Regularized Auto-Encoders Estimate Local Statistics

Guillaume Alain, Yoshua Bengio and Salah Rifai
Department of Computer Science and Operations Research
University of Montreal
Montreal, H3C 3J7, Quebec, Canada

Abstract

What do auto-encoders learn about the underlying data generating distribution? Recent work suggests that some auto-encoder variants do a good job of capturing the local manifold structure of the unknown data generating density. This paper clarifies some of these previous intuitive observations by showing that minimizing a particular form of regularized reconstruction error yields a reconstruction function that locally characterizes the shape of the data generating density. More precisely, we show that the auto-encoder captures the score (derivative of the log-density with respect to the input) or the local mean associated with the unknown data-generating density. This is the second result linking denoising auto-encoders and score matching, but in way that is different from previous work, and can be applied to the case when the auto-encoder reconstruction function does not necessarily correspond to the derivative of an energy function. The theorems provided here are completely generic and do not depend on the parametrization of the auto-encoder: they show what the auto-encoder would tend to if given enough capacity and examples. These results are for a contractive training criterion we show to be similar to the denoising auto-encoder training criterion with small corruption noise, but with contraction applied on the whole reconstruction function rather than just encoder. Similarly to score matching, one can consider the proposed training criterion as a convenient alternative to maximum likelihood, i.e., one not involving a partition function. Finally, we make the connection to existing sampling algorithms for such autoencoders, based on an MCMC walking near the high-density manifold.

1 Introduction

Machine learning is about capturing aspects of the unknown distribution from which the observed data are sampled (the *data-generating distribution*). For many learning algorithms and in particular in *manifold learning*, the focus is on identifying the regions (sets of points) in the space of examples where this distribution concentrates, i.e., which configurations of the observed variables are plausible.

Unsupervised *representation-learning* algorithms attempt to characterize the data-generating distribution through the discovery of a set of features or latent variables whose variations capture most of the structure of the data-generating distribution. In recent years, a number of unsupervised feature learning algorithms have been proposed that are based on minimizing some form of *reconstruction error*, such as auto-encoder and sparse coding variants (Olshausen and Field, 1997; Bengio *et al.*, 2007; Ranzato *et al.*, 2007; Jain and Seung, 2008; Ranzato *et al.*, 2008; Vincent *et al.*, 2008; Kavukcuoglu *et al.*, 2009; Rifai *et al.*, 2011a,b; Gregor *et al.*, 2011). An auto-encoder reconstructs the input through two stages, an encoder function f (which outputs a learned representation $h = f(x)$ of an example x) and a decoder function g, such that $g(f(x)) \approx x$ for most x sampled from the data-generating distribution. These feature learning algorithms can be *stacked* to form

deeper and more abstract representations. *Deep learning* algorithms learn multiple levels of representation, where the number of levels is data-dependent. There are theoretical arguments and much empirical evidence to suggest that when they are well-trained, deep learning algorithms (Hinton *et al.*, 2006; Bengio, 2009; Lee *et al.*, 2009; Salakhutdinov and Hinton, 2009; Bengio and Delalleau, 2011) can perform better than their shallow counterparts, both in terms of learning features for the purpose of classification tasks and for generating higher-quality samples.

Here we restrict ourselves to the case of continuous inputs $x \in \mathbb{R}^d$ with the data-generating distribution being associated with an unknown *target density* function, denoted p. Manifold learning algorithms assume that p is concentrated in regions of lower dimension (Cayton, 2005; Narayanan and Mitter, 2010), i.e., the training examples are by definition located very close to these high-density manifolds. In that context, the core objective of manifold learning algorithms is to identify where the density concentrates.

Some important questions remain concerning many of feature learning algorithms based on reconstruction error. *What is their training criterion learning about the input density*? Do these algorithms implicitly learn about the whole density or only some aspect? If they capture the essence of the target density, then can we formalize that link and in particular exploit it to *sample from the model*? This would turn these algorithms into *implicit density* models, which only define a density indirectly, e.g., through the estimation of statistics or through a generative procedure that converges to it. These are the questions to which this paper contributes.

The paper is divided in two sections, along with detailed appendices with proofs of the theorems. See the supplementary document to this article for the appendices. The first section makes a direct link between denoising auto-encoders (Vincent *et al.*, 2008) and contractive auto-encoders (Rifai *et al.*, 2011a), justifying the interest in the contractive training criterion studied in the rest of the paper. The second and main section regards the following question: when minimizing that criterion, *what does an auto-encoder learn about the data generating density*? The answer is that it estimates the *score* (first derivative of the log-density), i.e., the direction in which density is increasing the most, which also corresponds to the *local mean*, which is the expected value in a small ball around the current location. Finally, we briefly introduce the connection with a stochastic diffusion equation associated with the Langevin MCMC that motivates a sampling algorithm based on the estimated score.

The order of presentation for the results in the appendices follows the mathematical development. This differs slightly from the order in which we present results in this paper and it leaves certain gaps. This tradeoff was made so that any result in the paper could easily be looked up in the appendices by searching for the one bearing the same name or number.

2 Contractive and Denoising Auto-Encoders

Regularized auto-encoders (see Bengio *et al.* (2012b) for a review and a longer exposition) capture the structure of the training distribution thanks to the productive opposition between reconstruction error and a regularizer. An auto-encoder maps inputs x to an internal representation (or code) $f(x)$ through the encoder function f, and then maps back $f(x)$ to the input space through a decoding function g. The composition of f and g is called the reconstruction function r, with $r(x) = g(f(x))$, and a reconstruction loss function ℓ penalizes the error made, with $r(x)$ viewed as a prediction of x. When the auto-encoder is regularized, e.g., via a sparsity regularizer, a contractive regularizer (detailed below), or a denoising form of regularization (that we find below to be very similar to a contractive regularizer), the regularizer basically attempts to make r (or f) as simple as possible, i.e., as constant as possible, as unresponsive to x as possible. It means that f has to throw away some information present in x, or at least represent it with less precision. On the other hand, to make reconstruction error small, examples that are neighbors on a high-density manifold must be represented with sufficiently different values of $f(x)$ or $r(x)$. Otherwise, it would not be possible to distinguish and hence correctly reconstruct these examples. It means that the derivatives of $f(x)$ or $r(x)$ in the directions along the manifold must remain large, while the derivatives (of f or r) in the x-directions orthogonal to the manifold can be made very small. This is illustrated in Figure 1 below. In the case of Principal Components Analysis, one constrains the derivative to be exactly 0 in the directions orthogonal to the chosen projection directions, and around 1 in the chosen projection directions. In regularized auto-encoders, f is non-linear, meaning that it is allowed to choose differ-

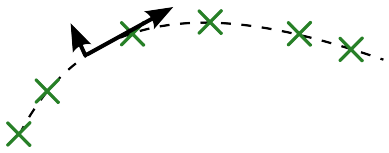


Figure 1: Regularization forces the auto-encoder to become less sensitive to the input, but minimizing reconstruction error forces it to remain sensitive to variations along the manifold of high density. Hence the representation and reconstruction end up capturing well variations on the manifold while mostly ignoring variations orthogonal to it.

