Brisk Kernel ICA[1]

Stefanie Jegelka[2], Arthur Gretton[3]

Abstract. Recent approaches to independent component analysis have used kernel independence measures to obtain very good performance in ICA, particularly in areas where classical methods experience difficulty (for instance, sources with near-zero kurtosis). In this chapter, we compare two efficient extensions of these methods for large-scale problems: random subsampling of entries in the Gram matrices used in defining the independence measures, and incomplete Cholesky decomposition of these matrices. We derive closed-form, efficiently computable approximations for the gradients of these measures, and compare their performance on ICA using both artificial and music data. We show that kernel ICA can scale up to larger problems than yet attempted, and that incomplete Cholesky decomposition performs better than random sampling.

1 Introduction

The problem of instantaneous independent component analysis involves the recovery of linearly mixed, independent sources, in the absence of information about the source distributions beyond their mutual independence (Hyvärinen et al., 2001). Classical approaches to this problem, which use as their independence criterion the sum of expectations of a fixed nonlinear function (or a small number of such functions) on each recovered source, scale well to large numbers of sources and samples. On the other hand, they only ensure local convergence in the vicinity of independence (Shen and Hüper, 2006a, give one such analysis for FastICA), and do not guarantee independent sources are recovered at the *global* optimum of the independence criterion. Statistical tests of independence should then be applied (as by Ku and Fine, 2005) to verify independent sources are recovered.

Ideally, we would prefer to optimise over a criterion that exhibits a global optimum at independence. This approach is adopted in several recent algorithms, including those of Stögbauer et al. (2004); Chen (2006); Learned-Miller and Fisher III (2003), who optimise the mutual information between the sources; and Eriksson and Koivunen (2003); Chen and Bickel (2005); Murata (2001), who optimise characteristic function-based independence measures. We focus in particular on the kernel approaches of Bach and Jordan (2002); Gretton et al. (2005a,b), which are already established as yielding excellent performance in small-scale (relatively few sources and samples) linear ICA problems. Kernel methods measure independence by computing statistics on the spectrum of covariance or correlation operators defined on reproducing kernel Hilbert spaces (RKHSs). In the present study, our independence measures are the Constrained Covariance (COCO), which is the spectral norm of the covariance operator, and the Hilbert-Schmidt Independence Criterion (HSIC), which is the Hilbert-Schmidt norm of this operator.

Two difficulties arise when kernel measures of independence are used for ICA in large-scale settings. First, when there are many samples, it becomes impractical to compute and store the matrices of kernel values, especially as these evolve in the course of optimising over the estimated unmixing matrix. Second, we are required to compute the gradient of the independence measures with respect to the mixing matrix, which is accomplished highly inefficiently in the kernel methods cited earlier: for m sources, each derivative requires $2m(m-1)$ evaluations of the independence measure (for a total cost of $O(m^4)$), which causes especially poor scaling for large numbers of sources.

To make kernel ICA relevant to large-scale problems, we must deal both with many independent sources and with a large sample size n. Our focus is on evaluating both sparse and low rank matrix approximation techniques to efficiently compute both our independence measures and their gradients: in particular, we must avoid ever computing or storing a complete Gram matrix, since the cost of $O(n^2)$ would make the algorithm unusable. Bach and Jordan (2002) employ the incomplete Cholesky decomposition to this purpose, following Fine and Scheinberg (2001), which results in a substantial reduction in cost. An alternative approach is to compute a random subsample

[1] When referring to this article, please cite the Book Chapter (Jegelka and Gretton, 2007)

[2] jegelka@tuebingen.mpg.de; MPI for Biological Cybernetics, Tübingen, Germany

[3] arthur.gretton@tuebingen.mpg.de; MPI for Biological Cybernetics, Tübingen, Germany

of the Gram matrix entries as proposed by Achlioptas and McSherry (2001), thus taking advantage of sparse matrix operations to cheaply obtain the dependence measures. As long as the spectra of the Gram matrices decay quickly, the spectral norm of the matrix product that defines COCO is well approximated by this random sampling (this is also true for the Hilbert-Schmidt norm defining HSIC, albeit to a lesser extent): in particular, error bounds established by Drineas et al. (2004) apply to our case. We also emphasise the broader implications of the problem we address, since it involves manipulating a product of Gram matrices so as to minimise its singular values: at this minimum, however, an accurate estimation of the product from low rank approximations is at its most difficult.

Additional efficiencies can be achieved in the gradient descent algorithm on the unmixing matrix. We investigate both local quadratic approximations to the independence measures, and steps of decreasing size along consecutive directions of steepest descent. Both strategies require significantly fewer evaluations of the independence measure than the Golden search used by Bach and Jordan (2002); Gretton et al. (2005a,b); Chen and Bickel (2005).

We begin our presentation in Section 2, where we introduce the problem of independent component analysis. In Section 3, we describe independence measures based on covariance operators in RKHSs, and discuss the two Gram matrix approximation strategies for these measures. We compute the gradients of our independence measures in Section 4, and show how these may likewise be efficiently computed using our approximation strategies. We give a breakdown of the computational cost of the proposed methods in Section 5. Finally, we present our experiments in Section 6.

2 Independent component analysis

We describe the goal of instantaneous independent component analysis (ICA), drawing mainly on (Hyvärinen et al., 2001; Cardoso, 1998), as well as the core properties of ICA explored by Comon (1994). We are given n samples $\boldsymbol{t} := (\mathbf{t}_1, \ldots, \mathbf{t}_n)$ of the m dimensional random vector $\mathbf{t}$, which are drawn independently and identically from the distribution $\mathrm{P}_{\mathbf{t}}$. The vector $\mathbf{t}$ is related to the random vector $\mathbf{s}$ (also of dimension m) by the linear mixing process

$$\mathbf{t} = \mathbf{B}\mathbf{s}, \tag{1}$$

where $\mathbf{B}$ is a matrix with full rank. We refer to our ICA problem as being *instantaneous* as a way of describing the dual assumptions that any observation $\mathbf{t}$ depends only on the sample $\mathbf{s}$ at that instant, and that the samples $\mathbf{s}$ are drawn independently and identically.

The components s_i of $\mathbf{s}$ are assumed to be mutually independent: this model codifies the assumption that the sources are generated by unrelated phenomena (for instance, one component might be an EEG signal from the brain, while another could be due to electrical noise from nearby equipment). Mutual independence (in the case where the random variables admit probability densities) has the following definition (Papoulis, 1991, p. 184):

Definition 1 (Mutual independence)**.** *Suppose we have a random vector* $\mathbf{s}$ *of dimension* m*. We say that the components* s_i *are* mutually *independent if and only if*

$$\mathbf{f}_{\mathbf{s}}(\mathbf{s}) = \prod_{i=1}^{m} \mathbf{f}_{\mathsf{s}_i}(s_i). \tag{2}$$

It follows easily that the random variables are pairwise *independent if they are* mutually *independent; i.e.* $\mathbf{f}_{\mathsf{s}_i}(s_i)\,\mathbf{f}_{\mathsf{s}_j}(s_j) = \mathbf{f}_{\mathsf{s}_i,\mathsf{s}_j}(s_i, s_j)$ *for all* $i \neq j$*. The reverse does not hold, however: pairwise independence does not guarantee mutual independence.*

Our goal is to recover $\mathbf{s}$ via an estimate $\mathbf{W}$ of the inverse of the matrix $\mathbf{B}$, such that the recovered vector $\mathbf{x} = \mathbf{W}\mathbf{B}\mathbf{s}$ has mutually independent components.[4] For the purpose of simplifying our discussion, we will assume that $\mathbf{B}$ (and hence $\mathbf{W}$) is an *orthogonal matrix*; in the case of arbitrary $\mathbf{B}$, the observations must first be decorrelated before an orthogonal $\mathbf{W}$ is applied. We will return to the problem of gradient descent on the orthogonal group in Section 4.5. Mutual independence is generally difficult to determine (Kankainen, 1995, gives one approach). In the case of linear mixing, however, we are able to find a unique optimal unmixing matrix $\mathbf{W}$ using only the *pairwise* independence between elements of $\mathbf{x}$, because it is in this case equivalent to recovering the *mutually* independent terms of $\mathbf{s}$ (up to permutation and scaling). This is due to Theorem 11 of Comon (1994).

[4] It turns out that the problem described above is indeterminate in certain respects. For instance, our measure of independence does not change when the ordering of elements in $\mathbf{x}$ is swapped, or when components of $\mathbf{x}$ are scaled by different constant amounts. Thus, source recovery takes place up to these invariances.

3 Independence measures based on RKHS covariance operators

We begin in Section 3.1 by introducing the problem variables, and describing covariance operators in RKHSs. Next, in Section 3.2, we show how the spectral and Hilbert-Schmidt norms of the covariance operators may easily be expressed as a function of the kernels of the RKHSs, and describe the relation to other independence measures in the literature. Finally, in Section 3.3, we cover approximation techniques based on subsampling and on the incomplete Cholesky decomposition, which will be used in computing our independence criteria and their derivatives.

3.1 Covariance operators and RKHSs

In this section, we provide the functional analytic background necessary in describing covariance operators between RKHSs. Our main purpose in this section is to introduce functional covariance operators, following Baker (1973); Fukumizu et al. (2004), who characterise these operators for general Hilbert spaces.

Consider a Hilbert space $\mathcal{F}$ of functions from $\mathcal{X}$ to $\mathbb{R}$, where $\mathcal{X}$ is a separable metric space. The Hilbert space $\mathcal{F}$ is an RKHS if at each $x \in \mathcal{X}$, the point evaluation operator $\delta_x : \mathcal{F} \to \mathbb{R}$, which maps $f \in \mathcal{F}$ to $f(x) \in \mathbb{R}$, is a bounded linear functional. To each point $x \in \mathcal{X}$, there corresponds an element $\phi(x) \in \mathcal{F}$ (we call ϕ the *feature map*) such that $\langle \phi(x), \phi(x') \rangle_{\mathcal{F}} = k(x, x')$, where $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a unique positive definite kernel. We also define a second RKHS $\mathcal{G}$ with respect to the separable metric space $\mathcal{Y}$, with feature map ψ and kernel $\langle \psi(y), \psi(y') \rangle_{\mathcal{G}} = l(y, y')$. We require $\mathcal{F}$ and $\mathcal{G}$ to be universal in the sense of Steinwart (2002), since this is a requirement of Theorem 6 (which guarantees the criteria we optimise are truly independence measures).

