Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)

Conservativeness of Untied Auto-Encoders

Daniel Jiwoong Im*
Montreal Institute
for Learning Algorithms
University of Montreal
Montreal, QC, H3C 3J7
imdaniel@iro.umontreal.ca

Mohamed Ishmael Belghazi*
HEC Montreal
3000 Ch de la Cte-Ste-Catherine
Montreal, QC, H3T 2A7
mohamed.2.belghazi@hec.ca

Roland Memisevic
Montreal Institute
for Learning Algorithms
University of Montreal
Montreal, QC, H3C 3J7
roland.memisevic@umontreal.ca

Abstract

We discuss necessary and sufficient conditions for an auto-encoder to define a conservative vector field, in which case it is associated with an energy function akin to the unnormalized log-probability of the data. We show that the conditions for conservativeness are more general than for encoder and decoder weights to be the same ("tied weights"), and that they also depend on the form of the hidden unit activation functions. Moreover, we show that contractive training criteria, such as denoising, enforces these conditions locally. Based on these observations, we show how we can use auto-encoders to extract the conservative component of a vector field.

Introduction

An auto-encoder is a feature learning model that learns to reconstruct its inputs by going though one or more capacity-constrained "bottleneck"-layers. The motivation for auto-encoders is that reconstructing input data will be possible only for a model that creates a faithful internal representation of the data. If that representation is forced through a bottleneck, like a low-dimensional or sparse hidden layer, then the model must have learned to compress (or equivalently, "understand") the data.

Since technically it defines a function $r : \mathbb{R}^n \to \mathbb{R}^n$, which maps data to its reconstruction, an auto-encoder can also be viewed as dynamical system, and minimizing reconstruction error can be viewed as a way to encourage the system to have fixed points at the data (Seung 1998). Recent renewed interest in the dynamical systems perspective led to a variety of results that help clarify the role of auto-encoders and their relationship to probabilistic models.

For example, (Vincent et al. 2008; Swersky et al. 2011) showed that training an auto-encoder to denoise corrupted inputs is closely related to performing score matching (Hyvärinen 2005) in an undirected model. Similarly, (Alain and Bengio 2014) showed that training the model to denoise inputs, or to reconstruct them under a suitable choice of regularization penalty, lets the auto-encoder approximate the derivative of the empirical data density. And (Kamyshanska 2013) showed that, regardless of training criterion, an auto-encoder whose weights are tied (decoder-weights are identical to the encoder weights) can be written as the derivative of a scalar "potential" or energy-function.

In general, a dynamical system that can be written as the derivative of such a scalar function is called *conservative*. When it is conservative, the autoencoder thus defines, via this scalar function, an energy landscape, which may viewed as an unnormalized log-density in the data space. As such, the autoencoder could in principle be turned into a probabilistic model of the data by simply exponentiating and normalizing this function. In fact, for sigmoid hidden units the potential function is identical to the free energy of an RBM (Kamyshanska 2013). This shows that rather than thinking of autoencoders and RBMs as two distinct types of model, one may think of them as merely two different *training schemes* to fit the same kind of density function to the data.

Besides rooting autoencoders in the realm of probabilistic and energy-based models, the energy function has been of interest because it may help explain how training of multi-layer neural networks may be possible via local, biologically plausible updates (see, for example, (Bengio 2014)).

For untied auto-encoders it has not been clear whether an energy function exists. It has also not been clear under which conditions an energy function exists or does not exist, or even how to define it in the case where decoder-weights differ from encoder weights.

In this paper, we describe necessary and sufficient conditions for the existence of an energy function ("conservativeness"), and we show that suitable learning criteria will lead to an auto-encoder that satisfies these conditions at least locally, near the training data. We verify our results experimentally. We also show how we can use an auto-encoder to extract the conservative part of a vector field.

Background

We will focus on auto-encoders of the form

$$r(\mathbf{x}) = Rh\big(W^{\mathrm{T}}\mathbf{x} + \mathbf{b}\big) + \mathbf{c} \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$ is an observation, W and R are $d \times n$ encoder and $n \times d$ decoder weights, respectively, $\mathbf{b}$ and $\mathbf{c}$ are biases, and $h(\cdot)$ is an elementwise hidden activation function. An auto-encoder can be identified with its vector field, $r(\mathbf{x}) - \mathbf{x}$, which is the set of vectors pointing from observations to

*Authors constributed equally.