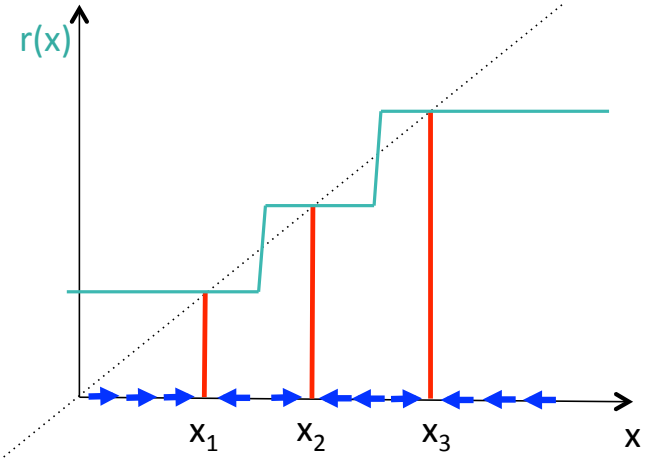


Figure 2: The reconstruction function $r(x)$ (in green) learned by an autoencoder on a 1-dimensional input with high capacity, minimizing reconstruction error *at the training examples* x_i (with in $r(x_i)$ in red) while trying to be as constant as possible otherwise (large λ). The figure is used to exagerate and illustrate the effect of the regularizer. The dotted line is the identity reconstruction (which might be obtained without the regularizer). The blue arrows shows the vector field of $r(x) - x$ pointing towards high density peaks as estimated by the model, and estimating the score (log-density derivative).

.

ent principal directions (those that are well represented, i.e., ideally the manifold tangent directions) at different x's, and this allows a regularized auto-encoder with non-linear encoder to capture non-linear manifolds. Figure 2 illustrates the extreme case when the regularization is very strong ($r(\cdot)$ wants to be nearly constant where density is high) in the special case where the distribution is highly concentrated at three points (three training examples). It shows the compromise between obtaining the identity function at the training examples and having a flat r near the trainign examples, yielding a $r(x) - x$ that points towards the high density points.

Here we show that the Denoising Auto-Encoder (Vincent *et al.*, 2008) with very small Gaussian corruption and squared error loss is actually a particular kind of Contractive Auto-Encoder (Rifai *et al.*, 2011a), contracting the whole auto-encoder reconstruction function rather than just the encoder, whose contraction penalty coefficient is the magnitude of the perturbation.

The Contractive Auto-Encoder or CAE (Rifai *et al.*, 2011a) is a particular form of regularized auto-encoder which is trained to minimize the following regularized reconstruction error:

$$\mathcal{L}_{CAE} = \mathbb{E}\left[\ell(x, r(x)) + \lambda \left\| \frac{\partial f(x)}{\partial x} \right\|_F^2\right] \tag{1}$$

where $r(x) = g(f(x))$ and $||A||_F^2$ is the sum of the squares of the elements of A. Both the squared loss $\ell(x, r) = \frac{1}{2}||x-r||^2$ and the cross-entropy loss $\ell(x, r) = -x \log r - (1-x) \log(1-r)$ have been used, but here we focus our analysis on the squared loss because of the easier mathematical treatment it allows. Note that success in minimizing the above criterion strongly depends on the parametriza-

tion of f and g and in particular on the tied weights constraint used, with $f(x) = \text{sigmoid}(Wx + b)$ and $g(h) = \text{sigmoid}(W^T h + c)$. The above regularizing term forces f (as well as g, because of the tied weights) to be contractive, i.e., to have singular values less than 1 [1]. Larger values of λ yield more contraction (smaller singular values) where it hurts reconstruction error the least, i.e., in the local directions where there are only little or no variations in the data. These typically are the directions orthogonal to the manifold of high density concentration.

The Denoising Auto-Encoder or DAE (Vincent *et al.*, 2008) is trained to minimize the following denoising criterion:

$$\mathcal{L}_{DAE} = \mathbb{E}\left[\ell(x, r(N(x)))\right] \tag{2}$$

where $N(x)$ is a stochastic corruption of x and the expectation is over the training distribution and the corruption noise source. Here we consider mostly the squared loss and Gaussian noise corruption, again because it is easier to handle them mathematically.

Theorem 1. *When using corruption noise $N(x) = x + \epsilon$ with*

$$\epsilon \sim \mathcal{N}\left(0, \sigma^2 I\right),$$

the objective function $\mathcal{L}_{DAE}$ is

$$\mathcal{L}_{DAE} = \frac{1}{2}\left(\mathbb{E}\left[\|x - r(x)\|^2\right] + \sigma^2 \mathbb{E}\left[\left\|\frac{\partial r(x)}{\partial x}\right\|_F^2\right]\right) + o(\sigma^2)$$

as $\sigma \to 0$.

Proof. With a Taylor expansion around x we have that

$$r(x + \epsilon) = r(x) + \frac{\partial r(x)}{\partial x}\epsilon + o(\sigma^2).$$

Substituting this into $\mathcal{L}_{DAE}$ we have that

$$\begin{aligned}
\mathcal{L}_{DAE} &= \mathbb{E}\left[\tfrac{1}{2}\left\|x - \left(r(x) + \tfrac{\partial r(x)}{\partial x}\epsilon + o(\sigma^2)\right)\right\|^2\right] \\
&= \tfrac{1}{2}\left(\mathbb{E}\left[\|x - r(x)\|^2\right] - 2E[\epsilon]^T \mathbb{E}\left[\tfrac{\partial r(x)}{\partial x}^T (x - r(x))\right]\right) \\
&\quad + \tfrac{1}{2} Tr\left(\mathbb{E}\left[\epsilon\epsilon^T\right] \mathbb{E}\left[\tfrac{\partial r(x)}{\partial x}^T \tfrac{\partial r(x)}{\partial x}\right]\right) + o(\sigma^2) \\
&= \tfrac{1}{2}\left(\mathbb{E}\left[\|x - r(x)\|^2\right] + \sigma^2 \mathbb{E}\left[\left\|\tfrac{\partial r(x)}{\partial x}\right\|_F^2\right]\right) + o(\sigma^2)
\end{aligned} \tag{3}$$

where in the second line we used the independance of the noise from x and properties of the trace, while in the last line we used $\mathbb{E}\left[\epsilon\epsilon^T\right] = \sigma^2 I$ and $\mathbb{E}[\epsilon] = 0$ by definition of ϵ. □

This derivation shows that *the DAE with small corruption is similar to a Contractive Auto-Encoder* but where the contraction is imposed explicitly on the whole reconstruction function $r(\cdot) = g(f(\cdot))$ rather than on $f(\cdot)$ alone (in the CAE there is a also a contractive effect on $g(\cdot)$ as a side effect of the parametrization with weights between $f(\cdot)$ and $g(\cdot)$).

This analysis motivates the study in the rest of this paper of the following training criterion: contractive penalty on the reconstruction plus squared reconstruction loss:

$$\mathcal{L} = \mathbb{E}\left[\|r(x) - x\|^2 + \lambda \left\|\frac{\partial r(x)}{\partial x}\right\|_F^2\right]. \tag{4}$$

This is an analytic version of the denoising criterion with small noise $\lambda = \sigma^2$, and also corresponds to a contractive auto-encoder with contraction on both f and g, i.e., on r.

[1] Note that an auto-encoder without any regularization would tend to find many leading singular values near 1 in order to minimize reconstruction error, i.e., preserve input norm in all the directions of variation present in the data.

3 Minimizing the Loss to Recover Local Features of $p(\cdot)$

3.1 Solution from Calculus of Variations

The central result of this paper is that in a non-parametric setting (without parametric constraints on r) the loss function defined by (4) can be solved asymptotically as $\lambda \to 0$. The exact meaning of this claim is made clearer in the following theorem.