Let $\mathrm{P}_{\mathsf{x},\mathsf{y}}(x, y)$ be a joint measure on $(\mathcal{X} \times \mathcal{Y}, \Gamma \times \Lambda)$ (here Γ and Λ are the Borel σ-algebras on $\mathcal{X}$ and $\mathcal{Y}$, respectively), with associated marginal measures P_{x} and P_{y} and random variables x and y. The covariance operator $C_{xy} : \mathcal{G} \to \mathcal{F}$ is defined such that for all $f \in \mathcal{F}$ and $g \in \mathcal{G}$,

$$\langle f, C_{xy} g \rangle_{\mathcal{F}} \quad = \quad \mathbf{E}_{\mathsf{x},\mathsf{y}} \left[[f(\mathsf{x}) - \mathbf{E}_{\mathsf{x}} [f(\mathsf{x})]] [g(\mathsf{y}) - \mathbf{E}_{\mathsf{y}} [g(\mathsf{y})]] \right].$$

In practice, we do not deal with the measure $\mathrm{P}_{\mathsf{x},\mathsf{y}}$ itself, but instead observe samples drawn independently according to it. We write an i.i.d. sample of size n from $\mathrm{P}_{\mathbf{x},\mathbf{y}}$ as $\boldsymbol{z} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, and likewise $\boldsymbol{x} := \{x_1, \ldots, x_n\}$ and $\boldsymbol{y} := \{y_1, \ldots y_n\}$. Finally, we define the Gram matrices $\mathbf{K}$ and $\mathbf{L}$ of inner products in $\mathcal{F}$ and $\mathcal{G}$, respectively, between the mapped observations above: here $\mathbf{K}$ has (i, j)th entry $k(x_i, x_j)$ and $\mathbf{L}$ has (i, j)th entry $l(y_i, y_j)$. The Gram matrices for the variables centred in their respective feature spaces are shown by Schölkopf et al. (1998) to be

$$\widetilde{\mathbf{K}} := \mathbf{HKH}, \quad \widetilde{\mathbf{L}} := \mathbf{HLH},$$

where

$$\mathbf{H} = \mathbf{I} - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^\top, \tag{3}$$

and $\mathbf{1}_n$ is an $n \times 1$ vector of ones.

3.2 Norms of the covariance operator

In this section, we introduce two norms of the covariance operator, and show how these may be computed using kernels. First, the constrained covariance (Gretton et al., 2005b).

Definition 2 (Constrained Covariance (COCO))**.** *Let $\mathcal{F}, \mathcal{G}$ be universal RKHSs, with F and G their respective unit balls, and $\mathrm{P}_{\mathsf{x},\mathsf{y}}$ be a probability measure. We define the* constrained covariance *as*

$$\mathrm{COCO}(\mathrm{P}_{\mathsf{x},\mathsf{y}}; \mathcal{F}, \mathcal{G}) := \sup_{f \in F, g \in G} \left[\mathrm{cov}(f(\mathsf{x}), g(\mathsf{y})) \right]. \tag{4}$$

Note that this is just the spectral norm of the covariance operator $\mathbf{C}_{xy}$.

The next lemma gives an empirical estimate of COCO.

Lemma 3 (Value of $\mathrm{COCO}(\boldsymbol{z}; F, G)$)**.** *Given n independent observations $\boldsymbol{z} := ((x_1, y_1), \ldots, (x_n, y_n)) \subset (\mathcal{X} \times \mathcal{Y})^n$, the empirical estimate of COCO is*

$$\mathrm{COCO}(\boldsymbol{z}; \mathcal{F}, \mathcal{G}) = \frac{1}{n} \sqrt{\|\widetilde{\mathbf{K}}\widetilde{\mathbf{L}}\|_2}, \tag{5}$$

where the matrix norm $\|\cdot\|_2$ *denotes the largest singular value. An equivalent* unnormalised *form is* $\mathrm{COCO}(\boldsymbol{z}; F, G) = \max_i \gamma_i$*, where* γ_i *are the solutions to the generalised eigenvalue problem*

$$\begin{bmatrix} \mathbf{0} & \widetilde{\mathbf{K}}\widetilde{\mathbf{L}} \\ \widetilde{\mathbf{L}}\widetilde{\mathbf{K}} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_i \\ \boldsymbol{\beta}_i \end{bmatrix} = \gamma_i \begin{bmatrix} \widetilde{\mathbf{K}} & \mathbf{0} \\ \mathbf{0} & \widetilde{\mathbf{L}} \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_i \\ \boldsymbol{\beta}_i \end{bmatrix}. \tag{6}$$

Our second independence measure, the Hilbert-Schmidt Independence Criterion (HSIC), is defined by Gretton et al. (2005a) as follows.

Definition 4 (HSIC)**.** *Given separable RKHSs* $\mathcal{F}, \mathcal{G}$ *and a joint measure* $\mathrm{P}_{\mathsf{x},\mathsf{y}}$ *over* $(\mathcal{X} \times \mathcal{Y}, \Gamma \times \Lambda)$*, we define the Hilbert-Schmidt Independence Criterion (HSIC) as the squared Hilbert-Schmidt norm of the associated cross-covariance operator* $\mathbf{C}_{xy}$*:*

$$\mathrm{HSIC}(\mathrm{P}_{\mathsf{x},\mathsf{y}}, \mathcal{F}, \mathcal{G}) := \|\mathbf{C}_{xy}\|^2_{\mathrm{HS}} \tag{7}$$

To compute an empirical estimate of this norm, we need to express HSIC in terms of kernel functions. This is achieved in the following lemma:

Lemma 5 (HSIC in terms of kernels)**.**

$$\begin{aligned} \mathrm{HSIC}(\mathrm{P}_{\mathsf{x},\mathsf{y}}; \mathcal{F}, \mathcal{G}) \quad := \quad & \mathbf{E}_{\mathsf{x},\mathsf{x}',\mathsf{y},\mathsf{y}'}\left[k(\mathsf{x},\mathsf{x}')l(\mathsf{y},\mathsf{y}')\right] + \mathbf{E}_{\mathsf{x},\mathsf{x}'}\left[k(\mathsf{x},\mathsf{x}')\right]\mathbf{E}_{\mathsf{y},\mathsf{y}'}\left[l(\mathsf{y},\mathsf{y}')\right] \\ & -2\mathbf{E}_{\mathsf{x},\mathsf{y}}\left[\mathbf{E}_{\mathsf{x}'}\left[k(\mathsf{x},\mathsf{x}')\right]\mathbf{E}_{\mathsf{y}'}\left[l(\mathsf{y},\mathsf{y}')\right]\right] \end{aligned}$$

Here $\mathbf{E}_{\mathsf{x},\mathsf{x}',\mathsf{y},\mathsf{y}'}[\cdot]$ *denotes the expectation over pairs* x,y *and* x',y' *drawn independently from* $\mathrm{P}_{\mathsf{x},\mathsf{y}}$*. A biased empirical estimate of the squared* $HSIC$ *is*

$$\mathrm{HSIC}^2(\boldsymbol{z}; \mathcal{F}, \mathcal{G}) := \frac{1}{n^2}\mathrm{tr}(\mathbf{KHLH}).$$

The bias in this estimate decreases as $1/n$*, and thus drops faster than the variance (which decreases as* $1/\sqrt{n}$*).*

It follows from the above lemma that the Hilbert-Schmidt norm of $\mathbf{C}_{xy}$ exists when the various expectations over the kernels are bounded. As discussed in the previous section, we only require pairwise independence for ICA: thus, we may sum either HSIC or COCO over all pairs of sources to obtain a cost function to minimise when solving for the unmixing matrix $\mathbf{W}$.

The empirical HSIC in Lemma 5 is well established in the independence testing literature, although its description in terms of covariance operators in RKHSs is more recent. The bivariate criterion was originally proposed by Feuerverger (1993), and a multivariate generalisation was established by Kankainen (1995); the original motivation of these statistics was as smoothed divergence measures between the joint characteristic function and the product of the marginal characteristic functions. Kankainen's multivariate independence measure was applied to ICA by Chen and Bickel (2005); Eriksson and Koivunen (2003). The former established the consistency of ICA methods based on Kankainen's measure, and both studies proposed (different) algorithms for its optimisation. Chen and Bickel's approach is closest to ours, and is further discussed in Section 4.5. Kankainen's measure was also applied by Achard et al. (2003) to recovering sources that are first linearly mixed, and then subject to nonlinear distortion.

Note that we have omitted several additional kernel independence measures. The kernel canonical correlation (Bach and Jordan, 2002) is similar to COCO, but represents the regularised spectral norm of the functional correlation operator, rather than the covariance operator. The kernel generalised variance (Bach and Jordan, 2002) and kernel mutual information (Gretton et al., 2005b) were shown by Gretton et al. (2005b) to upper bound the mutual information near independence (and to be tight at independence). Experiments by Gretton et al. (2005a), however, indicate that these methods do not outperform HSIC in linear ICA for large sample sizes, at least on the benchmark data of Bach and Jordan (2002). That said, it would be interesting to investigate ways to decrease the computational cost of these approaches.

Finally, we give a theorem that combines the results of Gretton et al. (2005a,b), and which justifies using both COCO and HSIC as independence measures.

Theorem 6 (COCO and HSIC are only zero at independence for universal kernels)**.** *Denote by* $\mathcal{F}, \mathcal{G}$ *RKHSs with universal kernels on the compact metric spaces* $\mathcal{X}$ *and* $\mathcal{Y}$*, respectively. Then both* $\mathrm{COCO}(\mathrm{P}_{\mathsf{x},\mathsf{y}}; \mathcal{F}, \mathcal{G}) = 0$ *and* $\mathrm{HSIC}(\mathrm{P}_{\mathsf{x},\mathsf{y}}; \mathcal{F}, \mathcal{G}) = 0$ *if and only if* x,y *are independent.*

3.3 Gram matrix approximation strategies

Both COCO and HSIC are matrix norms of a product of two centred Gram matrices. We address two main approaches to approximating these matrices. The first is sparsification by random subsampling of the matrix entries, and the second is incomplete Cholesky decomposition.

A simple and efficient approximation of matrix products was suggested by Drineas et al. (2004), based on results by Achlioptas and McSherry (2001): Replace each matrix $\mathbf{A}$ in the product by a sparse approximation $\mathbf{A}'$ with entries a'_{ij}. Assign a probability p_{ij} to each entry a_{ij} and let $a'_{ij} = a_{ij}/p_{ij}$ with probability p_{ij}, otherwise $a'_{ij} = 0$. We use a uniform probability p for all entries. The expected number of nonzero entries for an $n \times n$ matrix is then pn^2. This method is especially effective when the matrices have a strong spectral structure, since the subsampling of entries can be interpreted as adding a matrix with weak spectral structure, which will have little effect on the larger singular values of the original matrix.

Although we do not go into detail here, it is possible to provide theoretical guarantees on the closeness of both COCO and HSIC to their respective approximations (computed with this random sampling approach). Achlioptas and McSherry (2001, Theorem 3) provide probabilistic bounds on both the spectral and Frobenius norms of the difference between a matrix and its sparse approximation. Drineas et al. (2004) show how to generalise this result to convergence of the difference between a *product* of two subsampled matrices and the product of the two original matrices. Thus, since COCO is a spectral norm of a matrix product, and HSIC is the Frobenius norm of this same product, we can use the triangle inequality to obtain probabilistic bounds on the difference between the subsampled COCO and HSIC and their true values, as a function of the fraction of entries retained. Better convergence properties may be obtained by selecting matrix entries non-uniformly, with probability proportional to their size, according to the scheme of Drineas et al. (2004, Theorem 4). This would require computation of the entire Gram matrix, however.

