Kristian Nybo

Dimensionality reduction methods for fMRI analysis and visualization

School of Science

Thesis submitted for examination for the degree of Licentiate of Science in Technology.

Espoo September 6, 2015

Thesis supervisor and advisor:

Prof. Samuel Kaski

Aalto University
School of Science

AALTO-YLIOPISTO
PERUSTIETEIDEN KORKEAKOULU

LISENSIAATINTUTKIMUKSEN
TIIVISTELMÄ

Tekijä: Kristian Nybo

Työn nimi: Dimensionpudotusmenetelmiä fMRI-analyysissä ja visualisoinnissa

Päivämäärä: September 6, 2015 Kieli: Englanti Sivumäärä: 8+43

Tietojenkäsittelytieteen laitos

Professuuri: Tietojenkäsittelytiede Koodi: T-61

Valvoja ja ohjaaja: Prof. Samuel Kaski

Monilla aloilla esiintyy tarve korkeaulotteisen, kohinaisen datan analysoimiseen. Algoritminen dimensionpudotus tai muuttujanvalinta ovat usein sovellettavia lähestymistapoja, joko muuta analyysiä edeltävänä esikäsittelynä tai itsenäisenä analyysinä.

Tässä työssä käsitellään sekä dimensionpudotusta että muuttujanvalintaa, keskittyen erityisesti fMRI-dataan ja visualisointiin. Työssä esitellään kolme uutta algoritmia.

Ensimmäinen algoritmi käyttää harvaa kanonista korrelaatioanalyysi-mallia (CCA) ja koeärsykkeen korkeaulotteista piirre-esitystä olennaisten vokseleiden (muuttujien) valitsemiseen fMRI-kokeissa, joissa koehenkilöt altistetaan monimutkaiselle luonnolliselle ärsykkeelle, kuten esimerkiksi musiikille. Kokeet musiikkia ärsykkeenä käyttävän fMRI-kokeen kanssa osoittavat algoritmin löytävän tärkeitä vokseleita.

Toinen algoritmi, NeRV, on dimensionpudotusmenetelmä korkeaulotteisen datan visualisoimiseen hajontakuvion avulla. NeRV pohjautuu yksinkertaiseen abstraktiin malliin ihmisen tavalle tulkita hajontakuviota. Kokeet osoittavat NeRVin olevan perinteisiä menetelmiä parempi tämän visualisointimallin mielessä. Lisäksi NeRViä sovelletaan ensimmäisen algoritmin valitsemien fMRI-vokseleiden visuaaliseen analyysiin; analyysi sekä osoittaa NeRVin hyödyllisyyden käytännössä että tarjoaa uusia näkökulmia vokselinvalintatulosten ymmärtämiseen.

Kolmas algoritmi, LDA-NeRV, on NeRViä ja bayesiläistä latenttimuuttujamallia soveltava visualisointimenetelmä graafeille. Kokeet osoittavat LDA-NeRVin kykenevän visualisoimaan rakennetta, jota perinteiset visualisointimenetelmät eivät tuo esiin.

Avainsanat: funktionaalinen MRI, dimensionpudotus, muuttujanvalinta, visualisointi, graafien piirtäminen

AALTO UNIVERSITY
SCHOOL OF SCIENCE

ABSTRACT OF THE
LICENTIATE'S THESIS

Author: Kristian Nybo

Title: Dimensionality reduction methods for fMRI analysis and visualization

Date: September 6, 2015 Language: English Number of pages: 8+43

Department of Computer and Information Science

Professorship: Computer and Information Science Code: T-61

Supervisor and instructor: Prof. Samuel Kaski

The need to model and understand high-dimensional, noisy data sets is common in many domains these day, among them neuroimaging and fMRI analysis. Dimensionality reduction and variable selection are two common strategies for dealing with high-dimensional data, either as a pre-processing step prior to further analysis, or as an analysis step itself.

This thesis discusses both dimensionality reduction and variable selection, with a focus on fMRI analysis, visualization, and applications of visualization in fMRI analysis. Three new algorithms are introduced.

The first algorithm uses a sparse Canonical Correlation Analysis model and a high-dimensional stimulus representation to find relevant voxels (variables) in fMRI experiments with complex natural stimuli. Experiments on a data set involving music show that the algorithm successfully retrieves voxels relevant to the experimental condition.

The second algorithm, NeRV, is a dimensionality reduction method for visualizing high-dimensional data using scatterplots. A simple abstract model of the way a human studies a scatterplot is formulated, and NeRV is derived as an algorithm for producing optimal visualizations in terms of this model. Experiments show that NeRV is superior to conventional dimensionality reduction methods in terms of this model. NeRV is also used to perform a novel form of exploratory data analysis on the fMRI voxels selected by the first algorithm; the analysis simultaneously demonstrates the usefulness of NeRV in practice and offers further insights into the performance of the voxel selection algorithm.

The third algorithm, LDA-NeRV, combines a Bayesian latent-variable model for graphs with NeRV to produce one of the first principled graph drawing methods. Experiments show that LDA-NeRV is capable of visualizing structure that conventional graph drawing methods fail to reveal.

Keywords: functional MRI, dimensionality reduction, variable selection, visualization, graph drawing

Preface

This work was carried out at the Adaptive Informatics Research Centre of the Department of Information And Computer Science, in the Aalto University School of Science and Technology. I have also been a part of the Helsinki Institute for Information Technology (HIIT). My work has been funded by the Finnish Doctoral Programme in Computational Sciences (FICS).

I wish to thank my supervisor, professor Samuel Kaski, for his patient guidance and support during many years. I am also indebted to professor John-Shawe Taylor and Dr Janaina Mourao-Miranda for their time, energy and encouragement. I am grateful to all my other co-authors: Jarkko Venna, Jaakko Peltonen, Juuso Parkkinen, and Helena Aidos. Dr Arto Klami, Dr Maria Joao Rosa and Dr David Hardoon also contributed with advice and helpful discussions.

I wish to express my gratitude to all my colleagues and friends at the ICS department. I consider myself especially indebted to Dr Manuel Eugster, Jussi Gillberg, Dr Teemu Hirsimäki, Dr Krista Lagus, Dr Leo Lahti, Eemeli Leppäaho, Ilari Nieminen, Dr Janne Nikkilä, Dr Mari-Sanna Paukkeri, Veli Peltola, Ulpu Remes, Tommi Suvitaival, and Paul Wagner.

Espoo, September 6, 2015

Kristian Nybo

List of publications

1. Kristian Nybo, John Shawe-Taylor, Samuel Kaski, Janaina Mourao-Miranda. *Data-driven Selection of Relevant fMRI Voxels using Sparse CCA and Stability Selection.* Proceedings of the 3rd NIPS Workshop on Machine Learning and Interpretation in Neuroimaging 2013. Springer.

2. Jarkko Venna, Jaakko Peltonen, Kristian Nybo, Helena Aidos, Samuel Kaski. *Information Retrieval Perspective to Nonlinear Dimensionality Reduction for Data Visualization.* Journal of Machine Learning Research, 11:451–490, 2010.

3. Juuso Parkkinen, Kristian Nybo, Jaakko Peltonen, and Samuel Kaski. *Graph visualization with latent variable models.* In Proceedings of MLG-2010, the Eighth Workshop on Mining and Learning with Graphs, 94-–101, New York, NY, USA, 2010. ACM.