Theorem 2. *Let p be a probability density function that is continuously differentiable once and with support $\mathbb{R}^d$ (i.e. $\forall x \in \mathbb{R}^d$ we have $p(x) \neq 0$). Let $\mathcal{L}$ be the expected loss functional defined by*

$$\mathcal{L}(r) = \int_{\mathbb{R}^d} p(x) \left[\|r(x) - x\|_2^2 + \lambda \left\| \frac{\partial r(x)}{\partial x} \right\|_F^2 \right] dx \tag{5}$$

for $r : \mathbb{R}^d \to \mathbb{R}^d$ assumed to be continuously differentiable once, and $0 < \lambda \in \mathbb{R}$ used as factor to the penalty term.

Then the optimal function $r^*(x)$ that minimizes $\mathcal{L}$ is such that

$$r^*(x) = x + \lambda \frac{\partial \log p(x)}{\partial x} + o(\lambda) \quad \textit{as} \quad \lambda \to 0.$$

The proof is given in the appendix and uses the Euler-Lagrange equations from the calculus of variations.

The idea that the scaling factor $\lambda > 0$ of the penalty term is brought to very small values is related to Theorem 1 where the scaling factor σ is also studied for the asymptotic behavior as $\sigma \to 0$.

3.2 Empirical Loss

In an experimental setting, the expected loss (5) is replaced by the empirical loss

$$\hat{\mathcal{L}} = \frac{1}{N} \sum_{n=1}^{N} \left(\left\| r(x^{(n)}) - x^{(n)} \right\|_2^2 + \lambda \left\| \left. \frac{\partial r(x)}{\partial x} \right|_{x=x^{(n)}} \right\|_F^2 \right)$$

based on a sample $\left\{x^{(n)}\right\}_{n=1}^{N}$ drawn from $p(x)$.

Alternatively, the auto-encoder is trained online (by stochastic gradient updates) with a stream of examples $x^{(n)}$, which corresponds to performing stochastic gradient descent on the expected loss (5). In both cases we obtain an auto-encoder that approximately minimizes the expected loss.

An interesting question is the following: what can we infer from the data generating density when given an autoencoder reconstruction function $r(x)$?

The premise is that this autoencoder $r(x)$ was trained to approximately minimize a loss function that has exactly the form of (5) for some $\lambda > 0$. This is assumed to have been done through minimizing the empirical loss and the distribution p was only available indirectly through the samples $\left\{x^{(n)}\right\}_{n=1}^{N}$. We do not have access to p or to the samples. We have only $r(x)$ and maybe λ.

We will now discuss the usefulness of $r(x)$ based on different conditions such as the model capacity and the value of λ.

3.2.1 Perfect World Scenario

As a starting point, we will assume that we are in a perfect situation, i.e., with no constraint on r (non-parametric setting), an infinite amount of training data, and a perfect minimization. We will see what can be done to recover information about p in that ideal case. Afterwards, we will drop certain assumptions one by one and discuss the possible paths to getting back some information about p.

We use $r_\lambda(x)$ notation when we want to emphasize the fact that the value of $r(x)$ came from minimizing the loss with a certain fixed λ.

Suppose that $r_\lambda(x)$ was trained with an infinite sample drawn from p. Suppose also that it had infinite (or sufficient) model capacity and that it is capable of achieving the minimum of the loss function (5) while satisfying the constraints that $\frac{\partial r(x)}{\partial x}$ exists and is continuous. Suppose that we know the value of λ and that we are working in a computing environment of arbitrary precision (i.e. no rounding errors).

As shown by Theorem 2, we would be able to get numerically the values of $\frac{\partial \log p(x)}{\partial x}$ at any point $x \in \mathbb{R}^d$ by simply evaluating

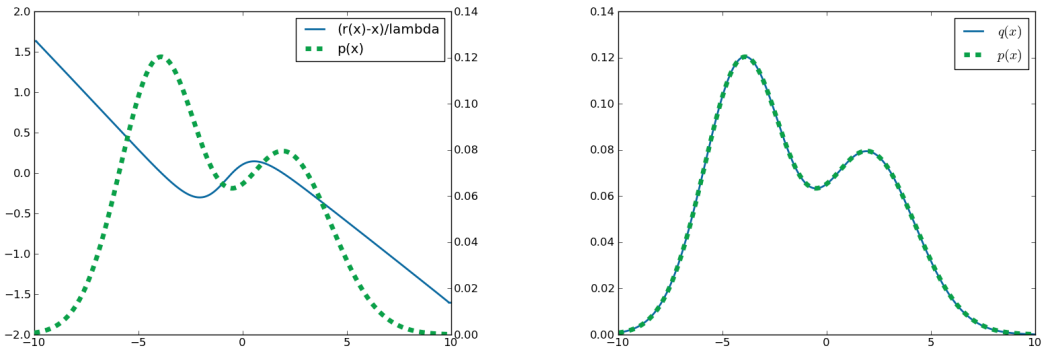
$$\frac{r_\lambda(x) - x}{\lambda} \rightarrow \frac{\partial \log p(x)}{\partial x} \quad \text{as} \quad \lambda \rightarrow 0. \tag{6}$$

In the setup described, we do not get to pick values of λ so as to take the limit $\lambda \rightarrow 0$. However, it is assumed that λ is already sufficiently small that the above quantity is close to $\frac{\partial \log p(x)}{\partial x}$ for all intents and purposes.

3.2.2 Simple Numerical Example

To give an example of this in one dimension, we will show what happens when we train a non-parametric model $\hat{r}(x)$ to minimize numerically the loss (5) relative to $p(x)$.

The distribution $p(x)$ studied is shown in Figure 3 and it was created to be simple enough to illustrate the mechanics.



(a) $p(x)$ and $(\hat{r}(x) - x)/\lambda$

(b) $p(x)$ and $q(x) \propto exp\left(\int^x \frac{\hat{r}(y)-y}{\lambda} dy\right)$

Figure 3: Left (a): the data generating density $p(x)$ along with the estimated score $\frac{r(x)-x}{\lambda}$. Right (b): in the one-dimensional case, one can directed convert the estimated score into an estimated density q by simply summing the score from left to right.

The model $\hat{r}(x)$ is fitted by dividing the interval $[-10, 10]$ into $M = 1000$ partition points $x_1, \ldots, x_M$ evenly separated by a distance Δ. We then minimize numerically the loss function

$$\sum_{i=1}^{M} p(x_i)\Delta \left(\hat{r}(x_i) - x_i\right)^2 + \lambda \sum_{i=1}^{M-1} p(x_i)\Delta \left(\frac{\hat{r}(x_{i+1}) - \hat{r}(x_i)}{\Delta}\right)^2.$$

Every value $\hat{r}(x_i)$ for $i = 1, \ldots, M$ is treated as a free parameter. Setting to 0 the derivative with respect to the $\hat{r}(x_i)$ yields a system of linear equations in M unknowns. It can be solved exactly and this solution is plotted in Figure 3(a).

As expected, we get into numerical instabilities as λ becomes too small, but for $\lambda = 10^{-4}$ we can verify that $\max_{x_i \in [-10,10]} |p(x) - q(x)| \leq 2.6 \times 10^{-4}$ for the properly normalized distribution $q(x) \propto \exp\left(\sum_{x_i < x} \frac{\hat{r}(x_i)-x_i}{\lambda}\right)$, and this is visually seen as a good fit to the data generating density p in Figure 3(b).

This example supports the claim of Theorem 2. That is, the minimizer of the loss 5 behaves as $x + \lambda \frac{\partial \log p(x)}{\partial x} + o(\lambda)$ as $\lambda \to 0$. This is illustrated by showing how the best numerical solution to a discretized version of the problem has exactly that behavior.