The other approach we consider, described by Williams and Seeger (2000), is to generate an index set I of length d having entries chosen without replacement from $\{1, \ldots, n\}$, and to approximate the Gram matrix as

$$\mathbf{K} \approx \mathbf{K}' := \mathbf{K}_{:,I}\mathbf{K}_{I,I}^{-1}\mathbf{K}_{I,:}, \tag{8}$$

where $\mathbf{K}_{:,I}$ is the Gram matrix with the rows unchanged, and the columns chosen from the set I; and $\mathbf{K}_{I,I}$ is the $d \times d$ submatrix with both rows and columns restricted to I. The question is then how to choose the subset I. Williams and Seeger (2000) propose to select the subset at random. Smola and Schölkopf (2000) greedily choose a subset of the sample points to minimise either the Frobenius norm of the difference between $\mathbf{K}$ and its approximation $\mathbf{K}'$ from these samples, or $\mathbf{tr}\,(\mathbf{K} - \mathbf{K}')$. The latter approach costs $O(pnd^2)$, where p are the number of sample points examined at each stage of the greedy search (the Frobenius approach is more expensive). Fine and Scheinberg (2001) likewise choose entries that minimise $\mathbf{tr}\,(\mathbf{K} - \mathbf{K}')$. Their incomplete Cholesky algorithm costs $O(nd^2)$: thus, we use this cheaper method.

We remark that Drineas and Mahoney (2005) generalise the approach of Williams and Seeger (2000), proposing a scheme for sampling *with* replacement from the columns of the Gram matrix. Although they provide performance guarantees for any sampling distribution (including the uniform), they find in practice that the sampling probability of a column should be proportional to its length for best performance (Drineas and Mahoney, 2005, Theorem 1). They also claim that no performance guarantees are yet established for efficient, non-uniform sampling without replacement (Drineas and Mahoney, 2005, p. 2158). Since computing the column length requires computing the entire Gram matrix for each evaluation of the kernel independence measure, this approach is too expensive for kernel ICA, both in terms of time and in storage.

4 Gradient descent on the orthogonal group

When optimising our kernel dependence measures in ICA, a fundamental building block in the gradient is the derivative of the Gram matrix with respect to the relevant unmixing matrix entries. We provide this gradient in the case of the Gaussian kernel in Section 4.1. We obtain the derivative of HSIC with respect to $\mathbf{W}$ in Section 4.2, and its incomplete Cholesky approximation in Section 4.3; we then give the COCO derivative and its approximations in Section 4.4. We show how to use these free gradients to perform gradient descent on the orthogonal group in Section 4.5. Finally, in Section 4.6, we describe different search strategies along the direction of steepest descent.

4.1 Gradient of the Gram matrix entries

We begin by providing the derivative of the Gram matrix entries with respect to a particular row $\mathbf{w}$ of the unmixing matrix, which we will require in the following sections. This derivative clearly depends on the particular kernel

used. We employ a Gaussian kernel here, but one could easily obtain the derivatives of alternative kernels: these can then be plugged straightforwardly into the gradients in the following sections.

Lemma 7 (Derivative of a Gaussian Gram matrix with respect to the unmixing matrix)**.** *Let* $\mathbf{K}$ *be the Gram matrix computed with a Gaussian kernel, and let* $\mathbf{w}$ *be an* $1 \times m$ *row of the unmixing matrix, such that*

$$k_{ij} = k(x_i, x_j) = \exp\left[\frac{-1}{2\sigma^2}\mathbf{w}^\top \mathbf{T}_{ij}\mathbf{w}\right],$$

where $\mathbf{T}_{ij} = (\mathbf{t}_i - \mathbf{t}_j)(\mathbf{t}_i - \mathbf{t}_j)^\top$*, and* $\mathbf{t}_i$ *is the* i*th sample of observations. Then the derivative of any* k_{ij} *with respect to* $\mathbf{w}$ *is*

$$\frac{\partial k_{ij}}{\partial \mathbf{w}} = -\frac{k_{ij}}{\sigma^2}\mathbf{w}^\top(\mathbf{t}_i - \mathbf{t}_j)(\mathbf{t}_i - \mathbf{t}_j)^\top.$$

Since the above derivative is a vector, we must establish how to write the derivative of the entire Gram matrix as a single matrix. This is done using the $\mathrm{vec}(\mathbf{A})$ operation, which stacks the columns of the matrix $\mathbf{A}$ on top of each other. Thus, the resulting differential is

$$d(\mathrm{vec}\mathbf{K}) = \underbrace{\begin{bmatrix} \partial k_{11}/\partial \mathbf{w} \\ \vdots \\ \partial k_{n1}/\partial \mathbf{w} \\ \partial k_{12}/\partial \mathbf{w} \\ \vdots \\ \partial k_{nn}/\partial \mathbf{w} \end{bmatrix}}_{\partial \mathrm{vec}(\mathbf{K})/\partial \mathbf{w}} d(\mathrm{vec}\ \mathbf{w}), \tag{9}$$

where obviously $d(\mathrm{vec}\,\mathbf{w}) = d\mathbf{w}$.

4.2 HSIC gradient and sparse approximation

We next give the HSIC derivative. Since HSIC for more than two sources is computed as a sum of pairwise dependencies, we need only express the derivative for a single pair of sources: the aggregate derivative is a sum of multiple such terms.

Theorem 8 (Derivative of HSIC^2)**.** *Let* $\mathbf{K}$ *and* $\mathbf{L}$ *be two* $n \times n$ *Gram matrices, corresponding to the respective rows* $\mathbf{w}$ *and* $\mathbf{v}$ *of the unmixing matrix. Then*

$$\frac{\partial \mathrm{tr}(\mathbf{KHLH})}{\partial \begin{bmatrix} \mathbf{w} \\ \mathbf{v} \end{bmatrix}} = \begin{bmatrix} \frac{\partial \mathrm{tr}(\mathbf{KHLH})}{\partial \mathbf{w}} \\ \frac{\partial \mathrm{tr}(\mathbf{KHLH})}{\partial \mathbf{v}} \end{bmatrix},$$

where

$$\frac{\partial \mathrm{tr}(\mathbf{KHLH})}{\partial \mathbf{w}} = (\mathrm{vec}(\mathbf{HLH}))^\top \begin{bmatrix} \frac{\partial k_{11}}{\partial \mathbf{w}} & \dots & \frac{\partial k_{n1}}{\partial \mathbf{w}} & \frac{\partial k_{21}}{\partial \mathbf{w}} & \dots & \frac{\partial k_{nn}}{\partial \mathbf{w}} \end{bmatrix}^\top.$$

We now address how this may be efficiently computed when the Gram matrices are sparsely sampled. The products in the gradients not only contain the Kernel matrices, but also the centred matrices $\widetilde{\mathbf{K}}$ and $\widetilde{\mathbf{L}}$. To again avoid the computation of all entries, we do not want to build sparse approximations from the centred matrices of full rank. Instead, there are two alternatives to approximate $\widetilde{\mathbf{K}}$: (i) center the sparse approximation $\mathbf{K}'$ so that $\widetilde{\mathbf{K}}' = \mathbf{H}\mathbf{K}'\mathbf{H}$, or, (ii) approximate $\widetilde{\mathbf{K}}$ by a sparse matrix based on the sparse $\mathbf{K}'$. The expectation of the result of (i) is exactly $\widetilde{\mathbf{K}}$. However, the approximation is no longer sparse if it is explicitly computed. For a sparse approximation of $\widetilde{\mathbf{K}}$, observe that the entries of the full matrix are

$$\tilde{k}_{ij} = k_{ij} - \frac{1}{n}\sum_{q=1}^{n} k_{iq} - \frac{1}{n}\sum_{q=1}^{n} k_{qj} + \frac{1}{n^2}\sum_{q,r=1}^{n} k_{qr}, \tag{10}$$

i.e. the original entries of $\mathbf{K}$ minus the row and column means plus the mean of all matrix entries. Let μ_i be the mean of all nonzero elements in row i, μ^j the mean of all nonzero entries in column j and μ the mean of all nonzero entries of the entire matrix. Then we set

$$\tilde{k}'_{ij} = \begin{cases} 0, & \text{if } k'_{ij} = 0 \\ k'_{ij} - \mu_i - \mu^j + \mu, & \text{otherwise.} \end{cases} \tag{11}$$

The row and column means of the resulting matrix will usually not be zero, but the matrix is sparse. With this latter scheme, we only need to compute those $\partial k_{ij}/\partial \mathbf{w}$ in the derivative in Theorem 8 where both k'_{ij} and l'_{ij} are nonzero. Therefore we use approximate centring for the sparse HSIC derivative. In addition, the partial derivatives of $\mathbf{K}$ can be computed on the fly in the computation of the product, without being stored.

4.3 Incomplete Cholesky approximation of HSIC gradient

When computing the incomplete Cholesky decomposition so as to approximate the independence measures, we need never explicitly evaluate the matrix inverse in (8) (see the algorithm in Figure 2 of Fine and Scheinberg, 2001). In computing the derivative, however, we must explicitly deal with this inverse. The differential of HSIC^2 is

$$\begin{aligned} d\mathbf{tr}\left(\widetilde{\mathbf{K}}'\widetilde{\mathbf{L}}'\right) &= \mathbf{tr}\left(\widetilde{\mathbf{K}}'d(\mathbf{L}')\right) + \mathbf{tr}\left(\widetilde{\mathbf{L}}'d(\mathbf{K}')\right) \\ &= \mathbf{tr}\left(\widetilde{\mathbf{K}}'d(\mathbf{L}')\right) + \mathbf{tr}\left(\widetilde{\mathbf{L}}'d(\mathbf{K}_{:,I}\mathbf{K}_{I,I}^{-1}\mathbf{K}_{I,:})\right) \\ &= \mathbf{tr}\left(\widetilde{\mathbf{K}}'d(\mathbf{L}')\right) + 2\mathrm{vec}(\widetilde{\mathbf{L}}'\mathbf{K}_{:,I}\mathbf{K}_{I,I}^{-1})^\top \mathrm{vec}(d\mathbf{K}_{:,I}) \\ &\quad -\mathrm{vec}(\mathbf{K}_{I,I}^{-1}\mathbf{K}_{I,:}\widetilde{\mathbf{L}}'\mathbf{K}_{:,I}\mathbf{K}_{I,I}^{-1})^\top \mathrm{vec}(d\mathbf{K}_{I,I}). \end{aligned} \tag{12}$$

where our expression for $d\mathbf{K}'$ is derived in the appendix of this chapter (we do not expand $d\mathbf{L}'$, but its expression is analogous). The matrix derivative $\partial\mathrm{vec}(\mathbf{K}_{:,I})/\partial\mathrm{vec}(\mathbf{w})$ has size $nd \times m$, and $\partial\mathrm{vec}(\mathbf{K}_{I,I})/\partial\mathrm{vec}(\mathbf{w})$ has size $d^2 \times m$, whereas the original matrix derivative from the differential (9) contains $n^2 \times m$ entries. If both kernel matrices are decomposed, (12) can be computed without ever taking a product with an $n \times n$ matrix, provided we order the matrix multiplications correctly.

