Improving Variational Auto-Encoders using Householder Flow

Jakub M. Tomczak, Max Welling
University of Amsterdam
J.M.Tomczak@uva.nl, M.Welling@uva.nl

Abstract

Variational auto-encoders (VAE) are scalable and powerful generative models. However, the choice of the variational posterior determines tractability and flexibility of the VAE. Commonly, latent variables are modeled using the normal distribution with a diagonal covariance matrix. This results in computational efficiency but typically it is not flexible enough to match the true posterior distribution. One fashion of enriching the variational posterior distribution is application of *normalizing flows*, *i.e.*, a series of invertible transformations to latent variables with a simple posterior. In this paper, we follow this line of thinking and propose a *volume-preserving flow* that uses a series of *Householder transformations*. We show empirically on MNIST dataset and histopathology data that the proposed flow allows to obtain more flexible variational posterior and highly competitive results comparing to other normalizing flows.

1 Variational Auto-Encoder

Let $\mathbf{x}$ be a vector of D observable variables, $\mathbf{z} \in \mathbb{R}^M$ a vector of stochastic latent units (variables) and let $p(\mathbf{x}, \mathbf{z})$ be a parametric model of the joint distribution. Given N datapoints $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ we typically aim at maximizing the marginal log-likelihood:

$$\ln p(\mathbf{X}) = \sum_{i=1}^{N} \ln p(\mathbf{x}_i), \tag{1}$$

with respect to parameters. This task could be troublesome due to intractability of the marginal likelihood, *e.g.*, when the model is parameterized by a neural network (NN). To overcome this issue one can introduce an *inference model* (an *encoder*) $q(\mathbf{z}|\mathbf{x})$ and optimize the variational lower bound:

$$\ln p(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}[\ln p(\mathbf{x}|\mathbf{z})] - \mathrm{KL}\big(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\big), \tag{2}$$

where $p(\mathbf{x}|\mathbf{z})$ is called a *decoder* and $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})$ is the *prior*. There are various ways of optimizing this lower bound but for continuous $\mathbf{z}$ this could be done efficiently through a re-parameterization of $q(\mathbf{z}|\mathbf{x})$ [9, 15]. Then the architecture is called a *variational auto-encoder* (VAE).

In practice, the inference model assumes a diagonal covariance matrix, *i.e.*, $q(\mathbf{z}|\mathbf{x}) = \mathcal{N}\big(\mathbf{z}|\boldsymbol{\mu}(\mathbf{x}), \mathrm{diag}(\boldsymbol{\sigma}^2(\mathbf{x}))\big)$, where $\boldsymbol{\mu}(\mathbf{x})$ and $\boldsymbol{\sigma}^2(\mathbf{x})$ are parameterized by the NN. However, the assumption about the diagonal posterior can be insufficient and not flexible enough to match the true posterior.

2 Improving posterior flexibility using *Normalizing Flows*

A (finite) *normalizing flow*, first formulated by [20, 21] and further developed by [14], is a powerful framework for building flexible posterior distribution by starting with an initial random variable with a simple distribution for generating $\mathbf{z}^{(0)}$ and then applying a series of invertible transformations $\mathbf{f}^{(t)}$, for $t = 1, \ldots, T$. As a result, the last iterate gives a random variable $\mathbf{z}^{(T)}$ that has a more flexible distribution. Once we choose transformations $\mathbf{f}^{(t)}$ for which the Jacobian-determinant can be computed, we aim at optimizing the following objective:

$$\ln p(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{z}^{(0)}|\mathbf{x})}\Big[\ln p(\mathbf{x}|\mathbf{z}^{(T)}) + \sum_{t=1}^{T} \ln \Big|\det \frac{\partial \mathbf{f}^{(t)}}{\partial \mathbf{z}^{(t-1)}}\Big|\Big] - \mathrm{KL}\big(q(\mathbf{z}^{(0)}|\mathbf{x})||p(\mathbf{z}^{(T)})\big). \tag{3}$$

In fact, the normalizing flow can be used to enrich the posterior of the VAE with small or even none modifications in the architecture of the encoder and the decoder.

There are two main kinds of normalizing flows, namely, *general normalizing flows* and *volume preserving flows*. The difference between these types of flow is in the manner how the Jacobian-determinant is handled. The general normalizing flows aim at formulating the flow for which the Jacobian-determinant is relatively easy to compute. On the contrary, the volume-preserving flows design series of transformations such that the Jacobian-determinant equals 1 while still it allows to obtain flexible posterior distributions. The reduced computational complexity is the main reason why volume-preserving flows are so appealing. The question is whether one can propose a series of transformations for which the Jacobian-determinant is equal one, they are cheap to calculate and are general enough to model flexible posteriors. In the next subsection we present a new volume-preserving flow that applies series of Householder transformations that we refer to as the *Householder flow*.

