Journal of Machine Learning Research 13 (2012) 307-361 Submitted 12/10; Revised 11/11; Published 2/12

NCE of Unnormalized Statistical Models , with Applications to Natural Image Statistics

Michael U. Gutmann MICHAEL.GUTMANN@HELSINKI.FI
Aapo Hyvärinen AAPO.HYVARINEN@HELSINKI.FI
Department of Computer Science
Department of Mathematics and Statistics
Helsinki Institute for Information Technology HIIT
University of Helsinki, Finland

Editor: Yoshua Bengio

Abstract

We consider the task of estimating, from observed data, a probabilistic model that is parameterized by a finite number of parameters. In particular, we are considering the situation where the model pdf is unnormalized . That is, the model is only specified up to the partition function . The partition function normalizes a model so that it integrates to one for any choice of the parameters. However, it is often impossible to obtain it in closed form. Gibbs distributions, Markov and multi-layer networks are examples of models where analytical normalization is often impossible . MLE can then not be used without resorting to numerical approximations which are often computationally expensive . We propose here a new objective function for the estimation of both normalized and unnormalized models . The basic idea is to perform nonlinear logistic regression to discriminate between the observed data and some artificially generated noise. With this approach , the normalizing partition function can be estimated like any other parameter. We prove that the new estimation method leads to a consistent (convergent) estimator of the parameters . For large noise sample sizes , the new estimator is furthermore shown to behave like the mle. In the estimation of unnormalized models , there is a trade -off between statistical and computational performance . We show that the new method strikes a competitive trade -off in comparison to other estimation methods for unnormalized models. As an application to real data, we estimate novel two-layer models of natural image statistics with spline nonlinearities.

Keywords: unnormalized models, partition function, computation, estimation, natural image statistics

1. Introduction

This paper is about parametric density estimation, where the general setup is as follows. A sample $X = (\mathbf{x}_1, \ldots, \mathbf{x}_{T_d})$ of a random vector $\mathbf{x} \in \mathbb{R}^n$ is observed which follows an unknown pdf p_d. The data-pdf p_d is modeled by a parameterized family of functions $\{p_m(.;\boldsymbol{\theta})\}_{\boldsymbol{\theta}}$ where $\boldsymbol{\theta}$ is a vector of parameters . It is commonly assumed that p_d belongs to this family . In other words , $p_d(.) = p_m(.;\boldsymbol{\theta}^\star)$ for some parameter $\boldsymbol{\theta}^\star$. The parametric density estimation problem is then about finding $\boldsymbol{\theta}_\star$ from the observed sample X.

Any estimate $\hat{\theta}$ must yield a properly normalized pdf $p_m(.;\hat{\boldsymbol{\theta}})$:

$$\int p_m(\mathbf{u};\hat{\boldsymbol{\theta}})\mathrm{d}\mathbf{u} = 1, \qquad p_m(.;\hat{\boldsymbol{\theta}}) \geq 0. \tag{1}$$

If the model $p_m(.;\boldsymbol{\theta})$ is such that the constraints hold for all $\boldsymbol{\theta}$, and not only $\hat{\boldsymbol{\theta}}$, we say that the model is normalized . If the model is specified such that the positivity constraint but not the normalization constraint is satisfied for all parameters , we say that the model is unnormalized . By assumption there is, however, at least one value of the parameters for which an unnormalized model integrates to one, namely $\boldsymbol{\theta}^\star$.

In order to highlight that a model, parameterized by $\boldsymbol{\alpha}$, is unnormalized, we denote it by ${p_m}^0(.;\boldsymbol{\alpha})$.

The partition function

$$Z(\boldsymbol{\alpha}) = \int p_m^0(\mathbf{u};\boldsymbol{\alpha})\mathrm{d}\mathbf{u}, \tag{2}$$

can be used to convert an unnormalized model $p_m^0(.;\boldsymbol{\alpha})$ into a normalized one: $p_m^0(.;\boldsymbol{\alpha})/Z(\boldsymbol{\alpha})$ integrates to one for every value of $\boldsymbol{\alpha}$. Examples of distributions which are often specified by means of an unnormalized model and the partition function are Gibbs distributions, Markov networks or multilayer networks. The function $\boldsymbol{\alpha} \mapsto Z(\boldsymbol{\alpha})$ is, however, defined via an integral. Unless $p_m^0(.;\boldsymbol{\alpha})$ has some particularly convenient form, the integral cannot be computed analytically so that the function $Z(\boldsymbol{\alpha})$ is not available in closed form. For low-dimensional problems, numerical integration can be used to approximate the function $Z(\boldsymbol{\alpha})$ to a very high accuracy but for high-dimensional problems this is computationally expensive. Our paper deals with density estimation in this case, that is, with density estimation when the computation of the partition function is analytically intractable and computationally expensive.

Several solutions for the estimation of unnormalized models which cannot be normalized in closed form have been suggested so far. Geyer (1994) proposed to approximate the calculation of the partition function by means of importance sampling and then to maximize the approximate log-likelihood (Monte Carlo maximum likelihood). Approximation of the gradient of the log-likelihood led to another estimation method (contrastive divergence by Hinton, 2002). Estimation of the parameter $\boldsymbol{\alpha}$ directly from an unnormalized model $p_m^0(.;\boldsymbol{\alpha})$ has been proposed by Hyvärinen (2005). This approach, called score matching, avoids the problematic integration to obtain the partition function altogether. All these methods need to balance the accuracy of the estimate and the time to compute the estimate.

In this paper,[1] we propose a new estimation method for unnormalized models. The idea is to consider Z, or $c = \ln 1/Z$, not any more as a function of $\boldsymbol{\alpha}$ but as an additional parameter of the model. That is, we extend the unnormalized model $p_m^0(.;\boldsymbol{\alpha})$ to include a normalizing parameter c and estimate

$$\ln p_m(.;\boldsymbol{\theta}) = \ln p_m^0(.;\boldsymbol{\alpha}) + c,$$

with parameter vector $\boldsymbol{\theta} = (\boldsymbol{\alpha}, c)$. The estimate $\hat{\boldsymbol{\theta}} = (\hat{\boldsymbol{\alpha}}, \hat{c})$ is then such that the unnormalized model $p_m^0(.;\hat{\boldsymbol{\alpha}})$ matches the shape of p_d, while $\hat{c}$ provides the proper scaling so that Eq (1) holds.

1. Preliminary versions were presented at AISTATS (Gutmann and Hyvärinen, 2010) and ICANN (Gutmann and Hyvärinen, 2009).

Unlike in the approach based on the partition function, we aim not at normalizing $p_m^0(.;\boldsymbol{\alpha})$ for all $\boldsymbol{\alpha}$ but only for $\hat{\boldsymbol{\alpha}}$. This avoids the problematic integration in the definition of the partition function $\boldsymbol{\alpha} \mapsto Z(\boldsymbol{\alpha})$. Such a separate estimation of shape and scale is, however, not possible for MLE. The reason is that the likelihood can be made arbitrarily large by setting the normalizing parameter c to larger and larger numbers. The new estimation method which we propose here is based on the maximization of a well defined objective function. There are no constraints in the optimization so that powerful optimization techniques can be employed. The intuition behind the new objective function is to learn to classify between the observed data and some artificially generated noise. We approach thus the density estimation problem, which is an unsupervised learning problem, via supervised learning. The new method relies on noise which the data is contrasted to, so that we will refer to it as "noise-contrastive estimation".

The paper is organized in four main sections. In Section 2, we present NCE and prove fundamental statistical properties such as consistency. In Section 3, we validate and illustrate the derived properties on artificial data. We use artificial data also in Section 4 in order to compare the new method to the aforementioned estimation methods with respect to their statistical and computational efficiency. In Section 5, we apply NCE to real data. We estimate two-layer models of natural images and also learn the nonlinearities from the data. This section is fairly independent from the other ones. The reader who wants to focus on natural image statistics may not need to go first through the previous sections. On the other hand, the reader whose interest is in estimation theory only can skip this section without missing pieces of the theory although the section provides, using real data, a further illustration of the workings of unnormalized models and the new estimation method. Section 6 concludes the paper.

2. NCE

This section presents the theory of NCE. In Section 2.1, we motivate NCE and relate it to supervised learning. The definition of NCE is given in Section 2.2. In Section 2.3, we prove that the estimator is consistent for both normalized and unnormalized models, and derive its asymptotic distribution. In Section 2.4, we discuss practical aspects of the estimator and show that, in some limiting case, the estimator performs as well as MLE.

2.1 Density Estimation by Comparison

Density estimation is much about characterizing properties of the observed data X. A convenient way to describe properties is to describe them relative to the properties of some reference data Y. Assume that the reference (noise) data Y is an i.i.d. sample $(\mathbf{y}_1, \ldots \mathbf{y}_{T_n})$ of a rv $\mathbf{y} \in \mathbb{R}^n$ with pdf p_n. A relative description of the data X is then given by the ratio p_d/p_n of the two density functions. If the reference distribution p_n is known, one can, of course, obtain p_d from the ratio p_d/p_n. In other words, if one knows the differences between X and Y, and also the properties of Y, one can deduce from the differences the properties of X.

Comparison between two data sets can be performed via classification: In order to discriminate between two data sets, the classifier needs to compare their properties. In the following, we show that training a classifier based on logistic regression provides a relative description of X in the form of an estimate of the ratio p_d/p_n.

augmented data

Denote by $U = (\mathbf{u}_1, \ldots, \mathbf{u}_{T_d+T_n})$ the union of X and Y, and assign to each data point $\mathbf{u}_t$ a binary class label C_t: $C_t = 1$ if $\mathbf{u}_t \in X$ and $C_t = 0$ if $\mathbf{u}_t \in Y$. In logistic regression, the posterior probabilities of the classes given the data are estimated. As the pdf p_d of the data $\mathbf{x}$ is unknown, we model the class-conditional probability $p(.|C=1)$ with $p_m(.;\boldsymbol{\theta})$.[2] The class-conditional probability densities are

augmented distr.

$$p(\mathbf{u}|C=1;\boldsymbol{\theta}) = p_m(\mathbf{u};\boldsymbol{\theta}), \qquad p(\mathbf{u}|C=0) = p_n(\mathbf{u}).$$

The prior probabilities are $P(C=1) = T_d/(T_d+T_n)$ and $P(C=0) = T_n/(T_d+T_n)$. The posterior probabilities for the classes are therefore

$$P(C=1|\mathbf{u};\boldsymbol{\theta}) = \frac{p_m(\mathbf{u};\boldsymbol{\theta})}{p_m(\mathbf{u};\boldsymbol{\theta}) + \nu p_n(\mathbf{u})}, \qquad P(C=0|\mathbf{u};\boldsymbol{\theta}) = \frac{\nu p_n(\mathbf{u})}{p_m(\mathbf{u};\boldsymbol{\theta}) + \nu p_n(\mathbf{u})}, \tag{3}$$

where $\nu := P(C=0)/P(C=1) \sim T_n/T_d$. $h(u; \theta) := P(C=1|\mathbf{u};\boldsymbol{\theta})$ 0/1-classifier of data U

$$G(\mathbf{u};\boldsymbol{\theta}) := \ln p_m(\mathbf{u};\boldsymbol{\theta}) - \ln p_n(\mathbf{u}), \quad \text{model-noise LR} \tag{4}$$

==>

$$h(\mathbf{u};\boldsymbol{\theta}) = r_\nu\left(G(\mathbf{u};\boldsymbol{\theta})\right), \tag{5}$$

where

$$r_\nu(u) = \frac{1}{1+\nu\exp(-u)} \quad = \text{expit}(u - \ln \nu) \tag{6}$$

The conditional log-likelihood of aug. dstr. $p(C \mid u, \theta)$

not joint log-likelihood!!!

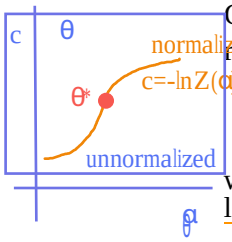
$$\ell(\boldsymbol{\theta}) \;=\; \sum_{t=1}^{T_d} \ln\left[h(\mathbf{x}_t;\boldsymbol{\theta})\right] + \sum_{t=1}^{T_n} \ln\left[1-h(\mathbf{y}_t;\boldsymbol{\theta})\right]. \quad = - \sum_t \text{Hb}(u_t, h(x_t, \Theta)) \tag{7}$$

Optimizing $\ell(\boldsymbol{\theta})$ wrt $\boldsymbol{\theta}$ leads to an estimate $G(.;\boldsymbol{\theta}^\wedge)$ of the log-ratio $\ln(p_d/p_n)$. That is, an approximate description of X relative to Y can be obtained by optimization of Eq (7). The sign-flipped objective function, $-\ell(\boldsymbol{\theta})$, is the cross-entropy error function (Bishop, 1995).

Density estimation, an unsupervised learning problem, can be performed by logistic regression, that is, supervised learning. Hastie et al. (2009, Chapter 14.2.4, pp. 495–497).

2. Classically, $p_m(.;\boldsymbol{\theta})$ would, in the context of this section, be a normalized pdf. In our paper, however, $\boldsymbol{\theta}$ may include a parameter for the normalization of the model.

2.2 Definition of the Estimator



Given an unnormalized statistical model $p_m^0(.;\boldsymbol{\alpha})$, we include for normalization an additional parameter c into the model:

$$\ln p_m(.;\boldsymbol{\theta}) = \ln p_m^0(.;\boldsymbol{\alpha}) + c,$$

where $\boldsymbol{\theta}=(\boldsymbol{\alpha},c)$. The parameter c scales the unnormalized model $p_m{}^0(.;\boldsymbol{\alpha})$ so that Eq (1) can be fulfilled. After learning, $\hat{c}$ provides an estimate for $\ln 1/Z(\hat{\boldsymbol{\alpha}})$.

$X = (\mathbf{x}_1,\ldots,\mathbf{x}_{T_d})$: the observed data set that consists of T_d independent observations of $\mathbf{x}\in\mathbb{R}^n$. $Y = (\mathbf{y}_1,\ldots,\mathbf{y}_{T_n})$: an artificially generated data set that consists of $T_n = \nu T_d$ independent observations of noise $\mathbf{y}\in\mathbb{R}^n$ with known distribution p_n. The estimator $\hat{\boldsymbol{\theta}}_T$ maximizes

$$J_T(\boldsymbol{\theta}) = \frac{1}{T_d}\left\{\sum_{t=1}^{T_d}\ln\left[h(\mathbf{x}_t;\boldsymbol{\theta})\right] + \sum_{t=1}^{T_n}\ln\left[1-h(\mathbf{y}_t;\boldsymbol{\theta})\right]\right\}, \tag{8}$$

where $h(.;\boldsymbol{\theta})$ was defined in (5). J_T is, up to the division by T_d, the log-likelihood in (7)

$$J_T(\boldsymbol{\theta}) = \frac{1}{T_d}\sum_{t=1}^{T_d}\ln\left[h(\mathbf{x}_t;\boldsymbol{\theta})\right] + \nu\frac{1}{T_n}\sum_{t=1}^{T_n}\ln\left[1-h(\mathbf{y}_t;\boldsymbol{\theta})\right]. \tag{9}$$

$h(.;\boldsymbol{\theta})$ -->0, $G(.;\boldsymbol{\theta})\to-\infty$; -->1, $G(.;\boldsymbol{\theta})\to\infty$. $J_T\leq 0$, =0, for all t, $h(\mathbf{x}_t;\boldsymbol{\theta})$ / $h(\mathbf{y}_t;\boldsymbol{\theta})$ -->1/0, respectively. Intuitively, this means that logistic regression has learned to discriminate between the two sets as well as possible.

2.3 Properties of the Estimator

We characterize here the behavior of the estimator $\hat{\boldsymbol{\theta}}_T$ for large sample sizes T_d and fixed ratio ν. Since ν is kept fixed, $T_n = \nu T_d$ will also increase as T_d increases. The weak law of large numbers shows that as T_d increases $J_T(\boldsymbol{\theta})$ converges in probability to J,

$$J(\boldsymbol{\theta}) := \mathrm{E}\left\{\ln\left[h(\mathbf{x};\boldsymbol{\theta})\right]\right\} + \nu\,\mathrm{E}\left\{\ln\left[1-h(\mathbf{y};\boldsymbol{\theta})\right]\right\}. \tag{10}$$

the objective J seen as a function of $f_m(.) = \ln p_m(.;\boldsymbol{\theta})$,

$$\tilde{J}(f_m) \;:=\; \mathrm{E}\left\{\ln\left[r_\nu\left(f_m(\mathbf{x})-\ln p_n(\mathbf{x})\right)\right]\right\} + \nu\,\mathrm{E}\left\{\ln\left[1-r_\nu\left(f_m(\mathbf{y})-\ln p_n(\mathbf{y})\right)\right]\right\} \tag{11}$$

We start the characterization of the estimator $\hat{\boldsymbol{\theta}}_T$ by describing the optimization landscape for f_m. The following theorem shows that the data-pdf p_d can be found by maximization of $\tilde{J}$, that is by learning a nonparametric classifier under the ideal situation of an infinite amount of data.

Theorem 1 (Nonparametric estimation) *$\tilde{J}$ attains a maximum at $f_m = \ln p_d$. There are no other extrema if $p_n >> p_d$.*

A fundamental point in the theorem is that the maximization is performed without any normalization constraint for f_m. This is in stark contrast to MLE, where $\exp(f_m)$ must integrate to one. With our objective function , no such constraints are necessary . The maximizing pdf is found to have unit integral automatically.

The positivity condition for p_n in the theorem tells us that the data-pdf p_d cannot be inferred at regions in the data space where there are no contrastive noise samples. For example, the estimation of a pdf p_d which is nonzero only on the positive real line by means of a noise distribution p_n that has its support on the negative real line is impossible. The positivity condition can be easily fulfilled by taking, for example, a Gaussian as cnd.

In practice, the amount of data is limited and a finite number of parameters $\boldsymbol{\theta} \in \mathbb{R}^m$ specify $p_m(.;\boldsymbol{\theta})$. This has two consequences for any estimation method that is based on optimization: First, it restricts the space where the data-pdf p_d is searched for. Second, it may introduce local maxima into the optimization landscape. For the characterization of the estimator in this situation, it is normally assumed that p_d follows the model, so that there is a $\boldsymbol{\theta}^\star$ with $p_d(.) = p_m(.;\boldsymbol{\theta}^\star)$.

Our second theorem shows that $\hat{\boldsymbol{\theta}}_T$, the value of $\boldsymbol{\theta}$ which (globally) maximizes J_T, converges to $\boldsymbol{\theta}^\star$. The correct estimate of p_d is thus obtained as the sample size T_d increases. For unnormalized models, the conclusion of the theorem is that maximization of J_T leads to the correct estimates for both the parameter $\boldsymbol{\alpha}$ in the unnormalized pdf $p^0_m(.;\boldsymbol{\alpha})$ and the normalizing parameter c.

Theorem 2 (Consistency) *If conditions (a) to (c) are fulfilled then $\hat{\boldsymbol{\theta}}_T$ converges in probability to $\boldsymbol{\theta}^\star$*

(a) $p_n >> p_d$

(b) $\sup_\theta |J_T(\boldsymbol{\theta}) - J(\boldsymbol{\theta})| \xrightarrow{P} 0$

(c) The matrix $\boldsymbol{\mathcal{I}}_\nu = \int \mathbf{g}(\mathbf{u})\mathbf{g}(\mathbf{u})^T P_\nu(\mathbf{u}) p_d(\mathbf{u}) \mathrm{d}\mathbf{u}$ *has full rank, where*

$$\mathbf{g}(\mathbf{u}) = \nabla_{\boldsymbol{\theta}} \ln p_m(\mathbf{u};\boldsymbol{\theta})|_{\boldsymbol{\theta}^\star}, \qquad P_\nu(\mathbf{u}) = \frac{\nu p_n(\mathbf{u})}{p_d(\mathbf{u}) + \nu p_n(\mathbf{u})}.$$

The proof is given in Appendix A.3. Condition (a) is inherited from Theorem 1. Conditions (b) and (c) have their counterparts in MLE (see for example Wasserman, 2004, Theorem 9.13): We need in (b) uniform convergence in probability of J_T to J; in MLE, uniform convergence of the log-likelihood to the Kullback-Leibler divergence is required likewise. Condition (c) assures that for large sample sizes, the objective function J_T becomes peaked enough around the true value $\boldsymbol{\theta}^\star$. This imposes a constraint on the model $p_m(.;\boldsymbol{\theta})$ via the vector $\mathbf{g}$. A similar constraint is required in MLE.

The next theorem describes the distribution of the estimation error $(\hat{\boldsymbol{\theta}}_T - \boldsymbol{\theta}^\star)$ for large sample sizes. The proof is given in Appendix A.4.

Theorem 3 (Asymptotic normality) $\sqrt{T_d}(\hat{\boldsymbol{\theta}}_T - \boldsymbol{\theta}^\star)$ *is asymptotically normal with mean zero and covariance matrix* $\boldsymbol{\Sigma}$*,*

$$\boldsymbol{\Sigma} = \boldsymbol{\mathcal{I}}_\nu^{-1} - \left(1 + \frac{1}{\nu}\right) \boldsymbol{\mathcal{I}}_\nu^{-1} \mathrm{E}(P_\nu \mathbf{g}) \mathrm{E}(P_\nu \mathbf{g})^T \boldsymbol{\mathcal{I}}_\nu^{-1},$$

where $\mathrm{E}(P_\nu \mathbf{g}) = \int P_\nu(\mathbf{u}) \mathbf{g}(\mathbf{u}) p_d(\mathbf{u}) \mathrm{d}\mathbf{u}$.

Corollary 4 *For large sample sizes T_d, the mse $\mathrm{E}\left(||\hat{\boldsymbol{\theta}}_T - \boldsymbol{\theta}^\star||^2\right)$ equals $\mathrm{tr}(\boldsymbol{\Sigma})/T_d$.*

■

2.4 Choosing the Noise

Theorem 3 shows that the noise distribution p_n and the ratio $\nu = T_n/T_d$ have an influence on the accuracy of the estimate $\hat{\boldsymbol{\theta}}_T$. A natural question to ask is what, from a statistical standpoint, the best choice of p_n and ν is. Our result on consistency (Theorem 2) also includes a technical constraint for p_n but this one is so mild that many distributions will satisfy it.

Theorem 2 shows that, for a given samples size T_d, P_ν -->1 as T_n is made larger and larger. This implies that for large ν, the covariance matrix $\boldsymbol{\Sigma}$ does not depend on the choice of p_n.

Corollary 5 *For $\nu \to \infty$, $\boldsymbol{\Sigma}$ is independent of the choice of p_n and*

$$\boldsymbol{\Sigma} = \boldsymbol{\mathcal{I}}^{-1} - \boldsymbol{\mathcal{I}}^{-1}\,\mathrm{E}(\mathbf{g})\,\mathrm{E}(\mathbf{g})^T\boldsymbol{\mathcal{I}}^{-1},$$

where $\mathrm{E}(\mathbf{g}) = \int \mathbf{g}(\mathbf{u})p_d(\mathbf{u})\mathrm{d}\mathbf{u}$ *and* $\boldsymbol{\mathcal{I}} = \int \mathbf{g}(\mathbf{u})\mathbf{g}(\mathbf{u})^T p_d(\mathbf{u})\mathrm{d}\mathbf{u}$.

The asymptotic distribution of the estimation error becomes thus independent from p_n. Hence, as the size of the contrastive -noise sample Y increases , the choice of the cnd becomes less and less important. Moreover, for normalized models, we have the result that the estimation error has the same distribution as the estimation error in MLE.

Corollary 6 *For normalized models, NCE is, $\nu \to \infty$, asymptotically Fisher-efficient for all p_n.*

Proof For normalized models, no normalizing parameter c is needed. In Corollary 5, the function $\mathbf{g}$ is then the score function as in MLE, and the matrix $\boldsymbol{\mathcal{I}}$ is the FIM. Since $\mathrm{E}(\mathbf{g})=0$, $\boldsymbol{\Sigma}$ is the inverse of the FIM.

■

The corollaries above give one answer to the question on how to choose the noise distribution p_n and the ratio ν: If ν is made large enough, the actual choice of p_n is not of great importance. Note that this answer considers only estimation accuracy and ignores the computational load associated with the processing of noise. In Section 4, we will analyze the trade-off between estimation accuracy and computation time.

given ν, one could try to find the noise distribution which minimizes MSE $\mathrm{E}||\hat{\boldsymbol{\theta}}_T - \boldsymbol{\theta}^\star||^2$. However, this minimization turns out to be quite difficult. Intuitively, one could think that a good candidate for the noise distribution p_n is a distribution which is close to the data distribution p_d. If p_n is too different from p_d, the classification problem might be too easy and would not require the system to learn much about the structure of the data. This intuition is partly justified by the following theoretical result:

Corollary 7 *If* $p_n = p_d$ *then* $\boldsymbol{\Sigma} = \left(1 + \frac{1}{\nu}\right)\left(\boldsymbol{\mathcal{I}}^{-1} - \boldsymbol{\mathcal{I}}^{-1}\,\mathrm{E}(\mathbf{g})\,\mathrm{E}(\mathbf{g})^T\boldsymbol{\mathcal{I}}^{-1}\right)$.