3.2.3 Missing λ

When we are in the same setting as in section 3.2.1 but the value of λ is unknown, we can modify (6) a bit and avoid dividing by λ. That is, for a trained reconstruction function $r(x)$ given to us we just take the quantity $r(x) - x$ and it should be approximatively the score *up to a multiplicative constant*.

$$r(x) - x \propto \frac{\partial \log p(x)}{\partial x}$$

Equivalently, if one estimates the density via an energy function (minus the unnormalized log density), then $x - r(x)$ estimates the gradient of the energy function.

We still have to assume that λ is small. Otherwise, if the unknown λ is too large we might get a poor estimation of the score.

3.2.4 Relation to Denoising Score Matching

Naturally, the driving assumption behind the above statements is still that $r(x)$ minimizes a loss of the form (5) for some $p(x)$ and λ. If it comes from any other procedure, or if it is limited in capacity (because we are in a parametric setting not including or close enough to the target function) and cannot achieve the minimum of (5), then more work is needed to be able to provide formal guarantees.

However, a reassuring and very related result was obtained by Vincent (2011). Motivated by the analysis of denoising auto-encoders, it is concerned with the case where we explicitly parametrize a score function $\psi(x)$ and we stochastically corrupt the original samples $x \sim p$ to obtain noisy samples $\tilde{x} \sim q_\sigma(\tilde{x}|x)$. In particular, the article analyses the case where q_σ adds Gaussian noise of variance σ^2 to x. The main result is that minimizing the expected square difference between $\psi(\tilde{x})$ and the score of $q_\sigma(\tilde{x}|x)$,

$$E_{x,\tilde{x}}[||\psi(\tilde{x}) - \frac{\partial q_\sigma(\tilde{x}|x)}{\partial \tilde{x}}||^2],$$

is equivalent to performing *score matching* (Hyvärinen, 2005) with estimator $\psi(\tilde{x})$ and target density $q_\sigma(\tilde{x}) = \int q_\sigma(\tilde{x}|x)p(x)dx$, where $p(x)$ generates the training samples x. Note that when a finite training set is used, $q_\sigma(\tilde{x})$ is simply a smooth of the empirical distribution (e.g. the Parzen density with Gaussian kernel of width σ). When the corruption noise is Gaussian, $\frac{q_\sigma(\tilde{x}|x)}{\partial \tilde{x}} = \frac{x - \tilde{x}}{\sigma^2}$, from which we can deduce that if we define a reconstruction function

$$r(\tilde{x}) = \tilde{x} + \sigma^2 \psi(\tilde{x}), \tag{7}$$

then the above expectation is equivalent to

$$E_{x,\tilde{x}}[||\frac{r(\tilde{x}) - \tilde{x}}{\sigma^2} - \frac{x - \tilde{x}}{\sigma^2}||^2] = \frac{1}{\sigma^2} E_{x,\tilde{x}}[||r(\tilde{x}) - x||^2]$$

which is the denoising criterion. This says that when the reconstruction function r is parametrized so as to correspond to the score ψ of a model density (as per eq. 7), the denoising criterion on r with Gaussian corruption noise is equivalent to score matching with respect to a smooth of the data generating density, i.e., a regularized form of score matching. Note that this regularization appears desirable, because matching the score of the empirical distribution (or an insufficiently smoothed version of it) could yield undesirable results when the training set is finite. Since score matching has been shown to be a consistent induction principle (Hyvärinen, 2005), it means that this *denoising score matching* (Vincent, 2011; Kingma and LeCun, 2010; Swersky *et al.*, 2011) criterion recovers the underlying density, up to the smoothing induced by the noise of variance σ^2. By making σ^2 small, we can make the estimator arbitrarily good (and we would expect to want to do that as the amount of training data increases). Note the correspondance of this conclusion with the results presented here, which show that (1) $\lambda = \sigma^2$ and (2) that minimizing the equivalent analytic criterion (based on a contraction penalty) estimates the score when λ is small. The difference is that our result holds even when r is not parametrized as per eq. 7, i.e., is not forced to correspond with the score function of a density.

3.3 Local Mean

What previous work on denoising and contractive auto-encoders suggest is that regularized auto-encoders can *capture the local structure of the density* through the value of the encoding (or reconstruction) function and its derivative. In particular, Rifai *et al.* (2012); Bengio *et al.* (2012a) argue that the first and second derivatives tell us in which directions it makes sense to randomly move while preserving or increasing the density, thereby justifying sampling procedures. This motivates us here to study so-called local moments as captured by the auto-encoder, and in particular the local mean, following the definitions introduced in Bengio *et al.* (2012a).

3.3.1 Definitions for Local Distributions

Let p be a continuous probability density function with support $\mathbb{R}^d$. That is, $\forall x \in \mathbb{R}^d$ we have that $p(x) \neq 0$. We define below the notion of a *local ball* $B_\delta(x_0)$, along with an associated *local density*, which is the normalized product of p with the indicator for the ball:

$$\begin{aligned} B_\delta(x_0) &= \{x \quad \text{s.t. } \|x - x_0\|_2 < \delta\} \\ Z_\delta(x_0) &= \int_{B_\delta(x_0)} p(x)dx \\ p_\delta(x|x_0) &= \frac{1}{Z_\delta(x_0)} p(x)\mathbb{I}\left(x \in B_\delta(x_0)\right) \end{aligned}$$

where $Z_\delta(x_0)$ is the normalizing constant required to make $p_\delta(x|x_0)$ a valid pdf for a distribution centered on x_0. The support of $p_\delta(x|x_0)$ is the ball of radius δ around x_0 denoted by $B_\delta(x_0)$. We stick to the 2-norm in terms of defining the balls $B_\delta(x_0)$ used, but everything could be rewritten in terms of another p-norm to have slightly different formulas.

We use the following notation for what will be referred to as the first two *local moments* (i.e. local mean and local covariance) of the random variable described by $p_\delta(x|x_0)$.

$$m_\delta(x_0) \stackrel{def}{=} \int_{\mathbb{R}^d} x p_\delta(x|x_0)dx$$

$$C_\delta(x_0) \stackrel{def}{=} \int_{\mathbb{R}^d} (x - m_\delta(x_0))(x - m_\delta(x_0))^T p_\delta(x|x_0)dx$$

Based on these definitions, one can prove (in appendix) the following theorem.

Theorem 3. *Let p be of class C^3 and represent a probability density function. Let $x_0 \in \mathbb{R}^d$ with $p(x_0) \neq 0$. Then we have that*

$$m_\delta(x_0) = x_0 + \delta^2 \frac{1}{d+2} \left.\frac{\partial \log p(x)}{\partial x}\right|_{x_0} + o\left(\delta^3\right).$$

This links the local mean of a density with the score associated with that density. Combining this theorem with Theorem 2, we obtain that the optimal reconstruction function $r^*(\cdot)$ also estimates the local mean:

$$m_\delta(x) - x = \frac{\delta^2}{\lambda(d+2)} \left(r^*(x) - x\right) \ + A(\delta) + \delta^2 B(\lambda) \tag{8}$$

for error terms $A(\delta), B(\lambda)$ such that

$$\begin{aligned} A(\delta) \in o(\delta^3) \quad &\text{as} \quad \delta \to 0, \\ B(\lambda) \in o(1) \quad &\text{as} \quad \lambda \to 0. \end{aligned}$$

This means that we can loosely estimate the *direction* to the local mean by the direction of the reconstruction:

$$m_\delta(x) - x \ \propto \ r^*(x) - x. \tag{9}$$

4 Sampling with Diffusion

The sampling algorithm from Bengio *et al.* (2012a), inspired from Rifai *et al.* (2012), proposes a way to sample from a distribution p if we have access to its local mean $m_\delta(x)$ and local covariance $C_\delta(x)$. Basically, these sampling algorithms proceed in steps where the next sample x_{t+1} is obtained by a sample from a Gaussian whose mean and covariance are respectively $m_\delta(x_t)$ and $C_\delta(x_t)$. What we have shown above is that the regularized auto-encoder estimates both the score and the local mean.