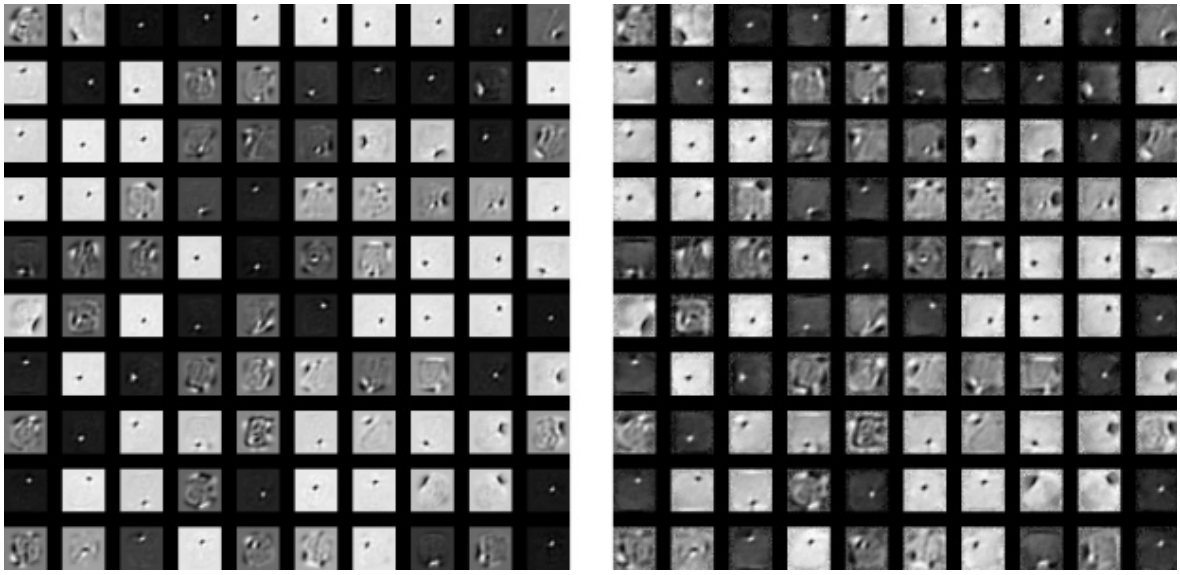


Figure 1: Encoder weights W (left) and decoder weights R^T (right).

their reconstructions. The vector field is called conservative if it can be written as the gradient of a scalar function $F(\mathbf{x})$, called potential or energy function:

$$r(\mathbf{x}) - \mathbf{x} = \nabla F(\mathbf{x}) \tag{2}$$

The energy function can be thought of as an unnormalized data density, and most common training criteria encourage large values near the data.

In the case where an energy function exists, we can integrate Eq. 2 to find it. (Kamyshanska 2013) show that for an auto-encoder with tied weights and real-valued observations the energy function takes the general form

$$F(\mathbf{x}) = \int h(\mathbf{u}) d\mathbf{u} - \frac{1}{2}\|\mathbf{x} - \mathbf{c}\|_2^2 + \text{const} \tag{3}$$

where $\mathbf{u} = W^{\mathrm{T}}\mathbf{x} + \mathbf{b}$ is an auxiliary variable and $h(\cdot)$ can be any elementwise activation function with known anti-derivative. For example, the energy function of an auto-encoder with sigmoid activation function is identical to the free energy of a Gaussian RBM (Hinton 2010):

$$F_{\text{sig}}(\mathbf{x}) = \sum_k \log\left(1 + \exp\left(W_{\cdot k}^T\mathbf{x} + b_k\right)\right) - \frac{1}{2}\|\mathbf{x} - \mathbf{c}\|_2^2 + \text{const} \tag{4}$$

A sufficient condition for the existence of an energy function is that the weights are tied (Kamyshanska 2013), but it has not been clear if this is also necessary. A peculiar phenomenon in practice is that it is very common for decoder and encoder weights to be "similar" (albeit not necessarily tied) in response to training. An example of this effect is shown in Figure 1[1]. This raises the question of why this happens, and whether the quasi-tying of weights has anything to do with the emergence of an energy function, and if yes, whether there is a way to compute the energy function despite the lack of exact symmetry. We shall address these questions in what follows.

Conservative auto-encoders

One of the central objectives of this paper is understanding the conditions for an auto-encoder to be conservative[2] and thus to have a well-defined energy function. In the following subsection we derive and explain said conditions.