Summary of publications and my contributions

Each publication in this thesis represents a joint effort of all its authors; I have summarized my key contributions to each publication below.

Publication 1. The original idea of using sparse CCA and stability selection came from JST, but I implemented the algorithm and developed it into its current form. I designed the first iteration of the classification experiment on a conceptual level together with my co-authors, but I carried out the experiment out and analyzed the results, and adjusted the design into its current form. The RSA experiment was my own idea. I wrote the manuscript, with comments from my co-authors.

Publication 2. I worked only on the first half of the paper, which deals with unsupervised visualization. I wrote the original first draft of this part of the paper and participated in fine-tuning the theoretical formulation of the neighbor retrieval task. I carried out the first iterations of unsupervised visualization experiments, instructed first by SK and JV and later by JaP. I designed and wrote the open-source C++ implementation of NeRV, Local MDS and some of the quality measures introduced in the paper together with JV. This implementation was also used in the experiments in this paper.

Publication 3. I carried out and wrote the literature review and participated in writing some of the other sections. I selected the comparison methods. I participated in designing, carrying out and interpreting the experiments. I designed and wrote scripts for importing output from some graph layout algorithms into Cytoscape. I was the first author in an earlier technical report, Technical Report TKK-ICS-R20, which this publication builds on; I wrote most of that manuscript.

Contents

1 Introduction

One of the fundamental problems in modern machine learning and data analysis is the *curse of dimensionality*, also known as the 'small n, large p' problem. Suppose that we are interested in estimating the values of p one-dimensional variables that interact with each other according to some model assumed to be known. Even for many relatively simple models, the number n of sample data vectors required for a reliable estimate increases superlinearly, possibly even exponentially, as a function of p (see, e.g., Bishop, 2007, page 33). To make matters worse, it is not uncommon for p to be orders of magnitude larger than n.

Fortunately data is rarely as high-dimensional as it might seem at first glance: often a data set can be transformed into a much lower-dimensional data set without losing much information. In some cases we may be able to determine that a large portion of variables are irrelevant for our task, in which case we can simply ignore them in further analysis; this approach is commonly referred to as *variable selection* or feature selection. In other cases none of the variables are entirely irrelevant, but strong dependencies among the variables nevertheless make most of them redundant. For example, we may have a 10000-dimensional data set whose vectors all lie approximately on a 10-dimensional hyperplane, in which case a simple change of coordinates can reduce the dimensionality of the data by three orders of magnitude while preserving most of the structure. Exploiting dependencies between variables to find a transformation to reduce dimensionality is generally termed *dimensionality reduction*. This thesis deals with both variable selection and dimensionality reduction.

Section 2 describes and expands upon the contributions of Publication 1. The main result is a novel variable selection algorithm for functional MRI data, specifically for experiments involving complex natural stimuli such as music. The proposed algorithm employs a sparse canonical correlation analysis (CCA) model, which itself can be interpreted as a dimensionality reduction method, to discover which variables are relevant. Preliminary experiments on fMRI data measured from subjects listening to music classified as 'happy' or 'sad' indicate that the algorithm successfully retrieves a small set of variables that are sufficient for deciding whether a subject is listening to 'happy' or 'sad' music. Notably the algorithm does not make any use of the emotional categories for the music samples, nor does it benefit from any kind of neuroscientific prior information: it learns the relevant voxels using only a set of features that are automatically extracted from the music using a toolbox.

Section 3 highlights some of the main contributions of Publication 2. Apart from being helpful in dealing with the curse of dimensionality in statistical modeling, dimensionality reduction methods have traditionally also been used for visualizing high-dimensional data. A natural way to visualize a two-dimensional data set with continuous variables is to draw a scatterplot. For higher-dimensional data sets we can use a dimensionality reduction method to reduce the dimensionality to two, and then draw a scatterplot. In general it is not possible to represent a high-dimensional data set perfectly in two dimensions, so every scatterplot represents a compromise. Presumably some details in the high-dimensional data are more important for a

good visualization than others; finding the best possible compromise requires defining what characteristics make up a 'good visualization' and using a dimensionality reduction method that triesto preserve those. Traditional dimensionality reduction methods, such as Principal Component Analysis (PCA; Hotelling, 1933), and manifold learning methods, like Isomap (Tenenbaum et al., 2000), have often been used and suggested for visualizing high-dimensional data via scatterplots, despite being designed to solve some other problem whose relation to visualization may be unclear. Section 3 formulates a simple abstract model of the way humans interpret scatter plots and describes a dimensionality reduction method, the Neighbor Retrieval Visualizer (NeRV), that is specifically designed to produce scatterplots that are optimal in terms of the model.

Section 4 introduces the main contribution of Publication 3, a novel, model-based graph visualization method called LDA-NeRV. Most graph drawing methods have not been formulated with an explicit goal: visualization quality is generally evaluated using certain established aesthetic criteria, such as the number of edge crossings, without stating what problem the visualization is supposed to solve and how the aesthetic criteria relate to that problem. Analysis of the criteria suggests that the underlying goal of traditional algorithms is to produce visualizations that try to show local structure for individual nodes as clearly as possible. In contrast, LDA-NeRV uses a Bayesian latent-variable model to learn a specific kind of global structure in the graph, and then applies NeRV to visualize that structure as captured by the estimated latent variables. Visualizations of two sample graphs demonstrate the benefits of using an appropriate model for visualizing graphs.

Section 5 combines ideas from Sections 2 and 3 in a previously unpublished case study. I use NeRV and PCA to visually explore fMRI voxel sets selected by the algorithm in Section 2. The exploratory data analysis provides some new insights into the experiment results in Section 2 and demonstrates why the abstract visualization task of 'neighbor retrieval' is a useful approximation for how a person actually studies a scatterplot. We also see how NeRV is different from PCA in practice.

Finally, Section 6 summarizes my conclusions regarding the work presented in this thesis.

2 Data-driven variable selection for fMRI data

Modelling whole-brain fMRI data is a prime example of a 'small n, large p' problem. For example, in the fMRI data set used in experiments in this section there are, for each of the 16 test subjects, 300 samples of 219727 real-valued variables. Each sample, or *volume*, measures the activity of the subject's entire brain during a period of 2 seconds. The area of the head is divided into a regular three-dimensional grid of volumetric pixels or *voxels*, and the scanner produces a separate measurement for each voxel. The 219727 variables correspond to the voxels that contain actual brain matter; areas such as the skull and the eyes are excluded.

In addition to being extremely high-dimensional, fMRI data is also very noisy. Naturally there is some scanner noise, but more crucially the scanner does not even measure brain activity directly. A typical voxel contains more than 5 million neurons, and as their activity levels change, so does the amount of blood flowing to that area; the scanner tracks these changes in blood flow via the so-called blood-oxygen-level-dependent (BOLD) contrast mechanism. Although it has traditionally been assumed that an increase in the BOLD signal indicates neuronal excitation — increased firing — it may in fact indicate inhibition as well (Logothetis, 2008). Furthermore, there is of course always some blood flowing everywhere in the brain, and the activations that fMRI analysis looks for show up as a mere 2% or 3% increase over the baseline level of the BOLD signal (Ashby, 2011). Finally, when using fMRI to study cognition, even if we had perfect measurements of every neuron in the brain, we would still have noise in the form of a great deal of brain activity that is unrelated to the particular aspect of cognition that we are trying to study. For example, if we are interested in the experience of listening to music, we have to assume that even the most avid listener's brain is simultaneously processing many other things in addition the music being heard. The last point is naturally equally true for all neuroimaging methods.