4.1 Diffusion equation as justification

Towards better understanding what happens when we use the estimated score or estimated local mean to drive random moves we can turn to stochastic differential equations. More precisely, we want a diffusion equation whose discretization would match the sampling methods described in Bengio *et al.* (2012a) and Rifai *et al.* (2012). Here we focus on the local mean and leave the study of the local covariance to future work, but we show that in principle the local mean or estimated score is already sufficient to sample from the underlying density.

We start from the presentation of such diffusion equations in Robert and Casella (1999). The Langevin diffusion X_t is defined by the stochastic differential equation

$$dX_t = \frac{1}{2}\nabla \log f(X_t) + dB_t \tag{10}$$

for B_t the standard Brownian motion. It has f as stationary distribution, which is in good part what we are interested in. The discretized form of this equation is

$$x^{(t+1)} = x^{(t)} + \frac{\sigma^2}{2}\nabla \log f(x^{(t)}) + \sigma\epsilon_t \tag{11}$$

where $\epsilon_t \sim \mathcal{N}(0, I)$ and σ^2 corresponds to the discretization size.

Without going into all the conditions required for this to work, we can see that this looks very similar to our situation if we truncate our expansion for $m_\delta(x)$, leaving only

$$\widehat{m_\delta}(x) = x + \frac{\delta^2}{d+2}\frac{\partial \log p(x)}{\partial x}, \tag{12}$$

If we assume that the local covariance is nearly a constant times the identity (which we are confident of being able to show), then the sampling algorithm from Bengio *et al.* (2012a) corresponds to discrete steps of the form

$$x^{(t+1)} = x^{(t)} + \frac{\delta^2}{d+2}\nabla \log p(x^{(t)}) + \sqrt{2\frac{\delta^2}{d+2}}\epsilon_t.$$

By taking $\sigma^2 = 2\frac{\delta^2}{d+2}$ we can see that we have a discretization just like (11) coming from (10). The stationary distribution in our case is p.

This shows a way in which we can implicitly estimate p from a trained regularized autoencoder, by drawing samples according to a Langevin MCMC using the estimated score. This is interesting because in the case where $r(x) - x$ is not constrained to be the derivative of some energy function and the data dimension $d > 1$, it is not clear how to recover an estimator of the underlying data generating distribution. This shows that an *implicit estimation* is possible through a data generating procedure.

5 Conclusion

We have shown that regularized auto-encoders such as the Denoising Auto-Encoder and a form of Contractive Auto-Encoder estimate local properties of the data generating density such as the score and local mean. This suggests that minimizing a regularized reconstruction error may be an alternative to maximum likelihood for unsupervised learning.

There are still many open questions that remain to be studied. Can the results presented here be generalized to other forms of regularization? What happens when r is parametric and λ is not tiny? What is the connection between the reconstruction function (and in particular its derivative) and the second derivative of the density or the second local moment (local covariance)? Geometric intuition suggests immediate answers to these questions. If the density is smooth, then we conjecture that the leading term of local covariance is simply proportional to the identity matrix. On the other hand, if the density concentrates near a low-dimensional manifold, then we conjecture that the leading eigenvectors of the local covariance coincide with the leading eigenvectors of the Jacobian of the reconstruction function and with the minor eigenvectors of the density Hessian.

As we further our understanding auto-encoders as estimating local properties of the data generating density, this suggests sampling algorithms as discussed in Bengio *et al.* (2012a); Rifai *et al.* (2012). However, several mathematical questions remain open in this respect. Whereas the standard Langevin MCMC only uses the first derivative of the density, we would like to formalize and establish corresponding Markov chains that also exploit the second order information. Indeed, our preliminary experiments on image data suggest that only using the score (local mean) and not the estimated local covariance yields substantially worse samples, compared to using the leading eigenvectors of the Jacobian of the reconstruction as well (to allow movement on the manifold and not just towards it).

References

Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, **2**(1), 1–127. Also published as a book. Now Publishers, 2009.

Bengio, Y. and Delalleau, O. (2011). On the expressive power of deep architectures. In *ALT'2011*.

Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H. (2007). Greedy layer-wise training of deep networks. In *NIPS 19*, pages 153–160. MIT Press.

Bengio, Y., Alain, G., and Rifai, S. (2012a). Implicit density estimation by local moment matching to sample from auto-encoders. Technical report, arXiv:1207.0057.

Bengio, Y., Courville, A., and Vincent, P. (2012b). Unsupervised feature learning and deep learning: A review and new perspectives. Technical report, arXiv:1206.5538.

Cayton, L. (2005). Algorithms for manifold learning. Technical Report CS2008-0923, UCSD.

Dacorogna, B. (2004). *Introduction to the Calculus of Variations*. World Scientific Publishing Company.

Gregor, K., Szlam, A., and LeCun, Y. (2011). Structured sparse coding via lateral inhibition. In *Advances in Neural Information Processing Systems (NIPS 2011)*, volume 24.

Hinton, G. E., Osindero, S., and Teh, Y. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, **18**, 1527–1554.

Hyvärinen, A. (2005). Estimation of non-normalized statistical models using score matching. *Journal of Machine Learning Research*, **6**, 695–709.

Jain, V. and Seung, S. H. (2008). Natural image denoising with convolutional networks. In *NIPS'08*, pages 769–776.

Kavukcuoglu, K., Ranzato, M., Fergus, R., and LeCun, Y. (2009). Learning invariant features through topographic filter maps. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR'09)*, pages 1605–1612. IEEE.

Kingma, D. and LeCun, Y. (2010). Regularized estimation of image statistics by score matching. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 1126–1134.

Lee, H., Grosse, R., Ranganath, R., and Ng, A. Y. (2009). Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In L. Bottou and M. Littman, editors, *ICML 2009*. ACM, Montreal (Qc), Canada.

Narayanan, H. and Mitter, S. (2010). Sample complexity of testing the manifold hypothesis. In *NIPS'2010*.

Olshausen, B. A. and Field, D. J. (1997). Sparse coding with an overcomplete basis set: a strategy employed by V1? *Vision Research*, **37**, 3311–3325.

Ranzato, M., Poultney, C., Chopra, S., and LeCun, Y. (2007). Efficient learning of sparse representations with an energy-based model. In *NIPS'06*, pages 1137–1144. MIT Press.

Ranzato, M., Boureau, Y.-L., and LeCun, Y. (2008). Sparse feature learning for deep belief networks. In *NIPS'07*, pages 1185–1192, Cambridge, MA. MIT Press.

Rifai, S., Vincent, P., Muller, X., Glorot, X., and Bengio, Y. (2011a). Contracting auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the Twenty-eight International Conference on Machine Learning (ICML'11)*.

Rifai, S., Dauphin, Y., Vincent, P., Bengio, Y., and Muller, X. (2011b). The manifold tangent classifier. In *NIPS'2011*. Student paper award.

Rifai, S., Bengio, Y., Dauphin, Y., and Vincent, P. (2012). A generative process for sampling contractive auto-encoders. In *ICML'2012*.

Robert, C. P. and Casella, G. (1999). *Monte Carlo Statistical Methods*. Springer.