■

For normalized models, for $\nu = 1$, $\boldsymbol{\Sigma}$ is two times the inverse of the FIM, and that for $\nu = 10$, the ratio is already down to 1.1. For a noise distribution that is close to the data distribution , we have thus even for moderate values of νsome guarantee that the MSE is reasonably close to the theoretical optimum.

To get estimates with a small estimation error, the foregoing discussion suggests the following

1. Choose noise for which an analytical expression for $\ln p_n$ is available.
2. Choose noise that can be sampled easily.
3. Choose noise that is in some aspect, for example wrt its covariance structure, similar to the data.
4. Make the noise sample size as large as computationally possible.

Suitable noise distributions : Gaussian , Gaussian mixture , or ICA distributions . Uniform distributions are also suitable as long as their support includes the support of the data distribution so that condition (a) in Theorem 2 holds.

3. Simulations to Validate and Illustrate the Theory

In this section,[3] we validate and illustrate the theoretical properties of NCE. In Section 3.1, we focus on the consistency of the estimator . In Section 3.2, we validate our theoretical results on the distribution of the estimation error, and investigate its dependency on the ratio νbetween noise and data sample size . In Section 3.3, we study how the performance of the estimator scales with the dimension of the data.

3.1 Consistency

For the illustration of consistency, we estimate the parameters of a zero mean mv Gaussian

$$\ln p_d(\mathbf{x}) = -\frac{1}{2}\mathbf{x}^T\boldsymbol{\Lambda}^\star\mathbf{x} + c^\star, \qquad c^\star = \left(+\frac{1}{2}\ln|\det\boldsymbol{\Lambda}^\star| - \frac{n}{2}\ln(2\pi)\right), \tag{12}$$

where $c^\star$normalizes p_d to integrate to one. The precision matrix $\boldsymbol{\Lambda}^\star$. The dim of $\mathbf{x}$ is $n = 5$.
As we are mostly interested estimation of unnormalized models

$$\ln p_m^0(\mathbf{x};\boldsymbol{\alpha}) = -\frac{1}{2}\mathbf{x}^T\boldsymbol{\Lambda}\mathbf{x}$$

without knowing how to normalize it in closed form. This unnormalized model is a pairwise Markov network with quadratic node and edge potentials (Koller and Friedman , 2009 , Chapter 7). The parameter vector $\boldsymbol{\alpha} \in \mathbb{R}^{15}$ contains the coefficients of the lower-triangular part of $\boldsymbol{\Lambda}$ as the matrix is symmetric. For NCE, we add an additional normalizing parameter c to the model.

3. Matlab code for this and the other sections can be downloaded from the homepage of the first author.

The model that we estimate is thus

$$\ln p_m(\mathbf{x};\boldsymbol{\theta}) = \ln p_m^0(\mathbf{x};\boldsymbol{\alpha}) + c.$$

The model has 16 parameters given by $\boldsymbol{\theta} = (\boldsymbol{\alpha}, c)$. They are estimated by maximization of the objective function $J_T(\boldsymbol{\theta})$ in Eq (8). We used N(0,1) for p_n. The optimization was performed with the nonlinear conjugate gradient algorithm of Rasmussen (2006).

3.1.1 RESULTS

The presented results are an average over 500 estimation problems where the true precision matrix $\mathbf{\Lambda}^\star$ was drawn at random with the condition number being controlled to be smaller than ten. The sampling of $\mathbf{\Lambda}^\star$ was performed by randomly sampling its eigenvalues and eigenvectors: We drew the eigenvalues from an uniform distribution on the interval $[0.1\ 0.9]$. The orthonormal matrix $\mathbf{E}$ with the eigenvectors was created by orthogonally projecting a matrix $\mathbf{M}$ with elements drawn independently from a standard Gaussian onto the set of orthonormal matrices: $\mathbf{E} = (\mathbf{M}\mathbf{M}^T)^{-1/2}\mathbf{M}$.

Figure 1(a) and (b) show the MSE for $\boldsymbol{\alpha}$, which contains the elements of the precision matrix $\mathbf{\Lambda}$, and the normalizing parameter c, respectively . The MSE as a function of the data sample size T_d decays linearly on a log-log scale. This illustrates our result of consistency of the estimator, stated as Theorem 2, as convergence in quadratic mean implies convergence in probability . The plots also show that taking more noise samples T_n than data samples T_d leads to more and more accurate estimates . The performance for nce with $\nu = T_n / T_d$ equal to one is shown in blue with circles as markers. For that value of ν, there is a clear difference compared to MLE (black triangles in Figure 1(a)). However , the accuracy of the estimate improves strongly for $\nu = 5$ (green squares) or $\nu = 10$ (red diamonds) where the performance is rather close to the performance of MLE.

Another way to visualize the results is by showing the K-L divergences between the 500 true and estimated distributions . Figure 2 shows boxplots of the divergences for $\nu = 1$ (blue) and $\nu = 10$ (red). The results for MLE are shown in black . In line with the visualization in Figure 1, the estimated distribution becomes closer to the true distribution as the sample size increases . Moreover , the divergences become clearly smaller as ν is increased from 1to 10.

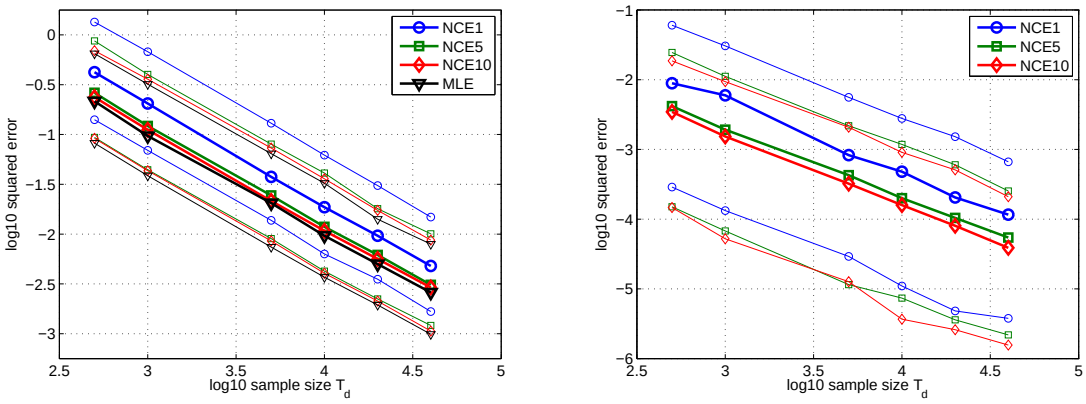
For unnormalized models, there is a subtlety in the computation of the divergence. With a validation set of size T_v, a sample version D_{KL} of the K-L div :

$$D_{\mathrm{KL}} = \frac{1}{T_v}\sum_{t=1}^{T_v} \ln p_d(\mathbf{x}_t) - \left(\frac{1}{T_v}\sum_{t=1}^{T_v} \ln p_m^0(\mathbf{x}_t;\hat{\boldsymbol{\alpha}}) + \ln 1/Z(\hat{\boldsymbol{\alpha}}) \right).$$

The first term is the rescaled log-likelihood (average, sign-inverted log-loss) for the true distribution. The term in parentheses is the rescaled log-likelihood L of the estimated model. In the estimation of unnormalized models, we do not assume to know the mapping $\boldsymbol{\alpha} \to Z(\boldsymbol{\alpha})$ so that L cannot be computed. With NCE, we can obtain an estimate ,

$$\hat{L} = \frac{1}{T_v}\sum_{t=1}^{T_v} \ln p_m^0(\mathbf{x}_t;\hat{\boldsymbol{\alpha}}) + \hat{c}, \tag{13}$$

Figure 2(a) shows that the estimated D_{KL} is sometimes negative which means that $L\hat{}$ is sometimes larger than the rescaled log-likelihood of the true distribution.



(a) Precision matrix

(b) Normalizing parameter

Figure 1: Validation of the theory of nce: Estimation errors for a 5D Gaussian distribution. Figures (a) and (b) show the MSE for the precision matrix $\mathbf{\Lambda}$ and the normalizing parameter c, respectively . The performance of NCE approaches the performance of MLE (black triangles) as the ratio $\nu = Tn / Td$ increases : the case of $\nu = 1$ is shown with blue circles , $\nu = 5$ with green squares , and $\nu = 10$ with red diamonds . The thicker curves are the median of the performance for 500 random precision matrices with cond < 10. The finer curves show the 0.9 and 0.1 quantiles of the logarithm of the squared estimation error.

This happens because $c\hat{}$ can be an over or underestimate of $\ln 1/Z(\boldsymbol{\alpha}\hat{})$. This result follows from Figure 2(b) where we have computed D_{KL} with the analytical expression for $\ln 1/Z(\boldsymbol{\alpha}\hat{})$, which is available for the Gaussian model considered here, see Eq (12).

3.2 Distribution of the Estimation Error

We validate and illustrate further properties of our estimator using the ICA model (see for example Hyvärinen et al., 2001b)

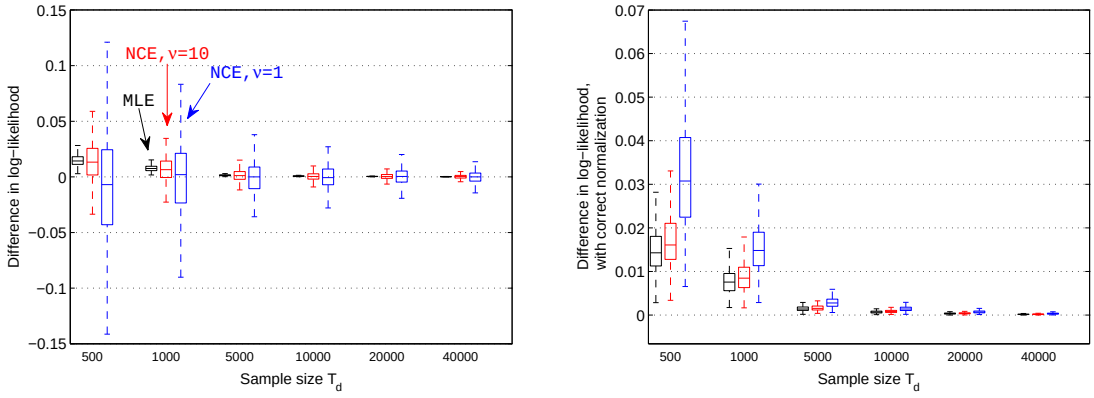
$$\mathbf{x} = \mathbf{A}\mathbf{s}. \tag{14}$$

In this subsection, $n = 4$, that is $\mathbf{x} \in \mathbb{R}^4$, and $\mathbf{A} = (\mathbf{a}_1, \ldots, \mathbf{a}_4)$ is a 4×4 mixing matrix. The sources in the vector $\mathbf{s} \in \mathbb{R}^4$ are identically distributed and independent from each other so that the data log-pdf $\ln p_d$ is

$$\ln p_d(\mathbf{x}) = \sum_{i=1}^{n} f(\mathbf{b}_i^\star \mathbf{x}) + c^\star. \tag{15}$$

The i-th row of the matrix $\mathbf{B}^\star = \mathbf{A}^{-1}$ is denoted by $\mathbf{b}_i^\star$. We consider here Laplacian sources of unit variance and zero mean. The nonlinearity f and the constant $c^\star$, which normalizes p_d to integrate to one, are in this case given by

$$f(u) = -\sqrt{2}|u|, \qquad c^\star = \ln|\det \mathbf{B}^\star| - \frac{n}{2}\ln 2. \tag{16}$$



(a) Estimated normalization

(b) Analytical normalization

Figure 2: Validation of the theory of nce: Distributions of the KL divergences between the true and estimated 5D Gaussians. For each sample size, from left to right, the results for MLE are shown in black, the results for NCE with $\nu = 10$ in red, and the results for $\nu = 1$ in blue. The size T_v of the validation set was 100000. For MLE, the results shown in Figures (a) and (b) are the same. For NCE, the divergences in Figure (a) were computed using the estimate $c\hat{}$ of $\ln 1/Z(\boldsymbol{\alpha}\hat{})$. In Figure (b), the analytical expression for $\ln 1/Z(\boldsymbol{\alpha}\hat{})$ was used.

As in Section 3.1, we apply NCE to the hypothetical situation where we want to estimate the unnormalized model

$$\ln p_m^0(\mathbf{x};\boldsymbol{\alpha}) = \sum_{i=1}^{n} f(\mathbf{b}_i\mathbf{x}) \tag{17}$$

without knowing how to normalize it in closed form. The parameter vector $\boldsymbol{\alpha} \in \mathbb{R}^{16}$ contains the elements of the row vectors $\mathbf{b}_i$. For NCE, we add an additional normalizing parameter c and estimate the model

$$\ln p_m(\mathbf{x};\boldsymbol{\theta}) = \ln p_m^0(\mathbf{x};\boldsymbol{\alpha}) + c,$$

with $\boldsymbol{\theta} = (\boldsymbol{\alpha}, c)$. As for the Gaussian case, we estimate $\boldsymbol{\theta}$ by maximizing $J_T(\boldsymbol{\theta})$ in Eq (8) with the nonlinear conjugate gradient algorithm of Rasmussen (2006). For the noise distribution p_n, we used a Gaussian distribution with covariance matrix given by the sample covariance of the data.

3.2.1 Results

In Figures 3 and 4, we illustrate Theorem 2 on consistency and Theorem 3 on the asymptotic distribution of the estimator, as well as its corollaries. The results are averages over 500 random estimation problems. The mixing matrices $\mathbf{A}$ were drawn at random by drawing their elements independently from a standard Gaussian and only accepting matrices which had a condition number smaller than ten.

Figure 3(a) and (b) show the MSE for $\boldsymbol{\alpha}$, corresponding to the mixing matrix, and the normalizing parameter c, respectively . As illustrated for the Gaussian case in Figure 1, this figure visualizes the consistency of NCE. Furthermore , we see again that making $\nu = T_n/T_d$ larger leads to a reduction of the error. The reduction gets, however , smaller as ν increases . On average , changing ν from 1(red curve with asterisks as markers) to 10(light blue squares) reduces the MSE for the mixing matrix by 53%; relative to $\nu = 10$, $\nu = 100$ (magenta diamonds) leads to a reduction of 18%. For c, the relative decrease in the MSE is 60% and 17%, respectively.

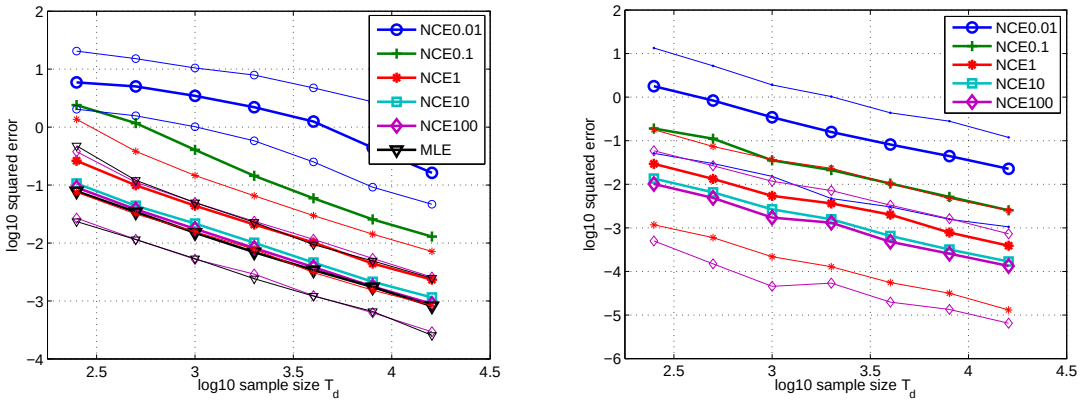
In Figure 4(a), we test the theoretical prediction of Corollary 4 that, for large samples sizes T_d, the MSE decays like $\mathrm{tr}\,\boldsymbol{\Sigma}/T_d$. The covariance matrix $\boldsymbol{\Sigma}$ can be numerically evaluated according to its definition in Theorem 3.[4] This allows for a prediction of the MSE that can be compared to the MSE obtained in the simulations. The figure shows that the MSE from the simulations (labelled "sim" in the figure) matches the prediction ("pred") for large T_d. Furthermore, we see again that for large ν, the performance of NCE is close to the performance of MLE. In other words, the trace of $\boldsymbol{\Sigma}$ is close to the trace of the FIM. Note that for clarity, we only show the curves for $\nu \in \{0.1, 1, 100\}$. The curve for $\nu = 10$ was, as in Figure 3(a) and (b), very close to the curve for $\nu = 100$.

In Figure 4(b), we investigate how $\mathrm{tr}\,\boldsymbol{\Sigma}$ (the asymptotic variance) depends on ν. Note that the covariance matrix $\boldsymbol{\Sigma}$ includes terms related to the parameter c. The FIM includes , in contrast to $\boldsymbol{\Sigma}$, only terms related to the mixing matrix. For better comparison with MLE, we show thus in the figure $\mathrm{tr}\boldsymbol{\Sigma}$ both with the contribution of the normalizing parameter c (blue squares) and without (red circles). For the latter case , the reduced trace of $\boldsymbol{\Sigma}$, which we will denote by $\mathrm{tr}\,\boldsymbol{\Sigma}_B$, approaches tr FIM . Corollary 6 stated that NCE is asymptotically Fisher-efficient for large ν if the normalizing constant is not estimated . Here , we see that this result also approximately holds for our unnormalized model where the normalizing constant needs to be estimated.

Figure 4(c) gives further details to which extent the estimation becomes more difficult if the model is unnormalized. We computed numerically the asymptotic variance $\mathrm{tr}\,\tilde{\boldsymbol{\Sigma}}$ if the model is correctly normalized, and compared it to the asymptotic variance $\mathrm{tr}\,\boldsymbol{\Sigma}_B$ for the unnormalized model. The figure shows the distribution of the ratio $\mathrm{tr}\,\boldsymbol{\Sigma}_B / \mathrm{tr}\,\tilde{\boldsymbol{\Sigma}}$ for different values of ν. Interestingly, the ratio is almost equal to one for all tested values of ν. Hence, additional estimation of the normalizing constant does not really seem to have had a negative effect on the accuracy of the estimates for the mixing matrix.

In Corollary 7, we have considered the hypothetical case where the noise distribution p_n is the same as the data distribution p_d. In Figure 4(d), we plot for that situation the asymptotic variance as a function of ν (green curve). For reference, we plot again the curve for Gaussian contrastive noise (red circles, same as in Figure 4(b)). In both cases, we only show the asymptotic variance $\mathrm{tr}\,\boldsymbol{\Sigma}_B$ for the parameters that correspond to the mixing matrix. The asymptotic variance for $p_n = p_d$ is, for a given value of ν, always smaller than the asymptotic variance for the case where the noise is Gaussian. However, by choosing ν large enough for the case of Gaussian noise, it is possible to get estimates which are as accurate as those obtained in the hypothetical situation where $p_n = p_d$. Moreover, for larger ν, the performance is the same for both cases: both converge to the performance of MLE.

4. See Appendix B.1 for the calculations in the special case of orthogonal mixing matrices.



(a) Mixing matrix

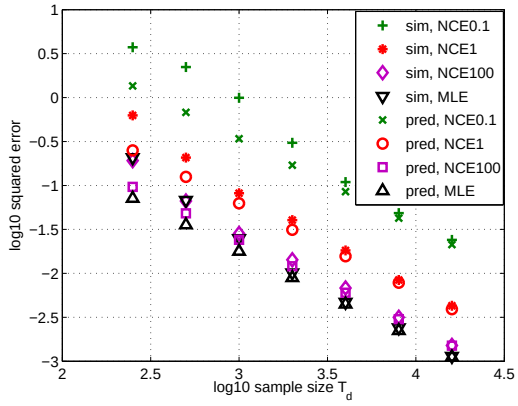
(b) Normalizing parameter

Figure 3: Validation of the theory of NCE: Estimation errors for an ICA model with four sources. Figures (a) and (b) show the mse for the mixing matrix $\mathbf{B}$ and the normalizing parameter c, respectively. The performance of NCE approaches the performance of MLE(black triangles) as the ratio $\nu = T_n/T_d$ increases: the case of $\nu = 0.01$ is shown with blue circles, $\nu = 0.1$ with green crosses, $\nu = 1$ with red asterisks, $\nu = 10$ with light blue squares, and $\nu = 100$ with magenta diamonds. The thicker curves are the median of the performance for 500 random precision matrices with condition number smaller than ten. The finer curves show the 0.9 and 0.1 quantiles of the logarithm of the squared estimation error. To increase readability of the plots, the quantiles for $\nu = 0.1$ and $\nu = 10$ are not shown.

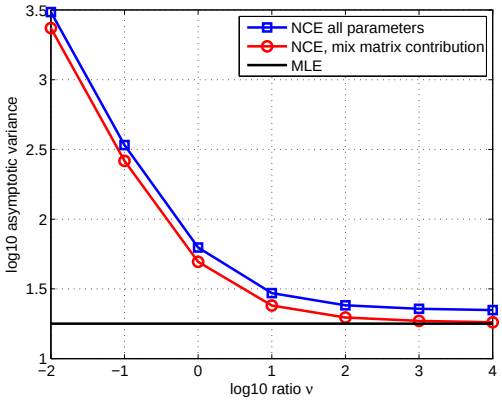
3.3 Scaling Properties

We use the ICA model from the previous subsection to study the behavior of NCE as the dimension n of the data increases. As before, we estimate the parameters by maximizing $J_T(\boldsymbol{\theta})$ in Eq (8) with the nonlinear conjugate gradient algorithm of Rasmussen(2006). Again, we use a Gaussian with the same covariance structure as the data as noise distribution p_n.

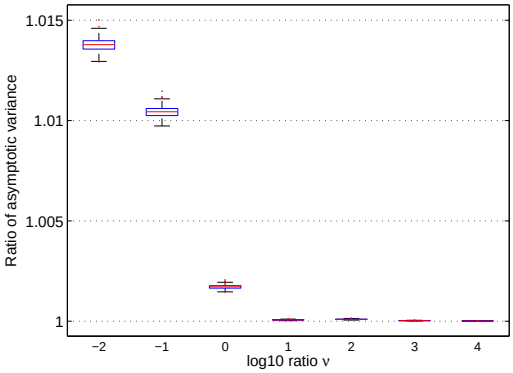
The randomly chosen $n \times n$ mixing matrices $\mathbf{A}$ are restricted to be orthogonal. Orthogonality is only used to set up the estimation problem; in the estimation, the orthogonality property is not used. A reason for this restriction is that drawing mixing matrices at random as in the previous subsection leads more and more often to badly conditioned matrices as the dimension increases. Another reason is that the estimation error for orthogonal mixing matrices depends only on the dimension n and not on the particular mixing matrix chosen, see Appendix B.1 for a proof. Hence, this restriction allows us to isolate the effect of dim n on the estimation accuracy.



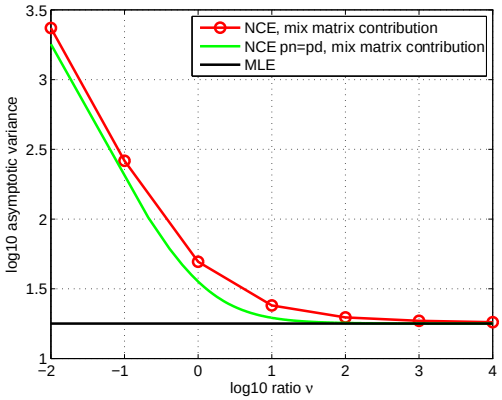
(a) Prediction of the MSE



(b) Asymptotic behavior



(c) Normalized vs unnormalized model

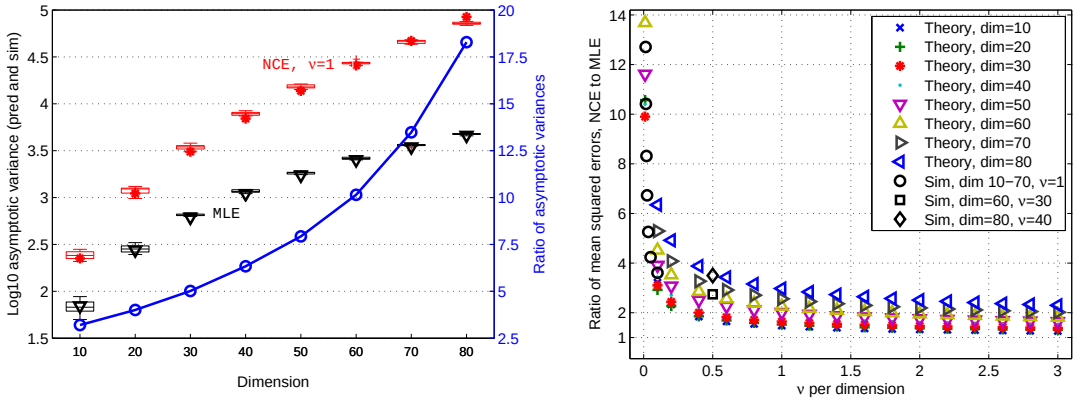


(d) Effect of the noise distribution

Figure 4: Validation of the theory of NCE: Estimation error for large sample sizes. Figure (a) shows that Corollary 4 correctly predicts the MSE for large samples sizes T_d. Figure (b) shows the asymptotic variance $\mathrm{tr}\,\boldsymbol{\Sigma}$ as a function of ν. Figure (c) shows a boxplot of the ratio between the asymptotic variance when the model is unnormalized and the asymptotic variance when the model is normalized. Figure (d) compares NCE with Gaussian noise to the hypothetical case where p_n equals the data distribution p_d. As in Figure 3, the curves in all figures but in Figure (c) are the median of the results for 500 random mixing matrices. The boxplot in Figure (c) shows the distribution for all the 500 matrices.