Conditions for conservative auto-encoders

Proposition 1. *Consider an m-hidden-layer auto-encoder defined as*

$$r(\mathbf{x};\theta) = W^{(m)}h^{(m)}\Big(W^{(m-1)}h^{(m-1)} \Big(\cdots W^{(1)}h^{(1)}(\mathbf{x})\cdots\Big) + \mathbf{c}^{(m-1)}\Big) + \mathbf{c}^{(m)},$$

where $\theta = \cup_{k=0}^{m}\theta^{(k)}$ such that $\theta^{(k)} = \{W^{(k)}, \mathbf{c}^{(k)}\}$ are the parameters of the model, and $h^{(k)}(\cdot)$ is a smooth element-wise activation function at layer k. Then the auto-encoder is said to be conservative over a smooth simply connected domain $K \subseteq \mathbb{R}^D$ if and only if its reconstruction's Jacobian $\frac{\partial r(\mathbf{x})}{\partial \mathbf{x}}$ is symmetric for all $\mathbf{x} \in K$.

[1] We found these kinds of behaviours not only for unwhitened, but also for binary data.

[2] The expressions, "conservative vector field" and "conservative auto-encoders" will be used interchangeably.

A formal proof is provided in the supplementary material.

A region K is said to be *simply connected* if and only if any simple curve in K can be shrunk to a point. It is not always the case that a region of $\mathbb{R}^D$ is simply connected. For instance, a curve surrounding a punctured circle in $\mathbb{R}^2$ cannot be continuously deformed to a point without crossing the punctured region. However, as long as we make the reasonable assumption that the activation function does not have a continuum of discontinuities, we should not run into trouble. This makes our analysis valid for activation functions with cusps such as ReLUs.

Throughout the paper, our focus will be on one–hidden-layer auto-encoders. Although the necessary and sufficient conditions for their conservativeness are a special case of the above proposition, it is worthwhile to derive them explicitly.

Proposition 2. *Let $r(x)$ be a one-hidden-layer auto-encoder with D dimensional inputs and H hidden units,*

$$r(\mathbf{x}) = Rh\left(W^T\mathbf{x} + \mathbf{b}\right) + \mathbf{c},$$

where $R, W, \mathbf{b}, \mathbf{c}$ are the parameters of the model. Then $r(\mathbf{x})$ defines a conservative vector field over a smooth simply connected domain $K \subseteq \mathbb{R}^D$ if and only if $RD_{h'}W^T$ is symmetric for all $\mathbf{x} \in K$ where $D_{h'} = \ diag\left(h'(\mathbf{x})\right)$.

Proof. Following proposition 1, an auto-encoder defines a conservative vector field if and only if its Jacobian is symmetric for all $\mathbf{x} \in K$.

$$\frac{\partial \mathbf{r}(\mathbf{x})}{\partial \mathbf{x}} = \left(\frac{\partial \mathbf{r}(\mathbf{x})}{\partial \mathbf{x}}\right)^T \tag{5}$$

By explicitly calculating the Jacobian, this is equivalent to

$$(\forall 1 \leq i < j \leq D) \sum_{l=0}^{H} \left(R_{jl}W_{li} - R_{il}W_{lj}\right) h'_l\left(\mathbf{x}\right) = 0 \tag{6}$$

Defining $D_{h'} = diag(h^{'}(\mathbf{x}))$, this holds if and only if

$$RD_{h'}W^T = WD_{h'}R^T \tag{7}$$

□

For one-hidden-layer auto-encoders with tied weights, Equation 7 holds regardless of the choice of activation function h and $\mathbf{x}$.

Corollary 1. *An auto-encoder with tied weights always defines a conservative vector field.*

Proposition 2 illustrates that the set of all one-layered tied auto-encoders is actually a subset of the set of all conservative one-layered auto-encoders. Moreover, the inclusion is strict. That is to say there are untied conservative auto-encoders that are not trivially equivalent to tied ones. As example, let us compare the parametrization of tied and conservative untied linear one-layered auto-encoders. $r_{untied}(\mathbf{x})$ in Eq. 7 defines a conservative vector field if and only $RW^T = WR^T$ which offers a richer parametrization than the tied linear auto-encoder $r_{tied}(\mathbf{x}) = WW^T\mathbf{x}$.

In the following section we explore in more detail and generality of the parametrization imposed by the conditions above.

Understanding the symmetricity condition

Note that if symmetry holds in the Jacobian of an auto-encoder's reconstruction function, then the vector field is conservative. A sufficient condition for symmetry of the Jacobian is that R can be written

$$R = CWD_{h'}E. \tag{8}$$

where C and E are symmetric matrices, and C commutes with $WD_{h'}ED_{h'}W^T$, as this will ensure symmetry of the partial derivatives:

$$\begin{aligned}\frac{\partial \mathbf{r}(\mathbf{x})}{\partial \mathbf{x}} &= RD_{h'}W^T = CWD_{h'}ED_{h'}W^T \\ &= WD_{h'}ED_{h'}W^TC = WD_{h'}R^T = \left(\frac{\partial \mathbf{r}(\mathbf{x})}{\partial \mathbf{x}}\right)^T.\end{aligned} \tag{9}$$

The case of tied weights ($R = W$) follows if C and E are the identity, since then $\frac{\partial \mathbf{r}(\mathbf{x})}{\partial \mathbf{x}} = RD_{h'}W^T = WD_{h'}W^T$.