In traditional fMRI analysis approaches like SPM, the problem of high dimensionality is side-stepped by modelling each voxel independently (see Section 2.1), so there is no need for a separate dimensionality reduction or variable selection step to reduce the dimensionality. Although these traditional approaches based on univariate modelling are still widely used, in the last 10 years true multi-variate models, generally called multi-voxel pattern analysis (MVPA) methods, have become very popular. In MVPA analysis it is standard to perform a separate variable selection or dimensionality reduction step before the main analysis (Norman et al., 2006; Mahmoudi et al., 2012).

Although generic dimensionality reduction methods like PCA can be used (Mahmoudi et al., 2012), variable selection is particularly justified for fMRI data because of *functional segregation*, the idea that a specific cognitive function can be attributed to specific locations in the brain. Much of neuroimaging research has concentrated on mapping cognitive functions to brain areas, and the principle of functional segregation has been firmly established (although it should be pointed out that the relationships between segregated areas, *functional integration*, is equally important for understanding brain function). (Friston, 1995) In other words, we know a priori

that provided that the experimental task is sufficiently specific, most of the relevant brain activity should be found in some small subset of the voxels. This suggests that we should expect variable selection to be a useful way of reducing dimensionality prior to other analysis, but it also means that variable selection algorithms can be interesting analysis methods on their own, because they can potentially tell us which areas of the brain are related to a particular task. Indeed, as we will see in Section 2.1, traditional analysis methods like SPM can be interpreted as variable selection methods.

This section proposes a new variable selection method specifically designed for fMRI experiments involving complex natural stimuli such as music. Section 2.1 briefly reviews existing approaches to variable selection. Section 2.3 motivates and describes the new algorithm. Section 2.4 attempts to tentatively validate the algorithm using a classification experiment on fMRI data measured from people listening to classical music. Finally, Section 2.5 evaluates the algorithm based on the experimental results and points out avenues for further research.

2.1 Existing approaches to variable selection

Statistical Parametric Mapping (Friston et al., 1994, SPM;) is arguably the most established approach to analyzing fMRI data and still very popular today. The most basic SPM analysis, sometimes called a 'first level analysis', produces a single-subject 'activation map' that shows which voxels display statistically significant activity relevant to the experimental condition(Amaro and Barker, 2006). Although SPM analysis can then proceed to analyze a wide variety of hypotheses, the foundation is essentially a single-subject variable selection method.

In SPM, each voxel is modelled independently of the others using a linear regression model, which transforms the problem of modelling a high-dimensional data set into a large number of independent univariate modelling problems. This bypasses the issue of high dimensionality.

SPM handles noise partly by imposing certain requirements on the design of the experiment in which the fMRI data is collected. The simplest possible experiment design is a *block design* with two different kinds of stimuli or conditions, the experimental condition and a control condition (Amaro and Barker, 2006, see, e.g.,). For example, if we wanted to find regions of the brain that are involved in recognizing human faces, the experimental condition could be "show the test subject a photo of a human face", and a possible control condition would be "show the test subject a photo of a landscape". Several photos in one category are shown in one contiguous block, and experimental blocks and control blocks alternate throughout the experiment. We can then define binary vectors $g_{\text{experimental}}$ and g_{control} such that $g_\cdot(t) = 1$ if the relevant condition is present at time t and $g_\cdot(t) = 0$ otherwise; SPM uses these variables as the regressors in its linear regression model for each voxel. Various standard statistical tests can then be applied to find voxels that are active during the experimental condition but not during the control condition.

The block design mitigates noise by severely restricting the degrees of freedom of the model's feature representation for the experimental condition. To continue

with the example from the previous paragraph, if we wanted to capture every detail of the experimental condition of viewing and recognizing faces, we would need a high-dimensional feature representation; a naive example would be a high-resolution bitmap image of the photo being shown. In the block design the experimental condition is described using a single categorical variable that can assume two values, 'face' or 'not a face'. Because the feature representation of the experimental condition is so simple, it is possible to obtain statistically significant results even with relatively few samples, noisy data, and plain linear regression. Due to the curse of dimensionality, a high-dimensional feature representation would require either impractically large numbers of samples or a more sophisticated model (see, e.g., Bishop, 2007, page 33).

The above is only a rough sketch of the simplest form of SPM, but the two main points hold in general: each voxel is modelled independently, and the experimental condition is described either using a single categorical variable or a one-dimensional real variable (Davis and Poldrack, 2013). The categories are generally decided before the data is collected to reflect the hypotheses that the researcher wants to test (Amaro and Barker, 2006). Although these properties simplify analysis and statistical inference as noted above, they also have their disadvantages.

The main disadvantage of modelling each voxel independently of other voxels is that we cannot discover groups of voxels that are individually insignificant but relevant as a whole. Just like MVPA methods in general have gained traction, several variable selection methods addressing this point have been proposed in the last 10 years. Kriegeskorte et al. (2006) proposed a model where each voxel is modeled together with voxels within a neighborhood (called a 'searchlight') of a certain radius, replacing univariate modelling with a kind of local multivariate model. Martino et al. (2008) used Recursive Feature Elimination to find the best voxels for SVM classification. Yamashita et al. (2008) selected voxels for a classification task using a sparse logistic regression classifier. Varoquaux et al. Varoquaux et al. (2012) recently introduced a voxel selection algorithm based on a randomized LASSO and stability selection.

The disadvantage of describing the experimental condition using a categorical variable is that it can be restrictive for some kinds of experiments and data analysis. When working with complex natural stimuli like music or movies, the stimulus is so rich that it is impossible to capture every interesting aspect with a single categorical variable. At the same time, a rich feature representation may be readily available; for example, in the case of music, many kinds of features can be easily extracted automatically from a digital recording using software like the MIR toolbox (Lartillot et al., 2008). The ability to retrieve all voxels that are relevant to any aspect of a rich stimulus can be desirable especially when performing variable selection as a preprocessing step for some other analysis. It can also be useful for exploring data to generate new hypotheses; this idea will be explored in Section 5.

The disadvantages of the conventional categorical feature representation for the experimental condition seem to have received less attention in voxel selection than the disadvantages of modelling voxels independently: all the voxel selection methods mentioned above assume the conventional feature representation. This was one of

the main motivations for the algorithm proposed in the next section.

Apart from algorithms, another traditional way to select voxels is to limit analysis to certain anatomical regions that are known to be relevant to the task based on current neuroscientific understanding (Norman et al., 2006). Although this approach can work well, the disadvantage is that we may exclude regions that are not yet known to be relevant for the task. We also need to solve the problem of locating each anatomical region in each individual subject's brain.

2.2 Voxel selection for rich feature descriptions

As noted in Section 2.1, most variable selection algorithms for fMRI assume that the experimental condition or stimulus is described using a categorical variable or a single real parameter, and there are situations where being able to select voxels based on a more general feature representation is desirable. The algorithm proposed here is designed specifically for fMRI experiments involving stimuli like music for which a high-dimensional, possibly real-valued feature representation is natural.