We must also account for the rapid decay of the spectrum of Gram matrices with Gaussian kernels (see e.g. the discussion in Appendix C of Bach and Jordan, 2002), since this can cause the inverse of $\mathbf{K}_{I,I}$ to be ill-conditioned. We therefore add a small ridge of 10^{-6} to $\mathbf{K}_{I,I}$, although we emphasise our algorithm is insensitive to this value.

4.4 COCO gradient and approximations

We now describe how to compute the gradient of COCO. The maximum singular value of a matrix $\mathbf{A}$ is the square root of the maximum eigenvalue of $\mathbf{A}^\top\mathbf{A}$. In other words, if we replace COCO by its fourth power (which has the same minimum of zero at independence), we wish to compute

$$\mathrm{COCO}^4(\boldsymbol{z}; \mathcal{F}, \mathcal{G}) = \frac{1}{n^4}\lambda_{\max}\left(\widetilde{\mathbf{K}}\,\widetilde{\mathbf{L}}^2\,\widetilde{\mathbf{K}}\right),$$

where $\lambda_{\max}$ is the maximum eigenvalue. The differential of this quantity is given in the following theorem, which is proved in Appendix A.2.

Theorem 9 (Differential of COCO^4)**.** *Let* $\mathbf{K}$ *and* $\mathbf{L}$ *be two* $n \times n$ *Gram matrices, corresponding to the respective rows* $\mathbf{w}$ *and* $\mathbf{v}$ *of the unmixing matrix. Then*

$$\begin{aligned} d\lambda_{\max}\left(\widetilde{\mathbf{K}}\,\widetilde{\mathbf{L}}^2\,\widetilde{\mathbf{K}}\right) &= \mathbf{u}^\top \otimes \mathbf{u}^\top \left(\left[\left(\widetilde{\mathbf{K}}\,\widetilde{\mathbf{L}}^2\right) \otimes \mathbf{H} + \mathbf{H} \otimes \left(\widetilde{\mathbf{K}}\,\widetilde{\mathbf{L}}^2\right)\right] d\mathrm{vec}\mathbf{K} \right. \\ &\quad \left. + \left[\left(\widetilde{\mathbf{K}}\,\widetilde{\mathbf{L}}\right) \otimes \widetilde{\mathbf{K}} + \widetilde{\mathbf{K}} \otimes \left(\widetilde{\mathbf{K}}\,\widetilde{\mathbf{L}}\right)\right] d\mathrm{vec}\mathbf{L}\right), \end{aligned}$$

where $\otimes$ *is the Kronecker product*[5]*, and* $\mathbf{u}$ *is the eigenvector of* $\widetilde{\mathbf{K}}\,\widetilde{\mathbf{L}}^2\,\widetilde{\mathbf{K}}$ *associated with* $\lambda_{\max}$.

[5]The Kronecker product is a special case of the tensor product for matrices and, for $\mathbf{A} = (a_{ij})_{ij}$ and $\mathbf{B} = (b_{kl})_{kl}$ defined in terms of block matrices as $\mathbf{A} \otimes \mathbf{B} = (a_{ij}\mathbf{B})_{ij}$.

method	error	std	eval. total	searches	eval./search
Golden search	0.58	0.02	214.04	17.92	11.89
Quadratic (ql)	0.58	0.02	44.96	11.16	3.94
Fixed (fl)	0.60	0.02	39.04	20.00	1.90
Fixed (fq)	0.63	0.03	33.84	17.72	1.85

Table 1: ICA performance for different line searches using the Cholesky-based gradient with 8 sources and 20,000 observations. 'Quadratic': quadratic approximation of the dependence measure along the gradient, 'Fixed (fl)': fixed scheme, t_0/j at iteration j, 'Fixed (fq)': fixed scheme, $1.35^{-j}t_0$. The numbers are averages over 25 runs with artificial sources as described by Bach and Jordan (2002) and Gretton et al. (2005b). The average error at initialization (result from Jade) was 1.24 with a standard error of 0.06. 'Eval. total': total number of dependence measure evaluations, 'searches': number of line searches (iterations), 'eval./search': average number of measure evaluations per line search. The fixed schemes lead to the fewest evaluations of the measure but a slightly higher error. The quadratic method is slightly more expensive in terms of measure evaluations but needs fewer iterations, i.e. fewer computations of the gradient. Golden search is most robust but also most time-consuming.

The above expression may be simplified further for efficient computation, giving

$$d\lambda_{\max}\left(\widetilde{\mathbf{K}}\,\widetilde{\mathbf{L}}^2\,\widetilde{\mathbf{K}}\right) = 2\mathbf{u}^\top\mathbf{H}(d\mathbf{L})\widetilde{\mathbf{K}}^2\widetilde{\mathbf{L}}\mathbf{u} + 2\mathbf{u}^\top\widetilde{\mathbf{L}}\widetilde{\mathbf{K}}(d\mathbf{K})\widetilde{\mathbf{L}}\mathbf{u}. \tag{13}$$

Each entry of the above COCO gradient is a sequence of matrix-vector products. Hence we replace the kernel matrices by their sparse counterparts, and implement the multiplication by $\mathbf{H}$ as a centring operation of the vector. In addition, we need the dominant eigenvector for the product of each pair of Gram matrices. We compute this from the sparse matrices as well, so that only sparse matrix multiplications occur. Likewise, to compute the Cholesky approximation of the above gradient, we need only replace $\mathbf{K}$ and $\mathbf{L}$ with their low rank approximations $\mathbf{K}'$ and $\mathbf{L}'$, and substitute the expressions for $d\mathrm{vec}\mathbf{K}'$ and $d\mathrm{vec}\mathbf{L}'$ from the previous section into (13).

4.5 Using the free gradient to obtain a constrained gradient on the orthogonal group

In the previous two sections, we gave expressions for the free gradient of COCO and HSIC with respect to the unmixing matrix $\mathbf{W}$. We now describe how to transform these into gradients on the orthogonal group, i.e. on the manifold described by $m \times m$ matrices $\mathbf{A}$ for which $\mathbf{A}^\top\mathbf{A} = \mathbf{I}$. Gradient descent for functions defined on this manifold is described by Edelman et al. (1998). Let $f(\mathbf{W})$ be the particular dependence functional (COCO or HSIC) on which we wish to do gradient descent. We have expressions for the derivative

$$\mathbf{G} := \frac{\partial f(\mathbf{W})}{\partial \mathbf{W}}.$$

First, we compute the direction of steepest descent *on the manifold* from the free gradient $\mathbf{G}$, which is

$$\Delta_{\max} := \mathbf{G} - \mathbf{W}\mathbf{G}^\top\mathbf{W}.$$

Then, if we use t to parameterise displacement along a geodesic in the direction $\Delta_{\max}$ from an initial matrix $\mathbf{W}(0)$, the resulting $\mathbf{W}(t)$ is given by

$$\mathbf{W}(t) = \mathbf{W}(0)\exp\left(\frac{1}{2}t\mathbf{W}(0)^\top\Delta_{\max}\right),$$

which is a matrix exponential.

4.6 Gradient descent techniques

The ICA algorithm is a simple gradient descent procedure based on the code from (Bach and Jordan), who find a good step width in the direction of the projected gradient by a Golden search. In experiments, this search technique robustly finds a step width that decreases the independence measures. This robustness, however, comes with a large number of evaluations of the measure per line search (see Table 1). Two alternatives reduce the number of measure evaluations by more than $3/4$ with comparable results. Note that displacement is always along the geodesic in the direction of the gradient, parameterised by the step width t.

Our first approach is to approximate the independence measure along the curve by a quadratic polynomial.[6] To do so, it suffices to evaluate the measure at three points t_1, t_2, t_3 on the curve, one of them being the current position. The minimum of the polynomial indicates the estimated best step width in the direction of the gradient. At each iteration, the distance of the t_i from the current solution decreases. This method requires at least three evaluations of the dependence measure, and possibly more if the procedure is repeated due to the polynomial being concave.

In our second approach, which is even simpler, the step width is decreased by a fixed amount at each iteration. The move by the current step size along the gradient is only made if it actually reduces the dependence measure. This fixed scheme only requires evaluation of the measure at the current position and at the target of the current step. Such a method is the most efficient per iteration, but also the most sensitive with respect to the initial step width and the decrease of the width at each iteration.

Table 1 shows example results for the Cholesky-based gradient with Golden search, a quadratic approximation of the dependence measure along the gradient, and two fixed schemes: a step width of t_0/j or $(1.35)^{-j}t_0$ at iteration j, with initial width t_0. In the case of Golden search, the measure is computed more than 200 times with about 12 evaluations per search, whereas the quadratic approximation only evaluates the measure about 45 times with fewer searches and fewer evaluations per search. The fixed methods yield slightly worse results, but compute the measure least often. Note, however, that they also lead to more searches than the quadratic method and occasionally even the Golden search (see Figure 4). For each search, the gradient is recomputed if there was a move in the previous iteration. More searches may therefore mean more computations of the gradient. In summary, there is a tradeoff between robustness and time, and the quadratic approximation seems to provide a reasonable balance.

Finally, we note another approach to optimization on the orthogonal group by Eriksson and Koivunen (2003), who express $\mathbf{W}$ as a product of Jacobi rotations, solving iteratively for each rotation angle using smoothed approximations of the objective (the smoothing decreases the computational cost compared with a line search over the exact objective). This last method was only tested on small-scale problems, however, and its scaling behaviour for larger problems is unclear.

5 Complexity analysis

The unconstrained gradient is an $m \times m$ matrix. Its ith row corresponds to the derivative of the independence measure with respect to the ith row $\mathbf{w}_i$ of $\mathbf{W}$. We express the multi-unit independence measure as the sum of pairwise measures between all $\mathbf{w}_i\mathbf{T}$ and $\mathbf{w}_j\mathbf{T}$, where $\mathbf{T}$ is the $m \times n$ matrix of observations. Therefore $m-1$ pairwise terms of the form in Theorem 8 contribute to the ith row of the gradient. Once the $m(m-1)/2 = O(m^2)$ terms are computed, it takes another $O(m^2)$ to sum them up for the complete gradient. We first analyse the complexity for each pairwise term in HSIC, then we proceed to COCO.

The first factor $\widetilde{\mathbf{L}}$ in Theorem 8 requires $O(n^2)$ time, and each of the n^2 rows of the second factor $O(m)$ time. Since matrix multiplication takes $O(nmk)$ time for an $n \times k$ by $k \times m$ multiplication, the entire term is computed in $O(n^2 + n^2m) = O(n^2m)$ time.

Now replace the kernel matrices by sparse approximations with $z \ll n^2$ nonzero elements each. A sparse first factor $\widetilde{\mathbf{L}}'$ needs $O(z)$ time, the second factor $O(mz)$, totalling in $O(mz)$ for the entire term.