Notice that $R = CWD_{h'}$ and $R = WD_{h'}E$ are further special cases of the condition $R = CWD_{h'}E$ when E is the identity (first case) or C is the identity (second case). Moreover, we can also find matrices E and C given the parameters W and R, which is shown in Section 1.2 of the supplementary material[3].

Conservativeness of trained auto-encoders

Following (Alain and Bengio 2014) we will first assume that the true data distribution is known and the auto-encoder is trained. We then analyze the conservativeness of auto-encoders around fixed points of the data manifold. After that, we will proceed to empirically investigate and explain the tendency of trained auto-encoders to become conservative away from the data manifold. Finally, we will use the obtained results to explain why the product of the encoder and decoder weights become increasingly symmetric in response to training.

Local Conservativeness

Let $r(\mathbf{x})$ be an auto-encoder that minimizes a contraction-regularized squared loss function averaged over the true data distribution p,

$$L_\epsilon(\mathbf{x}) = \int_{\mathbb{R}^d} p(\mathbf{x}) \left[\|r(\mathbf{x}) - \mathbf{x}\|_2^2 + \epsilon\|\frac{\partial r(\mathbf{x})}{\partial \mathbf{x}}\|_2^2\right] d\mathbf{x} \tag{10}$$

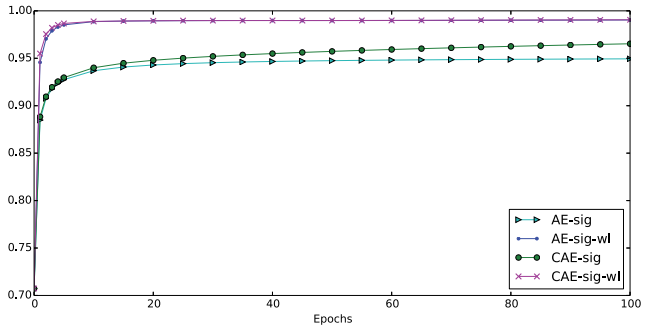
A point $\mathbf{x} \in \mathbb{R}^d$ is a fixed point of the auto-encoder if and only if $r(\mathbf{x}) = \mathbf{x}$.

Proposition 3. *Let $r(\mathbf{x})$ be an untied one-layer auto-encoder minimizing Equation 10. Then $r(\mathbf{x})$ is locally conservative as the contraction parameter $\boldsymbol{\epsilon}$ tends to zero.*

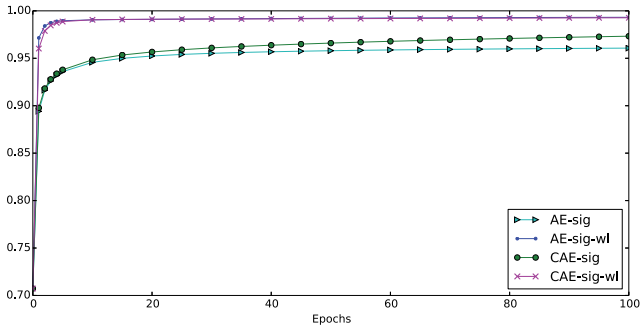
Taking a first order Taylor expansion of $r(\mathbf{x})$ around a fixed point $\mathbf{x}$ yields

$$r(\mathbf{x} + \boldsymbol{\epsilon}) = \mathbf{x} + \frac{\partial r(\mathbf{x})}{\partial \mathbf{x}}^T \boldsymbol{\epsilon} + o(\boldsymbol{\epsilon}) \text{ as } \boldsymbol{\epsilon} \to 0. \tag{11}$$

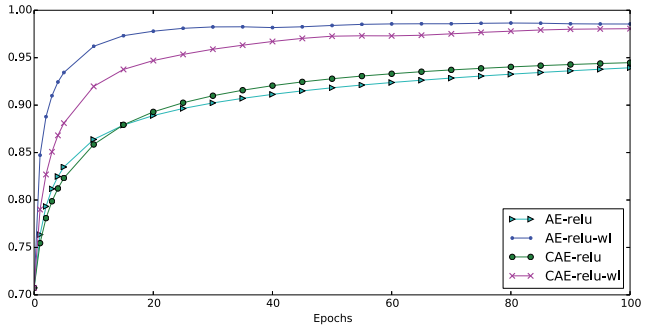
[3] www.uoguelph.ca/~imj/files/conservative_ae_supplementary.pdf