3.3.1 RESULTS

Figure 5(a) shows the asymptotic variance $\mathrm{tr}\,\boldsymbol{\Sigma}_B$ related to the mixing matrix as a function of the dim n. NCE with $\nu = T_n / T_d = 1$ is shown in red with asterisks as markers , MLE in black using triangles as markers. The markers show the theoretical prediction based on Corollary 4; the boxplots the simulation results for ten



(a) NCE and MLE in higher dimensions

(b) Relative estimation error and amount of noise per dimension

Figure 5: Investigating how NCE scales with the dimension of the data. Figure (a) shows the logarithm of the asymptotic variance for NCE ($\nu = Tn/Td = 1$, in red) and MLE (in black). The boxplots show simulation results; the asterisks and triangles theoretical predictions for NCE and MLE, respectively. The same figure shows the ratio of the two asymptotic variances (blue circles, right scale). Figure (b) plots the ratio of the mean squared errors of the two estimators as a function of ν per dimension n. The value of ν needs to be increased as the dimensions increases; a linear increase leads to acceptable results.

random mixing matrices with $T_d = 80000$. The simulation results match the predictions well, which validates the theory of NCE in large dimensions.

Since the number of parameters increases with larger n, it is natural that $\mathrm{tr}\,\mathbf{\Sigma}_B$ increases with n. However, for NCE, the increase is larger than for MLE. This is more clearly visible by considering the blue curve in Figure 5(a) (circles as markers, scale on the right axis). The curve shows the ratio between the asymptotic variance for noise-contrastive estimation and for MLE. By definition of the asymptotic variance, this ratio is equal to the ratio of the two estimation errors obtained with the two different methods. The ratio does not depend on the number of parameters and the sample size Td. It is hence a suitable performance indicator to investigate how NCE scales with the dimension n of the data. The plot shows that for fixed ν, the performance deteriorates as the dimension increases. In order to counteract this decline in performance, the parameter ν needs to be increased as the dimension increases.

Figure 5(b) shows the ratio of the squared errors as a function of ν/n where we varied n from ten to eighty dimensions as in Figure 5(a). Importantly, both theoretical results, where we numerically calculated the asymptotic variances, and simulation results show that for a reasonable performance in comparison to MLE, ν does not need to be increased exponentially as the dimension n increases; a linear increase with, for instance, $\nu \in [n/2\ n]$ suffices to lead to estimation errors of about 2-4 times of those that are obtained by estimating normalized models with MLE.

4. Investigating the Trade-Off between Statistical and Computational Performance

We have seen that for large ratios ν of noise sample size T_n to data sample size T_d, the estimation error for NCE behaves like the error in MLE. For large ν, however, the computational load becomes also heavier because more noise samples need to be processed . There is thus a trade -off between statistical and computational performance . Such a trade-off exists also in other estimation methods for unnormalized models. In this section, we investigate the trade-off in NCE, and compare it to the trade-off in MCMLE (Geyer, 1994), contrastive divergence (Hinton, 2002) and persistent contrastive divergence [5] (Younes, 1989; Tieleman, 2008), as well as score matching (Hyvarinen¨ , 2005).

In Section 4.1, we comment on the data which we use in the comparison. In Section 4.2, we review the different estimation methods with focus on the trade-off between statistical and computational performance. In Section 4.3, we point out the limitations of our comparison before presenting the simulation results in Section 4.4.

4.1 Data Used in the Comparison

For the comparison, we use artificial data which follows the ICA model in Equation (14) with the data log-pdf $\ln p_d$ being given by Equation (15). We set the dimension n to ten and use $T_d = 8000$ observations to estimate the parameters. In a first comparison, we assume Laplacian sources in the ICA model. The log-pdf $\ln p_d$ is then specified with Equation (16). Note that this log-pdf has a sharp peak around zero where it is not continuously differentiable. In a second comparison, we use sources that follow the smoother logistic density. The nonlinearity f and the log normalizing constant c^* in Equation (15) are in that case

$$f(u) = -2\ln\cosh\left(\frac{\pi}{2\sqrt{3}}u\right), \qquad c^* = \ln|\det\mathbf{B}^\star| + n\ln\left(\frac{\pi}{4\sqrt{3}}\right),$$

respectively. We are thus making the comparison for a relatively nonsmooth and smooth density. Both comparisons are based on 100 randomly chosen mixing matrices with condition number smaller than 10.

4.2 Estimation Methods Used in the Comparison

We introduce here briefly the different methods and comment on our implementation and choices of parameters.

4.2.1 NCE

To estimate the parameters, we maximize J_T in Eq(8). We use here a Gaussian noise density p_n with a covariance matrix equal to the sample covariance of the data. As before, J_T is maximized using the nonlinear conjugate gradient method of Rasmussen (2006). To map out the trade -off between statistical and computational performance, we measured the estimation error and the time needed to optimize J_T for $\nu \in \{1,2,5,10,20,50,100,200,400,1000\}$.

5. Persistent contrastive divergence= stochastic MLE.

4.2.2 MC MLE

For normalized models, an estimate for the parameters $\boldsymbol{\alpha}$ can be obtained by choosing them such that the probability of the observed data is maximized. This is done by maximization of

$$J_{\text{MLE}}(\boldsymbol{\alpha}) = \frac{1}{T_d}\sum_{t=1}^{T_d} \ln p_m^0(\mathbf{x}_t;\boldsymbol{\alpha}) - \ln Z(\boldsymbol{\alpha}). \tag{18}$$

If no analytical expression for the partition function $Z(\boldsymbol{\alpha})$ is available, IS can be used to numerically approximate $Z(\boldsymbol{\alpha})$ via its definition in Eq (2),

$$Z(\boldsymbol{\alpha}) \approx \frac{1}{T_n}\sum_{t=1}^{T_n} \frac{p_m^0(\mathbf{n}_t;\boldsymbol{\alpha})}{p_{\text{IS}}(\mathbf{n}_t)}.$$

The $\mathbf{n}_t$ are independent observations of "noise" ~ p_{IS}. Note that more sophisticated ways exist to numerically calculate the value of Z at a given $\boldsymbol{\alpha}$ (see Robert and Casella, 2004, in particular Chapter 3 and 4). The simple approach above leads to $J_{\text{IS}}(\boldsymbol{\alpha})$ known as MCML(Geyer, 1994),

$$J_{\text{IS}}(\boldsymbol{\alpha}) = \frac{1}{T_d}\sum_{t=1}^{T_d} \ln p_m^0(\mathbf{x}_t;\boldsymbol{\alpha}) - \ln\left(\frac{1}{T_n}\sum_{t=1}^{T_n} \frac{p_m^0(\mathbf{n}_t;\boldsymbol{\alpha})}{p_{\text{IS}}(\mathbf{n}_t)}\right).$$

We maximized $J_{\text{IS}}(\boldsymbol{\alpha})$ with the nonlinear conjugate gradient algorithm of Rasmussen (2006).

Like in NCE, there is a trade-off between statistical performance and running time: The larger T_n gets the better the approximation of the log-likelihood. Hence, the estimates become more accurate but the optimization of J_{IS} takes also more time. To map out the trade-off curve, we used the same values of $T_n = \nu T_d$ as in NCE, and also the same noise distribution, $p_{\text{IS}} = p_n$.

4.2.3 Contrastive Divergence

If J_{MLE} is maximized with a steepest ascent algorithm, the update rule for $\boldsymbol{\alpha}$ is

$$\boldsymbol{\alpha}_{k+1} = \boldsymbol{\alpha}_k + \mu_k \nabla_{\boldsymbol{\alpha}} J_{\text{MLE}}(\boldsymbol{\alpha})|_{\boldsymbol{\alpha}_k}, \tag{19}$$

where μ_k is the step-size. For the calculation of $\nabla_{\boldsymbol{\alpha}} J_{\text{MLE}}$, the gradient of the log partition function $\ln Z(\boldsymbol{\alpha})$ is needed, see Eq (18). Above, IS was used to evaluate $\ln Z(\boldsymbol{\alpha})$ and its gradient $\nabla_{\boldsymbol{\alpha}} \ln Z(\boldsymbol{\alpha})$. The gradient of the log partition function:

$$\nabla_{\boldsymbol{\alpha}} \ln Z(\boldsymbol{\alpha}) = \frac{\nabla_{\boldsymbol{\alpha}} Z(\boldsymbol{\alpha})}{Z(\boldsymbol{\alpha})} = \int \frac{p_m^0(\mathbf{n};\boldsymbol{\alpha})}{Z(\boldsymbol{\alpha})} \nabla_{\boldsymbol{\alpha}} \ln p_m^0(\mathbf{n};\boldsymbol{\alpha}) \mathrm{d}\mathbf{n}. \tag{20}$$

If we had data $\mathbf{n}_t$ at hand which follows the normalized model density $p_m^0(.;\boldsymbol{\alpha})/Z(\boldsymbol{\alpha})$, the last equation could be evaluated by taking the sample average. The parameter vector $\boldsymbol{\alpha}$ could then be learned based on Eq (19). In general, sampling from the model density is, however, only possible by means of McMC methods. In CD (Hinton, 2002), to compute $\boldsymbol{\alpha}_{k+1}$, Markov chains are started at the data points $\mathbf{x}_t$ and stopped after a few Monte Carlo steps before they actually reach the stationary distribution ${p_m}^0(.;\boldsymbol{\alpha}_k)/Z(\boldsymbol{\alpha}_k)$. The data points

$\mathbf{n}_t$ that are created in that way follow thus only approximately $p_m^0(.;\boldsymbol{\alpha}_k)/Z(\boldsymbol{\alpha}_k)$. For every update of $\boldsymbol{\alpha}$ the Markov chains are restarted from the $\mathbf{x}_t$. Note that this update rule for $\boldsymbol{\alpha}$ is not directly optimizing a known objective function.

In our implementation, we used HMC (see for example Neal, 2010) with a rejection ratio of 10% for the sampling (like in Teh et al., 2004; Ranzato and Hinton, 2010). There are then four tuning parameters for CD: The number of Monte Carlo steps, the number of "leapfrog" steps in HMC, the choice of the step sizes μ_k, as well as the number of data points $\mathbf{x}_t$ and noise points $\mathbf{n}_t$ used in each update step of $\boldsymbol{\alpha}$. The choice of the tuning parameters will affect the estimation error and the computation time. For our comparison here, we used contrastive divergence with one and three Monte Carlo steps (denoted by CD1 and CD3 in the figures below), together with either three or twenty leapfrog steps. Ranzato and Hinton (2010) used CD1 with twenty leapfrog steps (below denoted by CD1 20), while Teh et al. (2004) used CD1 30 to estimate unnormalized models from natural image data. For the μ_k, we considered constant step sizes, as well as linearly and exponentially decaying step sizes.[6] For each update step, we chose an equal number of data and noise points. We considered the case of using all data in each update step, and the case of using minibatches of only 100 randomly chosen data points.

We selected the step size μ_k and the number of data points used in each update by means of preliminary simulations on five data sets. We limited ourselves to CD with one Monte Carlo and three leapfrog steps (CD1 3). For both Laplacian and logistic sources, using mini-batches with an exponential decaying step size gave the best results. The results are reported below in Section 4.4. The use of minibatches led to faster estimation results without affecting their accuracy. Exponentially decaying step sizes are advocated by the theory of stochastic approximation; in some cases, however, linear decay was found to be more appropriate (Tieleman, 2008, Section 4.5). For Laplacian sources, the initial step size μ_0 was 0.005; for logistic sources, it was $\mu_0 = 0.01$. Note that in this selection of the tuning parameters, we used the true parameters to compute the estimation error. Clearly, this cannot be done in real applications since the true parameter values are not known. The choice of the tuning parameters must then solely be based on experience, as well as trial and error.

4.2.4 Persistent Contrastive Divergence

As CD, PCD (Younes, 1989; Tieleman, 2008) uses the update rule in Eq (19) together with an approximative evaluation of the integral in Eq(20) to learn the parameters $\boldsymbol{\alpha}$. The integral is also computed based on McMC sampling. Unlike CD, the Markov chains are not restarted at the data points $\mathbf{x}_t$. For the computation of $\boldsymbol{\alpha}_{k+1}$, the Markov chains are initialized with the samples $\mathbf{n}_t$ that were obtained in the previous iteration by running Markov chains $\rightarrow p_m{}^0(.;\boldsymbol{\alpha}_{k-1})/Z(\boldsymbol{\alpha}_{k-1})$. As in CD, the Markov chains are only run for a short time and stopped before having actually converged.

Since PCD differs from CD only by the initialization of the Markov chains, it has the same tuning parameters. As in CD, we used preliminary simulations to select suitable parameters: again, exponentially decaying step sizes μ_k together with minibatches of size 100 gave the best performance. The preliminary simulations yielded also the same initial step sizes μ_0 as in CD. It turned out, however,

6. Linear decay: $\mu_k = \mu_0(1 - k/maxIteration)$, exponential decay: $\mu_k = \mu_0 C/(C+k)$ with $C = 5000$.

that the number of leapfrog steps in PCD needs to be larger than in CD: using, for example, only three leapfrog steps as in CD resulted in a poor performance in terms of estimation accuracy. For the results reported below in Section 4.4, we used 20 and 40 leapfrog steps, together with one and three Monte Carlo steps.

4.2.5 SCORE MATCHING

In score matching (Hyvärinen, 2005), the parameter vector $\boldsymbol{\alpha}$ is estimated by minimization of the cost function J_{SM},

$$J_{\text{SM}}(\boldsymbol{\alpha}) = \frac{1}{T_d} \sum_{t=1}^{T_d} \sum_{i=1}^{n} \frac{1}{2} \Psi_i^2(\mathbf{x}_t; \boldsymbol{\alpha}) + \Psi_i'(\mathbf{x}_t; \boldsymbol{\alpha}).$$

The term $\Psi_i(\mathbf{x}; \boldsymbol{\alpha})$ is the derivative of the unnormalized model wrt $\mathbf{x}(i)$, the i-th element of $\mathbf{x}$,

$$\Psi_i(\mathbf{x}; \boldsymbol{\alpha}) = \frac{\partial \ln p_m^0(\mathbf{x}; \boldsymbol{\alpha})}{\partial \mathbf{x}(i)}.$$

The term $\Psi_i'(\mathbf{x}; \boldsymbol{\alpha})$ denotes the derivative of $\Psi_i(\mathbf{x}; \boldsymbol{\alpha})$ with respect to $\mathbf{x}(i)$. The presence of this derivative may make the objective function and its gradient algebraically rather complicated if a sophisticated model is estimated. For the ICA model with Laplacian sources, $\Psi_i(\mathbf{x}; \boldsymbol{\alpha})$ equals

$$\Psi_i(\mathbf{x}; \boldsymbol{\alpha}) = \sum_{j=1}^{n} -\sqrt{2}\text{sign}(\mathbf{b}_j \mathbf{x}) B_{ji} \tag{21}$$

which is not smooth enough to be used in score matching. Using the smooth approximation $\text{sign}(u) \approx \tanh(10u)$ is a way to obtain a smooth enough $\Psi_i(\mathbf{x}; \boldsymbol{\alpha})$ and $\Psi_i'(\mathbf{x}; \boldsymbol{\alpha})$. The optimization of J_{SM} is done by the nonlinear conjugate gradient algorithm of Rasmussen (2006). Note that, unlike the estimation methods considered above, score matching does not have a tuning parameter which controls the trade-off between statistical and computational performance. Moreover, score matching does not rely on sampling.

4.3 Limitations of the Comparison

For all considered methods but CD/PCD, the algorithm which is used to optimize the given objectives can be rather freely chosen . This choice will influence the trade -off between statistical and computational performance . Here , we use the optimization algorithm by Rasmussen (2006). Our results below show thus the trade -off of the different estimation methods in combination with this particular optimization algorithm. With this optimization algorithm, we used for each update all data. The algorithm is not suitable for stochastic optimization with minibatches (see for example Schraudolph and Graepel , 2002). Optimization based on mini - batches may well lead not only for (persistent) contrastive divergence to gains in speed but also for the other estimation methods , including nce.

It is well known that a Gaussian as noise (proposal) distribution is not the optimal choice for IS if the data has heavy tails (Wasserman , 2004, Chapter 24). Gaussian noise is not the optimal choice for NCE either. The presented results should thus not be considered as a general comparison of the two estimation methods perse . Importantly , however , the chosen setup allows one to assess how NCE behaves when the data has heavier tails than the noise, which is often the case in practice.

Finally, the reader may want to keep in mind that for other kinds of data, in particular also in very high dimensions, differences may occur.

4.4 Results

We first compare NCE with the methods for which we use the same optimization algorithm, that is MCMLE and score matching. Then, we compare it with CD/PCD.

4.4.1 Comparison with MC MLE and Score Matching

Figure 6 shows the comparison of (NCE, red squares), MCML (IS, blue circles) and score matching (SM, black triangles). The left panels show the simulation results in form of "result points" where the x-coordinate represents the time till the algorithm converged and the y- coordinate the estimation error at convergence. Convergence in the employed nonlinear conjugate gradient algorithm by Rasmussen (2006) means that the line search procedure failed twice in a row to meet the strong Wolfe-Powell conditions (Sun and Yuan, 2006, Chapter 2.5.2). For score matching, 100 result points corresponding to 100 different random mixing matrices are shown in each figure. For NCE and MCml, we used ten different values of ν so that for these methods, each figure shows 1000 result points. The panels on the right present the simulation result in a more schematic way. For NCE and MCml, the different ellipses represent the outcomes for different values of ν. Each ellipse contains 90% of the result points. We can see that increasing ν reduces the estimation error but it also increases the running time. For score matching, there is no such trade-off.

Figure 6(a) shows that for Laplacian sources, NCE outperforms the other methods in terms of the trade-off between statistical and computational performance. The large estimation error of score matching is likely to be due to the smooth approximation of the sign function in Eq(21). The figure also shows that NCE handles noise that has lighter tails than the data more gracefully than MCMLE. The reason is that the nonlinearity $h(\mathbf{u};\boldsymbol{\theta})$ in the objective function in Eq (8) is bounded even if data and noise distribution do not match well (see also Pihlaja et al., 2010).

For logistic sources, shown in Figure 6(b), nce and MCml perform equally. Score matching reaches its level of accuracy about 20 times faster than the other methods. NCE and MCml can, however, have a higher estimation accuracy than score matching if ν is large enough. Score matching can thus be considered to have a built-in trade-off between estimation performance and computation time: Computations are fast but the speed comes at the cost of not being able to reach an estimation accuracy as high as, for instance, NCE.

4.4.2 Comparison with Contrastive and Persistent Contrastive Divergence

Since CD/PCD do not have an objective function and given the randomness that is introduced by the minibatches, it is difficult to choose a reliable stopping criterion. Hence, we did not impose any stopping criterion but the maximal number of iterations. The two algorithms had always converged before this maximal number of iterations was reached in the sense that the estimation error did not visibly decrease any more.

We base our comparison on the estimation error as a function of the running time of the algorithm. This makes the comparison independent from the stopping criterion that is used in NCE. For NCE, the parameter ν controls the trade-off between computational and statistical performance ; for CD/PCD, it is the number of leapfrog steps and the number of Markov steps taken in each update . We compiled a trade-off curve for each of the one hundred estimation problems by taking at any time point the minimum estimation error over the various estimation errors that are obtained for different values of the trade -off parameters.[7] Figure 7 shows an example for NCE and CD. The distribution of the trade-off curves is shown in Figure 8. For large running times, the distribution of the estimation error is for all estimation methods similar to the one for MLE. For shorter running times, NCE is seen to have for Laplacian sources a better trade-off than the other methods. For logistic sources, however, the situation is reversed.

4.4.3 SUMMARY

The foregoing simulation results and discussion suggest that all estimation methods trade, in one form or the other, estimation accuracy against computation speed. In terms of this trade-off, NCE is particularly well suited for the estimation of data distributions with heavy tails . In case of thin tails , NCE performs similarly to MCml , and CD/PCD has a better trade-off. If the data distribution is particularly smooth and the model algebraically not too complicated , score matching may , depending on the required estimation accuracy, be the best option.

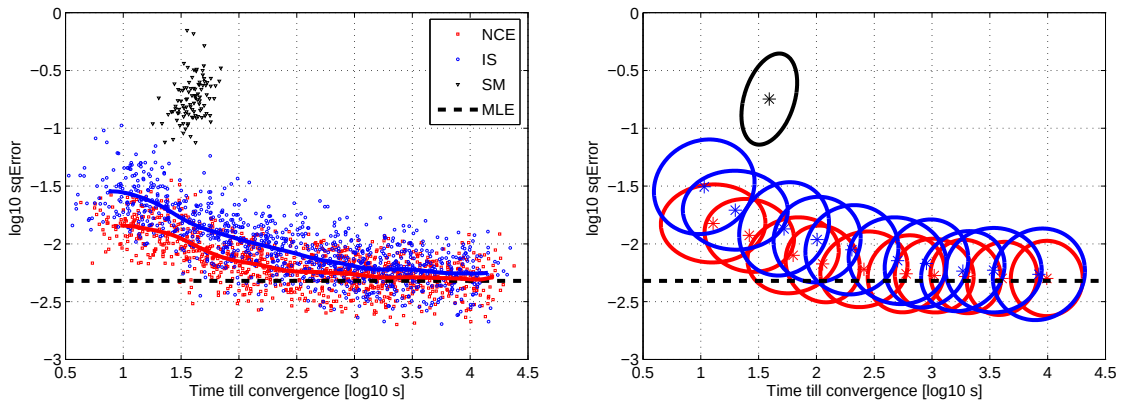
5. Simulations with Natural Images

In this section, we estimate with our new estimation method models of natural images. In the theory of NCE, we have assumed that all variables can be observed . NCE can thus not be used for models with latent variables which cannot be integrated out analytically . Such models occur for example in the work by Olshausen and Field (1996), Hyvarinen et al. (2001a), Karklin and Lewicki (2005), Lucke and Sahani (2008) and Osindero and Hinton (2008). We are here considering models which avoid latent variables. Recent models which are related to the models that we are considering here can be found in the work by Osindero et al. (2006), Koster and Hyvarinen (2010) and Ranzato and Hinton (2010). For a comprehensive introduction to natural image statistics, see for example the textbook by Hyvarinen et al. (2009).

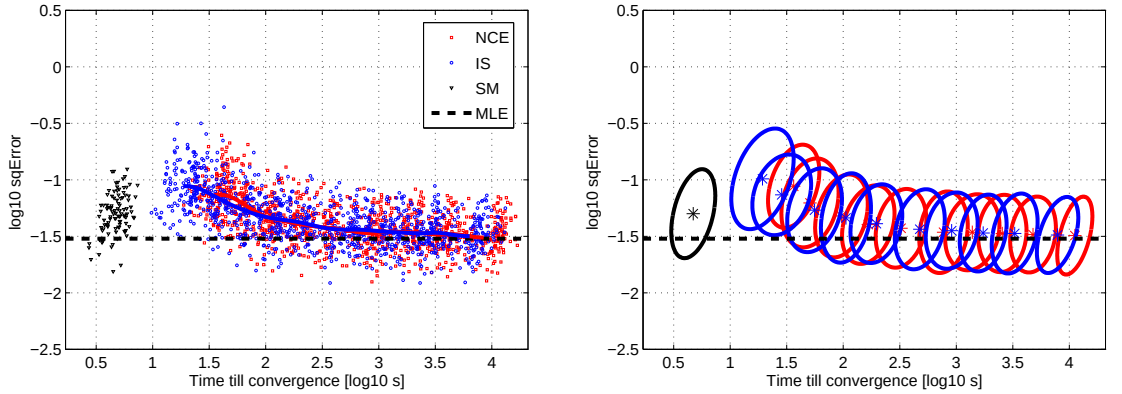
The presented models will consist of two processing layers, like in a multilayer neural network. The output of the network for a given input image gives the value of the model-pdf at that image. Because of the two processing layers, we call the models “two-layer models”.

We start with giving some preliminaries in Section 5.1. In Section 5.2, we present the settings of nce. In Section 5.3, we properly define the two-layer model and estimate a version with more than 50000 parameters . In Section 5.4, we present an extension of the model where the learned output nonlinearity of the network belongs to the flexible family of splines . The different models are compared in Section 5.5.