2.2.1 Canonical Correlation Analysis

At the core of the proposed voxel selection algorithm is a variant of the Canonical Correlation Analysis (CCA Hotelling, 1936) algorithm. The following overview is based on (Shawe-Taylor and Cristianini, 2004).

CCA operates on two data sets that are paired in the sense that they can in some sense be interpreted as two different views of the same phenomenon; for example, if we have fMRI measurements of a person listening to music as one view, then a feature representation of the music being listened to could be another view. CCA assumes that the two data sets have the same number of samples, so if the scanning interval for the fMRI view is two seconds, then each of the music feature view's samples should describe a two-second window of music. Furthermore, the ith sample of the music feature view should naturally describe the same two-second window as the ith fMRI scan. The two views do not need to have the same number of variables.

CCA finds two projection vectors, u_1 for the first view and v_1 for the second, such that the correlation beteen the projection of the first view onto u_1 and the projection of the second view onto v_1 is maximized. It can be shown that u_1 and v_1 can be found by computing a singular value decomposition on the cross-correlation matrix of the two views.

Conceptually, CCA could be used as a naive voxel selection method: we compute CCA using fMRI data as one view and stimulus features as the other, and interpret the projection vector returned by CCA for the fMRI view as a weight vector. A larger weight (in absolute value) indicates that the voxel is more relevant.

In practice there are two problems with this idea. The first problem is yet another incarnation of the curse of dimensionality: it can be shown that when the dimensionality of at least one view is sufficiently large compared with the number of samples, we can find perfectly correlated projections even if both views are pure uncorrelated noise. To ensure that the correlation found by CCA actually reflects a

real relationship between the two views, we need to add some form of regularization to the algorithm.

The second problem is that the basic form of CCA does not tend to produce *sparse* solutions (Hardoon and Shawe-Taylor, 2011). The CCA solution is not in general unique: we can find infinitely many pairs of projection vectors that all produce the maximal correlation. Even if the maximal correlation could be achieved by giving non-zero weights to only 100 voxels, the weight vector returned by basic CCA may well give a non-zero weight to every voxel in the brain.

2.2.2 Kernel CCA

It can be shown that to compute CCA for two views, we don't actually need access to the sample vectors themselves: it is enough to have a *kernel matrix* for each view, in which case the method is called Kernel CCA or KCCA (Lai and Fyfe, 2000; Shawe-Taylor and Cristianini, 2004). If a view has n samples x_i, a kernel matrix is an $n \times n$ matrix K such that the entry $K(i, j) = k(x_i, x_j)$, where $k(\cdot, \cdot)$ is a *kernel function*. A kernel function is a function for which it can be shown that, for all x, y, $k(x, y) = \langle \phi(x), \phi(y) \rangle_\nu$, where ν is some Hilbert space, $\langle \cdot, \cdot \rangle$ is the corresponding inner product, and ϕ is some mapping from the feature space to ν.

In other words, computing the kernel function for two samples is equivalent to first mapping them to some space ν and then taking their inner product there. The mapping can be highly nonlinear and ν can be infinite-dimensional, so kernel methods are potentially very powerful. Note that in KCCA each view has its own kernel, and the kernels can be completely different.

The simplest kernel function is the so-called linear kernel, where $k(x, y) = x^{\mathrm{T}} y$; we simply take the scalar product of the two feature vectors, so ϕ is the identity mapping and ν is the original feature space. If we use the linear kernel, KCCA is equivalent to normal CCA. Linear KCCA can still be useful if the number of samples is smaller than the dimensionalities of the feature spaces, because then the kernel matrix will be smaller than the covariance matrix, so matrix decompositions will be computationally cheaper.

For a detailed treatise of kernel CCA and kernel methods in general, see (Shawe-Taylor and Cristianini, 2004).

2.2.3 Sparse Canonical Correlation Analysis

The Sparse Canonical Correlation Analysis (SCCA; Hardoon and Shawe-Taylor, 2011) algorithm is a variant of CCA that finds sparse projection vectors.

A distinguishing feature of this particular CCA formulation is that one of the views is represented as an ordinary data matrix $X \in \mathbb{R}^{M \times N}$ (the *primal* view), where each column is one sample vector, but the other view is represented as a kernel matrix $K \in \mathbb{R}^{M \times M}$ (the *dual* view). Hardoon and Shawe-Taylor do not state an explicit reason for this asymmetry, but I suspect that there is a technical reason for it, which I will discuss below and in Section 2.5.

Like CCA, SCCA tries to find projection vectors $w \in \mathbb{R}^N$ and $e \in \mathbb{R}^M$ such that the projections $X^{\mathrm{T}} w$ and Ke are maximally correlated. The two problems

of basic CCA for voxel selection mentioned above, overfitting and lack of sparsity, are addressed by penalizing the L1 norms of w and e. Hardoon and Shawe-Taylor express the problem as the optimization problem

$$\min_{w,e} \|X^{\mathrm{T}}w - Ke\|_2^2 + \mu\|w\|_1 + \gamma\|e\|_1, \tag{1}$$

subject to the constraints $\|e\|_{\mathrm{inf}} = 1$ and $e \geq 0$ (element-wise). The constraint $\|e\|_{\mathrm{inf}} = 1$ is necessary to avoid the trivial solution $e = w = 0$. The element-wise non-negativity constraint $e \geq 0$ seems to be specific to the problem domain in whose context SCCA was introduced, and could in principle be dropped.

Equation 1 would be a convex optimization problem if it weren't for the constraint $\|e\|_{\mathrm{inf}} = 1$, which is convex but not affine; a convex optimization problem can have convex inequality constraints, but only affine equality constraints (Boyd and Vandenberghe, 2004). Hardoon and Shawe-Taylor circumvent the problem by replacing the constraint with the convex inequality constraint $\|e\|_{\mathrm{inf}} \leq 1$ and an additional affine equality constraint, $e_k = 1$, where e_k is the kth element of e, and k, the *seed index* is a parameter set by the user of the algorithm. In other words, the user must choose one sample that will receive a constant weight of 1 in the dual view.

The optimization problem is solved using a fast customized iterative algorithm that alternates between optimizing w and optimizing e. The regularization parameters μ and γ are determined automatically using an approach that has been shown to work well in practice; see (Hardoon and Shawe-Taylor, 2011) for more details. A MATLAB implementation of the iterative algorithm is available at `http://davidroihardoon.com/Professional/Code_files/SCCA2.m`. This implementation was used for the experiments in Section 2.4.3.

2.3 Applying SCCA to voxel selection

When applying SCCA to voxel selection, one should use the primal representation X for the fMRI scans and the dual representation K for the stimulus features. The reason for this is that the primal-view projection vector w is sparse in the variables (voxels), whereas the dual-view projection vector e is sparse in the *samples*.

The basic idea is to run SCCA and then select those voxels that have non-zero entries in the sparse weight vector w. In principle, the sparsity-inducing regularization of SCCA solves the two problems of performing voxel selection with CCA discussed in Section 2.2.1. In practice, however, the second problem is only partly solved. Although SCCA favors sparse linear combinations over non-sparse ones, the sparse linear combination that minimizes the cost function is still unlikely to be unique. The voxels with non-zero weight in the sparse weight vector w returned by SCCA should be *among* the most relevant, but we cannot know whether there are other equally relevant voxels, or whether some of the voxels implicated by w are more relevant than the others. For example, if w has 20 non-zero weights, 10 of those weights might correspond to voxels that contain information that isn't found in any other voxel in the brain, whereas the rest might contain information that could just as well be explained by selecting 10 voxels from a pool of 20000 voxels.