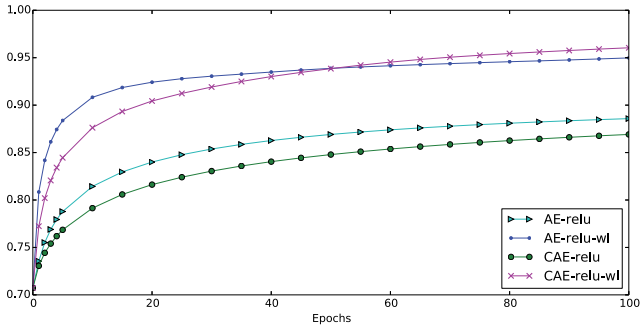
(a) Symmetricity for $RD_{h'}W$ with sigmoid units



(b) Symmetricity for RW with sigmoid units



(c) Symmetricity for $RD_{h'}W$ with ReLU units



(d) Symmetricity for RW with ReLU units

Figure 2: Symmetricity of $\frac{\partial r(\mathbf{x})}{\partial \mathbf{x}}$ and RW^T for sigmoid and ReLU activations over training time.

(Alain and Bengio 2014) show that the reconstruction $r(\mathbf{x}) - \mathbf{x}$ becomes an estimator of the score when $\|r(\mathbf{x}) - \mathbf{x}\|_2^2$ is small and the contraction parameters $\epsilon \to 0$. Hence around a fixed point we have

$$r(\mathbf{x}+\boldsymbol{\epsilon}) - \mathbf{x} = \boldsymbol{\epsilon}\frac{\partial \log(p(\mathbf{x}))}{\partial \mathbf{x}}, \quad \text{and} \tag{12}$$

$$\frac{\partial(r(\mathbf{x}+\boldsymbol{\epsilon}) - \mathbf{x})}{\partial \mathbf{x}} = \boldsymbol{\epsilon}\frac{\partial^2 \log(p(\mathbf{x}))}{\partial \mathbf{x}^2} \tag{13}$$

By explicitly expressing the Jacobian of the auto-encoder's dynamics $\frac{\partial r(\mathbf{x})-\mathbf{x}}{\partial x}$ and using the Taylor expansion of $r(\mathbf{x})$, we have

$$W^T D_{h'} R^T - I = \boldsymbol{\epsilon}\frac{\partial^2 \log(p(\mathbf{x}))}{\partial \mathbf{x}^2} \tag{14}$$

The Hessian of $\log p(\mathbf{x})$ being symmetric, Equation 14 illustrates that around fixed points, $RD_{h'}W$ is symmetric. In conjunction with *Proposition 2*, this shows that untied auto-encoders, when trained using a contractive regularizer, are locally conservative. Notice that when the auto-encoder is trained with patterns drawn from a continuous family, then the auto-encoder forms a continuous attractor that lies near the examples it is trained on (Seung 1998).

It is worth noting that dynamics around fixed points can be understood by analyzing the eigenvalues of the Jacobian. The latter being symmetric implies that its eigenvalues cannot have complex parts, which corresponds to the lack of oscillations in a conservative vector field. Moreover, in directions orthogonal to the fixed point, the eigenvalues of the reconstruction will be negative. Thus the fixed point is actually a sink.

Table 1: Symmetricity of ADW after training AEs with 500 units on MNIST for 100 epochs. We denote the auto-encoders with weight length constraints as '+wl'.

	ReLU	ReLU+wl	sig.	sig.+wl
AE	95.9%	98.7%	95.1%	99.1%
CAE	95.2%	98.6%	97.4%	99.1%

Empirical Conservativeness

We now empirically analyze the conservativeness of trained untied auto-encoders. To this end, we train an untied auto-encoder with 500 hidden units with and without weight length constraints[4] on the MNIST dataset. We measure symmetricity using $\text{sym}(A) = \frac{\|(A+A^T)/2\|^2}{\|A\|^2}$ which yields values between $[0, 1]$ with 1 representing complete symmetricity. Figure 2a and 2c shows the evolution of the symmetricity of $\frac{\partial r(\mathbf{x})}{\partial \mathbf{x}} = RD_{h'}W$ during training. For untied auto-encoders, we observe that the Jacobian becomes increasingly symmetric as training proceeds and hence, by *Proposition 2*, the

[4]Weight length constraints : $||\mathbf{w}_i||^2 = \alpha$ for all $i = 1 \cdots H$ and α is a constant term.

auto-encoder becomes increasingly conservative.