7. A comparison of CD and PCD for different settings can be found in Appendix C.1.



(a) Sources following a Laplacian density

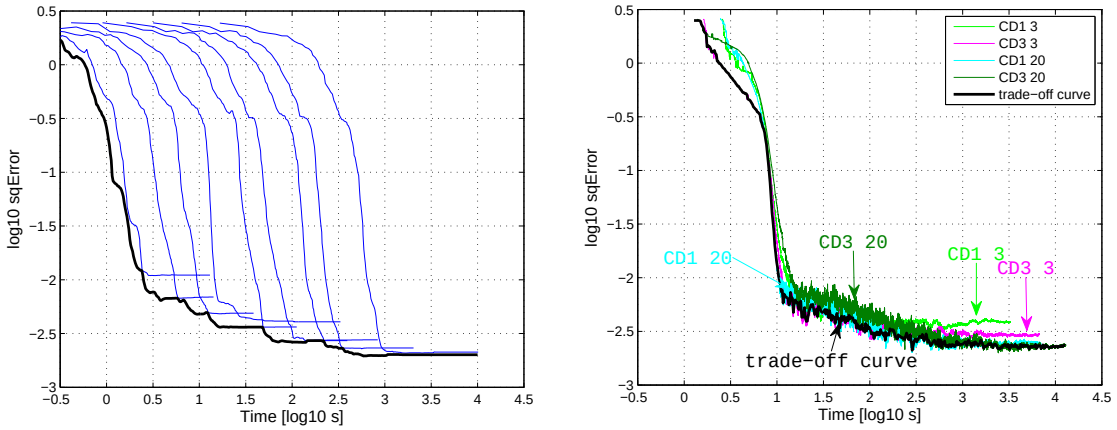


(b) Sources following a logistic density

Figure 6: Trade-off between statistical and computational performance for (NCE, red squares), MCML (IS, blue circles) and score matching (SM, black triangles). Each point represents the result of one simulation. Performing local linear kernel smoothing regression on the result points yields the thick curves. For NCE and MCML, the ten ellipses represent the outcomes for the ten different values of $\nu \in \{1,2,5,10,20,50,100,200,400,1000\}$. The ellipses were obtained by fitting a Gaussian to the distribution of the result points, each one contains 90% of the results points for a given ν. The asterisks mark their center. For an ICA model with Laplacian sources, NCE has the best trade-off between statistical and computational performance. For logistic sources, NCE and IS perform equally well. For medium estimation accuracy, score matching outperforms the other two estimation methods.

5.1 Data, Preprocessing and Modeling Goal

Our basic data are a random sample of $25\text{px} \times 25\text{px}$ image patches that we extracted from a subset of van Hateren's image database (van Hateren and van der Schaaf, 1998). The images in the subset showed wildlife scenes only. The sample size T_d is 160000.



(a) Noise-contrastive estimation

(b) Contrastive divergence

Figure 7: Example of a trade-off curve for NCE and CD.

(a) The different curves in blue show the estimation error which is obtained for the various values of ν. The thicker curve in black shows the trade-off curve. It is is obtained by taking at any time point the minimum estimation error. (b) The trade-off curve, shown in black, is similarly obtained by taking the minimum over the estimation errors which are obtained with different settings of CD.

As preprocessing, we removed from each image patch its average value (local mean, DC component), whitened the data and reduced the dimension from $d = 25 \cdot 25 = 625$ to $n = 160$. This retains 93% of the variance of the image patches. After dimension reduction, we additionally centered each data point and rescaled it to unit variance. In order to avoid division by small numbers, we avoided taking small variance patches. This gave our data $X = (\mathbf{x}_1, \ldots, \mathbf{x}_{T_d})$. Because of the centering and rescaling, each data point $\mathbf{x}_t$ satisfies

$$\sum_{k=1}^{n} \mathbf{x}_t(k) = 0, \qquad \frac{1}{n-1} \sum_{k=1}^{n} \mathbf{x}_t(k)^2 = 1. \tag{22}$$

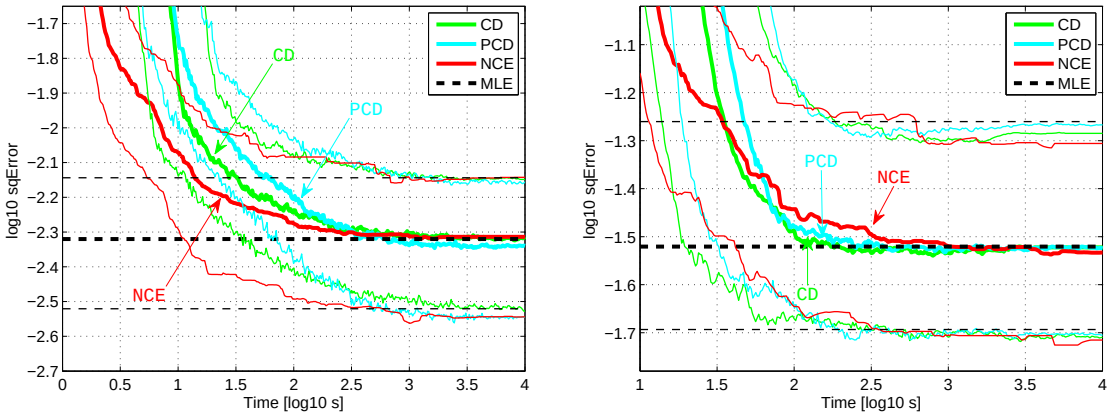
This means that each data point lies on the surface of a $n-1$ dimensional sphere $\mathbb{S}$.

This kind of preprocessing is a form of luminance and contrast gain control which aim at canceling out the effects of the lighting conditions (see for example Hyvärinen et al., 2009, Chapter 9, where also the statistical effects of such a preprocessing are analyzed). Centering and rescaling to unit variance has also been used in image quality assessment in order to access the structural component of an image, which is related to the reflectance of the depicted objects (Wang et al., 2004, in particular Section III.B). By modeling the data X, we are thus modeling the structure in the image patches.

Given a data point $\mathbf{x}_t$, we can reconstruct the original (vectorized) image patch via

$$\mathbf{i}_t = \mathbf{V}^- \mathbf{x}_t, \qquad \mathbf{V}^- = \mathbf{E}\mathbf{D}^{1/2}, \tag{23}$$

where $\mathbf{E}$ is the $d \times n$ matrix formed by the leading n eigenvectors of the covariance matrix of the image patches. The diagonal $n \times n$ matrix $\mathbf{D}$ contains the corresponding eigenvalues. The matrix



(a) Laplacian sources

(b) Logistic sources

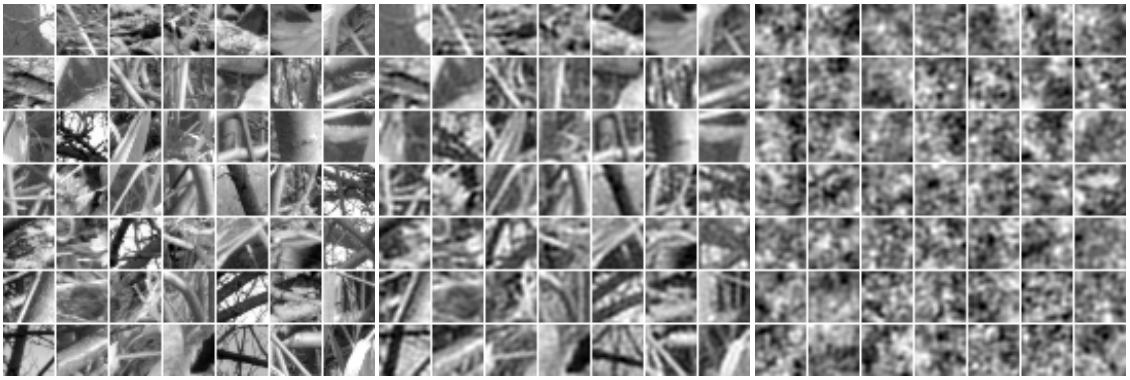
Figure 8: Distribution of the trade-off curves for CD(green), PCD(cyan), and NCE(red). The distribution of the estimation error for MLE is shown in black. The thick curves show the median, the finer curves the 0.9 and 0.1 quantiles.

$\mathbf{V}^-$ defined above is the pseudoinverse of the whitening matrix $\mathbf{V} = \mathbf{D}^{-1/2}\mathbf{E}^T$. Since the column vectors of $\mathbf{V}^-$ form a basis for a n dimensional subspace of $\mathbb{R}^d$, $\mathbf{x}$ is the coordinate vector of $\mathbf{i}$ with respect to that basis. The dimension reduction implies that the reconstruction cannot be perfect; the reconstruction can also only performed up to the scale and average value of the patch because of the the luminance and contrast gain control. Figure 9(a) shows examples of natural image patches after extraction from the data base; Figure 9(b) shows the corresponding reconstructions $\mathbf{i}$. Since all image patches in Figure 9 were rescaled to use the full colormap, the effects of luminance and contrast gain control are not visible. The effect of the dimension reduction is low-pass filtering.

5.2 Settings for NCE

Matlab code for the simulations is available from the authors' homepage so that our description here will not be exhaustive. All the models considered in the next subsections are estimated with noise-contrastive estimation. We learn the parameters by optimization of the objective J_T in Equation (8). The two-layer models are estimated by first estimating one-layer models. The learned parameters are used as initial values for the first layer in the estimation of the complete two-layer model. The second layer is initialized to small random values.

For the cnd p_n, we take a uniform distribution on the surface of the $n-1$ dimensional sphere $\mathbb{S}$ on which $\mathbf{x}$ is defined.[8] Examples of image patches with coordinates following p_n are shown in Figure 9(c). Samples from p_n can easily be created by sampling from N(0,1), followed by centering and rescaling such that Eq (22) holds. Since p_n is a constant, the log-ratio $G(.;\boldsymbol{\theta})$ in Eq (4) is up to an additive constant equal to



(a) Image patches (b) Reconstructions (c) Noise

Figure 9: (a) Natural image patches of size $25\text{px} \times 25\text{px}$. (b) Reconstructed image patches after preprocessing. These are examples of the image patches denoted by $\mathbf{i}$ in Equation (23) with coordinate vectors $\mathbf{x} \in \mathbb{R}^{160}$. (c) Noise images which are obtained via Equation (23) if the coordinates are uniformly distributed on the sphere $\mathbb{S}$. Comparison with Figure (b) shows that the coordinate vectors $\mathbf{x}$ for natural images are clearly not uniformly distributed on the sphere. In the next subsections, we model their distribution.

$\ln p_m(.;\boldsymbol{\theta})$,

$$G(.;\boldsymbol{\theta}) = \ln p_m(.;\boldsymbol{\theta}) + \text{constant}.$$

As pointed out in Section 2.2, $\boldsymbol{\theta}$ evolves in the maximization of J_T such that $G(\mathbf{u};\hat{\boldsymbol{\theta}}_T)$ is as large as possible for $\mathbf{u} \in X$ (natural images) but as small as possible for $\mathbf{u} \in Y$ (noise). For uniform noise, the same must thus also hold for $\ln p_m(\mathbf{u};\hat{\boldsymbol{\theta}}_T)$. This observation will be a useful guiding tool for the interpretation of the models below.

The factor $\nu = T_n/T_d$ was set to 10. We found that an iterative optimization procedure where we separate the data into subsets and optimize J_T for increasingly larger values of ν reduced computation time. The optimization for each ν is done with the nonlinear conjugate gradient method of Rasmussen (2006). The size of the subsets is rather large, for example 80000 in the simulation of the next subsection.[9] A more detailed discussion of this optimization procedure can be found in Appendix C.2.

5.3 Two-Layer Model with Thresholding Nonlinearities

The first model that we consider is

$$\ln p_m(\mathbf{x};\boldsymbol{\theta}) = \sum_{k=1}^{n} f(y_k;a_k,b_k) + c, \qquad y_k = \sum_{i=1}^{n} Q_{ki}(\mathbf{w}_i^T\mathbf{x})^2, \tag{24}$$

where f is a smooth, compressive thresholding function that is parameterized by a_k and b_k. See Figure 10 for details regarding the parameterization and the formula for f. The parameters $\boldsymbol{\theta}$ of

9. As pointed out in Section 4.3, the used nonlinear conjugate gradient algorithm is not suitable for stochastic optimization with small minibatches.

the model are the second-layer weights $Q_{ki} \geq 0$, the first-layer weights $\mathbf{w}_i \in \mathbb{R}^n$, the normalizing parameter $c \in \mathbb{R}$, as well as $a_k > 0$ and $b_k \in \mathbb{R}$ for the nonlinearity f. The definition of y_k shows that multiplying Q_{ki} by a factor γ_i^2 and $\mathbf{w}_i$ at the same time by the factor $1/\gamma_i$ does not change the value of y_k. There is thus some ambiguity in the parameterization which could be resolved by imposing a norm constraint either on the $\mathbf{w}_i$ or on the columns of the matrix $\mathbf{Q}$ formed by the weights Q_{ki}. It turned out that for the estimation of the model such constraints were not necessary. For the visualization and interpretation of the results, we chose γ_i such that all the $\mathbf{w}_i$ had norm one.

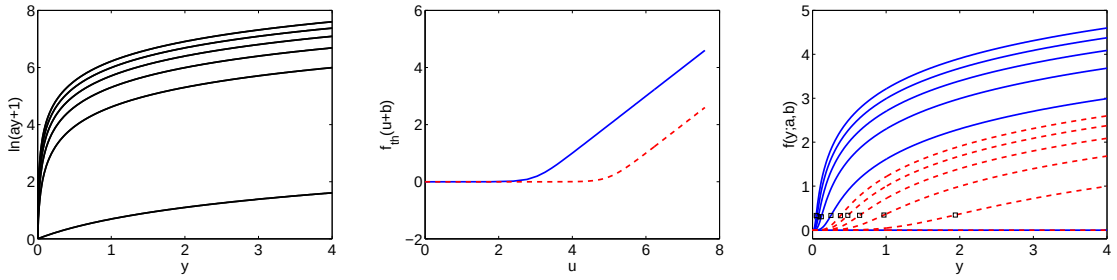
The motivation for the thresholding property of f is that, in line with Section 5.2, $\ln p_m(.;\boldsymbol{\theta})$ can easily be made large for natural images and small for noise. The y_k must just be above the thresholds for natural image input and below for noise. This occurs when the vectors $\mathbf{w}_i$ detect features (regularities) in the input which are specific to natural images, and when, in turn, the second-layer weights Q_{ki} detect characteristic regularities in the squared first-layer feature outputs $\mathbf{w}_i^T\mathbf{x}$. The squaring implements the assumption that the regularities in $\mathbf{x}$ and $-\mathbf{x}$ are the same so that the pdf of $\mathbf{x}$ should be an even function of the $\mathbf{w}^T{}_i\mathbf{x}$. Another property of the nonlinearity is its compressive log-like behavior for inputs above the threshold. The motivation for this is to "counteract" the squaring in the computation of y_k. The compression of large values of y_k leads to numerical robustness in the computation of $\ln p_m$.

A model like the one in Eq (24) has been studied before by Osindero et al. (2006) and Koster and Hyvärinen (2010). There are, however, a number of differences. The main difference is that in our case $\mathbf{x}$ lies on a sphere while in the cited work, $\mathbf{x}$ was defined in the whole space $\mathbb{R}n$. This difference allows us to use nonlinearities that do not decay asymptotically to $-\infty$ which is necessary if $\mathbf{x}$ is defi ned in $\mathbb{R}n$. A smaller difference is that we do not need to impose norm constraints to facilitate the learning of the parameters.

5.3.1 RESULTS

For the visualization of the first-layer feature detectors $\mathbf{w}_i$, note that the inner product $\mathbf{w}_i^T\mathbf{x}$ equals $(\mathbf{w}^T{}_i\mathbf{V})\mathbf{i} = \mathbf{w}\tilde{}^T{}_i\mathbf{i}$. The $\mathbf{w}_i \in \mathbb{R}^n$ are coordinate vectors wrt the basis given by the columns of $\mathbf{V}^-$, see Section 5.1, while the $\mathbf{w}\tilde{}_i \in \mathbb{R}^d$ are the coordinate vectors with respect to the pixel basis. The latter vectors can thus be visualized as images. This is done in Figure 11(a). Another way to visualize the fi rst-layer feature detectors $\mathbf{w}_i$ is to show the images which yield the largest feature output while satisfying the constraints in Eq(22). These optimal stimuli are proportional to $\mathbf{V}^-(\mathbf{w}_i - \langle \mathbf{w}_i \rangle)$, where $\langle \mathbf{w}_i \rangle \in \mathbb{R}$ is the average value of the elements in the vector $\mathbf{w}_i$, see Appendix B.2 for a proof. The optimal stimuli are shown in Figure 11(b). Both visualizations show that the first layer computes " Gabor-like" features, which is in line with previous research on natural image statistics.

Figure 12 shows a random selection of the learned second-layer weights Q_{ik}. Figure 12(a) shows that the weights are extremely sparse. The optimization started with the weights being randomly assigned to small values, with the optimization most of them shrank to zero; few selected ones, however, increased in magnitude. Note that this result was obtained without any norm constraints on $\mathbf{Q}$. From Figure 12(b), we see that the learned second-layer weights Q_{ik} are such that they combine first-layer features of similar orientation, which are centered at nearby locations ("complex cells"). The same figure shows also a condensed representation of the feature detectors using icons. This form of visualization is used in Figure 13 to visualize all the second-layer feature detectors.



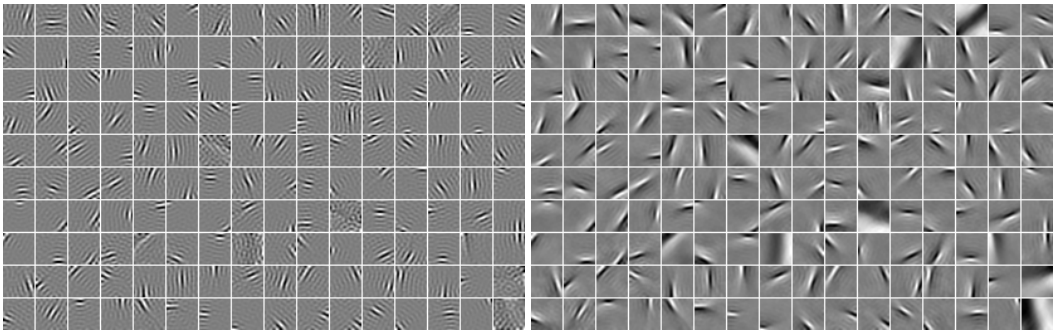
(a) Compression (b) Rectification (c) Resulting nonlinearity

Figure 10: Two-layer model with thresholding nonlinearities. The family of nonlinearities used in the modeling is $f(y;a,b) = f_{\text{th}}(\ln(ay+1)+b)$, $y \geq 0$. The parameterized function is composed of a compressive nonlinearity $\ln(ay+1)$, shown in Figure (a), and a smooth rectification function $f_{\text{th}}(u+b)$ shown in Figure (b). Figure (c) shows examples of $f(y;a,b)$ for different values of a and b. Parameter b sets the threshold, and parameter a controls the steepness of the function. Since the scale of the weights in Equation (24) is not restrained, the parameters a_k do not need to be learned explicitly. After learning, they can be identified by dividing y_k in Equation (24) by a_k so that its expectation is one for natural images. The formula for the thresholding function is $f_{\text{th}}(u) = 0.25\ln(\cosh(2u)) + 0.5u + 0.17$. The curves shown in blue are for $b = -3$ and $a \in \{1, 50, 100, 200, \ldots, 500\}$. For the dashed curves in red, $b = -5$. The small squares in Figure (c) indicate where f changes from convex to concave.

Figure 14(a) shows the learned nonlinearities $f(.;a_k,b_k)$. Note that we incorporated the learned normalizing parameter c as an offset c/n for each nonlinearity. The learned thresholding is similar for feature outputs of mid- and high-frequency feature detectors (black, solid curves). For the feature detectors tuned to low frequencies, the thresholds tend to be smaller (green, dashed curves). The nonlinearities in black are convex for arguments y smaller than two (see red rectangle in the figure). That is, they show a squashing behavior for $y < 2$. Looking at the distribution of the second-layer outputs y_k in Figure 14(b), we see that it is more likely that noise rather than natural images was the input when the second-layer feature outputs y_k are approximately between 0.5 and 2. In this regime, the squashing nonlinearities map thus more often the noise input to small values than natural images so that $\ln p_m(\mathbf{u};\hat{\boldsymbol{\theta}}_T)$ tends to be larger when input $\mathbf{u}$ is a natural image than when it is noise (see Section 5.2). One could, however, think that the thresholding nonlinearities are suboptimal because they ignore the fact that natural images lead, compared to the noise, rather often to y_k which are close to zero, see Figure 14(b). An optimal nonlinearity should, unlike the thresholding nonlinearities, assign a large value to both large and small y_k while mapping intermediate values of y_k to small numbers. The next subsection shows that such kinds of mappings emerge naturally when splines are used to learn the nonlinearities from the data.

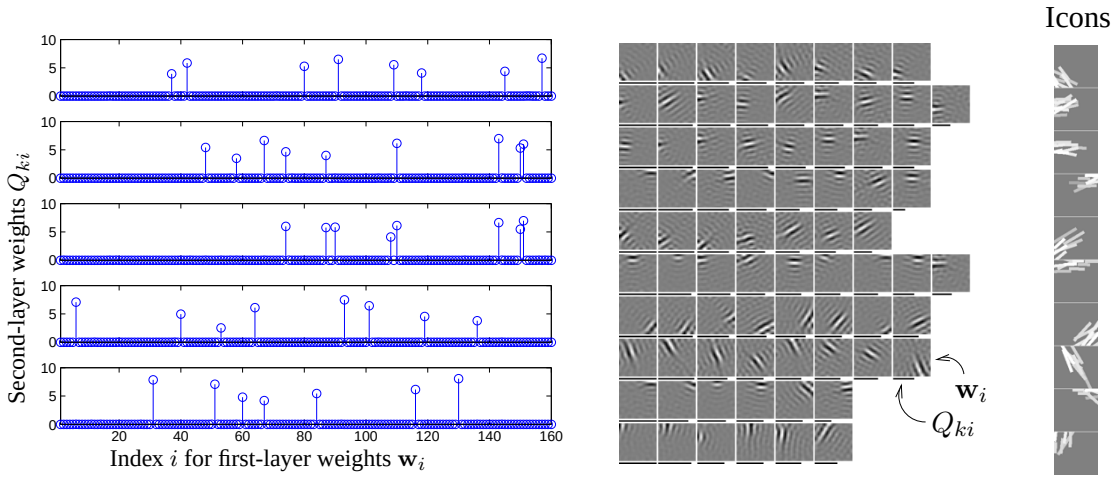
5.4 Two-Layer Model with Spline Nonlinearities

In the previous subsection, the family of nonlinearities f in Eq (24) was rather limited. Here, we look for f in the larger family of cubic splines where we consider the location of the knots to be fixed



(a) Feature detectors (b) Optimal stimuli

Figure 11: Two-layer model with thresholding nonlinearities: Visualization of the learned first-layer feature detectors $\mathbf{w}_i$. (a) The feature detectors in the pixel basis. (b) The corresponding optimal stimuli. The feature detectors in the first layer are "Gabor-like" (localized, oriented, bandpass). Comparison of the two figures shows that feature detectors which appear noisy in the pixel basis are tuned to low-frequency input.



(a) Raw result (b) Graphical visualization

Figure 12: Two-layer model with thresholding nonlinearities: Random selection of second layer units. (a) Second-layer weights Q_{ki} for five different k (five different rows of the matrix $\mathbf{Q}$) are shown. The weights are extremely sparse so that in the sum $\sum_{i=1}^{n} Q_{ki}(\mathbf{w}_i^T\mathbf{x})^2$ only few selected squared first-layer outputs are added together. (b) Every row shows one second-layer feature detector. The first-layer feature detectors $\mathbf{w}_i$ are shown as image patches like in Figure 11, and the black bar under each patch indicates the strength Q_{ki} by which a certain $\mathbf{w}_i$ is pooled by the k-th second-layer feature detector. The numerical values Q_{ki} for the first five rows are shown in Figure (a). The right-most column shows a condensed visualization. The icons were created by representing each first-layer feature by a bar of the same orientation and similar length as the feature, and then superimposing them with weights given by Q_{ki}.

