbor based algorithms for the Netflix prize problem. In *Proceedings of the 2008 ACM conference on recommender systems* (pp. 267–274). ACM.
Vincent, P. (2011). A connection between score matching and denoising autoencoders. *Neural Computation*, 1–14.
Vincent, P., Larochelle, H., Bengio, Y., & Manzagol, P. (2008). Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on machine learning* (pp. 1096–1103). ACM.