The $\partial\mathrm{vec}(\mathbf{K})/\partial\mathbf{w}_i$ type matrices in the Cholesky-based derivative (Equation (12)) cost $O(nmd)$, if $d \ll n$ columns are retained by the incomplete decomposition. In our implementation, we reuse the Cholesky decomposition from the dependence measure computation for each of the m Gram matrices, which takes $O(nmd^2)$. The remaining terms, including matrix inversion, are computable in $O(nd^2 + d^3)$ time. In sum, this pairwise term is in $O(nmd + nd^2 + d^3)$, or $O(nmd^2 + nd^2 + d^3)$ if the incomplete decomposition is included.

Thus, the complexity for each gradient is cubic in m. If we were to use full Gram matrices, the cost would be quadratic in n. However, the number of nonzero elements z and the number of columns d are much smaller than n. The former does not grow linearly in n, and the latter can be kept almost constant. Therefore the approximations reduce the quadratic cost. In addition, the space complexity of the matrices is reduced from $O(n^2)$ to $O(z)$ and $O(nd)$, and from $O(n^2m)$ to $O(zm)$ and $O(nmd)$ for $\partial\mathrm{vec}(\mathbf{K})/\partial\mathbf{w}_i$.

For the COCO gradient, the complexity of $\partial\mathrm{vec}(\mathbf{K})/\partial\mathbf{w}_i$ is the same as above. Each entry in the pairwise term in Theorem 9 further requires $O(n^2)$ for full matrices and $O(z+n)$ for sparse matrices, if the product is decomposed as matrix-vector products and $\mathbf{Hu}$ merely centres the vector $\mathbf{u}$. In total, we get $O(n^2m)$ for

[6] Our approach differs from Brent's method (Brent, 1973) in that the points used to fit the polynomial are always equally spaced, i.e. we do not perform golden sections. In addition, we move to the minimum of the polynomial even if the resulting displacement is outside the interval defined by the t_i, provided the dependence measure is lower there.

one pairwise term with full matrices and $O(n^2m^3)$ for the entire gradient. Sparse matrices reduce this cost to $O(m^3z + m^2(z+n)) = O(m^3z + nm^2)$.

The complexity of computing the maximum eigenvalue is more difficult to determine. The power method requires a repeated series of matrix-vector multiplications, but the speed of convergence is a function of the gap between the first and second eigenvalues (Golub and Van Loan, 1996, Section 7.3.1). Nevertheless, the complexity of the multiplication in the power method is reduced by replacing the full matrices by sparse approximations.

6 Experiments

In this section, we test our method on the data sets described by Gretton et al. (2005b), but with a much larger number of samples and sources. We compare with classical methods suited to medium and larger-scale problems, namely Jade (Cardoso, 1998) and FastICA (Hyvärinen et al., 2001). These also provide starting points for the optimisation. Comparisons of kernel ICA with several additional ICA methods are given by Gretton et al. (2005b) and Gretton et al. (2005a).

We begin in Section 6.1 with a measure of algorithm performance on ICA, the *Amari divergence* between the inverse of the true mixing matrix and the estimated inverse $\mathbf{W}$. After an outline of the experimental setup in Section 6.2, we compare our methods in the following sections on 20,000 observations of mixtures of 8 artificial sources, and 20,000 observations of 8 sources representing music from various genres. We also present results for larger scale data, i.e. 30,000 samples from 16 and 32 artificial sources.

6.1 Measurement of Performance

We use the Amari divergence, defined by Amari et al. (1996), as an index of ICA algorithm performance: this is an adaptation and simplification of a criterion proposed earlier by Comon (1994). Note that the properties of this quantity in Lemma 11 were not described by Amari et al. (1996), but follow from the proof of Comon (1994).

Definition 10 (Amari divergence)**.** *Let* $\mathbf{B}$ *and* $\mathbf{W}$ *be two* $n \times n$ *matrices, where* $\mathbf{B}$ *is the mixing matrix and* $\mathbf{W}$ *the estimated unmixing matrix (these need not be orthogonal here), and let* $\mathbf{D} = \mathbf{WB}$*. Then the Amari divergence between* $\mathbf{B}^{-1}$ *and* $\mathbf{W}$ *is*

$$\mathcal{D}(\mathbf{WB}) = \frac{100}{2n(n-1)} \left[\sum_{i=1}^{n} \left(\frac{\sum_{j=1}^{n} |d_{ij}|}{\max_j |d_{ij}|} - 1 \right) + \sum_{j=1}^{n} \left(\frac{\sum_{i=1}^{n} |d_{ij}|}{\max_i |d_{ij}|} - 1 \right) \right].$$

Although this measure is not a distance metric for general matrices $\mathbf{B}^{-1}$ and $\mathbf{W}$, it nonetheless possesses certain useful properties, as shown below.

Lemma 11 (Properties of the Amari divergence)**.** *The Amari divergence* $\mathcal{D}(\mathbf{WB})$ *between the* $n \times n$ *matrices* $\mathbf{B}^{-1}$*,* $\mathbf{W}$ *has the following properties:*

- $0 \leq \mathcal{D}(\mathbf{WB}) \leq 100$*. The factor of 100 is not part of the original definition of Amari et al. (1996), who defined the Amari divergence on* $[0, 1]$*.* In our experiments*, however, the Amari divergence was generally small, and we scaled it by 100 to make the results tables more readable.*
- *Let* $\mathbf{P}$ *be an arbitrary permutation matrix (a matrix with a single* 1 *in each row and column, and with remaining entries* 0*), and* $\mathbf{S}$ *be a diagonal matrix of non-zero scaling factors. Then* $\mathbf{W} = \mathbf{B}^{-1}$ *if and only if* $\mathcal{D}(\mathbf{WB}) = 0$*, or equivalently* $\mathcal{D}(\mathbf{WBSP}) = 0$ *or* $\mathcal{D}(\mathbf{SPWB}) = 0$*.*

The final property in the above lemma is particularly useful in the context of ICA, since it causes our performance measure to be invariant to output ordering ambiguity once the sources have been demixed.

6.2 Experimental Setup

We replace the gradient and line search in the downloadable code by Bach and Jordan with our gradients and line search methods. We do not apply the polishing step used by Bach and Jordan (2002); Gretton et al. (2005b).

In all experiments, the data is generated as follows. The m sources are chosen randomly from a particular data pool (artificial or music), and n samples are generated according to each of the m distributions. The $m \times n$ data matrix is multiplied by a random mixing matrix $\mathbf{B}$ with condition number between one and two, and the resulting mixed data is then whitened. We initialise our algorithm using Jade for 8 sources, and FastICA for 16 and 32 sources. If not stated otherwise, the experiments are repeated 25 times.

All experiments employ the Gaussian Kernel,

$$k(x, x') = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right), \tag{14}$$

with a width $\sigma = 0.5$. To stabilise the inversion of $\mathbf{K}_{I,I}$ for the Cholesky-based gradient, we add a ridge of 10^{-6}. The precision for the incomplete Cholesky decomposition is set to ηn, where n is the number of samples. For the artificial sources, we set $\eta = 10^{-4}$, and for the music signals, $\eta = 10^{-6}$. The sparse matrices have an expected percentage of 0.35% nonzero entries.

We test ICA both with HSIC and COCO, and different gradients and line search methods. The gradients we test are

- HSIC with sparse kernel matrices,
- HSIC with the Cholesky-based gradient, and
- COCO with sparse kernel matrices.

For the COCO gradient, we investigate two variations: (i) using the eigenvector as in Theorem 9, and (ii) replacing the eigenvector by a random vector. The latter saves the computation of the eigenvector, which is otherwise based on the sparse matrices (we will comment further on the consequences of this heuristic in practice). We combine these gradients with the following line search methods:

(g) Golden search,

(ql) Quadratic approximation of the dependence measure along the gradient, where at iteration j the measure is evaluated at three points that are t_0/j apart,

(qq) Quadratic approximation of the dependence measure along the gradient, where at iteration j the measure is evaluated at three points that are $2^{-j}t_0$ apart,

(fl) Fixed decay of the step width: t_0/j for iteration j,

(fq) Fixed decay of the step width: $(1.35)^{-j}t_0$ for iteration j,

(fq2) Fixed decay of the step width: $(1.1)^{-j}t_0$ for iteration j,

where t_0 is the initial step size. For the first 38 iterations, the step size is greater with (fq2) than with (fl). In the experiments presented below, there are usually less than 20 iterations. Table 2 summarises the abbreviations used in the plots and tables.

6.3 General Mixtures of Artificial Data

We first explore performance for a collection of 18 source distributions used by Bach and Jordan (2002) and Gretton et al. (2005b). These distributions include sub- and super-Gaussian sources, near-Gaussian distributions, and both uni- and multimodal distributions.

Table 3 and Figure 1 show results for 8 sources and $20{,}000$ samples. Each gradient is combined with a line search: Golden search, a quadratic approximation scheme, or a fixed scheme of step decay. We summarise the average computational cost of the various methods in Figure 3. The measurements were repeated five times for each of 25 mixtures from the artificial sources, on cluster nodes with 64 bit Opteron CPUs running Debian/GNU Linux 3.1. Our experiments lead to the following conclusions:

- The best combination of performance and cost is attained by HSIC with a Cholesky approximation and quadratic interpolation. More generally, for both HSIC and COCO, the performance of quadratic interpolation is very similar to Golden search, but the cost of the former is significantly lower.
- The sparse and Cholesky HSIC approximations have a similar cost, but the Cholesky approximation performs better.

Gradient	
HC	HSIC, Cholesky-based (Section 4.3)
HS	HSIC with sparse approximations (Section 4.2)
CS	COCO with sparse approximations (Section 4.4)
CR	COCO with sparse approximations, eigenvector $\mathbf{u}$ replaced by a random unit vector
Line Search	
g	Golden search
ql	Quadratic approximation of the dependence measure along the gradient, measure evaluated at points of distance t_0/j at iteration j with initial step size t_0
qq	Quadratic approximation of the dependence measure along the gradient, measure evaluated at points of distance $2^{-j}t_0$ at iteration j with initial step size t_0
fl	Fixed scheme: at iteration j, move t_0/j along the gradient if it improves the dependence measure
fq	Fixed scheme: at iteration j, move $(1.35)^{-j}t_0$ along the gradient if it improves the dependence measure
fq2	Fixed scheme: at iteration j, move $(1.1)^{-j}t_0$ along the gradient if it improves the dependence measure

Table 2: Abbreviations for the gradients and line search methods for the tables and figures.

– Sparse quadratic COCO with a random vector replacing the eigenvector performs remarkably well, with similar Amari error to sparse quadratic HSIC but significantly lower cost. In general, the random vector is better than the real eigenvector, since this appears to decrease the number of critical points (with zero derivative) that arise due to the inherent difficulty in sparsely approximating a matrix with small spectrum, while retaining the minimum at independence. It even outperforms the original method of Gretton et al. (2005b), which uses an incomplete Cholesky approximation and computes the gradient via finite differences. This sparse COCO approach might be worth pursuing for very large-scale problems, where incomplete Cholesky is no longer feasible.