To address this issue, the proposed voxel selection algorithm incorporates the idea of stability selection, as introduced by Meinshausen and Bühlmann (2010). The algorithm randomly subsamples 10% of the voxels and 66% of the samples and runs SCCA on those. This step is repeated 1000 times while keeping count of how many times each voxel received a non-zero weight relative to how many times it was included in the random subsample. This ratio can be interpreted as an empirical probability of relevance: if a voxel is almost always assigned a non-zero weight no matter what other voxels are available, it is more likely to be relevant than a voxel that only occasionally gets a non-zero weight. We include in our set of relevant voxels all voxels whose probability exceeds a certain threshold p_{thres}.

Due to the subsampling, each iteration of the stability selection procedure effectively trains SCCA on a different data set, and so the optimal seed index k could be different each time. An exhaustive search on each iteration would be prohibitively expensive. In the experiments described in Section 2.4 the music feature sample vectors were first clustered using K-means with 20 clusters. For each of these index clusters, SCCA and stability selection were performed on the full data set, as described in the previous paragraph, so that on each iteration a new seed index k was chosen randomly from the corresponding cluster. Thus 20 different sets of empirical voxel probabilities were obtained, one for each cluster; to create a single stable voxel set, every voxel whose probability exceeded p_{thres} in at least one of the 20 sets was included.

2.4 Validating the voxel selection method

The goal of the voxel selection method is to retrieve voxels that are relevant to the experimental condition. Validating the method is complicated by the fact that there are two distinct points of failure: the set of stimulus features and the algorithm itself. On the one hand, the stimulus features might not be rich enough to capture what's relevant; on the other hand, the algorithm itself might fail to find the voxels that are relevant to the features.

In an attempt to tentatively validate both the algorithm and the music features in a single experiment, a classification experiment was devised. The voxel selection method was used to select relevant voxels from a set of fMRI measurements of people listening to classical music, using as the feature view a set of features extracted automatically from digital recordings of the music. The experiment in which the fMRI data had been collected followed a traditional block design, with the block labels being 'happy music', 'sad music', 'neutral music' and 'rest'. Two classifiers for the labels 'happy' and 'sad' were trained, one using only the voxels selected as relevant by the proposed algorithm, and another using only a structurally similar but randomly picked set of voxels.

Because neither the voxel selection algorithm nor the feature extraction process use these block labels in any way, the block labels can be treated as an independent ground truth. If many different people perceive a certain set of songs as 'happy' and another set of songs as 'sad', there should be some structure in the songs that determines the emotional content, and a good feature representation might capture

that structure. Therefore if the selected voxels provide significantly better classification performance in the happy–sad task than the random voxels do, this would indicate that the automatically extracted music features successfully captured some information about the music, and that the algorithm successfully selected voxels that were relevant to the music features.

2.4.1 The data set

The data set used in the experiments was originally introduced in (Mitterschiffthaler et al., 2007). 16 healthy subjects listened to 20 extracts of orchestral classical music, each 30 seconds long. Each extract was labeled as either 'happy', 'sad' or 'neutral'. The labels were based on a separate pilot study where people were asked to rate 60 different pieces of music on a scale from 0 (very sad) to 100 (very happy). Of the 60 pieces, 20 of the most consistently rated pieces were chosen so that there were 5 happy pieces, 5 sad pieces and 10 neutral pieces. First happy pieces were played alternating with neutral pieces, and then sad pieces were played, again alternating with neutral pieces.

The scanning interval was 2 seconds, so there were altogether 300 samples for each subject. Scans taken during the silent resting periods between each music extract were omitted.

For details on data pre-processing, see (Mitterschiffthaler et al., 2007). For this experiment, a voxel mask toremove the eyes and the skull was applied, leaving 219,727 voxels per scan. The data set was randomly partitioned into 150 training samples and 150 test samples. The training set and the test set contained the same number of samples from each class. Voxel selection was performed only on the training data.

2.4.2 Music features

A 26-dimensional set of features was extracted from CD-quality music (encoded in .WAV format) using the MIR toolbox for MATLAB (Lartillot et al., 2008). The feature set used consists of all the low-level features (all features but pulse clarity, key clarity and tonal centroid) in the feature set recently used by Alluri et al. to study fMRI measurements from subjects listening to tango (Alluri et al., 2012). Music feature vectors within each 2-second window corresponding to one fMRI scan were then averaged to obtain one music feature sample for each fMRI scan. Finally the music samples were convolved with a standard hemodynamic response function.

2.4.3 SVM classification of fMRI scans as 'happy' or 'sad'

To obtain a two-class problem, the neutral samples were removed from the training and test data set defined in Section 2.4.1. For each subject, there were thus 75 training samples and 75 test samples. Separately for each subject, linear Support Vector Machines (SVM) (Cristianini and Shawe-Taylor, 2000) were trained using libSVM (Chang and Lin, 2011) on three different sets of variables:

Table 1: SVM happy–sad classification rates for various fixed (not cross-validated) values of p_{thres}.

Stability threshold p_{thres}	0.02	0.05	0.1	0.2	0.4	0.6	0.8
SVM accuracy (stable), %	83.53	82.85	82.77	83.19	81.50	77.28	68.50
SVM accuracy (random), %	82.56	75.57	74.66	72.09	69.30	64.54	57.92
Voxels used, % of full	15.95	6.40	2.79	0.96	0.20	0.043	0.0057

1. the full brain;
2. the subset of stable voxels chosen by applying the voxel selection algorithm to the training data;
3. and a random set of voxels generated by taking set #2 and moving each contiguous cluster of voxels to a random location in the brain (overlap allowed).

The random voxel set #3 was generated as described to obtain a random set that would be comparable in size and shape to set #2. It was found that if we simply sample N_2 voxels uniformly from the entire brain, where N_2 is the number of voxels in set #2, we obtain classification performance very similar to using the full brain, unless N_2 is extremely small. This is not surprising considering how strongly correlated adjacent fMRI voxels are: voxels sampled uniformly from the brain may well contain most of the information found in the entire brain.

In training the SVM, two parameters had to be set: the SVM regularization parameter C_{SVM} and the stability probability threshold p_{thres} for the voxel selection method. The parameters were selected using 5-fold cross-validation on the training data, with C_{SVM} taking values from 10^-9 to 10^-2 and p_{thres} taking values from 0 to 0.8. The utility function maximized under cross-validation was ((SVM classification accuracy in percent) $- \sigma$(percent of total voxels used). Here σ can be interpreted as a non-negative hyperparameter specified by the user to control the trade-off between classification accuracy and the number of voxels used. I set $\sigma = 7$ to favor sparsity over tiny performance improvements.

The classification accuracy for the SVM trained on all the voxels, averaged over subjects, was 82.10%.The corresponding accuracy for the SVM trained on stable voxels was 79.05%, and the stable voxels comprised on average 0.10% of the full voxel set (circa 200 voxels). The accuracy for the randomly translated clusters was 65.60% (averaged over 10 different random sets).