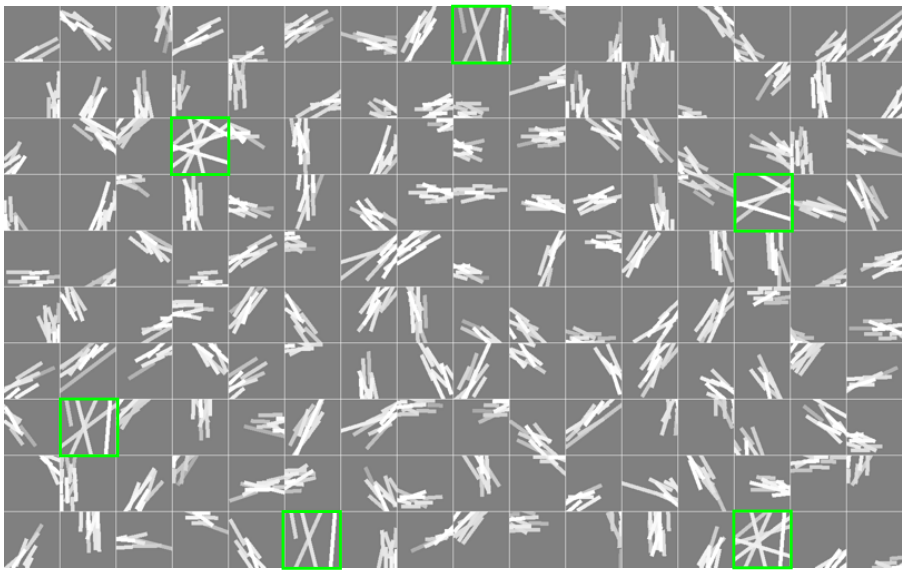
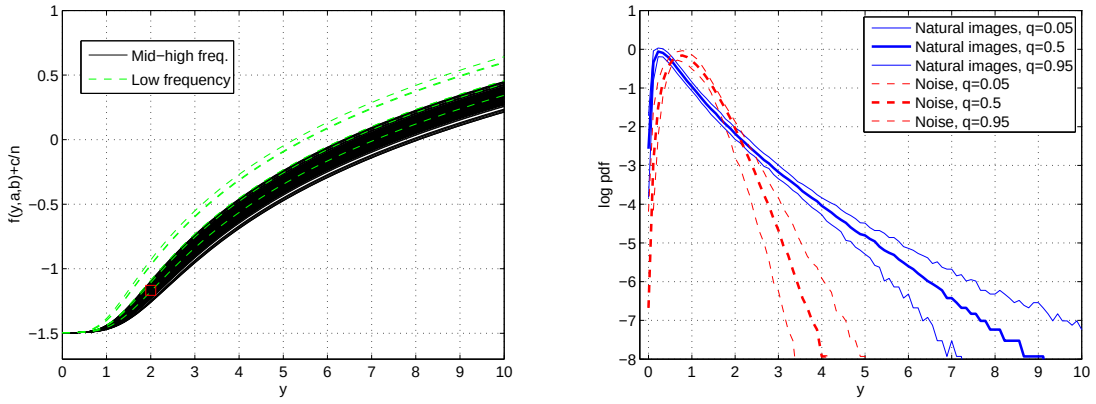


Figure 13: Two-layer model with thresholding nonlinearities: Visualization of the first- and second-layer feature detectors with icons. In the second layer, first-layer features of similar orientations are pooled together. See Figure 12 for details of how the icons were created. The feature detectors marked with a green frame are tuned to low frequencies.



(a) Learned nonlinearities

(b) Distribution of second-layer outputs y_k

Figure 14: Two-layer model with thresholding nonlinearities: Learned nonlinearities and interpretation. Natural images tend to have larger second-layer outputs y_k than noise input since the two processing layers, visualized in Figures 11 to 13, detect structure inherent to natural images. Thresholding the y_k provides a way to assign to natural images large values in the model-pdf and to noise small values. In Figure (a), the nonlinearities acting on pooled low-frequency feature detectors are shown in green (dashed lines), those for medium and high frequency feature detectors in black (solid lines). The bold curves in Figure (b) show the median, the other curves the 5% and 95% quantiles. The solid curves in blue relate to natural images, the dashed curves in red to noise. As explained in Figure 10, the y_k have expectation one for natural images.

(regression splines represented with B-spline basis functions, Hastie et al., 2009, Chapter 5).

The model that we consider here is

$$\ln p_m(\mathbf{x};\boldsymbol{\theta}) = \sum_{k=1}^{n} f(y_k; a_1, a_2, \ldots) + c, \qquad y_k = \sum_{i=1}^{n} Q_{ki}(\mathbf{w}_i^T \mathbf{x})^2. \tag{25}$$

The difference between this and the model of the previous subsection is that the output nonlinearity f is a cubic spline. Part of the parameters $\boldsymbol{\theta}$ are thus as previously the $\mathbf{w}_i \in \mathbb{R}^n$, $Q_{ki} \geq 0$, and $c \in \mathbb{R}$. Additional parameters are the $a_i \in \mathbb{R}$ which are the coefficients of the B-spline basis functions of the cubic spline f. As before, we denote the matrix formed by the Q_{ki} by $\mathbf{Q}$.

For the modeling of the nonlinearity f, we must define its domain, which is the range of its arguments y_k. A way to control the range of y_k is to constrain the norm of the columns of $\mathbf{Q}$ and also to constrain the vectors $\mathbf{w}_k$ such that

$$\max_i \mathrm{E}\left\{(\mathbf{w}_i^T \mathbf{x})^2\right\} = 1, \tag{26}$$

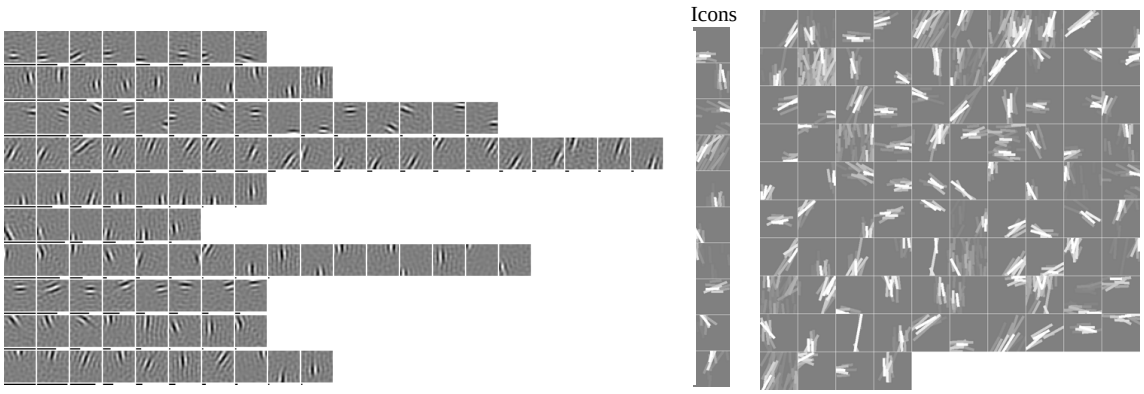
where the expectation is taken over the natural images.

We estimated the model in Equation (25) by first estimating a spline-based one-layer model which is presented in Appendix C.3. In brief, in this model, we did not square the first-layer feature outputs $\mathbf{w}_i^T \mathbf{x}$ and the matrix $\mathbf{Q}$ was the identity. The arguments of the spline nonlinearity f were thus the feature outputs $\mathbf{w}_i^T \mathbf{x}$ without additional processing. The learned nonlinearity is shown in Figure 16(a). In the following, we denote it by f_1. In Appendix C.3, we point out that the shape of f_1 is closely related to the sparsity of the feature outputs when natural images are the input. Because f_1 is an even function, and because of the squaring in the definition of y_k, we initialized f for the estimation of the two-layer model as $f(u) = f_1(\sqrt{u})$. This function is shown in Figure 16(b) (blue, dashes). The learned $\mathbf{w}_i$ of the one-layer model were used as initial points for the estimation of the two-layer model. The Q_{ki} were randomly initialized to small values. It turned out that imposing Equation (26) was enough for the learning to work and no norm constraint for the columns of $\mathbf{Q}$ was necessary. The results were very similar whether there were norm constraints or not. In the following, we report the results without any norm constraints.

5.4.1 Results

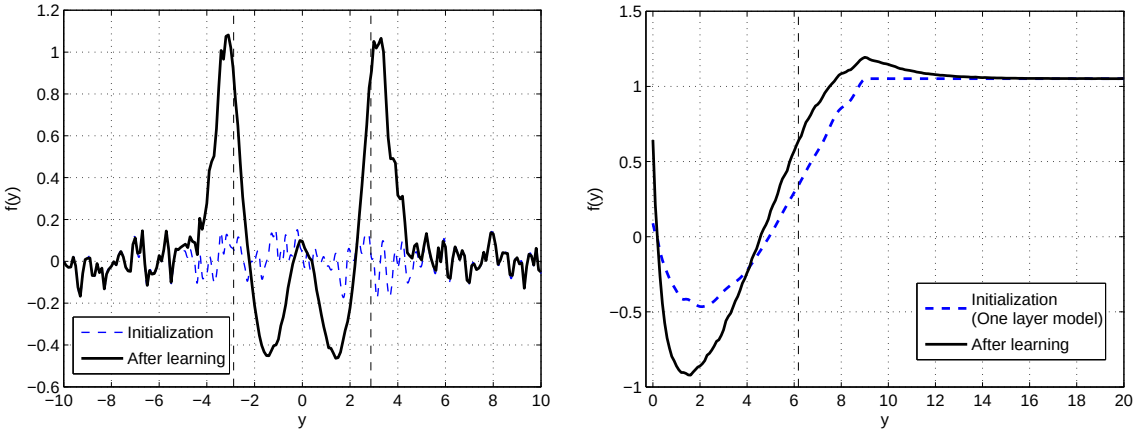
Figure 15 visualizes the learned parameters $\mathbf{w}_i$ and Q_{ki} in the same way as in Figures 12 and 13 for the two-layer model with thresholding nonlinearities. The learned feature extraction stage is qualitatively very similar, up to two differences. The first difference is that many second-layer weights Q_{ki} shrank to zero: 66 out of 160 rows of the matrix $\mathbf{Q}$ had so small values that we could omit them while accounting for 99.9% of the sum $\sum_{ki} Q_{ki}$. The second difference is that the pooling in the second layer is sometimes less sparse. In that case, the second layer still combines first-layer feature detectors of the same orientation but they are not all centered at the same location.

The learned nonlinearity f is shown in Figure 16(b) (black, solid). The nonlinearity from the one-layer model, shown in blue as a dashed curve, is altered so that small and large inputs are assigned to larger numbers while intermediate inputs are mapped to smaller numbers. Compared to the thresholding nonlinearities from the previous subsection, the learned nonlinearity has also for small inputs large outputs. Since the second-layer feature outputs y_k are sparser (that is, more often very small or large) for natural images than for the noise, the shape of the learned nonlinearity



(a) Pooling in the second layer (b) Representation with icons

Figure 15: Two-layer model with spline nonlinearities. (a) Random selection of the learned second-layer units. (b) Representation of all the learned second-layer feature detectors as iconic images.



(a) Learned nonlinearity, one-layer model (b) Learned nonlinearity, two-layer model

Figure 16: Two-layer model with spline nonlinearities. (a) Learned nonlinearity (black, solid) and its random initialization (blue, dashes) for the one-layer model. The learned nonlinearity is used as starting point in the learning of the two-layer model. (b) Learned nonlinearity (black, solid) and its initialization (blue, dashes) for the two-layer model. The dashed vertical lines indicate the 99% quantile for all the feature outputs for natural images. Due to the lack of training examples, the nonlinearities should not be considered valid beyond these lines.

implies that the estimated model assigns more often a higher probability density to natural images than to the noise.

5.5 Model Comparison

We have estimated models for natural images, both with thresholding nonlinearities and with splines. We make here a simple model comparison.

A quantitative comparison is done by calculating for ten validation sets the value of the objective function J_T of noise-contrastive estimation (see Equation (8) for the definition). The sample size of each validation set was $T_v = 100000$, and ν was set to 10, as in the estimation of the models. For the same validation data, we also computed the performance measure $\hat{L} = 1/T_v \sum_t \ln p_m(\mathbf{x}_t; \hat{\boldsymbol{\theta}}_T)$, which is an estimate for the rescaled log-likelihood, see Equation (13) in Section 3.1. As pointed out there, $\hat{L}$ is only an estimate of the rescaled log-likelihood because $\hat{c}$, which is an element of the parameter vector $\hat{\boldsymbol{\theta}}_T$, is used instead of the correct normalizing constant. Both J_T and the log-likelihood have the property that models which fit the data better have a higher score.

Comparing the structure of data points which are considered likely by the different models is a way to make a qualitative model comparison. Another approach would be to sample from the models, which we do in Appendix C.5. In order to get the likely points, we drew random samples that followed the noise distribution p_n (uniform on the sphere), and used them as initial points in the optimization of the various log-densities $\ln p_m(\mathbf{x}; \hat{\boldsymbol{\theta}}_T)$ with respect to $\mathbf{x}$ under the constraint of Equation (22). We used the same initial points for all models and visualized the likely points $\hat{\mathbf{x}}$ via Equation (23) as images $\hat{\mathbf{i}} = \mathbf{V}^- \hat{\mathbf{x}}$.

The ICA model with Laplacian sources is a simple model for natural images. It has previously also been used to model natural images after they have been projected on a sphere (Hyvärinen et al., 2009, Chapter 9). The unnormalized model has been defined in Section 3.2 in Equation (17) and consists of one processing layer with the fixed nonlinearity $f(u) = -\sqrt{2}|u|$. We include it in our comparison and refer to it as one-layer model with "Laplacian nonlinearity".

5.5.1 Results

Table 1 shows that the spline-based two-layer model of Section 5.4 gives, on average, the largest value of the objective function J_T, and also L_T. To investigate the merits of the spline output-nonlinearity, we fixed the feature extraction stage of the thresholding model in Section 5.3 and learned only the nonlinearity f using splines (for details, see Appendix C.4). The resulting model, labeled "refinement" in the table, performs nearly as good as the best model. The one-layer models with thresholding or Laplacian nonlinearities have the smallest objectives J_T and L_T. The two models achieve the objectives in different, complimentary ways. For the thresholding model, the absolute value of the feature outputs $\mathbf{w}_i^T \mathbf{x}$ must be large to yield a large objective while for the model with the Laplacian nonlinearity $f(\mathbf{w}_i^T \mathbf{x}) = -\sqrt{2}|\mathbf{w}_i^T \mathbf{x}|$, the feature outputs must have small absolute values. The two models consider thus different aspects of the, for natural images, typically sparse feature outputs $\mathbf{w}_i^T \mathbf{x}$. The one-layer model with spline nonlinearity combines both aspects, see Figure 16(a), and yields also a higher score in the comparison. The same reason explains why spline-based two-layer models have higher scores than the two-layer model with the thresholding nonlinearity.

Figure 17 shows the likely data points from the various models p_m. The models with large objectives in Table 1 lead to image patches with particularly clear structure. The emergence of structure can be explained in terms of sparse coding since image patches which lead to sparse activations of the feature detectors are typically highly structured. Sparseness of the feature outputs

	One-layer model			Two-layer model		
	Thresholding	Laplacian	Spline	Thresholding	Refinement	Spline
J_T, av	-1.871	-1.518	-1.062	-0.8739	-0.6248	**-0.6139**
J_T, std	0.0022	0.0035	0.0030	0.0029	0.0030	0.0037
L_T, av	-223.280	-222.714	-219,786	-220.739	-213.303	**-212.598**
L_T, std	0.0029	0.0077	0.0137	0.0088	0.0282	0.0273

Table 1: Quantitative model comparison. The objective J_T of noise-contrastive estimation, see Equation (8), and the estimate $\hat{L}$ of the (rescaled) log-likelihood, see Equation (13), are used to measure the performance. Larger values indicate better performance. The table gives the average (av) and the standard deviation (std) for ten validation sets. All models are defined on a sphere and learned with noise-contrastive estimation. The features for the one-layer models with thresholding and Laplacian nonlinearity are not shown in the paper. The "one-layer, thresholding" model is identical to the "two-layer, thresholding" model when the second layer is fixed to the identity matrix. With Laplacian nonlinearity we mean the function $f(u) = -\sqrt{2}|u|$. The "two-layer, thresholding" model has been presented in Section 5.3, and the "two-layer, spline" model in Section 5.4. The "one-layer, spline" and "two-layer, refinement" models are presented in the Appendix C.3 and C.4, respectively.

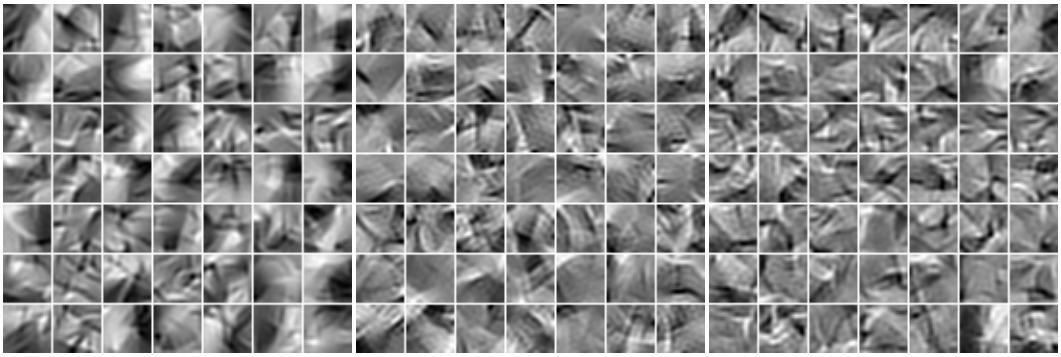
is facilitated by the nonlinearities in the models, and through the competition between the features by means of the sphere-constraint on the coordinates $\mathbf{x}$, as specified in Equation (22).

6. Conclusions

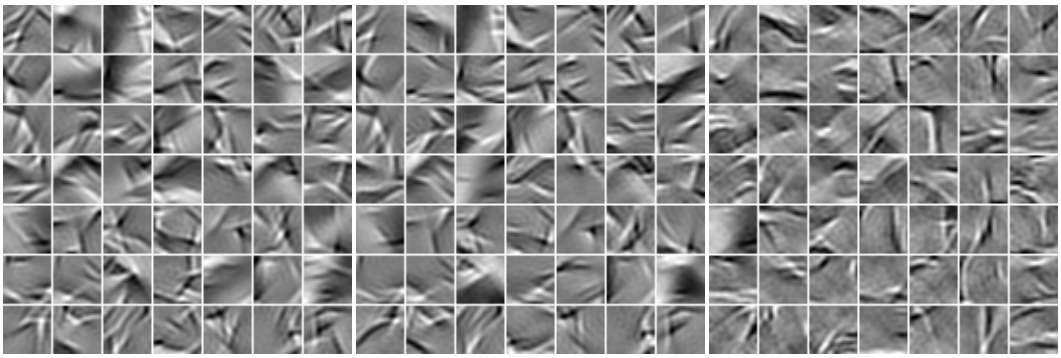
In this paper, we have considered the problem of estimating unnormalized statistical models for which the normalizing partition function cannot be computed in closed form. Such models cannot be estimated by maximization of the likelihood without resorting to numerical approximations which are often computationally expensive. The main contribution of the paper is a new estimation method for unnormalized models. A further contribution is made in the modeling of natural image statistics.

We have proven that our new estimation method, NCE, provides a consistent estimator for both normalized and unnormalized statistical models. The assumptions that must be fulfilled to have consistency are not stronger than the assumptions that are needed in MLE. We have further derived the asymptotic distribution of the estimation error which shows that, in the limit of arbitrarily many contrastive noise samples, the estimator performs like the maximum likelihood estimator. The new method has a very intuitive interpretation in terms of supervised learning: The estimation is performed by discriminating between the observed data and some artificially generated noise by means of logistic regression.

All theoretical results were illustrated and validated on artificial data where ground truth is known. We have also used artificial data to assess the balance between statistical and computational performance. In particular, we have compared the new estimation method to a number of other estimation methods for unnormalized models: Simulations suggest that NCE stries a highly competitive trade-off. We have used the mse of the estimated param-



(a) One-layer, thresholding (b) One-layer, Laplacian (c) One-layer, spline



(d) Two-layer, thresholding (e) Two-layer, refinement (f) Two-layer, spline

Figure 17: Likely points under the learned models for natural images. See caption of Table 1 for information on the models.

eters as statistical performance measure. It should be noted that this is only one possible criterion among many (see Hyvärinen, 2008, for a recently proposed alternative measure of performance).

NCE as presented here extends the previous definition given by Gutmann and Hyvarinen (2010) since it allows for more noise samples than data points. We have also previously considered such a generalization (Pihlaja et al., 2010). Unlike in that preliminary version, our method here is asymptotically Fisher-efficient for all admissible noise densities when the number of noise samples becomes arbitrarily large. Pihlaja et al. (2010) has established links of NCE to ISwhich remain valid for this paper.

We applied NCE to the modeling of natural images. Besides validating the method on a large two-layer model, we have, as a new contribution to the understanding of natural image statistics, presented spline-based extensions: In previous models, the output nonlinearity in the pdf was hand-picked. Here, we have parameterized it as a spline and learned it from the data. The statistical models were all unnormalized and had several ten-thousands of parameters which demonstrates that our new method can handle demanding estimation problems.

Acknowledgments

Yoshua Bengio, Ian Goodfellow, Pascal Vincent, Geoffrey Hinton, Nicolas Le Roux, Marc'Aurelio Ranzato, and Ilya Sutskever. This work was funded by the Centre-of-Excellence in Algorithmic Data Analysis and the Computational Sciences program, both of the Academy of Finland.

Appendix A. Proofs of the Theorems

A.1 Preliminaries

$$r_\nu(u) := \frac{1}{1+\nu\exp(-u)},$$

which was introduced in Equation (6):

$$\begin{aligned}
1-r_\nu(u) &= r_{\frac{1}{\nu}}(-u) \\
\frac{\partial r_\nu(u)}{\partial u} &= r_{\frac{1}{\nu}}(-u)r_\nu(u) \\
\frac{\partial}{\partial u}\ln r_\nu(u) &= r_{\frac{1}{\nu}}(-u) \\
\frac{\partial^2}{\partial u^2}\ln r_\nu(u) &= -r_{\frac{1}{\nu}}(-u)r_\nu(u) \\
\frac{\partial}{\partial u}\ln[1-r_\nu(u)] &= -r_\nu(u) \\
\frac{\partial^2}{\partial u^2}\ln[1-r_\nu(u)] &= -r_{\frac{1}{\nu}}(-u)r_\nu(u)
\end{aligned}$$

The functions $h(\mathbf{u};\boldsymbol{\theta}) = r_\nu\left(G(\mathbf{u};\boldsymbol{\theta})\right)$ and $1-h(\mathbf{u};\boldsymbol{\theta}) = r_{\frac{1}{\nu}}\left(-G(\mathbf{u};\boldsymbol{\theta})\right)$ are equal to

$$h(\mathbf{u};\boldsymbol{\theta}) = \frac{p_m(\mathbf{u};\boldsymbol{\theta})}{p_m(\mathbf{u};\boldsymbol{\theta})+\nu p_n(\mathbf{u})}, \qquad 1-h(\mathbf{u};\boldsymbol{\theta}) = \frac{\nu p_n(\mathbf{u})}{p_m(\mathbf{u};\boldsymbol{\theta})+\nu p_n(\mathbf{u})}, \tag{27}$$

see Eq (3). ==>

$$\nu p_n(\mathbf{u})r_\nu\left(G(\mathbf{u};\boldsymbol{\theta})\right) = \frac{\nu p_n(\mathbf{u})p_m(\mathbf{u};\boldsymbol{\theta})}{p_m(\mathbf{u};\boldsymbol{\theta})+\nu p_n(\mathbf{u})}, \tag{28}$$

$$p_d(\mathbf{u})r_{\frac{1}{\nu}}\left(-G(\mathbf{u};\boldsymbol{\theta})\right) = \frac{\nu p_n(\mathbf{u})p_d(\mathbf{u})}{p_m(\mathbf{u};\boldsymbol{\theta})+\nu p_n(\mathbf{u})}, \tag{29}$$

Taylor expansions

$$\begin{aligned}\ln r_\nu(u+\epsilon u_1+\epsilon^2 u_2) &= \ln r_\nu(u)+\epsilon r_{\frac{1}{\nu}}(-u)u_1+ \\ &\quad \epsilon^2\left[r_{\frac{1}{\nu}}(-u)u_2-\frac{1}{2}r_{\frac{1}{\nu}}(-u)r_\nu(u)u_1^2\right]+ \\ &\quad O(\epsilon^3), \qquad (30)\\ \ln\left[1-r_\nu(u+\epsilon u_1+\epsilon^2 u_2)\right] &= \ln[1-r_\nu(u)]-\epsilon r_\nu(u)u_1+ \\ &\quad \epsilon^2\left[-r_\nu(u)u_2-\frac{1}{2}r_{\frac{1}{\nu}}(-u)r_\nu(u)u_1^2\right]+ \\ &\quad O(\epsilon^3). \qquad (31)\end{aligned}$$

A.2 Proof of Theorem 1

For clarity of the proof, we state an important stepping stone as a lemma.

A.2.1 Lemma

Lemma 8 *For $\epsilon>0$ and $\phi(\mathbf{x})$ a perturbation of the log-pdf $f_m(\mathbf{x})=\ln p_m(\mathbf{x})$,*

$$\begin{aligned}\tilde{J}(f_m+\epsilon\phi) &= \tilde{J}(f_m)+\epsilon\int[p_d(\mathbf{u})r_{\frac{1}{\nu}}(-f_m(\mathbf{u})+\ln p_n(\mathbf{u}))- \\ &\quad \nu p_n(\mathbf{u})r_\nu(f_m(\mathbf{u})-\ln p_n(\mathbf{u}))]\phi(\mathbf{u})\mathrm{d}\mathbf{u}- \\ &\quad \frac{\epsilon^2}{2}\int r_{\frac{1}{\nu}}(-f_m(\mathbf{u})+\ln p_n(\mathbf{u}))r_\nu(f_m(\mathbf{u})-\ln p_n(\mathbf{u})) \\ &\quad (p_d(\mathbf{u})+\nu p_n(\mathbf{u}))\phi(\mathbf{u})^2\mathrm{d}\mathbf{u}+O(\epsilon^3).\end{aligned}$$

Gateaux derivation

Proof making use of Eq (30) and (31) with $u=f_m(\mathbf{x})-\ln p_n(\mathbf{x})$, $u_1=\phi(\mathbf{x})$ and $u_2=0$.