Salakhutdinov, R. and Hinton, G. (2009). Deep Boltzmann machines. In *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics (AISTATS 2009)*, volume 8.

Swersky, K., Ranzato, M., Buchman, D., Marlin, B., and de Freitas, N. (2011). On score matching for energy based models: Generalizing autoencoders and simplifying deep learning. In *Proc. ICML'2011*. ACM.

Vincent, P. (2011). A connection between score matching and denoising autoencoders. *Neural Computation*, **23**(7), 1661–1674.

Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. (2008). Extracting and composing robust features with denoising autoencoders. In *ICML'08*, pages 1096–1103. ACM.

6 Appendix

6.1 Calculus of Variations

Theorem 2. *Let p be a probability density function that is continuously differentiable once and with support $\mathbb{R}^d$ (i.e. $\forall x \in \mathbb{R}^d$ we have $p(x) \neq 0$). Let $\mathcal{L}$ be the loss function defined by*

$$\mathcal{L}(r) = \int_{\mathbb{R}^d} p(x) \left[\|r(x) - x\|_2^2 + \lambda \left\| \frac{\partial r(x)}{\partial x} \right\|_F^2 \right] dx$$

for $r : \mathbb{R}^d \to \mathbb{R}^d$ assumed to be continuously differentiable once, and $0 < \lambda \in \mathbb{R}$ used as factor to the penalty term.

Then the optimal function $r(x)$ that minimizes $\mathcal{L}$ is such that

$$r(x) = x + \lambda \frac{\partial \log p(x)}{\partial x} + o(\lambda) \quad as \quad \lambda \to 0.$$

Proof. To demonstrate this, we make use of the Euler-Lagrange equation from the Calculus of Variations. We would refer the reader to either (Dacorogna, 2004) or Wikipedia for more on the topic. Let

$$f(x_1, \ldots, x_n, r, r_{x_1}, \ldots, r_{x_n}) = p(x) \left[\|r(x) - x\|_2^2 + \lambda \left\| \frac{\partial r(x)}{\partial x} \right\|_F^2 \right]$$

where $x = (x_1, \ldots, x_d)$, $r(x) = (r_1(x), \ldots, r_d(x))$ and $r_{x_i} = \frac{\partial f}{\partial x_i}$.

We can rewrite the loss $\mathcal{L}(r)$ more explicitly as

$$\begin{aligned} \mathcal{L}(r) &= \int_{\mathbb{R}^d} p(x) \left[\sum_{i=1}^d (r_i(x) - x_i)_2^2 + \lambda \sum_{i=1}^d \sum_{j=1}^d \frac{\partial r_i(x)}{\partial x_j}^2 \right] dx \\ &= \sum_{i=1}^d \int_{\mathbb{R}^d} p(x) \left[(r_i(x) - x_i)_2^2 + \lambda \sum_{j=1}^d \frac{\partial r_i(x)}{\partial x_j}^2 \right] dx \end{aligned} \tag{13}$$

to observe that the components $r_1(x), \ldots, r_d(x)$ can each be optimized separately.

The Euler-Lagrange equation to be satisfied at the optimal $r : \mathbb{R}^d \to \mathbb{R}^d$ is

$$\frac{\partial f}{\partial r} = \sum_{i=1}^{d} \frac{\partial}{\partial x_i} \frac{\partial f}{\partial r_{x_i}}.$$

In our situation, the expressions from that equation are given by

$$\frac{\partial f}{\partial r} = 2(r(x) - x)p(x)$$

$$\frac{\partial f}{\partial r_{x_i}} = 2\lambda p(x) \begin{bmatrix} \frac{\partial r_1}{\partial x_i} & \frac{\partial r_2}{\partial x_i} & \cdots & \frac{\partial r_d}{\partial x_i} \end{bmatrix}^T$$

$$\begin{aligned} \frac{\partial}{\partial x_i}\left(\frac{\partial f}{\partial r_{x_i}}\right) &= 2\lambda \frac{\partial p(x)}{\partial x_i} \begin{bmatrix} \frac{\partial r_1}{\partial x_i} & \frac{\partial r_2}{\partial x_i} & \cdots & \frac{\partial r_d}{\partial x_i} \end{bmatrix}^T \\ &\quad + 2\lambda p(x) \begin{bmatrix} \frac{\partial^2 r_1}{\partial x_i^2} & \frac{\partial^2 r_2}{\partial x_i^2} & \cdots & \frac{\partial^2 r_d}{\partial x_i^2} \end{bmatrix}^T \end{aligned}$$

and the equality to be satisfied at the optimum becomes

$$(r(x) - x)p(x) = \lambda \sum_{i=1}^{d} \begin{bmatrix} \frac{\partial p(x)}{\partial x_i}\frac{\partial r_1}{\partial x_i} + p(x)\frac{\partial^2 r_1}{\partial x_i^2} \\ \vdots \\ \frac{\partial p(x)}{\partial x_i}\frac{\partial r_d}{\partial x_i} + p(x)\frac{\partial^2 r_d}{\partial x_i^2} \end{bmatrix}. \tag{14}$$

As equation (13) hinted, the expression (14) can be decomposed into the different components $r_k(x) : \mathbb{R}^d \to \mathbb{R}$ that make r. For $k = 1, \ldots, d$ we get

$$(r_k(x) - x_k)p(x) = \lambda \sum_{i=1}^{d} \left(\frac{\partial p(x)}{\partial x_i}\frac{\partial r_k(x)}{\partial x_i} + p(x)\frac{\partial^2 r_k(x)}{\partial x_i^2} \right).$$

As $p(x) \neq 0$ by hypothesis, we can divide all the terms by $p(x)$ and note that $\frac{\partial p(x)}{\partial x_i}/p(x) = \frac{\partial \log p(x)}{\partial x_i}$.

We get

$$r_k(x) - x_k = \lambda \sum_{i=1}^{d} \left(\frac{\partial \log p(x)}{\partial x_i}\frac{\partial r_k(x)}{\partial x_i} + \frac{\partial^2 r_k(x)}{\partial x_i^2} \right). \tag{15}$$

This first thing to observe is that when $\lambda = 0$ the solution is just $r_k(x) = x_k$, which translates into $r(x) = x$. This is not a surprise because it represents the perfect reconstruction value that we get when we the penalty term vanishes in the loss function.

The question is now that happens as $\lambda > 0$ gets close to 0. If we could solve (15) analytically, we would simply take the solution, compute its derivative with respect to λ, evaluate it at $\lambda = 0$, and that would be our answer. In a way, we are saying that

$$r(x) = x + \lambda h(x) + g(\lambda, x) \tag{16}$$

for some function $h(x) : \mathbb{R}^d \to \mathbb{R}^d$ that we want to identify, along with some remainder term $g(\lambda, x)$ that involves higher powers of λ. That is, $g(\lambda, x)$ is such that

$$\lim_{\lambda\to 0}\frac{1}{\lambda}g(\lambda,x)=0,\quad \lim_{\lambda\to 0}\frac{1}{\lambda}\frac{\partial g(\lambda,x)}{\partial x}=0,\quad \lim_{\lambda\to 0}\frac{1}{\lambda}\frac{\partial^2 g(\lambda,x)}{\partial x^2}=0.$$