3 Householder Flow

3.1 Motivation

In general, any full-covariance matrix $\mathbf{\Sigma}$ can be represented by the eigenvalue decomposition using eigenvectors and eigenvalues:

$$\mathbf{\Sigma} = \mathbf{U}\mathbf{D}\mathbf{U}^{\top}, \tag{4}$$

where $\mathbf{U}$ is an orthogonal matrix with eigenvectors in columns, $\mathbf{D}$ is a diagonal matrix with eigenvalues. In the case of the vanilla VAE, it would be tempting to model the matrix $\mathbf{U}$ to obtain a full-covariance matrix. The procedure would require a linear transformation of a random variable using an orthogonal matrix $\mathbf{U}$. Since the absolute value of the Jacobian determinant of an orthogonal matrix is 1, for $\mathbf{z}^{(1)} = \mathbf{U}\mathbf{z}^{(0)}$ one gets $\mathbf{z}^{(1)} \sim \mathcal{N}(\mathbf{U}\boldsymbol{\mu}, \mathbf{U}\,\mathrm{diag}(\boldsymbol{\sigma}^2)\,\mathbf{U}^{\top})$. If $\mathrm{diag}(\boldsymbol{\sigma}^2)$ coincides with true $\mathbf{D}$, then it would be possible to resemble the true full-covariance function. Hence, the main goal would be to model the orthogonal matrix of eigenvectors.

Generally, the task of modelling an orthogonal matrix in a principled manner is rather non-trivial. However, first we notice that any orthogonal matrix can be represented in the following form [2, 19]:

Theorem 1. (The Basis-Kernel Representation of Orthogonal Matrices)
For any $M \times M$ orthogonal matrix $\mathbf{U}$ there exist a full-rank $M \times K$ matrix $\mathbf{Y}$ (the basis) and a nonsingular (triangular) $K \times K$ matrix $\mathbf{S}$ (the kernel), $K \leq M$, such that:

$$\mathbf{U} = \mathbf{I} - \mathbf{Y}\mathbf{S}\mathbf{Y}^{\top}. \tag{5}$$

The value K is called the *degree* of the orthogonal matrix. Further, it can be shown that any orthogonal matrix of degree K can be expressed using the product of Householder transformations [2, 19], namely:

Theorem 2. *Any orthogonal matrix with the basis acting on the K-dimensional subspace can be expressed as a product of exactly K Householder transformations:*

$$\mathbf{U} = \mathbf{H}_K \mathbf{H}_{K-1} \cdots \mathbf{H}_1, \tag{6}$$

where $\mathbf{H}_k = \mathbf{I} - \mathbf{S}_{kk}\mathbf{Y}_{\cdot k}(\mathbf{Y}_{\cdot k})^{\top}$, for $k = 1, \ldots, K$.

Theoretically, Theorem 2 shows that we can model any orthogonal matrix in a principled fashion using K Householder transformations. Moreover, the Householder matrix $\mathbf{H}_k$ is *orthogonal* matrix itself [7]. Therefore, this property and the Theorem 2 put the Householder transformation as a perfect candidate for formulating a volume-preserving flow that allows to approximate (or even capture) the true full-covariance matrix.

3.2 Definition

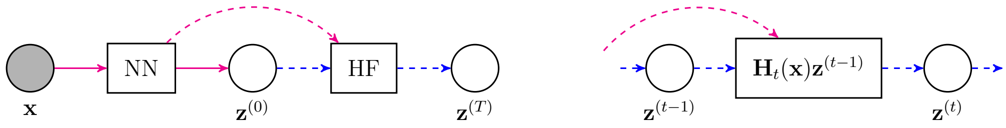
The *Householder transformation* is defined as follows. For a given vector $\mathbf{z}^{(t-1)}$ the reflection hyperplane can be defined by a vector (a *Householder vector*) $\mathbf{v}_t(\mathbf{x}) \in \mathbb{R}^M$ that is orthogonal to the hyperplane, and the reflection of this point about the hyperplane is [7]:

$$\mathbf{z}^{(t)} = \Big(\mathbf{I} - 2\frac{\mathbf{v}_t(\mathbf{x})\mathbf{v}_t(\mathbf{x})^\top}{||\mathbf{v}_t(\mathbf{x})||^2}\Big)\mathbf{z}^{(t-1)} \tag{7}$$

$$= \mathbf{H}_t(\mathbf{x})\mathbf{z}^{(t-1)}, \tag{8}$$

where $\mathbf{H}_t(\mathbf{x}) = \mathbf{I} - 2\frac{\mathbf{v}_t(\mathbf{x})\mathbf{v}_t(\mathbf{x})^\top}{||\mathbf{v}_t(\mathbf{x})||^2}$ is called the *Householder matrix*.