– Although they are cheap, methods using fixed step widths give mixed performance when HSIC is the dependence measure, doing well with Cholesky, but poorly with random sampling. These methods are also hard to tune, being sensitive to the initial step width choice (see Section 6.5). We were not able to obtain useful results using fixed step widths with sparse COCO.

Our methods also scale up to larger data sets. Table 4 shows results for $30{,}000$ samples from 16 and 32 sources. We use FastICA with the kurtosis-based nonlinearity (similar to Jade) instead of Jade for initialisation. The results demonstrate the Amari error at initialisation can be halved.

6.4 Number of Iterations and Dependence Measure Evaluations

We now give a more detailed breakdown of the behaviour of the various gradient descent procedures, as it contributes to their overall computational cost. Figure 2 illustrates the total number of dependence measure evaluations and iterations of the gradient descent for the results in Table 3 and Figure 1. Golden search is robust, but requires significantly more evaluations of the dependence measure. Performance is similar with a quadratic interpolation, but the latter computes the measure far less often. With fixed decay schemes, the measure is evaluated even fewer times.

During most iterations, we recompute the gradient, so the average number of iterations indicates how often the gradient is evaluated. Golden search is again more expensive than a quadratic approximation, except when COCO with a random vector replacing the eigenvector is used. The fixed schemes lead to more iterations than the other searches with the Cholesky-based gradient, but fewer with the sparse gradients. That said, sparse methods work poorly, or not at all, with fixed schemes.

6.5 Sensitivity to Step Sizes

Each line search method is initialised with a certain step size to start with. Figure 4 shows the sensitivity of the various methods with the Cholesky-based HSIC gradient, for initial step widths from 100 to 0.01. The results with Golden search are consistent over the entire range of initial widths, because of the expensive search at each

gradient	g	ql	qq	fl	fq	fq2
HC	0.58 ± 0.02	0.58 ± 0.02	0.59 ± 0.02	0.60 ± 0.02	0.63 ± 0.03	0.66 ± 0.04
HS	0.72 ± 0.02	0.72 ± 0.02	0.78 ± 0.02	1.21 ± 0.06	1.14 ± 0.06	1.14 ± 0.06
CS	1.17 ± 0.06	1.17 ± 0.06	1.19 ± 0.06	—	—	—
CR	0.74 ± 0.03	0.77 ± 0.03	0.72 ± 0.03	—	—	—

Table 3: ICA performance (mean Amari error for 25 runs) for 8 sources and 20,000 observations with sparse and Cholesky-based gradients and various line searches (see table Table 2 for the abbreviations). The sparsity of the approximating kernel matrices is 0.35% and the mean error of the initialization is 1.24 ± 0.06. The Cholesky-based gradient leads to the best overall results with the lowest variance. The fixed-step approach did not work for sparse COCO.

m	n	FastICA	HCql	iterations	dep. meas. eval.
16	30,000	1.12 ± 0.05	0.52 ± 0.02	18.47	74.84
32	30,000	1.10 ± 0.02	0.54 ± 0.01	20.00	80.64

Table 4: Results for larger scale Kernel ICA: Cholesky-based HSIC gradient with a quadratic approximation of the dependence measure along the gradient to find a suitable step width. The algorithm was initialised with the result of FastICA. The last two columns show the average number of iterations of the gradient descent and the average number of times the dependence measure was computed.

iteration. The quadratic approximation is fairly robust, except for the large start width of 100. Fixed decay schemes, on the other hand, leave little freedom to adapt the step width and thus depend on a good initialization. The quadratic approximation may be a reasonable balance between good performance and parameter tuning.

6.6 Audio Signal Demixing

The second type of data we investigate are brief extracts from various music genres chosen from the ICA benchmark data of (Pearlmutter). The music clips consist of 5 second segments, sampled at 11 kHz with a precision of 8 bits. Adjacent samples of the audio signals are certainly not independent, but ICA algorithms have still succeeded in demixing such signals. As in (Gretton et al., 2005b), the signals are randomly permuted to reduce the statistical dependence of adjacent samples.

Figure 5 displays results with the Cholesky-based HSIC gradient and all our line searches. We only present the best result over the variation of initial step widths. In this case, the simpler search methods lead to even better results than Golden search. With the quadratic approximation, the slower decrease of the interval between the test points along the gradient (HCql) yields better results than the faster shrinking (HCqq). Likewise, the slowest converging step width decay (HCfq2) decreases the error more than the other fixed schemes. Slower convergence means more iterations in the fixed schemes. This is not true of the quadratic approximation, however, because the step width is only indirectly influenced by the shrinking: HCqq performed 10.24 iterations on average, whereas HCql only needed an average of 4.80 iterations.

7 Conclusions and Future Directions

We have shown that kernel ICA may be applied to large-scale problems (up to 32 sources and 30,000 samples in our experiments), using a variety of techniques that substantially reduce its computational cost. These include

- efficient approximations of the independence measures and their derivatives, using both sparse approximations of the Gram matrices and incomplete Cholesky decomposition of these matrices. Prior approaches to computing the derivatives in kernel ICA were especially expensive, scaling as $O(m^4)$ (where m is the number of sources); our methods are $O(m^3)$;

- better search strategies along the direction of descent, which require far fewer independence measure evaluations than the Golden search used in previous implementations.

As an interesting consequence of our experiments, we observe that the Cholesky approximation performs better than sparsely sampling the matrix entries (at similar computational cost). Thus, while sparse approximation may be the only feasible option for yet larger scale problems ($> 10^6$ samples), the greedy approximation strategy implemented via the Cholesky approach is to be preferred on the sample sizes we investigated.

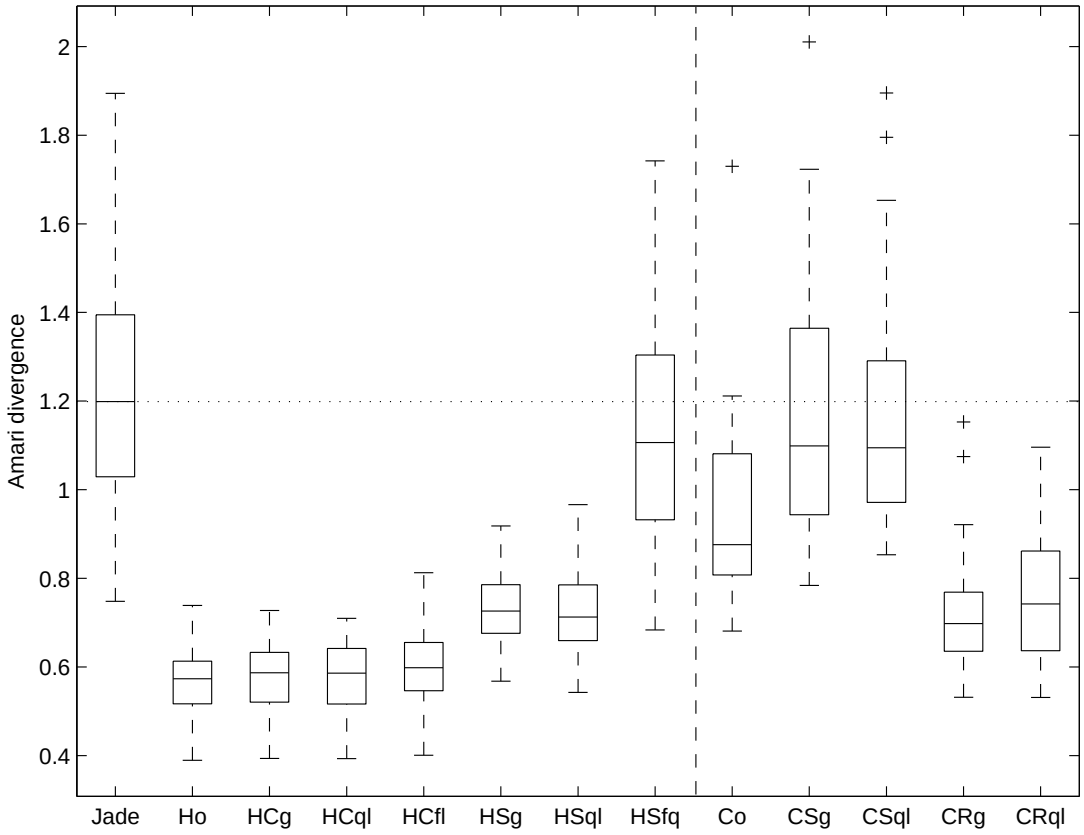

Figure 1: Amari error for 8 artificial sources and 20,000 observations for the sparse and Cholesky-based gradients (25 runs). For HSIC, we show the performance with Golden search, one quadratic interpolation method, and one fixed decay scheme; for COCO, we display the Golden and quadratic approaches. Jade was used to initialise the search. 'Ho' and 'Co' denote the original HSIC and COCO algorithms, respectively (Golden search, gradients computed by finite differences).

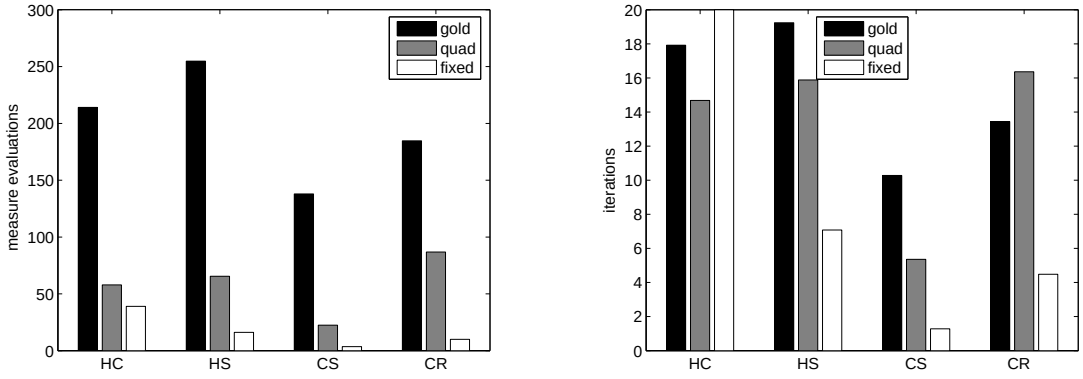

Figure 2: Average number of dependence measure evaluations (left) and iterations of the gradient descent (right) for the experiments in Table 3 and Figure 1. Golden search leads to the most computations of the dependence measure for all gradients, and the fixed decay schemes are the cheapest in this respect. Results are mixed in the case of the total number of gradient descent steps taken: there is generally little difference between the Golden and quadratic algorithms, while the fixed step size requires fewer iterations in three of four cases.