The contractive auto-encoder tends more towards complete symmetricity than the unregularized one. Their symmetricity scores plateau at 0.974 and 0.951 respectively. It is interesting to note that auto-encoders with weight length constraints yield sensibly higher symmetricity scores as shown in Table 1. The details of the experiments and further interpretations are provided in the supplementary material.

To explicitly confirm the conservativeness of the auto-encoder in 2D, we monitor the curl of the vector field during training. In our experiments, we created three 2D synthetic datasets by adding gaussian white noise to the parametrization of a line, a circle, and a spiral. As shown in Figure 3, we notice that the curl decreases sharply during training, further demonstrating how untied auto-encoders become more conservative during training. More results on the line, circle, and spiral synthetic datasets can be found in the supplementary material.

Symmetricity of weights product

The product of weight RW^T tends to become increasingly symmetric during training. This behavior is more marked for sigmoid activations than for ReLUs as shown in Figures 2b and 2d. This can be explained by considering the symmetricity of the Jacobian. After training, it holds that

$$\sum_{l=1}^{H}(R_{il}W_{lj} - R_{jl}W_{li})h'_l(\mathbf{x}) \approx 0, \quad \forall 1 \leq i,j \leq d \quad (15)$$

This implies that the activations of sigmoid hidden units, at least for training data points, are independent of $h'(\mathbf{x})$ or constant.

As shown in the supplementary material, most hidden unit activities are concentrated in the highest curvature region when training with weight length constraints. This forces $h_l(\mathbf{x})$ to be concentrated on high curvature regions of the sigmoid activation. This may be due to either $h'_l(\mathbf{x})$ being nearly constant for all l given $\mathbf{x}$, or $h'_l(\mathbf{x})$ being close to linearly independent. In both cases, the Jacobian becomes close to the identity, and hence $RW^T \approx WR^T$.

Decomposing the Vector Field

In this section, we consider finding the closest conservative vector field, in a least square sense, to a non-conservative vector field. Finding this vector field is of great practical importance in many areas of science and engineering (Bhatia et al. 2013). Here we show that conservative auto-encoders can provide a powerful, deep learning based perspective onto this problem.

The fundamental theorem of vector calculus, also known as Helmhotz decomposition states that any vector field in $\mathbb{R}^3$ can be expressed as the orthogonal sum of an irrotational and a solenoidal field. The Hodge decomposition is a generalization of this result to high dimensional space (James 1966). A complete statement of the result requires careful analysis of boundary conditions as well as differential form formalism. But since 1-forms correspond to vector fields, and our interest lies in the latter, we abuse notation to state the result in the special case of 1-forms as

$$\omega = d\alpha + \delta\beta + \gamma \quad (16)$$

where d is the exterior derivative, δ the co-differential, and $\Delta\gamma = 0$.[5] This means that any 1-form (vector field) can be orthogonally decomposed into a direct sum of a scalar, solenoidal, and harmonic component.

This shows that it is always theoretically possible to get the closest conservative vector field, in the least square sense, to a non-conservative one. When applied to auto-encoders, this guarantees the existence of a best approximate energy function for any untied conservative auto-encoder. For a more detailed background on the vector field decomposition we refer to the supplementary material.

Extracting the Conservative Vector Field through Learning

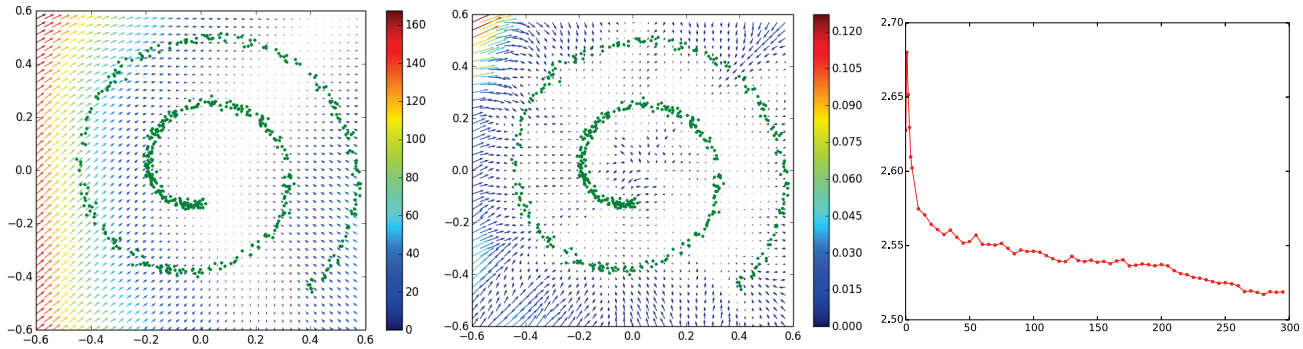
Although the explicit computation of the projection might be theoretically possible in special cases, we propose to find the best approximate conservative vector through *learning*. There are several advantages to learning the conservative part of a vector field: i) Learning the scalar vector field component α from some vector field ω with an auto-encoder is straightforward due to the intrinsic tendency of the trained auto-encoder to become conservative, ii) although there is a large body of literature to explicitly compute the projections, these methods are highly sensitive to boundary conditions (Bhatia et al. 2013) while learning based methods eschew this difficulty.