With that $r(x)=x+\lambda h(x)+g(\lambda,x)$, we will focus on the k-th component

$$r_k(x)=x_k+\lambda h_k(x)+g_k(\lambda,x).$$

The first and second derivatives with respect to x_i are

$$\begin{aligned}\frac{\partial r_k(x)}{\partial x_i} &= \mathbb{I}(k=i)+\lambda\frac{\partial h_k(x)}{\partial x_i}+\frac{\partial g_k(\lambda,x)}{\partial x_i}\\ \frac{\partial^2 r_k(x)}{\partial x_i^2} &= 0+\lambda\frac{\partial^2 h_k(x)}{\partial x_i^2}+\frac{\partial^2 g_k(\lambda,x)}{\partial x_i^2}.\end{aligned}$$

Working on both sides of equation (15), we get

$$\begin{aligned}\lambda h_k(x)+g_k(x,\lambda) &= \lambda\sum_{i=1}^{d}\left(\frac{\partial\log p(x)}{\partial x_i}\left(\mathbb{I}(k=i)+\lambda\frac{\partial h_k(x)}{\partial x_i}+\frac{\partial g_k(\lambda,x)}{\partial x_i}\right)+\lambda\frac{\partial^2 h_k(x)}{\partial x_i^2}+\frac{\partial^2 g_k(\lambda,x)}{\partial x_i^2}\right)\\ &= \lambda\frac{\partial\log p(x)}{\partial x_k}+\lambda\sum_{i=1}^{d}\left(\frac{\partial\log p(x)}{\partial x_i}\left(\lambda\frac{\partial h_k(x)}{\partial x_i}+\frac{\partial g_k(\lambda,x)}{\partial x_i}\right)+\lambda\frac{\partial^2 h_k(x)}{\partial x_i^2}+\frac{\partial^2 g_k(\lambda,x)}{\partial x_i^2}\right).\end{aligned}$$

Dividing both sides by λ and taking the limit $\lambda\to 0$ we eliminate almost all the terms and are left with

$$h_k(x)=\frac{\partial\log p(x)}{\partial x_k}.$$

This means that (16) can be rewritten as

$$r(x)=x+\lambda\frac{\partial\log p(x)}{\partial x}+o(\lambda)\quad\text{as}\quad\lambda\to 0,$$

which completes the proof. □

6.2 Asymptotic formulas for localised moments

Proposition 1. *Let p be of class C^2 and let $x_0\in\mathbb{R}^d$. Then we have that*

$$Z_\delta(x_0)=\delta^d\frac{\pi^{d/2}}{\Gamma(1+d/2)}\left[p(x_0)+\delta^2\frac{Tr(H(x_0))}{2(d+2)}+o(\delta^3)\right]$$

where $H(x_0)=\left.\frac{\partial^2 p(x)}{\partial x^2}\right|_{x=x_0}$. Moreover, we have that

$$\frac{1}{Z_\delta(x_0)}=\delta^{-d}\frac{\Gamma(1+d/2)}{\pi^{d/2}}\left[\frac{1}{p(x_0)}-\delta^2\frac{1}{p(x_0)^2}\frac{Tr(H(x_0))}{2(d+2)}+o(\delta^3)\right].$$

Proof.

$$\begin{aligned}
Z_\delta(x_0) &= \int_{B_\delta(x_0)} \left[p(x_0) + \left.\frac{\partial p(x)}{\partial x}\right|_{x_0} (x - x_0) + \frac{1}{2!}(x - x_0)^T H(x_0)(x - x_0) \right. \\
&\quad \left. + \frac{1}{3!} D^{(3)} p(x_0)(x - x_0) + o(\delta^3) \right] dx \\
&= p(x_0) \int_{B_\delta(x_0)} dx \; + \; 0 \; + \; \frac{1}{2} \int_{B_\delta(x_0)} (x - x_0)^T H(x_0)(x - x_0) dx \; + \; 0 \; + \; o(\delta^{d+3}) \\
&= p(x_0) \delta^d \frac{\pi^{d/2}}{\Gamma\left(1 + d/2\right)} + \delta^{d+2} \frac{\pi^{d/2}}{4\Gamma\left(2 + d/2\right)} \mathrm{Tr}\left(H(x_0)\right) + o(\delta^{d+3}) \\
&= \delta^d \frac{\pi^{d/2}}{\Gamma\left(1 + d/2\right)} \left[p(x_0) + \delta^2 \frac{\mathrm{Tr}(H(x_0))}{2(d+2)} + o(\delta^3) \right]
\end{aligned}$$

We use Proposition 3 to get that trace come up from the integral involving $H(x_0)$. The expression for $1/Z_\delta(x_0)$ comes from the fact that, for any $a, b > 0$ we have that

$$\begin{aligned}
\frac{1}{a + b\delta^2 + o(\delta^3)} &= \frac{a^{-1}}{1 + \frac{b}{a}\delta^2 + o(\delta^3)} = \frac{1}{a} \left(1 - (\frac{b}{a}\delta^2 + o(\delta^3)) + o(\delta^4) \right) \\
&= \frac{1}{a} - \frac{b}{a^2}\delta^2 + o(\delta^3) \quad \text{as } \delta \to 0.
\end{aligned}$$

by using the classic result from geometric series where $\frac{1}{1+r} = 1 - r + r^2 - \ldots$ for $|r| < 1$.

Now we just apply this to

$$\frac{1}{Z_\delta(x_0)} = \delta^{-d} \frac{\Gamma\left(1 + d/2\right)}{\pi^{d/2}} \frac{1}{\left[p(x_0) + \delta^2 \frac{\mathrm{Tr}(H(x_0))}{2(d+2)} + o(\delta^3) \right]}$$

and get the expected result.

□

Theorem 3. *Let p be of class C^3 and represent a probability density function. Let $x_0 \in \mathbb{R}^d$ with $p(x_0) \neq 0$. Then we have that*

$$m_\delta(x_0) = x_0 + \delta^2 \frac{1}{d+2} \left.\frac{\partial \log p(x)}{\partial x}\right|_{x_0} + o\left(\delta^3\right).$$

Proof. The leading term in the expression for $m_\delta(x_0)$ is obtained by transforming the x in the integral into a $x - x_0$ to make the integral easier to integrate.

$$m_\delta(x_0) = \frac{1}{Z_\delta(x_0)} \int_{B_\delta(x_0)} x p(x) dx = x_0 + \frac{1}{z_\delta(x_0)} \int_{B_\delta(x_0)} (x - x_0) p(x) dx.$$

Now using the Taylor expansion around x_0

$$\begin{aligned}
m_\delta(x_0) &= x_0 + \frac{1}{Z_\delta(x_0)} \int_{B_\delta(x_0)} (x - x_0) \left[p(x_0) + \left.\frac{\partial p(x)}{\partial x}\right|_{x_0} (x - x_0) \right. \\
&\quad \left. + \frac{1}{2}(x - x_0)^T \left.\frac{\partial^2 p(x)}{\partial x^2}\right|_{x_0} (x - x_0) + o(\|x - x_0\|^2) \right] dx.
\end{aligned}$$

Remember that $\int_{B_\delta(x_0)} f(x)dx = 0$ whenever we have a function f is anti-symmetrical (or "odd") relative to the point x_0 (i.e. $f(x - x_0) = f(-x - x_0)$). This applies to the terms $(x - x_0)p(x_0)$ and $(x - x_0)(x - x_0)\left.\frac{\partial^2 p(x)}{\partial x^2}\right|_{x=x_0}(x - x_0)^T$. Hence we use Proposition 2 to get

$$\begin{aligned} m_\delta(x_0) &= x_0 + \frac{1}{Z_\delta(x_0)} \int_{B_\delta(x_0)} \left[(x - x_0)^T \left.\frac{\partial p(x)}{\partial x}\right|_{x_0} (x - x_0) + o(\|x - x_0\|^3) \right] dx \\ &= x_0 + \frac{1}{Z_\delta(x_0)} \left(\delta^{d+2} \frac{\pi^{\frac{d}{2}}}{2\Gamma\left(2 + \frac{d}{2}\right)} \right) \left.\frac{\partial p(x)}{\partial x}\right|_{x_0} + o(\delta^3). \end{aligned}$$