■

A.2.2 Proof of the Theorem

Proof A necessary condition for optimality is that in the expansion of $J\tilde{}(f_m+\epsilon\phi)$, G derivation =0 iff

$$p_d(\mathbf{u})r_{\frac{1}{\nu}}(-f_m(\mathbf{u})+\ln p_n(\mathbf{u}))=\nu p_n(\mathbf{u})r_\nu(f_m(\mathbf{u})-\ln p_n(\mathbf{u})).$$

With Eq (28) and (29), this implies that $J\tilde{}$ has an extremum at p_m iff

$$\frac{\nu p_n(\mathbf{u})p_d(\mathbf{u})}{p_m(\mathbf{u})+\nu p_n(\mathbf{u})}=\frac{\nu p_n(\mathbf{u})p_m(\mathbf{u})}{p_m(\mathbf{u})+\nu p_n(\mathbf{u})}.$$

<=> $\nu>0, p_m(\mathbf{u})=p_d(\mathbf{u})$ where $p_n(\mathbf{u})\neq 0$. At points where $p_n(\mathbf{u})=0$, the eq is trivially fulfilled. Hence, $p_m=p_d$, or $f_m=\ln p_d$, leads to an extremum of $J\tilde{}$.

Inserting $f_m = \ln p_d$ into $\tilde{J}$ in Lemma 8 leads to

$$\tilde{J}(\ln p_d + \epsilon\phi) \quad = \quad \tilde{J}(\ln p_d) - \frac{\epsilon^2}{2}\left\{\int \frac{\nu p_n(\mathbf{u}) p_d(\mathbf{u})}{p_d(\mathbf{u}) + \nu p_n(\mathbf{u})} \phi(u)^2 \mathrm{d}\mathbf{u}\right\} + O(\epsilon^3).$$

Since the term of order ϵ^2 is negative for all choices of ϕ, the extremum is a maximum. The assumption that $p_n(\mathbf{u}) \not>> p_d(\mathbf{u})$ shows that $f_m = \ln p_d$ is the only extremum.

■

A.3 Proof of Theorem 2 (Consistency)

A.3.1 LEMMATA

The Taylor expansions in Eq (30) and (31) are used to prove the following lemma which is like Lemma 8 for $\tilde{J}$ but for the objective function J in Eq (10).

Lemma 9 *For $\epsilon > 0$ and $\boldsymbol{\varphi} \in \mathbb{R}^m$,*

$$\begin{aligned} J(\boldsymbol{\theta} + \epsilon\boldsymbol{\varphi}) \quad = \quad & J(\boldsymbol{\theta}) + \epsilon \int u_1 \left[p_d(\mathbf{u})(1 - h(\mathbf{u};\boldsymbol{\theta})) - \nu p_n(\mathbf{u}) h(\mathbf{u};\boldsymbol{\theta})\right] \mathrm{d}\mathbf{u} + \\ & \epsilon^2 \left\{ \int -\frac{1}{2} u_1^2 (1 - h(\mathbf{u};\boldsymbol{\theta})) h(\mathbf{u};\boldsymbol{\theta}) \left(p_d(\mathbf{u}) + \nu p_n(\mathbf{u})\right) \mathrm{d}\mathbf{u} + \right. \\ & \left. \int u_2 \left(p_d(\mathbf{u})(1 - h(\mathbf{u};\boldsymbol{\theta})) - \nu p_n(\mathbf{u}) h(\mathbf{u};\boldsymbol{\theta})\right) \mathrm{d}\mathbf{u} \right\} + O(\epsilon^3), \end{aligned}$$

where

$$\begin{aligned} u_1 \quad &= \quad \boldsymbol{\varphi}^T \mathbf{g}(\mathbf{u};\boldsymbol{\theta}), \\ u_2 \quad &= \quad \frac{1}{2} \boldsymbol{\varphi}^T \mathbf{H}_G(\mathbf{u};\boldsymbol{\theta}) \boldsymbol{\varphi}. \end{aligned}$$

$\mathbf{g}(\mathbf{u};\boldsymbol{\theta}) = \nabla G(\mathbf{u};\boldsymbol{\theta})$, and $\mathbf{H}_G$ = the Hessian matrix of $G(\mathbf{u};\boldsymbol{\theta})$ wrt $\boldsymbol{\theta}$.

Proof With the definition of J in Eq (10),

$$\begin{aligned} J(\boldsymbol{\theta} + \epsilon\boldsymbol{\varphi}) \quad = \quad & \int \ln\left[r_\nu\left(G(\mathbf{u};\boldsymbol{\theta} + \epsilon\boldsymbol{\varphi})\right)\right] p_d(\mathbf{u}) \mathrm{d}\mathbf{u} + \\ & \nu \int \ln\left[1 - r_\nu\left(G(\mathbf{u};\boldsymbol{\theta} + \epsilon\boldsymbol{\varphi})\right)\right] p_n(\mathbf{u}) \mathrm{d}\mathbf{u}. \end{aligned}$$

Developing $G(\mathbf{u};\boldsymbol{\theta} + \epsilon\boldsymbol{\varphi})$ till terms of order ϵ^2 yields

$$G(\mathbf{u};\boldsymbol{\theta} + \epsilon\boldsymbol{\varphi}) = G(\mathbf{u};\boldsymbol{\theta}) + \epsilon \boldsymbol{\varphi}^T \mathbf{g}(\mathbf{u};\boldsymbol{\theta}) + \epsilon^2 \frac{1}{2} \boldsymbol{\varphi}^T \mathbf{H}_G(\mathbf{u};\boldsymbol{\theta}) \boldsymbol{\varphi} + O(\epsilon^3).$$

Defining u_1 and u_2 as in the lemma, we obtain

$$\ln r_\nu\left(G(\mathbf{u};\boldsymbol{\theta} + \epsilon\boldsymbol{\phi})\right) = \ln r_\nu\left(G(\mathbf{u};\boldsymbol{\theta}) + \epsilon u_1 + \epsilon^2 u_2 + O(\epsilon^3)\right).$$

Using now the Taylor expansions in Eq (30) and (31) for $u = G(\mathbf{u};\boldsymbol{\theta})$,

■

Lemma 10 *If p_n >> p_d and*

$$\boldsymbol{\mathcal{I}}_\nu = \int \mathbf{g}(\mathbf{u})\mathbf{g}(\mathbf{u})^T P_\nu(\mathbf{u}) p_d(\mathbf{u}) \mathrm{d}\mathbf{u}$$

is full rank (PD), where

$$\begin{aligned} P_\nu(\mathbf{u}) &= \frac{\nu p_n(\mathbf{u})}{p_d(\mathbf{u}) + \nu p_n(\mathbf{u})}, \\ \mathbf{g}(\mathbf{u}) &= \nabla_{\boldsymbol{\theta}} \ln p_m(\mathbf{u};\boldsymbol{\theta})|_{\boldsymbol{\theta}=\boldsymbol{\theta}^\star}, \end{aligned}$$

then

$$J(\boldsymbol{\theta}^\star) > J(\boldsymbol{\theta}) \quad \boldsymbol{\theta} \neq \boldsymbol{\theta}^\star.$$

Proof A necessary condition for optimality is that in the expansion of $J(\boldsymbol{\theta} + \epsilon\boldsymbol{\varphi})$ in Lemma 9, the term of order $\epsilon = 0$ for any $\boldsymbol{\varphi}$ iff

$$p_d(\mathbf{u})(1 - h(\mathbf{u};\boldsymbol{\theta})) = \nu p_n(\mathbf{u}) h(\mathbf{u};\boldsymbol{\theta}),$$

<==>

$$\frac{\nu p_n(\mathbf{u}) p_d(\mathbf{u})}{p_m(\mathbf{u};\boldsymbol{\theta}) + \nu p_n(\mathbf{u})} = \frac{\nu p_n(\mathbf{u}) p_m(\mathbf{u};\boldsymbol{\theta})}{p_m(\mathbf{u};\boldsymbol{\theta}) + \nu p_n(\mathbf{u})},$$

where we have used Eq (28) and (29) as in the proof for Lemma 8. The assumption that $\nu > 0$ and $p_d(.) = p_m(.;\boldsymbol{\theta}^\star)$ together with the above equation => the term of order $\epsilon = 0$ if $\boldsymbol{\theta} = \boldsymbol{\theta}^\star$.

The objective function

$$\begin{aligned} J(\boldsymbol{\theta}^\star + \epsilon\boldsymbol{\varphi}) &= J(\boldsymbol{\theta}^\star) - \frac{\epsilon^2}{2} \int u_1^2 (1 - h(\mathbf{u};\boldsymbol{\theta}^\star)) h(\mathbf{u};\boldsymbol{\theta}^\star) \\ &\quad (p_d(\mathbf{u}) + \nu p_n(\mathbf{u}))\, \mathrm{d}\mathbf{u} + O(\epsilon^3). \end{aligned}$$

The terms $h(\mathbf{u};\boldsymbol{\theta}^\star)$ and $1 - h(\mathbf{u};\boldsymbol{\theta}^\star)$ are with Equation (27)

$$h(\mathbf{u};\boldsymbol{\theta}^\star) = \frac{p_d(\mathbf{u})}{p_d(\mathbf{u}) + \nu p_n(\mathbf{u})}, \qquad 1 - h(\mathbf{u};\boldsymbol{\theta}^\star) = \frac{\nu p_n(\mathbf{u})}{p_d(\mathbf{u}) + \nu p_n(\mathbf{u})}.$$

==>

$$J(\boldsymbol{\theta}^\star + \epsilon\boldsymbol{\varphi}) = J(\boldsymbol{\theta}^\star) - \frac{\epsilon^2}{2} \boldsymbol{\varphi}^T \left[\int \mathbf{g}(\mathbf{u})\mathbf{g}(\mathbf{u})^T P_\nu(\mathbf{u}) p_d(\mathbf{u}) \mathrm{d}\mathbf{u} \right] \boldsymbol{\varphi} + O(\epsilon^3)$$

by inserting the definition of u_1 evaluated at $\boldsymbol{\theta}^\star$. The term of order ϵ^2 defines the nature of the extremum at $\boldsymbol{\theta}^\star$. If $\boldsymbol{\mathcal{I}}_\nu > 0$, $J(\boldsymbol{\theta}^\star)$ is a maximum.

By definition of *J*, for a suitable noise density p_n , J attains a global maximum at $\boldsymbol{\theta}^\star$.

■

A.3.2 Proof of the Theorem

The proof of consistency goes along the same lines as MLE (Wasserman, 2004, Chapter 9).

Proof of consistency, given $\epsilon > 0$, $P(||\hat{\boldsymbol{\theta}}_T - \boldsymbol{\theta}^\star|| > \epsilon) \rightarrow 0$ as $T_d \rightarrow \infty$. In what follows, it is sometimes useful to make the underlying probability space explicit and write $P(||\hat{\boldsymbol{\theta}}_T - \boldsymbol{\theta}^\star|| > \epsilon)$.

by Lemma 10, $J(\hat{\boldsymbol{\theta}})$ is a global maximum, $||\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}|| > \epsilon$ ==> there is a $\delta(\epsilon)$ such that $J(\boldsymbol{\theta}) < J(\boldsymbol{\theta}^\star) - \delta(\epsilon)$. Hence,

$$\{||\hat{\boldsymbol{\theta}}_T - \boldsymbol{\theta}^\star|| > \epsilon\} \subset \{J(\hat{\boldsymbol{\theta}}_T) < J(\boldsymbol{\theta}^\star) - \delta(\epsilon)\}$$

and thus

$$P(||\hat{\boldsymbol{\theta}}_T - \boldsymbol{\theta}^\star|| > \epsilon) < P(J(\hat{\boldsymbol{\theta}}_T) < J(\boldsymbol{\theta}^\star) - \delta(\epsilon)). \tag{32}$$

Next, we investigate what happens to $P(J(\hat{\boldsymbol{\theta}}_T) < J(\boldsymbol{\theta}^\star) - \delta(\epsilon))$, $T_d \rightarrow \infty$.

Fact -

$$\begin{aligned} J(\boldsymbol{\theta}^\star) - J(\hat{\boldsymbol{\theta}}_T) &= J(\boldsymbol{\theta}^\star) - J_T(\boldsymbol{\theta}^\star) + J_T(\boldsymbol{\theta}^\star) - J(\hat{\boldsymbol{\theta}}_T) \\ &\leq J(\boldsymbol{\theta}^\star) - J_T(\boldsymbol{\theta}^\star) + J_T(\hat{\boldsymbol{\theta}}_T) - J(\hat{\boldsymbol{\theta}}_T) \end{aligned}$$

as $\hat{\boldsymbol{\theta}}_T$ maximizes J_T

==> $$|J(\boldsymbol{\theta}^\star) - J(\hat{\boldsymbol{\theta}}_T)| \leq 2 \sup_{\boldsymbol{\theta}} |J(\boldsymbol{\theta}) - J_T(\boldsymbol{\theta})|,$$

==> $$P(|J(\boldsymbol{\theta}^\star) - J(\hat{\boldsymbol{\theta}}_T)| > \delta) \leq P(2 \sup_{\boldsymbol{\theta}} |J(\boldsymbol{\theta}) - J_T(\boldsymbol{\theta})| > \delta).$$

Using the assumption that $J_T(\boldsymbol{\theta})$ converges in probability uniformly over $\boldsymbol{\theta}$ to $J(\boldsymbol{\theta})$, we obtain that for large T_d

$$P(|J(\boldsymbol{\theta}^\star) - J(\hat{\boldsymbol{\theta}}_T)| > \delta) < \epsilon_2$$

As $J(\boldsymbol{\theta}^\star) \geq J(\boldsymbol{\theta})$ for any $\boldsymbol{\theta}$, we have

$$P(J(\hat{\boldsymbol{\theta}}_T) < J(\boldsymbol{\theta}^\star) - \delta) < \epsilon_2$$

for any $\epsilon_2 > 0$.

■

A.4 Proof of Theorem 3 (Asymptotic Normality)

A.4.1 LEMMATA

Lemma 11

$$0 \quad = \quad \nabla_{\boldsymbol{\theta}} J_T(\boldsymbol{\theta}^\star) + \mathbf{H}_J(\boldsymbol{\theta}^\star)(\hat{\boldsymbol{\theta}}_T - \boldsymbol{\theta}^\star) + O(||\hat{\boldsymbol{\theta}}_T - \boldsymbol{\theta}^\star||^2)$$

where

$$\begin{aligned}
\nabla_{\boldsymbol{\theta}} J_T(\boldsymbol{\theta}^\star) &= \frac{1}{T_d}\sum_{t=1}^{T_d}(1-h(\mathbf{x}_t;\boldsymbol{\theta}^\star))\mathbf{g}(\mathbf{x}_t) - \nu\frac{1}{T_n}\sum_{t=1}^{T_n}h(\mathbf{y}_t;\boldsymbol{\theta}^\star)\mathbf{g}(\mathbf{y}_t), \\
\mathbf{H}_J(\boldsymbol{\theta}^\star) &= \frac{1}{T_d}\sum_{t=1}^{T_d}\Big\{-(1-h(\mathbf{x}_t;\boldsymbol{\theta}^\star))h(\mathbf{x}_t;\boldsymbol{\theta}^\star)\mathbf{g}(\mathbf{x}_t)\mathbf{g}(\mathbf{x}_t)^T + \\
&\quad (1-h(\mathbf{x}_t;\boldsymbol{\theta}^\star))\mathbf{H}_G(\mathbf{x}_t;\boldsymbol{\theta}^\star)\} - \\
&\quad \nu\frac{1}{T_n}\sum_{t=1}^{T_n}\Big\{(1-h(\mathbf{y}_t;\boldsymbol{\theta}^\star))h(\mathbf{y}_t;\boldsymbol{\theta}^\star)\mathbf{g}(\mathbf{y}_t)\mathbf{g}(\mathbf{y}_t)^T + \\
&\quad h(\mathbf{y}_t;\boldsymbol{\theta}^\star)\mathbf{H}_G(\mathbf{y}_t;\boldsymbol{\theta}^\star)\}.
\end{aligned}$$

Proof Using the chain rule, it follows from the relations in Section A.1 that

$$\begin{aligned}
\nabla_{\boldsymbol{\theta}} \ln h(\mathbf{x}_t;\boldsymbol{\theta}) &= (1-h(\mathbf{x}_t;\boldsymbol{\theta}))\mathbf{g}(\mathbf{x}_t;\boldsymbol{\theta}) \\
\nabla_{\boldsymbol{\theta}} \ln [1-h(\mathbf{y}_t;\boldsymbol{\theta})] &= -h(\mathbf{y}_t;\boldsymbol{\theta})\mathbf{g}(\mathbf{y}_t;\boldsymbol{\theta}).
\end{aligned}$$

==>

$$\boxed{\nabla_{\boldsymbol{\theta}} J_T(\boldsymbol{\theta}) = \frac{1}{T_d}\sum_{t=1}^{T_d}(1-h(\mathbf{x}_t;\boldsymbol{\theta}))\mathbf{g}(\mathbf{x};\boldsymbol{\theta}) - \nu\frac{1}{T_n}\sum_{t=1}^{T_n}h(\mathbf{y}_t;\boldsymbol{\theta})\mathbf{g}(\mathbf{y}_t;\boldsymbol{\theta}).}$$

As $\hat{\boldsymbol{\theta}}_T$ is the value of $\boldsymbol{\theta}$ which maximizes $J_T(\boldsymbol{\theta})$, must $= 0$. ==>

$$0 = \nabla_{\boldsymbol{\theta}} J_T(\boldsymbol{\theta}^\star) + \mathbf{H}_J(\boldsymbol{\theta}^\star)(\hat{\boldsymbol{\theta}^{\star}}_T - \boldsymbol{\theta}^\star) + O(||\hat{\boldsymbol{\theta}^{\star}}_T - \boldsymbol{\theta}^\star||^2). \quad /$$

The k-th row of the Hessian $\mathbf{H}_J(\boldsymbol{\theta})$ is $\nabla_{\boldsymbol{\theta}} F_k(\boldsymbol{\theta})^T$ where F_k is the k-th element of the vector $\nabla_{\boldsymbol{\theta}} J_T$. Denoting by g_k the k-th element of the score function $\mathbf{g}$, we have

$$\begin{aligned}
\nabla_{\boldsymbol{\theta}} F_k(\boldsymbol{\theta}) &= \frac{1}{T_d}\sum_{t=1}^{T_d}\{-\nabla_{\boldsymbol{\theta}} h(\mathbf{x}_t;\boldsymbol{\theta})g_k(\mathbf{x}_t;\boldsymbol{\theta}) + (1-h(\mathbf{x}_t;\boldsymbol{\theta}))\nabla_{\boldsymbol{\theta}} g_k(\mathbf{x}_t;\boldsymbol{\theta})\} \\
&\quad -\nu\frac{1}{T_n}\sum_{t=1}^{T_n}\{\nabla_{\boldsymbol{\theta}} h(\mathbf{y}_t;\boldsymbol{\theta})g_k(\mathbf{y}_t;\boldsymbol{\theta}) + h(\mathbf{y}_t;\boldsymbol{\theta})\nabla_{\boldsymbol{\theta}} g_k(\mathbf{x}_t;\boldsymbol{\theta})\}.
\end{aligned}$$

Using the chain rule, it follows from the relations in Section A.1 that

$$\nabla_{\boldsymbol{\theta}} h(\mathbf{u};\boldsymbol{\theta}) \;=\; (1-h(\mathbf{u};\boldsymbol{\theta}))h(\mathbf{u};\boldsymbol{\theta})\mathbf{g}(\mathbf{u};\boldsymbol{\theta}).$$

Hence,

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} F_k(\boldsymbol{\theta}) \;=\; & \frac{1}{T_d}\sum_{t=1}^{T_d}\{-(1-h(\mathbf{x}_t;\boldsymbol{\theta}))h(\mathbf{x}_t;\boldsymbol{\theta})\mathbf{g}(\mathbf{x}_t;\boldsymbol{\theta})g_k(\mathbf{x}_t;\boldsymbol{\theta})+\\ & (1-h(\mathbf{x}_t;\boldsymbol{\theta}))\nabla_{\boldsymbol{\theta}} g_k(\mathbf{x}_t;\boldsymbol{\theta})\}-\\ & \nu\frac{1}{T_n}\sum_{t=1}^{T_n}\{(1-h(\mathbf{y}_t;\boldsymbol{\theta}))h(\mathbf{y}_t;\boldsymbol{\theta})\mathbf{g}(\mathbf{y}_t;\boldsymbol{\theta})g_k(\mathbf{y}_t;\boldsymbol{\theta})+\\ & h(\mathbf{y}_t;\boldsymbol{\theta})\nabla_{\boldsymbol{\theta}} g_k(\mathbf{y}_t;\boldsymbol{\theta})\},\end{aligned}$$

■

Lemma 12 *$\mathbf{H}_J(\boldsymbol{\theta}^\star)$ converges in probability to $-\boldsymbol{\mathcal{I}}_\nu$, the sample size $T_d \to\infty$.*

Proof $T_n = \nu T_d \to \infty$. As the sample sizes become arbitrarily large, the sample averages become integration over the corresponding densities so that

$$\begin{aligned}\lim_{T_d\to\infty}\mathbf{H}_J(\boldsymbol{\theta}^\star) \;\xrightarrow{P}\; & -\int(1-h(\mathbf{u};\boldsymbol{\theta}^\star))h(\mathbf{u};\boldsymbol{\theta}^\star)\mathbf{g}(\mathbf{u})\mathbf{g}(\mathbf{u})^T(p_d(\mathbf{u})+\nu p_n(\mathbf{u}))\mathrm{d}\mathbf{u}+\\ & \int((1-h(\mathbf{u};\boldsymbol{\theta}^\star))p_d(\mathbf{u})-h(\mathbf{u};\boldsymbol{\theta}^\star)\nu p_n(\mathbf{u}))\,\mathbf{H}_G(\mathbf{u};\boldsymbol{\theta}^\star)\mathrm{d}\mathbf{u}.\end{aligned}$$

With Eq (28) and (29), we have

$$\begin{aligned}(1-h(\mathbf{u};\boldsymbol{\theta}^\star))p_d(\mathbf{u}) \;&=\; h(\mathbf{u};\boldsymbol{\theta}^\star)\nu p_n(\mathbf{u}), \qquad (33)\\ (1-h(\mathbf{u};\boldsymbol{\theta}^\star))h(\mathbf{u};\boldsymbol{\theta}^\star)(p_d(\mathbf{u})+\nu p_n(\mathbf{u})) \;&=\; \frac{\nu p_n(\mathbf{u})p_d(\mathbf{u})}{p_d(\mathbf{u})+\nu p_n(\mathbf{u})}.\end{aligned}$$

Hence,

$$\lim_{T_d\to\infty}\mathbf{H}_J(\boldsymbol{\theta}^\star) \;\xrightarrow{P}\; -\int\frac{\nu p_n(\mathbf{u})p_d(\mathbf{u})}{p_d(\mathbf{u})+\nu p_n(\mathbf{u})}\mathbf{g}(\mathbf{u})\mathbf{g}(\mathbf{u})^T\mathrm{d}\mathbf{u},$$

which is $-\boldsymbol{\mathcal{I}}_\nu$.

■

Lemma 13 $\mathrm{E}\nabla_{\boldsymbol{\theta}} J_T(\boldsymbol{\theta}^\star) = 0$.