Table 1 shows classification rates for various stability thresholds p_{thres}, when the same threshold is used for every subject instead of selecting the parameter using cross-validation. We can see that the classification rate is actually even slightly better than the full-brain classification rate for a large range of values of p_{thres}. For very small values of p_{thres} — corresponding to a large set of voxels — the randomized voxel clusters performed almost as well as the stable voxels, most likely because the voxel subsets in this range were so large (15–38% of all voxels) that a randomized voxel set was likely to contain many relevant voxels (overlap was allowed in the randomization procedure).

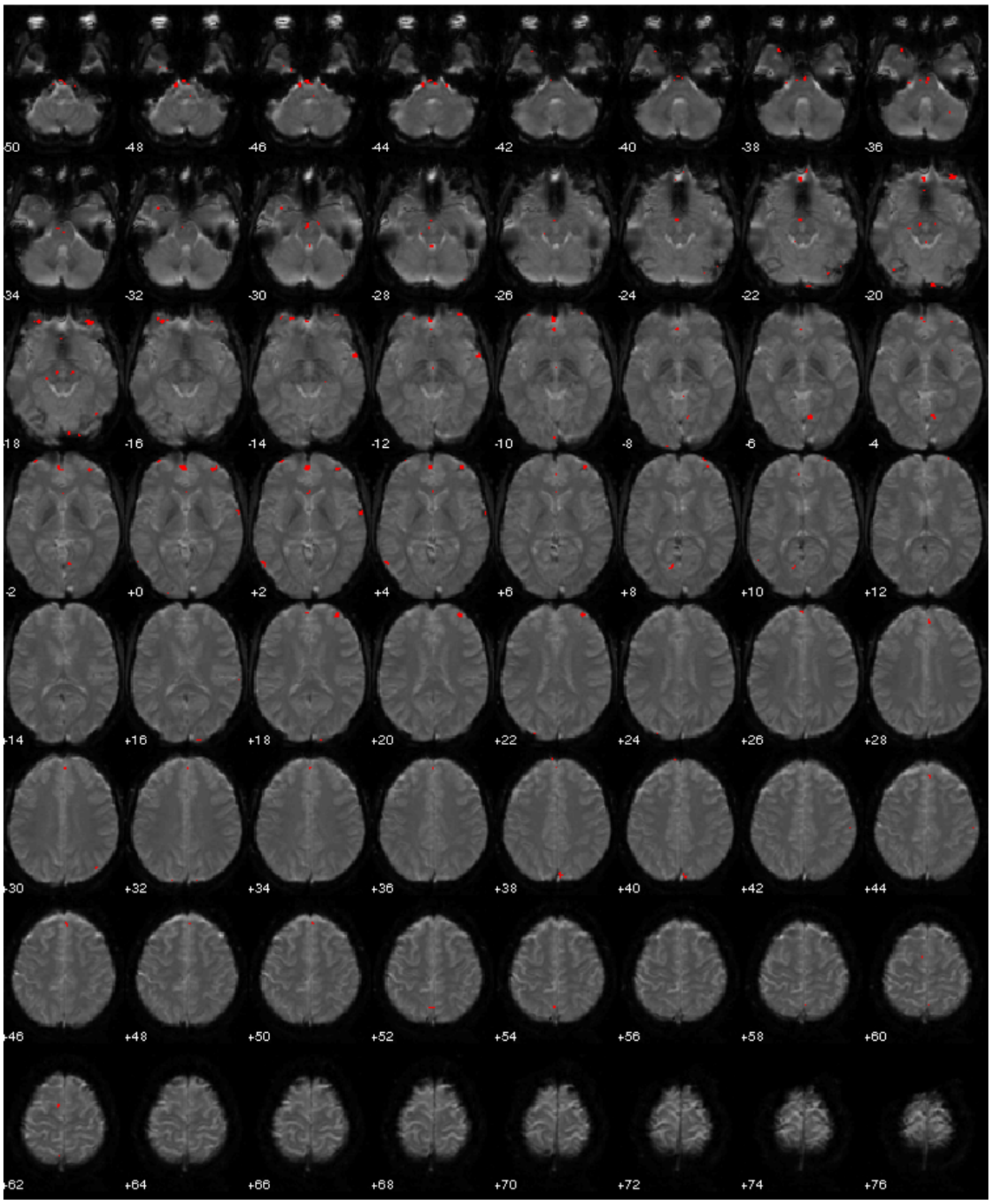


Figure 1: A visualization of a stable voxel set produced by the proposed voxel selection algorithm. This stable voxel set is studied further using exploratory visualization in Section 5. Stable voxels are displayed as tiny red squares; most of the red patches contain several voxels. The voxels were obtained from Subject 6 using a stability threshold p_{thres} of 0.4. There are altogether 535 stable voxels.

2.5 Discussion

Voxels selected by the proposed algorithm perform significantly better than the randomized voxels in the happy–sad classification task, which suggests that the music features successfully capture some meaningful structure in the music, and that the voxel selection algorithm finds voxels relevant to that structure. A classification experiment was used to validate the algorithm mainly because the only easily available independent ground truth happened to be in the form of class labels. The algorithm was not compared with other recently proposed voxel selection methods because they use the class labels for selecting voxels, which would have made a fair comparison impossible.

I was surprised to see randomized voxels produce such high classification rates; when I first saw the results, I assumed that I must have accidentally used some test data to train the classifier, but that was not the case. It seems that the difference between 'happy' music and 'sad' music can to some extent be seen almost everywhere in the brain; otherwise it is difficult to explain how a handful of small clusters from random locations in the brain could give a classification rate significantly higher than 50% in a balanced two-class classification problem. Because random voxels perform so well, this classification task is not ideal for validating a voxel selection algorithm. A task where relevant information is more localized in the brain could offer stronger evidence.

I do not know whether the performance of randomized voxels in fMRI classification has been studied in other contexts; if the good performance generalizes beyond this particular experimental condition and data set, this would imply that comparisons with random voxel sets should be included in every fMRI study that wants to make claims about the significance of a voxel set based on its performance in a classification or prediction task.

Some time after carrying out the classification experiments in Section 2.4.3, I discovered what appears to be a small bug in the MATLAB SCCA implementation due to Hardoon and Shawe-Taylor that was used in the experiments. The Karush-Kuhn Tucker conditions for optimality (see, e.g., Boyd and Vandenberghe, 2004) did not seem to hold for the solutions returned by the implementation. I wrote an alternative implementation (see Appendix A) that solves the convex optimization problem using the well-known CVX package (Grant and Boyd, 2014, 2008). Due to time constraints I have not repeated the entire set of voxel selection experiments with the new implementation, but in a brief comparison with data from one randomly selected subject the new and old SCCA implementations seemed to produce nearly identical solution vectors (in terms of the L2 norm), so it is possible that the bug did not have a significant effect on the voxel selection experiments. Nevertheless, I would recommend using the new implementation in future experiments. Uurtio et al. (2015) tried both SCCA implementations for studying microbe-environment interactions and found that the new implementation performed more robustly than the original: the correlations were more stable with respect to the sparsity parameter, and when measured by permutation tests, high correlations had higher statistical significance (Uurtio, 2015).