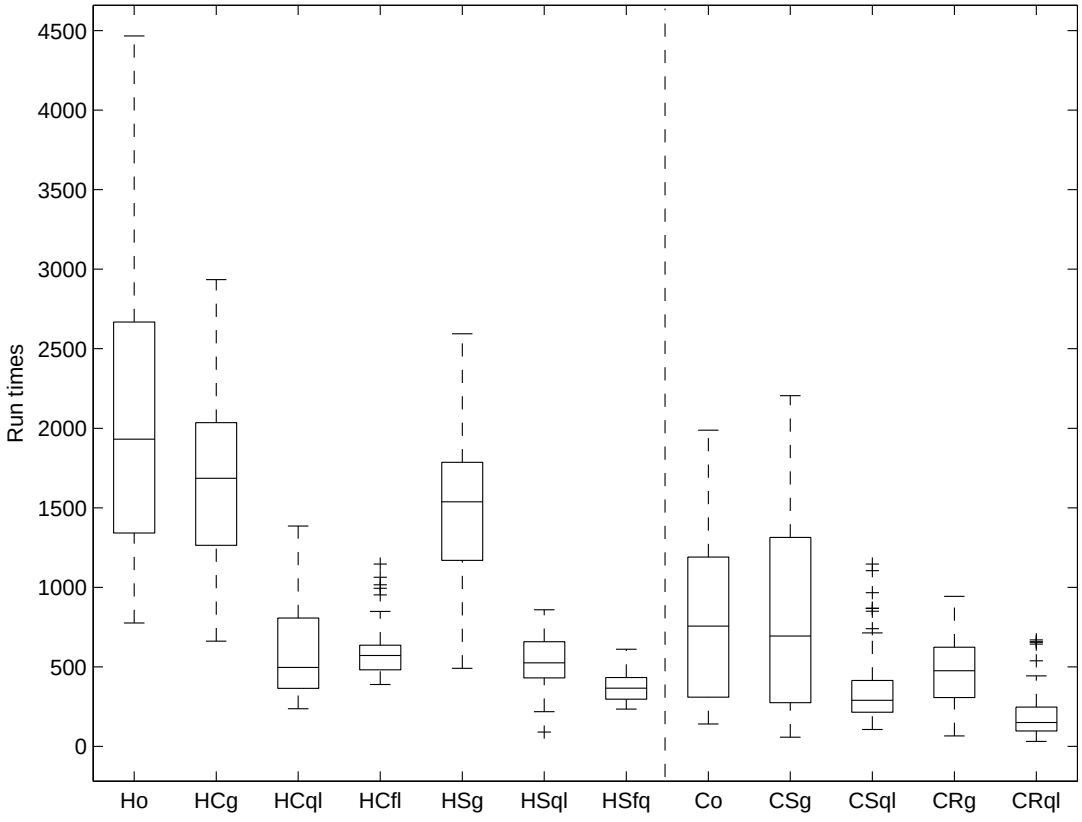

Figure 3: Running times for 8 sources and 20,000 observations of artificial data. The number of iterations and dependence measure evaluations are reflected in the times: Golden Search is usually the slowest method for a given gradient. 'Ho' and 'Co' refer to the original algorithms, with gradients computed via finite differences over all possible Jacobi rotations. The approximated analytic gradients compare favorably to the original method.

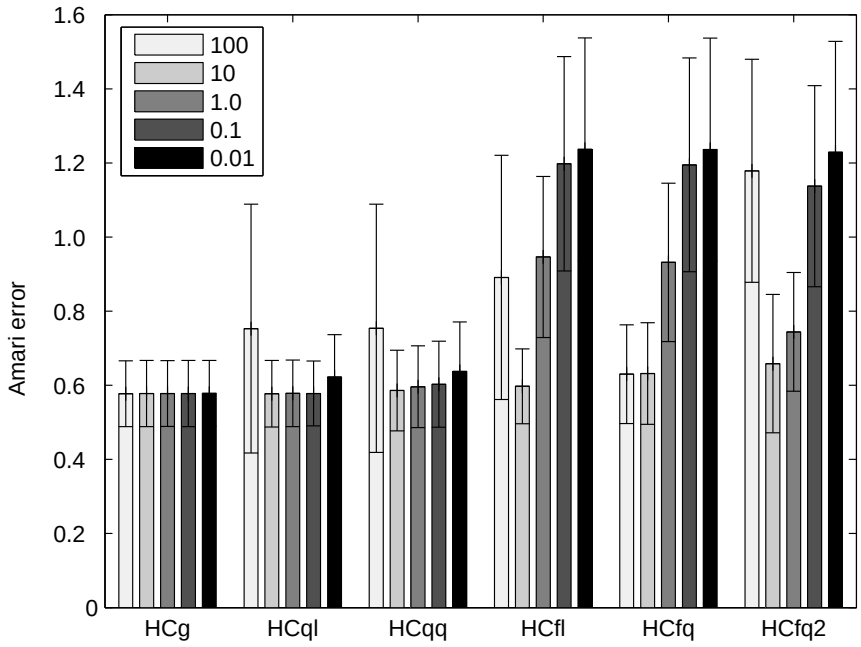

Figure 4: Influence of the initial step width on ICA performance for the Cholesky-based HSIC gradient (8 sources, 20,000 observations of artificial data). Golden search (HCg) is most robust with respect to the initial step size, and the quadratic approximation (HCql, HCqq) is fairly robust within a certain range. The fixed schemes of step decay (HCfl, HCfq, HCfq2) are most sensitive to step width, and may require parameter tuning. The data are averages over 25 runs, and the error bars represent the standard deviation.

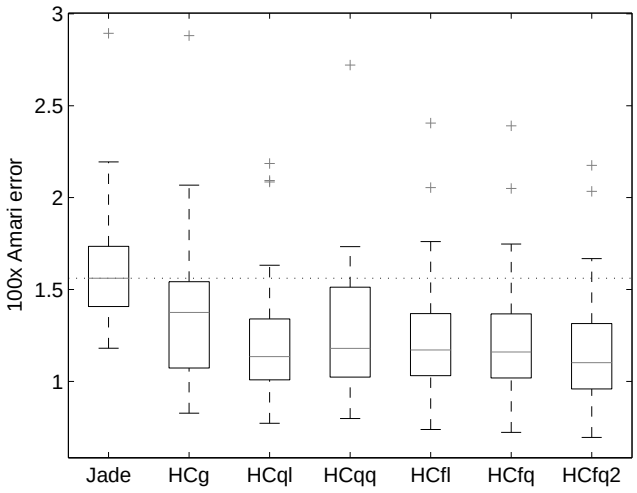

Figure 5: Amari error for the Cholesky-based HSIC gradient with audio signals (8 sources, 20,000 samples).

A number of additional strategies exist to further reduce the cost and improve performance of kernel ICA. One could combine the approximated gradients with a rapid stochastic gradient descent algorithm, such as stochastic meta-descent (SMD) by Vishwanathan et al. (2006). This should further speed convergence, since SMD includes information about the curvature. In addition, the linear cost per iteration makes SMD particularly applicable to large data sets.

Geometric optimisation on the orthogonal group may also improve the algorithm. Recently, an approximate Newton-like method has successfully been applied to one-unit nonparametric ICA (Shen et al., 2006). The similar, fully parallelised method on the orthogonal group applies to multi-unit ICA (Shen and Hüper, 2006b). The extension of these Newton-like geometric methods to multi-unit kernel ICA is therefore of interest.

Acknowledgement

We thank Francis Bach and Aiyou Chen, for providing us with their ICA implementations; Dimitris Achlioptas and Matthias Seeger, for helpful discussions regarding sparse and low rank matrix approximation; and Christian Walder, for providing us with his sparse matrix multiplication code.

A Appendix

A.1 Differential of the incomplete Cholesky approximation to HSIC

In this appendix, we obtain the differential of the incomplete Cholesky approximation to HSIC^2 described in Section 4.3. Recall that the differential of this low rank approximation is

$$d\mathbf{tr}\left(\widetilde{\mathbf{K}}'\widetilde{\mathbf{L}}'\right) = \mathbf{tr}\left(\widetilde{\mathbf{K}}' d(\mathbf{L}')\right) + \mathbf{tr}\left(\widetilde{\mathbf{L}}' d(\mathbf{K}')\right). \tag{15}$$

The differential of $\mathbf{K}' = \mathbf{K}_{:,I}\mathbf{K}_{I,I}^{-1}\mathbf{K}_{I,:}$ is

$$d\mathbf{K}' = d(\mathbf{K}_{:,I})\mathbf{K}_{I,I}^{-1}\mathbf{K}_{I,:} - \mathbf{K}_{:,I}\mathbf{K}_{I,I}^{-1}(d\mathbf{K}_{I,I})\mathbf{K}_{I,I}^{-1}\mathbf{K}_{I,:} + \mathbf{K}_{:,I}\mathbf{K}_{I,I}^{-1}d(\mathbf{K}_{I,:}). \tag{16}$$

Making this replacement in the second term $\mathbf{tr}\left(\widetilde{\mathbf{L}}' d(\mathbf{K}')\right)$ from (15) yields

$$\begin{aligned}
&\mathbf{tr}\left(\widetilde{\mathbf{L}}' d(\mathbf{K}')\right) \\
&= \mathbf{tr}\left(\widetilde{\mathbf{L}}' \mathrm{d}\mathbf{K}_{:,I}\mathbf{K}_{I,I}^{-1}\mathbf{K}_{I,:}\right) - \mathbf{tr}\left(\widetilde{\mathbf{L}}'\mathbf{K}_{:,I}\mathbf{K}_{I,I}^{-1}(d\mathbf{K}_{I,I})\mathbf{K}_{I,I}^{-1}\mathbf{K}_{I,:}\right) + \mathbf{tr}\left(\widetilde{\mathbf{L}}'\mathbf{K}_{:,I}\mathbf{K}_{I,I}^{-1}d\mathbf{K}_{I,:}\right) \\
&= \mathbf{tr}\left(\widetilde{\mathbf{L}}' d\mathbf{K}_{:,I}\mathbf{K}_{I,I}^{-1}\mathbf{K}_{I,:}\right) - \mathbf{tr}\left(\widetilde{\mathbf{L}}'\mathbf{K}_{:,I}\mathbf{K}_{I,I}^{-1}(d\mathbf{K}_{I,I})\mathbf{K}_{I,I}^{-1}\mathbf{K}_{I,:}\right) \\
&\quad + \mathbf{tr}\left((d\mathbf{K}_{:,I}\mathbf{K}_{I,I}^{-1}\mathbf{K}_{I,:})^\top \widetilde{\mathbf{L}}'^\top\right) \\
&= \mathbf{tr}\left(\widetilde{\mathbf{L}}' d\mathbf{K}_{:,I}\mathbf{K}_{I,I}^{-1}\mathbf{K}_{I,:}\right) - \mathbf{tr}\left(\widetilde{\mathbf{L}}'\mathbf{K}_{:,I}\mathbf{K}_{I,I}^{-1}(d\mathbf{K}_{I,I})\mathbf{K}_{I,I}^{-1}\mathbf{K}_{I,:}\right) + \mathbf{tr}\left(\widetilde{\mathbf{L}}' d\mathbf{K}_{:,I}\mathbf{K}_{I,I}^{-1}\mathbf{K}_{I,:}\right) \\
&= 2\,\mathbf{tr}\left(\mathbf{K}_{I,I}^{-1}\mathbf{K}_{I,:}\widetilde{\mathbf{L}}' d\mathbf{K}_{:,I}\right) - \mathbf{tr}\left(\mathbf{K}_{I,I}^{-1}\mathbf{K}_{I,:}\widetilde{\mathbf{L}}'\mathbf{K}_{:,I}\mathbf{K}_{I,I}^{-1}d\mathbf{K}_{I,I}\right) \\
&= 2\,\mathrm{vec}(\widetilde{\mathbf{L}}'\mathbf{K}_{:,I}\mathbf{K}_{I,I}^{-1})^\top \mathrm{vec}(d\mathbf{K}_{:,I}) - \mathrm{vec}(\mathbf{K}_{I,I}^{-1}\mathbf{K}_{I,:}\widetilde{\mathbf{L}}'\mathbf{K}_{:,I}\mathbf{K}_{I,I}^{-1})^\top \mathrm{vec}(d\mathbf{K}_{I,I}),
\end{aligned}$$

where we make use of the symmetry of the Gram matrices. The other term $\mathbf{tr}\left(\widetilde{\mathbf{K}}' d(\mathbf{L}')\right)$ in (15) is equivalent.