Now, looking at the coefficient in front of $\left.\frac{\partial p(x)}{\partial x}\right|_{x_0}$ in the first term, we can use Proposition 1 to rewrite it as

$$\frac{1}{Z_\delta(x_0)} \left(\delta^{d+2} \frac{\pi^{\frac{d}{2}}}{2\Gamma\left(2 + \frac{d}{2}\right)} \right) = \delta^{-d} \frac{\Gamma(1 + d/2)}{\pi^{d/2}} \left[\frac{1}{p(x_0)} - \delta^2 \frac{1}{p(x_0)^2} \frac{\mathrm{Tr}(H(x_0))}{2(d+2)} + o(\delta^3) \right] \delta^{d+2} \frac{\pi^{\frac{d}{2}}}{2\Gamma\left(2 + \frac{d}{2}\right)}$$

$$= \delta^2 \frac{\Gamma\left(1 + \frac{d}{2}\right)}{2\Gamma\left(2 + \frac{d}{2}\right)} \left[\frac{1}{p(x_0)} - \delta^2 \frac{1}{p(x_0)^2} \frac{\mathrm{Tr}(H(x_0))}{2(d+2)} + o(\delta^3) \right] = \delta^2 \frac{1}{p(x_0)} \frac{1}{d+2} + o(\delta^3).$$

There is no reason the keep the $-\delta^4 \frac{\Gamma\left(1+\frac{d}{2}\right)}{2\Gamma\left(2+\frac{d}{2}\right)} \frac{1}{p(x_0)^2} \frac{\mathrm{Tr}(H(x_0))}{2(d+2)}$ in the above expression because the asymptotic error from the remainder term in the main expression is $o(\delta^3)$. That would swallow our exact expression for δ^4 and make it useless.

We end up with

$$m_\delta(x_0) = x_0 + \delta^2 \frac{1}{d+2} \left.\frac{\partial \log p(x)}{\partial x}\right|_{x_0} + o(\delta^3).$$

□

6.3 Integration on balls and spheres

This result comes from *Multi-dimensional Integration : Scary Calculus Problems* from Tim Reluga (who got the results from *How to integrate a polynomial over a sphere* by Gerald B. Folland).

Theorem 4. *Let* $B = \left\{ x \in \mathbb{R}^d \,\middle|\, \sum_{j=1}^d x_j^2 \leq 1 \right\}$ *be the ball of radius* 1 *around the origin. Then*

$$\int_B \prod_{j=1}^d |x_j|^{a_j} \, dx = \frac{\prod \Gamma\left(\frac{a_j+1}{2}\right)}{\Gamma\left(1 + \frac{d}{2} + \frac{1}{2}\sum a_j\right)}$$

for any real numbers $a_j \geq 0$.

Corollary 1. *Let* B *be the ball of radius* 1 *around the origin. Then*

$$\int_B \prod_{j=1}^d x_j^{a_j} dx = \begin{cases} \frac{\prod \Gamma\left(\frac{a_j+1}{2}\right)}{\Gamma\left(1+\frac{d}{2}+\frac{1}{2}\sum a_j\right)} & \text{if all the } a_j \text{are even integers} \\ 0 & \text{otherwise} \end{cases}$$

for any non-negative integers $a_j \geq 0$. *Note the absence of the absolute values put on the* $x_j^{a_j}$ *terms.*

Corollary 2. *Let $B_\delta(0) \subset \mathbb{R}^d$ be the ball of radius δ around the origin. Then*

$$\int_{B_\delta(0)} \prod_{j=1}^{d} x_j^{a_j} dx = \begin{cases} \delta^{d+\sum a_j} \frac{\prod \Gamma\left(\frac{a_j+1}{2}\right)}{\Gamma\left(1+\frac{d}{2}+\frac{1}{2}\sum a_j\right)} & \text{if all the } a_j \text{are even integers} \\ 0 & \text{otherwise} \end{cases}$$

for any non-negative integers $a_j \geq 0$. Note the absence of the absolute values on the $x_j^{a_j}$ terms.

Proof. We take the theorem as given and concentrate here on justifying the two corollaries.

Note how in Corollary 1 we dropped the absolute values that were in the original Theorem 4. In situations where at least one a_j is odd, we have that the function $f(x) = \prod_{j=1}^{d} x_j^{a_j}$ becomes odd in the sense that $f(-x) = -f(x)$. Because of the symmetrical nature of the integration on the unit ball, we get that the integral is 0 as a result of cancellations.

For Corollary 2, we can rewrite the integral by changing the domain with $y_j = x_j/\delta$ so that

$$\delta^{-\sum a_j} \int_{B_\delta(0)} \prod_{j=1}^{d} x_j^{a_j} dx = \int_{B_\delta(0)} \prod_{j=1}^{d} (x_j/\delta)^{a_j} dx = \int_{B_1(0)} \prod_{j=1}^{d} y^{a_j} \delta^d dy.$$

We pull out the δ^d that we got from the determinant of the Jacobian when changing from dx to dy and Corollary 2 follows.

□

Proposition 2. *Let $v \in \mathbb{R}^d$ and let $B_\delta(0) \subset \mathbb{R}^d$ be the ball of radius δ around the origin. Then*

$$\int_{B_\delta(0)} y < v, y > dy = \left(\delta^{d+2} \frac{\pi^{\frac{d}{2}}}{2\Gamma\left(2+\frac{d}{2}\right)} \right) v$$

where $< v, y >$ is the usual dot product.

Proof. We have that

$$y < v, y > \quad = \quad \begin{bmatrix} v_1 y_1^2 \\ \vdots \\ v_d y_d^2 \end{bmatrix}$$

which is decomposable into d component-wise applications of Corollary 2. This yields the expected result with the constant obtained from $\Gamma\left(\frac{3}{2}\right) = \frac{1}{2}\Gamma\left(\frac{1}{2}\right) = \frac{1}{2}\sqrt{\pi}$.

□

Proposition 3. *Let $H \in \mathbb{R}^{d\times d}$ and let $B_\delta(x_0) \subset \mathbb{R}^d$ be the ball of radius δ around $x_0 \in \mathbb{R}^d$. Then*

$$\int_{B_\delta(x_0)} (x-x_0)^T H(x-x_0) dx = \delta^{d+2} \frac{\pi^{d/2}}{2\Gamma\left(2+d/2\right)} \textit{trace}\left(H\right).$$

Proof. First, by substituting $y = (x - x_0)/\delta$ we have that this is equivalent to showing that

$$\int_{B_1(0)} y^T H y dy = \frac{\pi^{d/2}}{2\Gamma\left(2+d/2\right)} \text{trace}\left(H\right).$$

This integral yields a real number which can be written as

$$\int_{B_1(0)} y^T H y dy = \int_{B_1(0)} \sum_{i,j} y_i H_{i,j} y_j dy = \sum_{i,j} \int_{B_1(0)} y_i y_j H_{i,j} dy.$$

Now we know from Corollary 2 that this integral is zero when $i \neq j$. This gives

$$\sum_{i,j} H_{i,j} \int_{B_1(0)} y_i y_j dy = \sum_i H_{i,i} \int_{B_1(0)} y_i^2 dy = \text{trace}\,(H)\, \frac{\pi^{d/2}}{2\Gamma\,(2 + d/2)}.$$

□