Some technical aspects of the voxel selection algorithm could be improved. In particular, needing to set the seed index k is burdensome. As mentioned in Section 2.2.3, the peculiar constraint $e_k = 1$ exists to avoid the trivial solution $e = w = 0$ without making the problem non-convex. Using my alternative SCCA implementation as a basis, I have briefly experimented with some alternative constraints, but have not found a constraint that would reliably prevent the trivial solution without causing overfitting.

Another aspect of the SCCA algorithm that could be changed is the element-wise non-negativity constraint $e \geq 0$. The constraint seems somewhat arbitrary for this problem domain; why would we not allow negative weights for the music feature samples? It seems that in this case the constraint is just an unnecessary restriction on the solution space. Removing the constraint amounts to removing one line in the alternative SCCA implementation in Appendix A.

Finally, the set of music features used deserves further investigation. I chose this set of music features as a starting point because a superset of these features had been successfully used in a different kind of fMRI study (Alluri et al., 2012). It is quite possible that a different set of music features would be better for voxel selection with SCCA.

3 Dimensionality reduction for visualization

Dimensionality reduction methods are often used for visualizing high-dimensional data by reducing the dimensionality to 2 and drawing a scatterplot. It is in general impossible to represent a high-dimensional data set perfectly in two dimensions. Thus producing a good scatterplot requires defining 'good' quantitatively and optimizing that quantity. Most dimensionality reduction methods as well as the measures used to evaluate them, however, are not based on an explicit definition of a 'good' visualization. A survey of 69 papers on dimensionality reduction from the years 2000–2006 found that 28 of the papers only presented visualizations of sample data sets as a proof of quality (Venna, 2007). The papers that did use quantitative measures mostly measured one of two things: how well the two-dimensional projection preserves pairwise distances in the original data; or how well a classification algorithm trained on the projection performs. These measures could be measures of visualization quality if the scatterplot were used for classification or assessing distances between points, but otherwise the connection to visualization is vague.

Section 3.1 defines an abstract visualization task, *visual neighbor retrieval*, that can be used to model some tasks that humans perform when they study scatterplots. A quantitative measure of visualization quality naturally follows from this visualization task. The quantitative measure can be turned into a differentiable cost function that can be optimized directly, which gives rise to a dimensionality reduction method for visualizing similarity relationships, the Neighbor Retrieval Visualizer (NeRV). Here I describe the method in abstract; in Section 5 I will apply it in practice and demonstrate how the abstract visualization task translates into real exploratory data analysis.

Graphs are important in many fields, and like high-dimensional data sets, are often explored by visualizing. Like dimensionality reduction methods used to visualize high-dimensional data sets, graph visualization methods have traditionally been evaluated by showing sample plots or by computing quantitative measures that are only vaguely related to visualization. Section 4 describes a graph layout algorithm, based on NeRV and a generative model for graphs, that is specifically designed to visualize certain kinds of graphs.

3.1 NeRV

When a person looks at a scatterplot where each data point is represented by an identical glyph, they are inclined to assume that points that are close to each other on the display are similar and that points that are far away from each other are dissimilar. In the psychology of vision, this perceptual organizing principle is known as the Gestalt law of proximity. A related concept is the spatial concentration principle, which states that regions with similar element density are perceptually grouped (Ware, 2004). The spatial concentration principle is illustrated in Figure 2.

3.1.1 Viewing scatter plots interpreted as an information retrieval task

One simple way to model human perception based on the principles above is to imagine that it uses the visualization to perform a simple information retrieval task: given a sample i, use the visualization to find that sample's neighbors, the set Q_i. Q_i could be defined simply as the points that lie within a certain radius r of sample i in the visualization, or the k nearest neighbors for some fixed k, or perhaps some more intricate model that incorporates the spatial concentration principle.

If we analogously define the set P_i as the neighbors of sample i in the input data, we can identify two possible types of errors in this visual information retrieval task. First, a sample could be in Q_i but not in P_i; this would be a *false positive*, a sample that appears similar to i in the visualization, but is in fact dissimilar in the real data. Second, a sample could be in P_i but not in Q_i; this would be a *miss*, a sample that appears dissimilar to i in the visualization, but is in fact dissimilar in the data. If we assign a fixed cost C_{FP} to false positives and a fixed cost C_{MISS} to misses, we can define a cost function for the visualization with respect to sample i:

$$E_i = N_{\mathrm{FP},i} C_{\mathrm{FP}} + N_{\mathrm{MISS},i} C_{\mathrm{MISS}}, \tag{2}$$

where $N_{\mathrm{FP},i}$ is the number of false positives for sample i, and $N_{\mathrm{MISS},i}$ is the number of misses.

It can be shown that Equation 2 can be interpreted as a weighted sum of precision and recall for a single query; see Publication 2 for details. Averaging E_i over all samples would then give a cost function for the entire visualization.

Although the abstract task of retrieving a single point's neighbors from the visualization probably has very little to do with how human visual processing works, it can still be a useful model for assessing visualization quality. Many real analysis tasks can be thought of in terms of multiple single-point neighbor queries. For example, if I look at Figure 2 and want to find clusters, then if I know that the visualization has high precision, I can be relatively certain that any clusters that I see in the visualization are actually clustered in the high-dimensional data as well. If the visualization has low recall, the visualization may have dispersed or split up some clusters.

3.1.2 Probabilistic neighbor retrieval

Equation 2 is the conceptual basis for NeRV, but as E_i is not a smooth function, optimizing it directly would be difficult. From a modelling perspective the idea of having fixed sets of neighbors P_i and Q_i, where each neighbor is equally relevant and each point outside the neighborhood is completely irrelevant, is also somewhat inflexible. For example, in Figure 2, what should Q_x be? The cluster marked 'a' could be one natural answer based on perceived clustering. But in Figure 3 one might be inclined to include cluster 'b' as well. The relationships between 'x', 'a', and 'b' are identical in Figures 2 and 3, but the context is different, and the context affects how those relationships are perceived.

Instead of having a fixed set of neighbors, NeRV uses a probabilistic model of neighborhood, where each sample j is assigned a non-zero (but possibly extremely

small) probability of being the neighbor of sample i. The probability of sample j being a neighbor of sample i in the visualization is defined as

$$q_{j|i} = \frac{\exp\left(-\frac{\|\mathbf{y}_i - \mathbf{y}_j\|^2}{\sigma_i^2}\right)}{\sum_{k \neq i} \exp\left(-\frac{\|\mathbf{y}_i - \mathbf{y}_k\|^2}{\sigma_i^2}\right)}, \tag{3}$$

where $\mathbf{y}_k$ is the coordinate vector of sample k in the visualization, and $\|\cdot\|$ is the L2 norm. Analogously, the probability of sample j being a neighbor of sample i in the original data is defined as

$$p_{j|i} = \frac{\exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\sigma_i^2}\right)}{\sum_{k \neq i} \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_k\|^2}{\sigma_i^2}\right)}, \tag{4}$$

where $\mathbf{x}_k$ is the sample vector in the original high-dimensional data set. The exponentials in the numerators of Equations 3 and 4 ensure that the neighborhood probability drops off very quickly as the distance increases, but σ_i can be set to adjust the rate for a more flexible concept of neighborhood. By default σ_i is set to the value that gives the distribution $p_{\cdot|i}$ an entropy equal to $\log k$, where k is a rough upper limit for the number of neighbors specified by the user.