A.2 Differential of COCO

In this appendix, we obtain the differential of COCO^4 in Theorem 9 with respect to two rows of the demixing matrix $\mathbf{W}$. We again use the formalism of Magnus and Neudecker (1999, p. 175): if $\mathbf{F}(\mathbf{X})$ is a matrix function of the matrix $\mathbf{X}$, then

$$d\mathrm{vec}\mathbf{F}(\mathbf{X}) = (\mathrm{D}\mathbf{F}(\mathbf{X}))d\mathrm{vec}\mathbf{X},$$

where $\mathrm{D}\mathbf{F}(\mathbf{X})$ is the matrix of derivatives. We also need the result of Magnus and Neudecker (1999, p. 180): if λ is an eigenvalue of a symmetric, real valued matrix $\mathbf{A}$ with associated eigenvector $\mathbf{u}$, then

$$\begin{aligned}
\mathrm{D}\lambda(\mathbf{A}) &= \frac{\partial \lambda}{\partial (\mathrm{vec}\mathbf{A})^\top} \\
&= \mathbf{u}^\top \otimes \mathbf{u}^\top,
\end{aligned}$$

where $\otimes$ is the Kronecker product. Finally, using

$$\mathrm{vec}(\mathbf{ABC}) = (\mathbf{C}^\top \otimes \mathbf{A})\mathrm{vec}\mathbf{B},$$

we compute the differential:

$$\begin{aligned}
& d\mathrm{vec}\lambda_{\max}\left(\widetilde{\mathbf{K}}\,\widetilde{\mathbf{L}}^2\,\widetilde{\mathbf{K}}\right) \\
&= d\lambda_{\max}\left(\widetilde{\mathbf{K}}\,\widetilde{\mathbf{L}}^2\,\widetilde{\mathbf{K}}\right) \\
&= \frac{\partial}{\partial \mathrm{vec}\mathbf{A}}\,\lambda_{\max}(\mathbf{A})|_{\mathbf{A}=\widetilde{\mathbf{K}}\,\widetilde{\mathbf{L}}^2\,\widetilde{\mathbf{K}}}\, d\mathrm{vec}(\widetilde{\mathbf{K}}\,\widetilde{\mathbf{L}}^2\,\widetilde{\mathbf{K}}) \\
&= \mathbf{u}^\top \otimes \mathbf{u}^\top \mathrm{vec}\,\Big[\mathbf{H}\,(d\mathbf{K})\,\mathbf{H}\widetilde{\mathbf{L}}^2\,\widetilde{\mathbf{K}} + \widetilde{\mathbf{K}}\,\mathbf{H}\,(d\mathbf{L})\,\mathbf{H}\widetilde{\mathbf{L}}\,\widetilde{\mathbf{K}} \\
&\qquad + \widetilde{\mathbf{K}}\,\widetilde{\mathbf{L}}\mathbf{H}\,(d\mathbf{L})\,\mathbf{H}\,\widetilde{\mathbf{K}} + \widetilde{\mathbf{K}}\,\widetilde{\mathbf{L}}^2\mathbf{H}\,(d\mathbf{K})\,\mathbf{H}\Big] \\
&= \mathbf{u}^\top \otimes \mathbf{u}^\top \Big(\Big[\Big(\widetilde{\mathbf{K}}\,\widetilde{\mathbf{L}}^2\Big) \otimes \mathbf{H} + \mathbf{H} \otimes \Big(\widetilde{\mathbf{K}}\,\widetilde{\mathbf{L}}^2\Big)\Big]\, d\mathrm{vec}\mathbf{K} \\
&\qquad + \Big[\Big(\widetilde{\mathbf{K}}\,\widetilde{\mathbf{L}}\Big) \otimes \widetilde{\mathbf{K}} + \widetilde{\mathbf{K}} \otimes \Big(\widetilde{\mathbf{K}}\,\widetilde{\mathbf{L}}\Big)\Big]\, d\mathrm{vec}\mathbf{L}\Big),
\end{aligned}$$

where we use $\mathbf{H} = \mathbf{HH}$ to absorb the extra centring matrices.

References

S. Achard, D. T. Pham, and C. Jutten. Quadratic dependence measure for nonlinear blind source separation. In *4th International Conference on ICA and BSS*, 2003.

D. Achlioptas and F. McSherry. Fast computation of low-rank matrix approximations. In *Proceedings of the 33rd ACM Symposium on Theory of Computing*, pages 611–618, 2001.

S.-I. Amari, A. Cichoki, and H. Yang. A new learning algorithm for blind signal separation. In *Advances in Neural Information Processing Systems*, volume 8, pages 757–763. MIT Press, 1996.

F. R. Bach and M. I. Jordan. Kernel independent component analysis. *Journal of Machine Learning Research*, 3: 1–48, 2002.

F. R. Bach and M. I. Jordan. Kernel independent component analysis - (matlab code, version 1.1). `http://www.cs.berkeley.edu/~fbach/kernel-ica/index.htm`.

C. R. Baker. Joint measures and cross-covariance operators. *Transactions of the American Mathematical Society*, 186:273–289, 1973.

R. P. Brent. *Algorithms for Minimization without Derivatives*. Prentice Hall, 1973.

J.-F. Cardoso. Blind signal separation: statistical principles. *Proceedings of the IEEE*, 90(8):2009–2026, 1998.

A. Chen. Fast kernel density independent component analysis. In *Proceedings of 6th international conference on ICA and BSS*, volume 3889 of *Lecture Notes in Computer Science*, pages 24 – 31, Heidelberg, 2006. Springer.

A. Chen and P. Bickel. Consistent independent component analysis and prewhitening. *IEEE Transactions on Signal Processing*, 53(10):3625–3632, 2005.

P. Comon. Independent component analysis, a new concept? *Signal Processing*, 36:287–314, 1994.

P. Drineas and M. W. Mahoney. On the Nyström method for approximating a gram matrix for improved kernel-based learning. *JMLR*, 6:2153–2175, 2005.

P. Drineas, R. Kannan, and M. W. Mahoney. Fast Monte Carlo algorithms for matrices I: Approximating matrix multiplication. Technical Report YALEU/DCS/TR-1270, Yale, 2004.

A. Edelman, T. A. Arias, and S. T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM Journal on Matrix Analysis and Applications*, 20(2):303–353, 1998.

J. Eriksson and V. Koivunen. Characteristic-function-based independent component analysis. *Signal Processing*, 83(10):2195 – 2208, 2003.

A. Feuerverger. A consistent test for bivariate dependence. *International Statistical Review*, 61(3):419–433, 1993.

S. Fine and K. Scheinberg. Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research*, 2:243–264, Dec 2001.

K. Fukumizu, F. R. Bach, and M. I. Jordan. Dimensionality reduction for supervised learning with reproducing kernel Hilbert spaces. *Journal of Machine Learning Research*, 5:73–99, 2004.

G. H. Golub and C. F. Van Loan. *Matrix Computations*. John Hopkins University Press, Baltimore, MD, 3rd edition, 1996.

A. Gretton, O. Bousquet, A. J. Smola, and B. Schölkopf. Measuring statistical dependence with Hilbert-Schmidt norms. In *Proc. Intl. Conf. on Algorithmic Learning Theory*, pages 63–78, 2005a.

A. Gretton, R. Herbrich, A. J. Smola, O. Bousquet, and B. Schölkopf. Kernel methods for measuring independence. *Journal of Machine Learning Research*, 6:2075–2129, 2005b.

A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. John Wiley, 2001.

S. Jegelka and A. Gretton. *Large Scale Kernel Machines*, chapter Brisk Kernel ICA, pages 225–250. MIT Press, 2007.

A. Kankainen. *Consistent Testing of Total Independence Based on the Empirical Characteristic Function*. PhD thesis, University of Jyväskylä, 1995.

C. Ku and T. L. Fine. Testing for stochastic independence: application to blind source separation. *IEEE Transactions on Signal Processing*, 53(5):1815 – 1826, 2005.

E. Learned-Miller and J. Fisher III. ICA using spacings estimates of entropy. *Journal of Machine Learning Research*, 4:1271–1295, 2003.

J. R. Magnus and H. Neudecker. *Matrix Differential Calculus with Applications in Statistics and Econometrics (Revised Edition)*. John Wiley and Sons, 1999.

N. Murata. Properties of the empirical characteristic function and its application to testing for independence. In *Proceedings of 3rd International Conference on ICA and Blind Source Separation*, pages 19–24, 2001.

A. Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw Hill, New York, 1991.

B. A. Pearlmutter. Music samples to illustrate the context-sensitive generalisation of ICA. URL `http://www.cs.unm.edu/~bap/demos.html`.

B. Schölkopf, A. J. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

H. Shen and K. Hüper. Local convergence analysis of FastICA. In *Proceedings of 6th international conference on ICA and BSS*, volume 3889 of *Lecture Notes in Computer Science*, pages 893 – 900, Heidelberg, 2006a. Springer.

H. Shen and K. Hüper. Newton-like methods for parallel independent component analysis. In *IEEE International Workshop on Machine Learning for Signal processing (MLSP 2006)*, 2006b.

H. Shen, K. Hüper, and A. Smola. Newton-like methods for nonparametric independent component analysis. In *International Conference on Neural Information Processing*, 2006. to appear.

A. J. Smola and B. Schölkopf. Sparse greedy matrix approximation for machine learning. In P. Langley, editor, *Proc. Intl. Conf. Machine Learning*, pages 911–918, San Francisco, 2000. Morgan Kaufmann Publishers.

I. Steinwart. On the influence of the kernel on the consistency of support vector machines. *Journal of Machine Learning Research*, 2:67–93, 2002.

H. Stögbauer, A. Kraskov, S. Astakhov, and P. Grassberger. Least dependent component analysis based on mutual information. *Phys. Rev. E*, 70(6):066123, 2004.

S. V. N. Vishwanathan, N. N. Schraudolph, and A. J. Smola. Step size adaptation in reproducing kernel hilbert space. *Journal of Machine Learning Research*, 7:1107–1133, 2006.

C. K. I. Williams and M. Seeger. Using the Nystrom method to speed up kernel machines. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, 2000.