The most important property of $\mathbf{H}_t(\mathbf{x})$ is that it is an orthogonal matrix and hence the absolute value of the Jacobian determinant is equal 1. This fact significantly simplifies the objective (3) because $\ln\left|\det\frac{\partial \mathbf{H}_t(\mathbf{x})\mathbf{z}^{(t-1)}}{\partial \mathbf{z}^{(t-1)}}\right| = 0$, for $t = 1, \ldots, T$. Starting from a simple posterior with the diagonal covariance matrix for $\mathbf{z}^{(0)}$, the series of T linear transformations given by (7) defines a new type of volume-preserving flow that we refer to as the *Householder flow* (HF). The vectors $\mathbf{v}_t(\mathbf{x})$, $t = 1, \ldots, T$, are produced by the encoder network along with means and variances. The idea of the Householder flow is schematically presented in Figure 1. Once the encoder returns the set of Householder vectors, the Householder flow requires T linear operations to produce a sample from a more flexible posterior with (approximate) full-covariance matrix.



(a) encoder network + Householder Flow

(b) one step of the Householder Flow

Figure 1: A schematical representation of the encoder network with the Householder flow. (a) The general architecture of the VAE+HF: The encoder returns means and variances for the posterior and a set of Householder vectors that are used to formulate the Householder flow. (b) A single step of the Householder flow that uses linear Householder transformation. In both figures solid lines correspond to the encoder network and the dashed lines are additional quantities required by the HF.

4 Related Work

In [14] invertible linear normalizing flows with known Jacobian determinant were proposed. These are easy to calculate, *i.e.*, the determinant of the Jacobian can be analytically computed, however, many such transformations are needed to capture high-dimensional dependencies. A different approach relies on *volume-preserving flows*, for which the absolute value of Jacobian determinant is equal 1, such as, Non-linear Independent Components Estimation (NICE) [5], Hamiltonian Variational Inference (HVI) [17] and linear Inverse Autoregressive Flow (linIAF) or its non-linear version (nlIAF) [10].

The HF is similar in spirit to the linIAF where the linear transformation is also applied but it is the lower triangular inverse Cholesky matrix with ones on the diagonal instead of the Householder matrix. Nevertheless, the motivation of the linIAF differs from ours completely.

The Householder transformations (reflections) were also exploited in the context of learning recurrent neural nets [1]. They were used for modelling unitary weights instead of more flexible variational posterior, however, these served as a component for representing a matrix of eigenvectors, similarly to our approach.

Acknowledgments

The research conducted by Jakub M. Tomczak was funded by the European Commission within the Marie Skłodowska-Curie Individual Fellowship (Grant No. 702666, ”Deep learning and Bayesian inference for medical imaging”).

References

[1] Martin Arjovsky, Amar Shah, and Yoshua Bengio. Unitary evolution recurrent neural networks. *ICML*, 2016.

[2] Christian Bischof and Xiaobai Sun. On orthogonal block elimination. *Argonne National Laboratory, Argonne, IL, Tech. Rep. MCS-P450-0794*, 1994.

[3] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.

[4] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

[5] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

[6] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, volume 9, pages 249–256, 2010.

[7] Alston S Householder. Unitary triangularization of a nonsymmetric matrix. *Journal of the ACM (JACM)*, 5(4):339–342, 1958.

[8] Diederik Kingma and Jimmy Ba. ADAM: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[9] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[10] Diederik P Kingma, Tim Salimans, Rafał Józefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improving variational inference with inverse autoregressive flow. *NIPS*, 2016.

[11] Yann LeCun, Corinna Cortes, and Christopher JC Burges. The MNIST database of handwritten digits, 1998.

[12] Yingzhen Li and Richard E Turner. Rényi divergence variational inference. *arXiv preprint arXiv:1602.02311*, 2016.

[13] Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. *ICML*, pages 1747–1756, 2016.

[14] Danilo Rezende and Shakir Mohamed. Variational Inference with Normalizing Flows. In *ICML*, pages 1530–1538, 2015.

[15] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

[16] Ruslan Salakhutdinov and Iain Murray. On the quantitative analysis of deep belief networks. In *ICML*, pages 872–879. ACM, 2008.

[17] Tim Salimans, Diederik P Kingma, and Max Welling. Markov chain Monte Carlo and Variational Inference: Bridging the gap. In *ICML*, pages 1218–1226, 2015.

[18] Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder variational autoencoders. *arXiv preprint arXiv:1602.02282*, 2016.

[19] Xiaobai Sun and Christian Bischof. A basis-kernel representation of orthogonal matrices. *SIAM Journal on Matrix Analysis and Applications*, 16(4):1184–1196, 1995.

[20] EG Tabak and Cristina V Turner. A family of nonparametric density estimation algorithms. *Communications on Pure and Applied Mathematics*, 66(2):145–164, 2013.

[21] Esteban G Tabak, Eric Vanden-Eijnden, et al. Density estimation by dual ascent of the log-likelihood. *Communications in Mathematical Sciences*, 8(1):217–233, 2010.

[22] Dustin Tran, Rajesh Ranganath, and David M Blei. The Variational Gaussian Process. *ICLR*, 1050:23, 2016.