The advantage of deep learning based methods over existing approaches, such as matrix-valued radial basis function kernels (Macedo and Castro 2008), is that they can be trained on very large amounts of data. To the best of our knowledge, this is the first application of neural networks to the problem of extracting the conservative part of any vector field (effectively recovering the scalar part of Eq. 16).

Two Dimensional space As a proof of concept, we first extract the conservative part of a two dimensional vector field $F(x,y) = (-x+y, -x-y)$. The field corresponds to a spiralling sink. We train an untied auto-encoder with 1000 ReLU units for 500 epochs using BFGS over an equally spaced grid of 100 points in each dimension. Figure 4 shows how the conservative part is perfectly recovered.

High Dimensional space We also conducted experiments with high dimensional vector fields. We created a continum of vector fields by considering convex combinations of a conservative and a non-conservative field. The former is obtained by training a tied auto-encoder on MNIST and the latter by setting the parameters of an auto-encoder to random values. That is, we have $(W_i, R_i) = \beta(W_0, R_0) + (1-\beta)(W_K, R_K)$ where (W_0, R_0) is the non-conservative auto-encoder and (W_K, R_K) is the conservative auto-encoder. We repeatedly train a tied auto-encoder on this continuum

[5] For Laplace-deRham, $\Delta = d\delta + \delta d$. Standard Δ on 1-forms is $d\delta$.



(a) Initial vector field (b) Final vector field (c) Magnitude of curl during training

Figure 3: Initial and final vector field after training an untied auto-encoder on the spiral dataset.

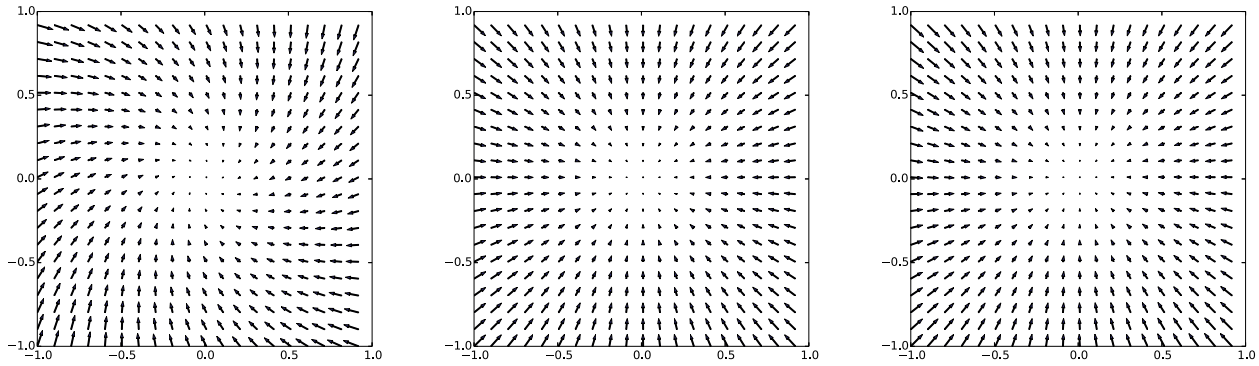


Figure 4: Vector field learning by tied (Middle) and united (Right) auto-encoder on 2D unconservative vector field (Left).