Proof

$$\begin{aligned}
\mathrm{E}\nabla_{\boldsymbol{\theta}} J_T(\boldsymbol{\theta}^\star) &= \frac{1}{T_d}\sum_{t=1}^{T_d} \mathrm{E}\,\mathbf{g}(\mathbf{x}_t)(1-h(\mathbf{x}_t;\boldsymbol{\theta}^\star)) - \nu\frac{1}{T_n}\sum_{t=1}^{T_n}\mathrm{E}\,\mathbf{g}(\mathbf{y}_t)h(\mathbf{y}_t;\boldsymbol{\theta}^\star) \\
&= \mathrm{E}\,\mathbf{g}(\mathbf{x})(1-h(\mathbf{x};\boldsymbol{\theta}^\star)) - \nu\,\mathrm{E}\,\mathbf{g}(\mathbf{y})h(\mathbf{y};\boldsymbol{\theta}^\star) \\
&= \int \mathbf{g}(\mathbf{u})(1-h(\mathbf{u};\boldsymbol{\theta}^\star))p_d(\mathbf{u})\mathrm{d}\mathbf{u} - \nu\int \mathbf{g}(\mathbf{u})h(\mathbf{u};\boldsymbol{\theta}^\star)p_n(\mathbf{u})\mathrm{d}\mathbf{u},
\end{aligned}$$

from the i.i.d. assumption of the sample X and Y. =>

$$\mathrm{E}\nabla_{\boldsymbol{\theta}} J_T(\boldsymbol{\theta}^\star) = \int \mathbf{g}(\mathbf{u})\left((1-h(\mathbf{u};\boldsymbol{\theta}^\star))p_d(\mathbf{u}) - h(\mathbf{u};\boldsymbol{\theta}^\star)\nu p_n(\mathbf{u})\right)\mathrm{d}\mathbf{u},$$

■

Lemma 14 $\mathrm{Var}\,\nabla_{\boldsymbol{\theta}} J_T(\boldsymbol{\theta}^\star) = \frac{1}{T_d}\left(\boldsymbol{\mathcal{I}}_\nu - \left(1+\frac{1}{\nu}\right)\mathrm{E}(P_\nu\mathbf{g})\,\mathrm{E}(P_\nu\mathbf{g})^T\right)$,

where $\boldsymbol{\mathcal{I}}_\nu$, P_ν and $\mathbf{g}$ were defined in Lemma 10, and the expectation is taken over the data-pdf p_d.

Proof As $\mathrm{E}\nabla_{\boldsymbol{\theta}} J_T(\boldsymbol{\theta}^\star) = 0$, the variance is $\mathrm{E}\nabla_{\boldsymbol{\theta}} J_T(\boldsymbol{\theta}^\star)\nabla_{\boldsymbol{\theta}} J_T(\boldsymbol{\theta}^\star)^T$. Multiplying out gives

$$\begin{aligned}
\mathrm{Var}\,\nabla_{\boldsymbol{\theta}} J_T(\boldsymbol{\theta}^\star) &= \frac{1}{T_d^2}\mathrm{E}\left[\sum_{t=1}^{T_d}(1-h(\mathbf{x}_t;\boldsymbol{\theta}^\star))\mathbf{g}(\mathbf{x}_t)\sum_{t=1}^{T_d}(1-h(\mathbf{x}_t;\boldsymbol{\theta}^\star))\mathbf{g}(\mathbf{x}_t)^T\right] - \\
&\quad \frac{1}{T_d^2}\mathrm{E}\left[\sum_{t=1}^{T_d}(1-h(\mathbf{x}_t;\boldsymbol{\theta}^\star))\mathbf{g}(\mathbf{x}_t)\sum_{t=1}^{T_n}h(\mathbf{y}_t;\boldsymbol{\theta}^\star)\mathbf{g}(\mathbf{y}_t)^T\right] - \\
&\quad \frac{1}{T_d^2}\mathrm{E}\left[\sum_{t=1}^{T_n}h(\mathbf{y}_t;\boldsymbol{\theta}^\star)\mathbf{g}(\mathbf{y}_t)\sum_{t=1}^{T_d}(1-h(\mathbf{x}_t;\boldsymbol{\theta}^\star))\mathbf{g}(\mathbf{x}_t)^T\right] + \\
&\quad \frac{1}{T_d^2}\mathrm{E}\left[\sum_{t=1}^{T_n}h(\mathbf{y}_t;\boldsymbol{\theta}^\star)\mathbf{g}(\mathbf{y}_t)\sum_{t=1}^{T_n}h(\mathbf{y}_t;\boldsymbol{\theta}^\star)\mathbf{g}(\mathbf{y}_t)^T\right].
\end{aligned}$$

Since the samples are all independent from each other, we have

$$\begin{aligned}
\mathrm{Var}\,\nabla_{\boldsymbol{\theta}} J_T(\boldsymbol{\theta}^\star) &= \frac{1}{T_d^2}\sum_{t=1}^{T_d}\mathrm{E}\left[(1-h(\mathbf{x}_t;\boldsymbol{\theta}^\star))^2\mathbf{g}(\mathbf{x}_t)\mathbf{g}(\mathbf{x}_t)^T\right] + \\
&\quad \frac{1}{T_d^2}\sum_{\substack{t,\tau=1\\ t\neq\tau}}^{T_d}\mathrm{E}\left[(1-h(\mathbf{x}_t;\boldsymbol{\theta}^\star))\mathbf{g}(\mathbf{x}_t)\right]\mathrm{E}\left[(1-h(\mathbf{x}_\tau;\boldsymbol{\theta}^\star))\mathbf{g}(\mathbf{x}_\tau)^T\right] - \\
&\quad \frac{1}{T_d^2}\sum_{t=1}^{T_d}\sum_{\tau=1}^{T_n}\mathrm{E}\left[(1-h(\mathbf{x}_t;\boldsymbol{\theta}^\star))\mathbf{g}(\mathbf{x}_t)\right]\mathrm{E}\left[h(\mathbf{y}_\tau;\boldsymbol{\theta}^\star)\mathbf{g}(\mathbf{y}_\tau)^T\right] - \\
&\quad \frac{1}{T_d^2}\sum_{t=1}^{T_n}\sum_{\tau=1}^{T_d}\mathrm{E}\left[h(\mathbf{y}_t;\boldsymbol{\theta}^\star)\mathbf{g}(\mathbf{y}_t)\right]\mathrm{E}\left[(1-h(\mathbf{x}_\tau;\boldsymbol{\theta}^\star))\mathbf{g}(\mathbf{x}_\tau)^T\right] + \\
&\quad \frac{1}{T_d^2}\sum_{\substack{t,\tau=1\\ t\neq\tau}}^{T_n}\mathrm{E}\left[h(\mathbf{y}_t;\boldsymbol{\theta}^\star)\mathbf{g}(\mathbf{y}_t)\right]\mathrm{E}\left[h(\mathbf{y}_\tau;\boldsymbol{\theta}^\star)\mathbf{g}(\mathbf{y}_\tau)^T\right] + \\
&\quad \frac{1}{T_d^2}\sum_{t=1}^{T_n}\mathrm{E}\left[h(\mathbf{y}_t;\boldsymbol{\theta}^\star)^2\mathbf{g}(\mathbf{y}_t)\mathbf{g}(\mathbf{y}_t)^T\right].
\end{aligned}$$

As we assume that all $\mathbf{x}_t$, and also $\mathbf{y}_t$, are identically distributed, the above expression simplifies to

$$\begin{aligned}
\mathrm{Var}\,\nabla_{\boldsymbol{\theta}} J_T(\boldsymbol{\theta}^\star) &= \frac{1}{T_d}\int(1-h(\mathbf{u};\boldsymbol{\theta}^\star))^2\mathbf{g}(\mathbf{u})\mathbf{g}(\mathbf{u})^T p_d(\mathbf{u})\mathrm{d}\mathbf{u} + \\
&\quad \frac{T_d^2-T_d}{T_d^2}\mathbf{m}_x\mathbf{m}_x^T - \frac{T_dT_n}{T_d^2}\mathbf{m}_x\mathbf{m}_y^T - \\
&\quad \frac{T_dT_n}{T_d^2}\mathbf{m}_y\mathbf{m}_x^T + \frac{T_n^2-T_n}{T_d^2}\mathbf{m}_y\mathbf{m}_y^T + \\
&\quad \frac{T_n}{T_d^2}\int h(\mathbf{u};\boldsymbol{\theta}^\star)^2\mathbf{g}(\mathbf{u})\mathbf{g}(\mathbf{u})^T p_n(\mathbf{u})\mathrm{d}\mathbf{u}, \qquad (34)
\end{aligned}$$

where

$$\begin{aligned}
\mathbf{m}_x &= \int(1-h(\mathbf{u};\boldsymbol{\theta}^\star))\mathbf{g}(\mathbf{u})p_d(\mathbf{u})\mathrm{d}\mathbf{u}, \\
\mathbf{m}_y &= \int h(\mathbf{u};\boldsymbol{\theta}^\star)\mathbf{g}(\mathbf{u})p_n(\mathbf{u})\mathrm{d}\mathbf{u}.
\end{aligned}$$

Denoting by A the sum of the first and last line of Eq (34), we have

$$A = \frac{1}{T_d}\int\mathbf{g}(\mathbf{u})\mathbf{g}(\mathbf{u})^T\left[(1-h(\mathbf{u};\boldsymbol{\theta}^\star))^2 p_d(\mathbf{u}) + h(\mathbf{u};\boldsymbol{\theta}^\star)^2\nu p_n(\mathbf{u})\right]\mathrm{d}\mathbf{u}$$

since $T_n = \nu T_d$. Now, Eq (27) and $p_m(\mathbf{u};\boldsymbol{\theta}^\star) = p_d(\mathbf{u})$ ==>

$$\begin{aligned}
(1-h(\mathbf{u};\boldsymbol{\theta}^\star))^2 p_d(\mathbf{u}) + h(\mathbf{u};\boldsymbol{\theta}^\star)^2\nu p_n(\mathbf{u}) &= \frac{\nu p_n(\mathbf{u})p_d(\mathbf{u})}{p_d(\mathbf{u})+\nu p_n(\mathbf{u})} \\
&= P_\nu p_d(\mathbf{u}),
\end{aligned}$$

==>

$$\begin{aligned} A &= \frac{1}{T_d}\int \mathbf{g}(\mathbf{u})\mathbf{g}(\mathbf{u})^T P_\nu p_d(\mathbf{u})\mathrm{d}\mathbf{u} \\ &= \frac{1}{T_d}\boldsymbol{\mathcal{I}}_\nu. \end{aligned}$$

Denote by B the second line of Eq (34). Rearranging the terms, we have

$$\begin{aligned} B &= \mathbf{m}_x \int \left[(1-h(\mathbf{u};\boldsymbol{\theta}^\star))p_d(\mathbf{u}) - h(\mathbf{u};\boldsymbol{\theta}^\star)\nu p_n(\mathbf{u})\right]\mathbf{g}(\mathbf{u})^T\mathrm{d}\mathbf{u} - \\ &\quad \frac{1}{T_d}\mathbf{m}_x\mathbf{m}_x^T. \end{aligned} \tag{35}$$

Again, Eq (27) and $p_m(\mathbf{u};\boldsymbol{\theta}^\star) = p_d(\mathbf{u})$ ==>

$$\begin{aligned} (1-h(\mathbf{u};\boldsymbol{\theta}^\star))p_d(\mathbf{u}) &= h(\mathbf{u};\boldsymbol{\theta}^\star)\nu p_n(\mathbf{u}) \\ &= \frac{\nu p_n(\mathbf{u})p_d(\mathbf{u})}{p_d(\mathbf{u})+\nu p_n(\mathbf{u})} \\ &= P_\nu p_d(\mathbf{u}), \end{aligned}$$

so that the first line in Eq (35) is zero and

$$\mathbf{m}_x = \int P_\nu \mathbf{g}(\mathbf{u})p_d(\mathbf{u})\mathrm{d}\mathbf{u}.$$

Thus

$$B = -\frac{1}{T_d}\int P_\nu \mathbf{g}(\mathbf{u})p_d(\mathbf{u})\mathrm{d}\mathbf{u}\int P_\nu \mathbf{g}(\mathbf{u})^T p_d(\mathbf{u})\mathrm{d}\mathbf{u}.$$

Denote by C the third line of Eq (34). Rearranging the terms, we have with $T_n = \nu T_d$

$$C = -\frac{\nu}{T_d}\mathbf{m}_y\mathbf{m}_y^T + \nu\mathbf{m}_y(\nu\mathbf{m}_y^T - \mathbf{m}_x^T).$$

The term $\nu\mathbf{m}_y$ is with Equation (27) and $p_m(\mathbf{u};\boldsymbol{\theta}^\star) = p_d(\mathbf{u})$

$$\nu\mathbf{m}_y = \int P_\nu \mathbf{g}(\mathbf{u})p_d(\mathbf{u})\mathrm{d}\mathbf{u},$$

so that $\nu\mathbf{m}_y = \mathbf{m}_x$, and hence

$$\begin{aligned} C &= -\frac{1}{\nu T_d}(\nu\mathbf{m}_y)(\nu\mathbf{m}_y^T) \\ &= \frac{1}{\nu}B. \end{aligned}$$

All in all, the variance $\mathrm{Var}\,\nabla_{\boldsymbol{\theta}} J_T(\boldsymbol{\theta}^\star)$ is thus

$$\begin{aligned} \mathrm{Var}\,\nabla_{\boldsymbol{\theta}} J_T(\boldsymbol{\theta}^\star) &= A+B+C \\ &= \frac{1}{T_d}\left(\boldsymbol{\mathcal{I}}_\nu - \left(1+\frac{1}{\nu}\right)\mathrm{E}\left(P_\nu\mathbf{g}\right)\mathrm{E}\left(P_\nu\mathbf{g}^T\right)\right), \end{aligned}$$

where

$$\mathrm{E}\left(P_\nu\mathbf{g}\right) = \int P_\nu \mathbf{g}(\mathbf{u})p_d(\mathbf{u})\mathrm{d}\mathbf{u}.$$

■

A.4.2 PROOF OF THE THEOREM

We are now ready to give the proof of Theorem 3.

Proof Up to terms of order $O(||\hat{\boldsymbol{\theta}}_T - \boldsymbol{\theta}^\star||^2)$, we have with Lemma 11

$$\sqrt{T_d}(\hat{\boldsymbol{\theta}}_T - \boldsymbol{\theta}^\star) = -\mathbf{H}_J^{-1}\sqrt{T_d}\nabla_{\boldsymbol{\theta}} J_T(\boldsymbol{\theta}^\star).$$

By Lemma 12, $\mathbf{H}_J \xrightarrow{P} -\boldsymbol{\mathcal{I}}_\nu$ for large sample sizes T_d. Using Lemma 13 and Lemma 14, we see that

$$\sqrt{T_d}\nabla_{\boldsymbol{\theta}} J_T(\boldsymbol{\theta}^\star)$$

converges in distribution to a normal distribution of mean zero and covariance matrix

$$\boldsymbol{\mathcal{I}}_\nu - \left(1 + \frac{1}{\nu}\right) \mathrm{E}(P_\nu \mathbf{g})\, \mathrm{E}(P_\nu \mathbf{g})^T,$$

which implies that $\sqrt{T_d}(\hat{\boldsymbol{\theta}}_T - \boldsymbol{\theta}^\star)$ converges in distribution to a normal distribution of mean zero and covariance matrix $\boldsymbol{\Sigma}$,

$$\boldsymbol{\Sigma} = \boldsymbol{\mathcal{I}}_\nu^{-1} - \left(1 + \frac{1}{\nu}\right) \boldsymbol{\mathcal{I}}_\nu^{-1}\, \mathrm{E}(P_\nu \mathbf{g})\, \mathrm{E}(P_\nu \mathbf{g})^T \boldsymbol{\mathcal{I}}_\nu^{-1}.$$

■

Appendix B. Calculations

The following sections contain calculations needed in Section 3.3 and Section 5.3.

B.1 Theory, Section 3.3: Asymptotic Variance for Orthogonal ICA Model

We calculate here the asymptotic covariance matrix of the estimation error for an orthogonal ICA model when a Gaussian distribution is used as noise distribution in NCE. This result is used to make the predictions about the estimation error in Section 3.3. The calculations show that the asymptotic variance does not depend on the mixing matrix but only on the dim of the data. Simila calculations can be used to show that this also holds for MLE.

A rv $\mathbf{x}$ following an ICA model with orthogonal mixing matrix $\mathbf{A} = (\mathbf{a}_1 \dots \mathbf{a}_n)$ has the distribution:

$$p_d(\mathbf{x}) = \frac{1}{Z} \prod_{i=1}^{n} f(\mathbf{a}_i^T \mathbf{x}),$$

where Z is the partition function. By orthogonality of $\mathbf{A}$,

$$p_d(\mathbf{A}\mathbf{x}) = \frac{1}{Z} \prod_{i=1}^{n} f(x_i),$$

$= p_s(\mathbf{x})$ the distribution of the sources $\mathbf{s}$ of the ICA model. Also by orthogonality of $\mathbf{A}$, the noise distribution p_n with the same covariance as $\mathbf{x}$ is the snd. In particular, $p_n(\mathbf{A}\mathbf{x}) = p_n(\mathbf{x})$.

For the calculation of the asymptotic variance, we need to compute the matrix $\boldsymbol{\mathcal{I}}_\nu$ which occurs in Theorem 2, $\boldsymbol{\mathcal{I}}_\nu = \int \mathbf{g}(\mathbf{u})\mathbf{g}(\mathbf{u})^T P_\nu(\mathbf{u}) p_d(\mathbf{u}) \mathrm{d}\mathbf{u}$. With the above data and noise distribution, $P_\nu(\mathbf{u})$ has the property that

$$\begin{aligned} P_\nu(\mathbf{Au}) &= \frac{\nu p_n(\mathbf{Au})}{p_d(\mathbf{Au}) + \nu p_n(\mathbf{Au})} \\ &= \frac{\nu p_n(\mathbf{u})}{p_s(\mathbf{u}) + \nu p_n(\mathbf{u})}. \end{aligned}$$

Hence $P_\nu(\mathbf{Au})$ does not depend on $\mathbf{A}$. Below, we will denote $P_\nu(\mathbf{Au})$ by $\tilde{P}_\nu(\mathbf{u})$. For the ICA model, the vector $\mathbf{g}(\mathbf{u})$ has the form

$$\mathbf{g}(\mathbf{u}) = (\mathbf{g}_1(\mathbf{u}), \ldots, \mathbf{g}_n(\mathbf{u}), g_c(\mathbf{u}))^T$$

where $\mathbf{g}_i(\mathbf{u}) = \nabla_{\mathbf{a}_i} \ln p_m(\mathbf{u}) = f'(\mathbf{a}_i^T \mathbf{u})\mathbf{u}$ and $g_c(\mathbf{u}) = \partial_c \ln p_m(\mathbf{u}) = 1$. By orthogonality of $\mathbf{A}$, we have

$$\mathbf{g}_i(\mathbf{Au}) = \mathbf{A} f'(u_i)\mathbf{u}.$$

We denote the vector $f'(u_i)\mathbf{u}$ by $\tilde{\mathbf{g}}_i(\mathbf{u})$ so that $\mathbf{g}_i(\mathbf{Au}) = \mathbf{A}\tilde{\mathbf{g}}_i(\mathbf{u})$. Hence,

$$\mathbf{g}(\mathbf{Au}) = \boldsymbol{\mathcal{A}}(\tilde{\mathbf{g}}_1(\mathbf{u}), \ldots, \tilde{\mathbf{g}}_n(\mathbf{u}), 1)^T$$

where $\boldsymbol{\mathcal{A}}$ is a block-diagonal matrix with n matrices $\mathbf{A}$ on the diagonal and a single 1 in the $(n+1)$-th slot. As a shorthand, we will denote $\mathbf{g}(\mathbf{Au})$ by $\boldsymbol{\mathcal{A}}\tilde{\mathbf{g}}(\mathbf{u})$.

With these preliminaries, using the change of variables $\mathbf{u} = \mathbf{Av}$,

$$\begin{aligned} \boldsymbol{\mathcal{I}}_\nu &= \int p_d(\mathbf{u})\mathbf{g}(\mathbf{u})\mathbf{g}(\mathbf{u})^T P_\nu(\mathbf{u})\mathrm{d}\mathbf{u} \\ &= \int p_s(\mathbf{v})\boldsymbol{\mathcal{A}}\tilde{\mathbf{g}}(\mathbf{v})\tilde{\mathbf{g}}(\mathbf{v})^T \boldsymbol{\mathcal{A}}^T \tilde{P}_\nu(\mathbf{v})\mathrm{d}\mathbf{v} \\ &= \boldsymbol{\mathcal{A}}\tilde{\boldsymbol{\mathcal{I}}}_\nu \boldsymbol{\mathcal{A}}^T, \end{aligned}$$

where the matrix

$$\tilde{\boldsymbol{\mathcal{I}}}_\nu = \int p_s(\mathbf{v})\tilde{\mathbf{g}}(\mathbf{v})\tilde{\mathbf{g}}(\mathbf{v})^T \tilde{P}_\nu(\mathbf{v})\mathrm{d}\mathbf{v}$$

does not depend on the mixing matrix $\mathbf{A}$ but only on the distribution of the sources $\mathbf{s}$, the noise distribution p_n, and ν. Moreover, by orthogonality of $\mathbf{A}$, the inverse of $\boldsymbol{\mathcal{I}}_\nu$ is given by

$$\boldsymbol{\mathcal{I}}_\nu^{-1} = \boldsymbol{\mathcal{A}}\tilde{\boldsymbol{\mathcal{I}}}_\nu^{-1}\boldsymbol{\mathcal{A}}^T.$$

The same reasoning shows that

$$\int p_d(\mathbf{u})P_\nu(\mathbf{u})\mathbf{g}(\mathbf{u})\mathrm{d}\mathbf{u} = \boldsymbol{\mathcal{A}}\int p_s(\mathbf{v})\tilde{\mathbf{g}}(\mathbf{v})\tilde{P}_\nu(\mathbf{v})\mathrm{d}\mathbf{v},$$

which we will denote below by $\boldsymbol{\mathcal{A}}\tilde{\mathbf{m}}$. Again, $\tilde{\mathbf{m}}$ does not depend on $\mathbf{A}$. Hence, the asymptotic covariance matrix $\boldsymbol{\Sigma}$,

$$\boldsymbol{\Sigma} = \boldsymbol{\mathcal{I}}_\nu^{-1} - \left(1 + \frac{1}{\nu}\right)\boldsymbol{\mathcal{I}}_\nu^{-1}\,\mathrm{E}(P_\nu \mathbf{g})\,\mathrm{E}(P_\nu \mathbf{g})^T \boldsymbol{\mathcal{I}}_\nu^{-1},$$

in Theorem 3 is for the ICA model with orthogonal mixing matrix $\mathbf{A}$ given by

$$\mathbf{\Sigma}_{\text{ortICA}} = \boldsymbol{\mathcal{A}} \left[\tilde{\boldsymbol{\mathcal{I}}}_{\nu}^{-1} - \left(1 + \frac{1}{\nu}\right) \tilde{\boldsymbol{\mathcal{I}}}_{\nu}^{-1} \tilde{\mathbf{m}} \tilde{\mathbf{m}}^T \tilde{\boldsymbol{\mathcal{I}}}_{\nu}^{-1} \right] \boldsymbol{\mathcal{A}}^T.$$

The block matrix $\boldsymbol{\mathcal{A}}$ is orthogonal since $\mathbf{A}$ is orthogonal. The asymptotic variance, that is the trace of $\mathbf{\Sigma}_{\text{ortICA}}$, does hence not depend on $\mathbf{A}$.

B.2 Natural Images, Section 5.3: Optimal Stimuli

We show here that the optimal stimulus, namely the image which yields the largest feature output for feature $\mathbf{w}$ while satisfying the sphere constraints in Eq (22), is proportional to $\mathbf{V}^-(\mathbf{w} - \langle \mathbf{w} \rangle)$.

$\langle \mathbf{w} \rangle$:= the average value of the elements $\mathbf{w}$.

Each coordinate vector $\mathbf{x}$ defines an image $\mathbf{i} = \mathbf{V}^-\mathbf{x}$, see Eq (23). The optimal image is thus $\mathbf{i}^* = \mathbf{V}^-\mathbf{x}^*$ where $\mathbf{x}^*$ is the solution to the optimization problem

$$\max_{\mathbf{x}} \mathbf{w}^T\mathbf{x}$$

subject to $\sum_{k=1}^n \mathbf{x}(k) = 0$ and $1/(n-1)\sum_{k=1}^n \mathbf{x}(k)^2 = 1$, which are the constraints in Equation (22). The Lagrangian associated with this constrained optimization problem is

$$L(\mathbf{x}, \lambda, \omega) = \mathbf{w}^T\mathbf{x} - \lambda \left(\frac{1}{n-1} \sum_{k=1}^n \mathbf{x}(k)^2 - 1 \right) - \omega \sum_{k=1}^n \mathbf{x}(k)$$

The maximizing $\mathbf{x}^*$ is $\mathbf{x}^* = (n-1)/(2\lambda)(\mathbf{w} - \omega)$. Taking ω such that the constraint $\sum_{k=1}^n \mathbf{x}^*(k) = 0$ is fulfilled gives

$$\mathbf{x}^* = \frac{n-1}{2\lambda}(\mathbf{w} - \langle \mathbf{w} \rangle).$$

Hence, the optimal image $\mathbf{i}^*$ is proportional to $\mathbf{V}^-(\mathbf{w} - \langle \mathbf{w} \rangle)$.