Algorithm 1 Learning to approximate a conservative field with an auto-encoder

procedure ($\mathcal{D}$ be a data set)
  Let (W_0, R_0) be a random weights for AE.
  Let (W_K, R_K) be trained AE on $\mathcal{D}$.
  Generate F_i $\forall i = 1 \cdots K$ as follows:
  - $(W_i, R_i) = \beta(W_0, R_0) + (1 - \beta)(W_K, R_K)$
  - Sample $\mathbf{x}_i$ from uniform distributon in the data space.
  - $\mathcal{F}_i = \{(\mathbf{x}_i, r(\mathbf{x}_i))\text{for} i = 1 \cdots N\}$

  for each vector field F_i, **do**
    Train a tied Auto-encoder on F_i
    Compute $E(\mathbf{x})$ where $\mathbf{x} \in \mathcal{D}$
    Compute $E(\tilde{\mathbf{x}})$ where $\tilde{\mathbf{x}} \sim$ Binomial
    Count number of $E(\mathbf{x}) > E(\tilde{\mathbf{x}})$.

in order to learn its conservative part. The pseudocode for the experiment is presented in Algorithm 1.

Figure 5 shows the mean squared error as a function of training epoch for different values of β. We observe that the auto-encoder's loss function decreases as β gets closer to 1. This is due to the auto-encoder only being able to learn the conservative component of the vector field.

We then compare the unnormalized model evidence of the auto-encoders. The comparison is based on computing the potential energy of auto-encoders given two points at a time. These two points are from the MNIST and a corrupted version of the latter using salt and pepper noise. We count the number of times where $E(\mathbf{x}) > E(\mathbf{x}_{\text{rand}})$. Given that the weights (W_K, R_K) of the conservative auto-encoder are obtained by training it on MNIST, the potential energy at MNIST data points should be higher than that at the corrupted MNIST data points. However, this does not hold for

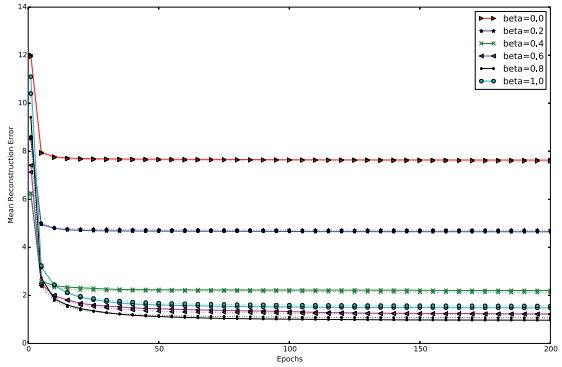


Figure 5: Learning curves for tied (dashed) and untied (solid) auto-encoders.

$\beta < 1$. Even for $\beta = 0.6$, we can recover the conservative component of the vector field up to 93% . Thus, we conclude

Table 2: The fraction of observations with $E(\mathbf{x}) > E(\mathbf{x}_{\text{rand}})$ for different β values.

β	0.0	0.2	0.4	0.6	0.8	1.0
CVF=Tied AE	0.5036	0.7357	0.9338	0.98838	0.9960	0.9968
CVF=Untied AE	0.5072	0.7496	0.9373	0.98595	0.9958	0.9968

that the tied auto-encoder is able to learn the conservative component of the vector field. The procedure is detailed in Algorithm 1.

Table 2 shows that, on average, the auto-encoder potential energy increasingly favors the original MNIST examples over the corrupted ones as the vector field F_i moves from 0 to K. "CVF=Tied AE" refers to conservative vector field F_K trained by the tied auto-encoder and "CVF=Untied AE" refers to conservative vector field F_K learned by the untied auto-encoder.

References

Alain, G., and Bengio, Y. 2014. What regularized auto-encoders learn from the data generating distribution. In *Internaton Conference on Learning Representations*.

Bengio, Y. 2014. How auto-encoders could provide credit assignment in deep networks via target propagation. *arXiv preprint arXiv:1407.7906*.

Bhatia, H.; Norgard, G.; Pascucci, V.; and Bremer, P.-T. 2013. The helmtholz-hodge decomposition-a survey. *IEEE Transactions on visualization and computer graphics* 19:1386–1404.

Hinton, G. 2010. A practical guide to training restricted boltzmann machines, version 1. *Momentum* 9.

Hyvärinen, A. 2005. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research* 6.

James, R. 1966. Advanced calculus. In *Advanced Calculus*. Belmont, CA: Wadsworth.

Kamyshanska, H. 2013. On autoencoder scoring. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Macedo, I., and Castro, R. 2008. Numerical solution of the naiver-stokes equations. In *Technical report*.

Seung, H. 1998. Learning continuous attractors in recurrent networks. In *Proceedings of the Neural Information Processing Systems (NIPS)*, 654–660.

Swersky, K.; Buchman, D.; Marlin, B.; and de Freitas, N. 2011. On autoencoders and score matching for energy based models. In *International Conference on Machine Learning (ICML)*.

Vincent, P.; Larochelle, H.; Bengio, Y.; and Manzagol, P. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning (ICML)*.