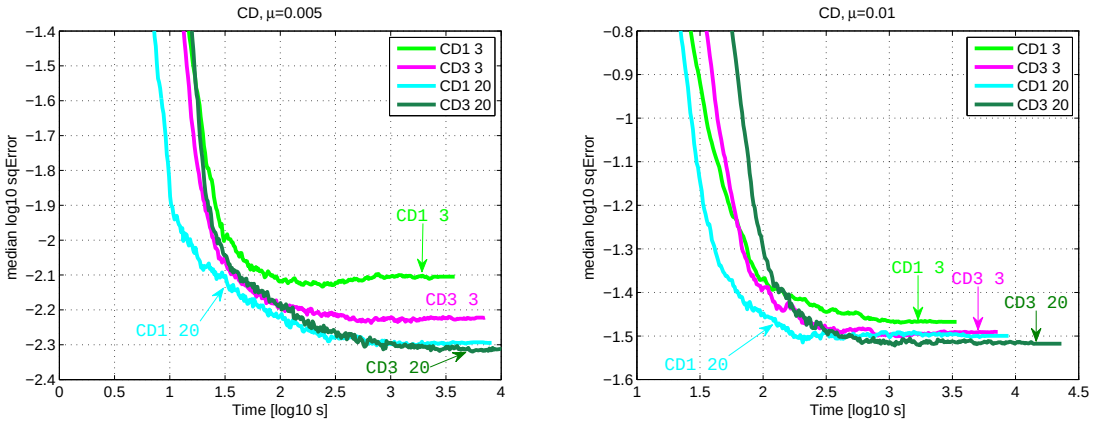
If we had a norm constraint on $\mathbf{i}$ instead of the constraints in Eq (22), the Lagrangian would be

$$\tilde{L}(\mathbf{x}, \lambda) = \mathbf{w}^T\mathbf{x} - \lambda \left(\sum_{k=1}^n \mathbf{x}(k)^2 d_k - 1 \right)$$

where we have used that $\mathbf{i}^T\mathbf{i} = \mathbf{x}^T\mathbf{V}^{-T}\mathbf{V}^-\mathbf{x} = \mathbf{x}^T\mathbf{D}\mathbf{x}$. The $n \times n$ matrix $\mathbf{D}$ is diagonal with the eigenvalue d_k of the covariance matrix of the natural image patches as k-th element. The optimal $\mathbf{x}$ would thus be $\tilde{\mathbf{x}}^* = 1/(2\lambda)\mathbf{D}^{-1}\mathbf{w}$ so that the optimal image $\tilde{\mathbf{i}}^*$ would be proportional to $\mathbf{V}^-\mathbf{D}^{-1}\mathbf{w} = \mathbf{E}\mathbf{D}^{-1/2}\mathbf{w} = \mathbf{V}^T\mathbf{w}$, for which we have used the notation $\tilde{\mathbf{w}}$ in Section 5.3. Since the eigenvalues d_k fall off with the spatial frequency f (like $1/f^2$, see for example Hyvärinen et al., 2009, Chapter 5.6) the norm constraint on $\mathbf{i}$ punishes low frequencies more heavily than the constraints in Equation (22). As a consequence, the $\tilde{\mathbf{w}}$, which are shown in Figure 11(a), are tuned to high frequencies while the optimal stimuli $\mathbf{i}^*$, shown in Figure 11(b), contain more low frequency components.

Appendix C. Further Simulation Results

The following sections contain additional simulation results related to Section 4 and 5.



(a) Laplacian sources

(b) Logistic sources

Figure 18: Trade-off between statistical and computational performance for CD.While the algorithms were running , measurements of the estimation error at a given time were made . The time variable indicates thus the time since the algorithm was started. Note the difference to Figure 6 where the time indicates the time-till- convergence . The plots show the median performance over the 100 estimation problems. CDx $y$refers to CD with x Monte Carlo steps, each using y leapfrog steps.

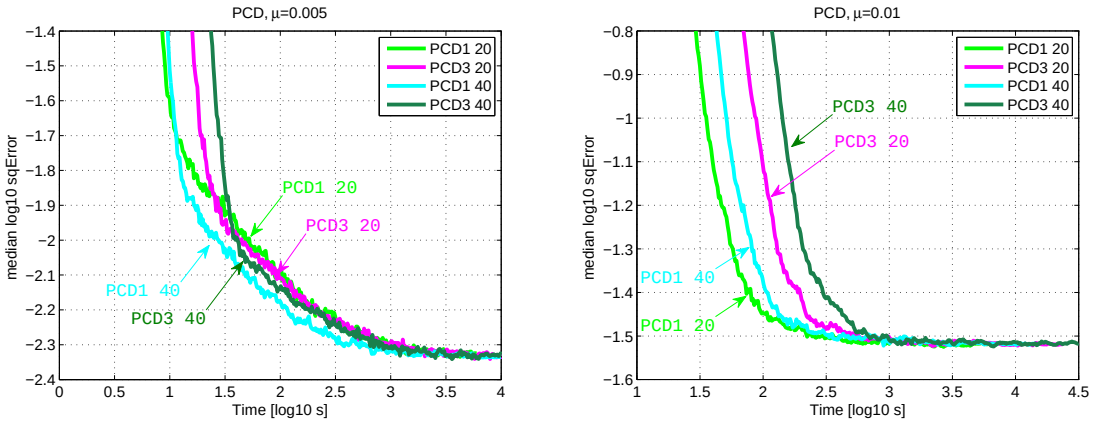
C.1 Trade-Off, Section 4: Comparison of the Different Settings of Contrastive and Persistent Contrastive Divergence

We compare here the different settings of contrastive and persistent contrastive divergence. Since the two estimations methods do not have an objective function, and given the randomness that is introduced by the minibatches, choosing a reliable stopping criterion is difficult. Hence, we did not impose any stopping criterion but the maximal number of iterations. The algorithms had always converged before this maximal number of iterations was reached, in the sense that the estimation error did not visibly decrease any more. In real applications, where the true parameters are not known, assessing convergence based on the estimation error is, however, clearly not possible.

C.1.1 Results

Figure 18 shows that for CD , using 20 leapfrog steps gives better results than using only three leapfrog steps . A trade-off between computation time and accuracy is visible : running the Markov chains for three Markov steps (CD3 20, in dark green) yields more accurate estimates than running them for one Markov step (CD1 20, in cyan) but the computations take also longer.

Figure 19 shows that for the tested schemes of PCD , using one Markov step together with 40 leapfrog steps (PCD1 40, in cyan) is the preferred choice for Laplacian sources; for logistic sources, it is PCD1 20 (shown in light green).



(a) Laplacian sources

(b) Logistic sources

Figure 19: Trade-off between statistical and computational performance for PCD. The results are plotted in the same way as for CD in Figure 18.

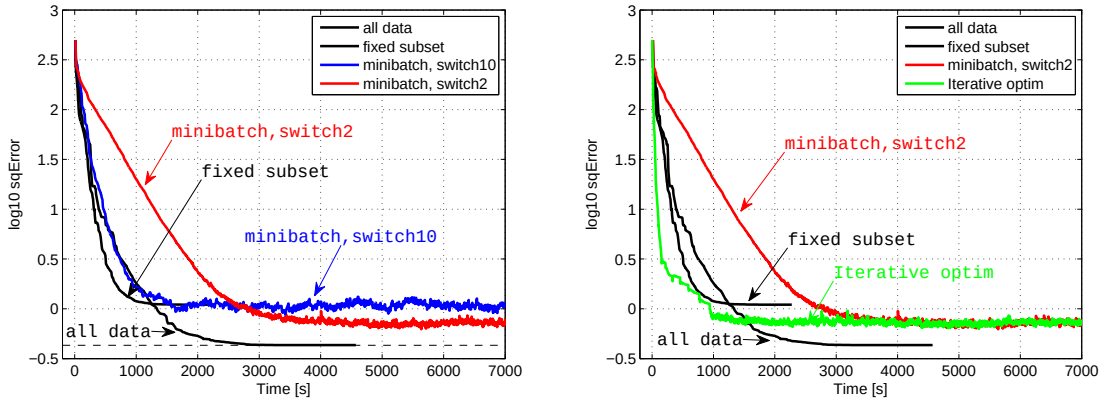
C.2 Natural Images, Section 5: Reducing Computation Time in the Optimization

The objective function J_T in Equation (8) is defined through an sample average. In an iterative optimization scheme, not all the data may be used to compute the average. The reason for using a smaller subset of the data can lie in memory considerations or in the desire to speed up the computations. We analyze here what statistical cost (reduction of estimation accuracy) such a optimization scheme implies. Furthermore, we show that optimizing J_T for increasingly larger values of ν reduces computation time without affecting estimation accuracy. The presented results were obtained by using the the nonlinear conjugate gradient algorithm of Rasmussen (2006) for the optimization.

As working example, we consider the unnormalized Gaussian distribution of Section 3.1 for $n = 40$. Estimating the precision matrix and the normalizing parameter means estimating 821 parameters. We use $T_d = 50000$, and $\nu = 10$. We assume further that, for whatever reason, it is not feasible to work with all the data points at the same time but only with $\tilde{T}_d = 25000$ samples (although for the present example, it is of course possible to use all the data).

C.2.1 Results

The lower black curve in Figure 20(a) shows the performance for the hypothetical situation where we could use all the data. The MSE reaches the level which Corollary 4 predicts (dashed horizontal line). This is the smallest error which can be obtained with nce for $\nu = 10$ and $T_d = 50000$. The upper black curve in the same figure shows the MSE when only a fixed subset with $\tilde{T}_d = 25000$ data points is used in the optimization. This clearly leads to less precise estimates. The performance can, however , be improved by randomly choosing a new subset of size $\tilde{T}_d$ after two updates of the parameters (red curve). The improved performance comes, however, at the cost of slowing down convergence. If the resampling of the



(a) Increasing accuracy

(b) Increasing accuracy and speed of convergence

Figure 20: Analysis of the optimization strategy in Section 5. See Section C.2 for details.

subset is switched at a lower rate, for example, after 10 updates, the speed of convergence stays the same but the accuracy does not improve (blue curve).

Figure 20(b) shows the proposed optimization strategy, which we also use in Section 5 for the simulations with natural image data: We iteratively optimize J_T for increasingly larger values of ν. Whenever we increase ν to $\nu+1$, we also take a new subset. When ν reaches its maximal value, which is here $\nu = 10$, we switch the subset after two parameter updates. For the other values of ν, we switch the subsets at a lower rate of 50 iterations. The results for this optimization strategy are shown in green (curve labelled "iterative optim"). It speeds up convergence while achieving the same precision as in the optimization with resampled subsets of size $\tilde{T}_d$ alone (red curve in Figures (a) and (b)). By resampling new subsets, all the data are actually used in the optimization. However, the estimation accuracy is clearly worse than when all the data are used at once (as in the lower black curve). Hence, there is room for improvement in the way the optimization is performed.

C.3 Natural Images, Section 5.4: Details for the Spline-Based One-Layer Model

The one-layer model that we consider here is

$$\ln p_m(\mathbf{x};\boldsymbol{\theta}) = \sum_{k=1}^{n} f(\mathbf{w}_k^T\mathbf{x};a_1,a_2,\ldots) + c,$$

where the nonlinearity f is a cubic spline. While the two-layer models in Section 5.3 and Section 5.4 were hardcoded to assign the same value to $\mathbf{x}$ and $-\mathbf{x}$, here, no symmetry assumption is made. The parameters are the feature weights $\mathbf{w}_k \in \mathbb{R}^n$, $c \in \mathbb{R}$ for the normalization of the pdf, as well as the $a_i \in \mathbb{R}$ for the parameterization of the nonlinearity f. For the modeling of the nonlinearity, its domain needs to be defined. Its domain is related to the range of its arguments $\mathbf{w}_k^T\mathbf{x}$. To avoid ambiguities in the model specification, we constrain the vectors as in Equation (26). Defining f as a cubic spline on the whole real line is impossible since the number of parameters a_i would become intractable. With the constraint in Equation (26), it is enough to define f only on the interval $[-10\ 10]$ as a cubic spline. For that, we use a knot sequence with an equal spacing of 0.1. Outside

the interval, we define f to stay constant. With these specifications, we can write f in terms of B-spline basis functions with 203 coefficients $a_1, \ldots, a_{203}$.

C.3.1 RESULTS

The learned features are "Gabor-like" (results not shown). We observed, however, a smaller number of feature detectors that are tuned to low frequencies. Figure 16(a) in Section 5.4 shows the learned nonlinearity f (black solid curve) and the random initialization (blue dashed curve). The dashed vertical lines indicate the interval where 99% of the feature outputs occur for natural image input. The learned nonlinearity should thus only be considered valid on that interval. The nonlinearity has two striking properties: First, it is an even function. Note that no such constraint was imposed, so the symmetry of the nonlinearity is due to the symmetry in the natural images. This result validates the symmetry assumption inherent in the two-layer models. It also updates a previous result of ours where we have searched for f in a more restrictive space of functions and no symmetric nonlinearity emerged (Gutmann and Hyvärinen, 2009). Second, f is not monotonic. The shape of f is closely related to the sparsity of the feature outputs $\mathbf{w}_k^T\mathbf{x}$. Since the absolute values of the feature outputs are often very large or very small in natural images, f tends to map natural images to larger numbers than the noise input. This means that the model assigns more often a higher probability density to natural images than to the noise.

C.4 Natural Images, Section 5.5: Refinement of the Thresholding Model

We are taking here a simple approach to the estimation of a two-layer model with spline nonlinearity f: We leave the feature extraction layers that were obtained for the thresholding model in Section 5.3 fixed, and learn only the cubic spline f. The model is thus

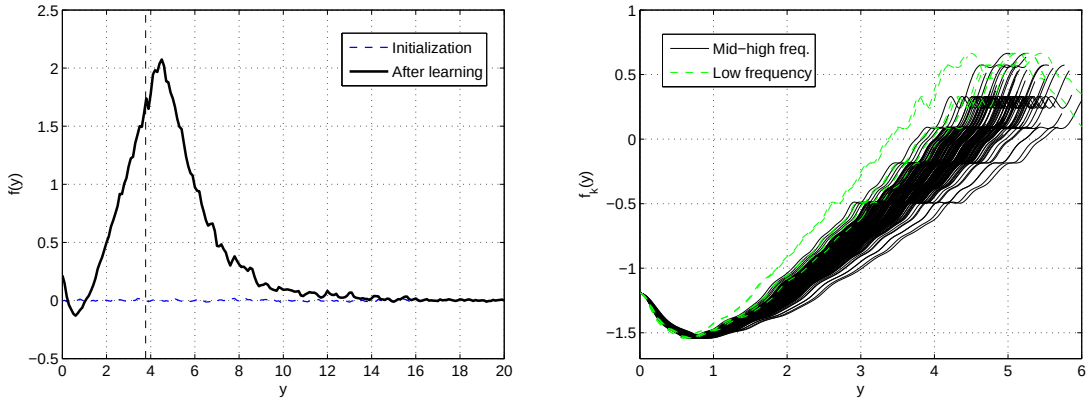
$$\ln p_m(\mathbf{x};\boldsymbol{\theta}) = \sum_{k=1}^{n} f(y_k; a_1, a_2, \ldots) + c, \qquad\qquad y_k = \sum_{i=1}^{n} Q_{ki}(\mathbf{w}_i^T\mathbf{x})^2,$$

where the vector $\boldsymbol{\theta}$ contains the parameters a_i for f and the normalizing parameter c. The knots of the spline are set to have an equal spacing of 0.1 on the interval $[0\ 20]$. Outside that interval, we define f to stay constant. With that specification, we can write f in terms of 203 B-spline basis functions. The parameter vector $\boldsymbol{\theta} \in \mathbb{R}^{204}$ contains then the 203 coefficients for the basis functions and the parameter c.

C.4.1 RESULTS

Figure 21(a) shows the learned nonlinearity (black solid curve) and its random initialization (blue dashed curve). The dashed vertical line around $y = 4$ indicates the border of validity of the nonlinearity since 99% of the y_k fall, for natural image input, to the left of the dashed line. The salient property of the emerging nonlinearity is the "dip" after zero which makes f non-monotonic, as the nonlinearity which emerged in Section 5.4. Figure 21(b) shows the effective nonlinearities f_k when the different scales of the second layer outputs y_k and the normalizing parameter c are taken into account, as we have done in Figure 14(a). We calculated the scale σ_k by taking the average value of y_k over the natural images. The different scales σ_k then define different nonlinearities. Incorporating the normalizing parameter c into the nonlinearity, we obtain the set of effective nonlinearities $f_k(y)$,

$$f_k(y) = f(\sigma_k y) + c/n,\ k = 1, \ldots n. \tag{36}$$



(a) Learned nonlinearity

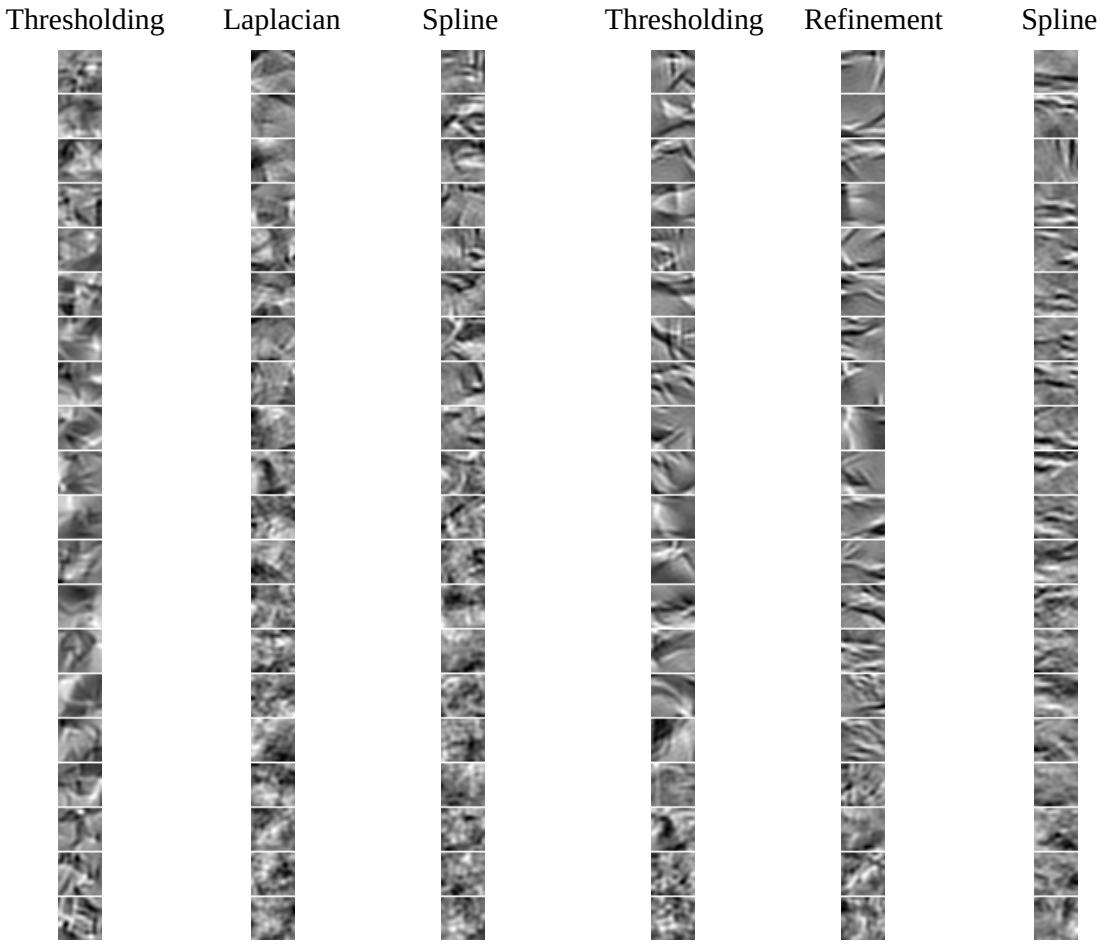
(b) Learned effective nonlinearities

Figure 21: Refinement of the thresholding model of Section 5.3. Only the nonlinearity was learned, the features were kept fixed. The features are shown in Figures 11 to 13. (a) Learned spline (black solid curve) and the initialization (blue dashed curve). The dashed vertical line indicates the border of validity of the learned nonlinearity since 99% of the y_k fall, for natural image input, to the left of it. (b) The different scales of the y_k give rise to a set of effective nonlinearities f_k, as defined in Equation (36). Nonlinearities acting on low-frequency feature detectors are shown in green (dashed lines), the others in black (solid lines), as in Figure 14(a).

For the nonlinearities f_k, the dip occurs between zero and two. Inspection of Figure 14(b) shows that the optimal nonlinearities f_k take, unlike the thresholding nonlinearities, the distribution of the second-layer outputs y_k fully into account. The region where the dip occurs is just the region where noise input is more likely than natural image input. This means that the model is assigning more often a higher probability density to natural images than to the noise.

C.5 Natural Images, Section 5.5: Samples from the Different Models

In Figure 17, we compared images which are considered likely by the different models. In Figure 22, we show samples that we drew from the models using Markov chains (HMC). Since the models are defined on a sphere, we constrained the Hamilitonian dynamics by projecting the states after each leapfrog step back onto the sphere. The number of leapfrog steps was set to 100, and the rejection rate to 0.35 (Neal, 2010, Section 4.4, p.30). The top row shows the most likely samples while the bottom row show the least likely ones. The least likely samples appear similar for all models. For the more probable ones, however, the two-layer models lead to more structured samples than the one-layer models.



(a) One-layer models (b) Two-layer models

Figure 22: Sampling from the learned models of natural images. Figure (a) shows samples from the one-layer models, Figure (b) shows samples from the two-layer models. The samples are sorted so that the top ones are the most likely ones while those at the bottom are the least probable ones. See caption of Table 1 in Section 5.5 for information on the models used. Samples of the training data and the noise are shown in Figure 9 in Section 5.1.

References

C. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

C.J. Geyer. On the convergence of Monte Carlo maximum likelihood calculations. *Journal of the Royal Statistical Society, Series B (Methodological)*, 56(1):261–274, 1994.

M. Gutmann and A. Hyvärinen. Learning features by contrasting natural images with noise. In *Proceedings of the 19th International Conference on Artificial Neural Networks (ICANN)*, volume 5769 of *Lecture Notes in Computer Science*, pages 623–632. Springer Berlin / Heidelberg, 2009.

M. Gutmann and A. Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 9 of *JMLR W&CP*, pages 297–304, 2010.

T. Hastie, R. Tibshirani, and J.H. Friedman. *The Elements of Statistical Learning*. Springer, 2009.

G. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002.

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

A. Hyvärinen. Optimal approximation of signal priors. *Neural Computation*, 20:3087–3110, 2008.

A. Hyvärinen, P.O. Hoyer, and M. Inki. Topographic independent component analysis. *Neural Computation*, 13(7):1527–1558, 2001a.

A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wiley-Interscience, 2001b.

A. Hyvärinen, J. Hurri, and P.O. Hoyer. *Natural Image Statistics*. Springer, 2009.

Y. Karklin and M. Lewicki. A hierarchical Bayesian model for learning nonlinear statistical regularities in nonstationary natural signals. *Neural Computation*, 17:397–423, 2005.

D. Koller and N. Friedman. *Probabilistic Graphical Models*. MIT Press, 2009.

U. Köster and A. Hyvärinen. A two-layer model of natural stimuli estimated with score matching. *Neural Computation*, 22(9):2308–2333, 2010.

J. Lücke and M. Sahani. Maximal causes for non-linear component extraction. *Journal of Machine Learning Research*, 9:1227–1267, 2008.

R.M. Neal. *Handbook of Markov Chain Monte Carlo*, chapter MCMC using Hamiltonian Dynamics. Chapman & Hall /CRC Press, 2010.

B.A. Olshausen and D.J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996.

S. Osindero and G. Hinton. Modeling image patches with a directed hierarchy of Markov random fields. In *Advances in Neural Information Processing Systems 20*, pages 1121–1128. MIT Press, 2008.

S. Osindero, M. Welling, and G. E. Hinton. Topographic product models applied to natural scene statistics. *Neural Computation*, 18 (2):381–414, 2006.

M. Pihlaja, M. Gutmann, and A. Hyvärinen. A family of computationally efficient and simple estimators for unnormalized statistical models. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 442–449. AUAI Press, 2010.

M.A. Ranzato and G. Hinton. Modeling pixel means and covariances using factorized third-order Boltzmann machines. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2551–2558, 2010.

C.E. Rasmussen. Conjugate gradient algorithm, Matlab code version 2006-09-08. Downloaded from http://learning.eng.cam.ac.uk/carl/code/minimize/minimize.m. 2006.

C.P. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer, 2nd edition, 2004.

N.N. Schraudolph and T. Graepel. Towards stochastic conjugate gradient methods. In *Proceedings of the 9th International Conference on Neural Information Processing (ICONIP)*, volume 2, pages 853–856, 2002.

W. Sun and Y. Yuan. *Optimization Theory and Methods: Nonlinear Programming*. Springer, 2006.

Y. Teh, M. Welling, S. Osindero, and G. Hinton. Energy-based models for sparse overcomplete representations. *Journal of Machine Learning Research*, 4:1235–1260, 2004.

T. Tieleman. Training restricted Boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1064–1071, 2008.

J. H. van Hateren and A. van der Schaaf. Independent component filters of natural images compared with simple cells in primary visual cortex. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 265(1394):359–366, 1998.

Z. Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

L. Wasserman. *All of Statistics*. Springer, 2004.

L. Younes. Parametric inference for imperfectly observed Gibbsian fields. *Probability Theory and Related Fields*, 82(4):625–645, 1989.