The actual cost function optimized by NeRV is defined as

$$\begin{aligned} E_{\text{NeRV}} &= \lambda \mathbb{E}_i[D(p_i, q_i)] + (1-\lambda)\mathbb{E}_i[D(q_i, p_i)] \\ &\propto \lambda \sum_i \sum_{j \neq i} p_{j|i} \log \frac{p_{j|i}}{q_{j|i}} + (1-\lambda) \sum_i \sum_{j \neq i} q_{j|i} \log \frac{q_{j|i}}{p_{j|i}} \end{aligned} \tag{5}$$

where $D(p, q)$ is the (asymmetric) Kullback-Leibler divergence and λ is a parameter set by the user. It can be shown that $D(p_i, q_i)$ is a generalization of recall in the sense that the two are equivalent if we define p_i and q_i appropriately; see Publication 2 for details. Similarly, $D(q_i, p_i)$ is a generalization of precision. Therefore λ allows the user to control the trade-off between precision and recall.

Figure 4 illustrates the trade-off between precision and recall. Embedding **A** was computed with a value of λ close to 0, strongly emphasizing precision. This results in the sphere being cut open and folded out. If we pick any point in the visualization, every point in its vicinity is also nearby in the original data, so there are no false positives. On the other hand, points on opposite sides of the 'seam' along which the sphere was cut end up far away from each other, so there are some misses. In contrast, embedding **B** was computed with a value of λ close to 1, strongly emphasizing recall. This results in the sphere being squashed flat. Now points in the original data are always relatively close to each other, so there are fewer misses, but there are also many false positives due to points on opposite sides of the sphere ending up next to each other.

Extensive experiments in Publication 2 show that NeRV universally produces better precision and recall than other dimensionality reduction and manifold learning methods, which suggests that it is a good method for visualization tasks where those are reasonable measures. In Section 5 I will apply NeRV to results from the voxel selection experiments in Section 2 to demonstrate its usefulness in real-world exploratory data analysis.

Figure 2: Illustration of the spatial concentration principle; drawn based on an illustration in (Ware, 2004). The point marked ‘x’ is perceived as being part of cluster ‘a’ rather than cluster ‘b’, even though it is no further from the points in ‘b’ than points in ‘b’ are from each other.

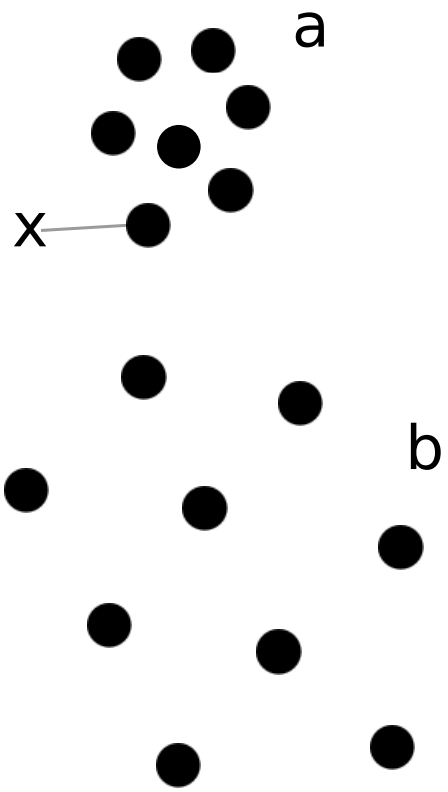
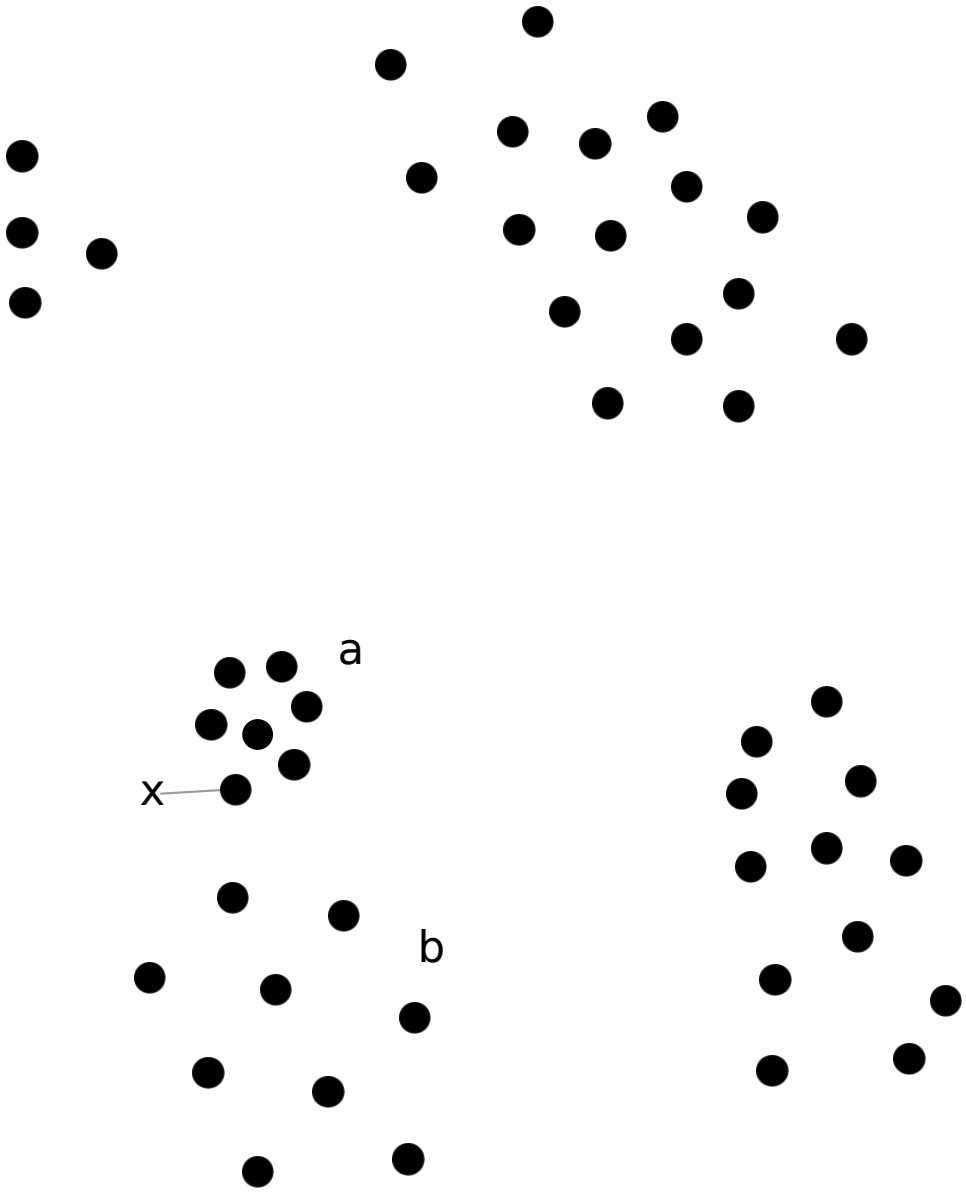


Figure 3: The scatter plot from Figure 2 with more points added.



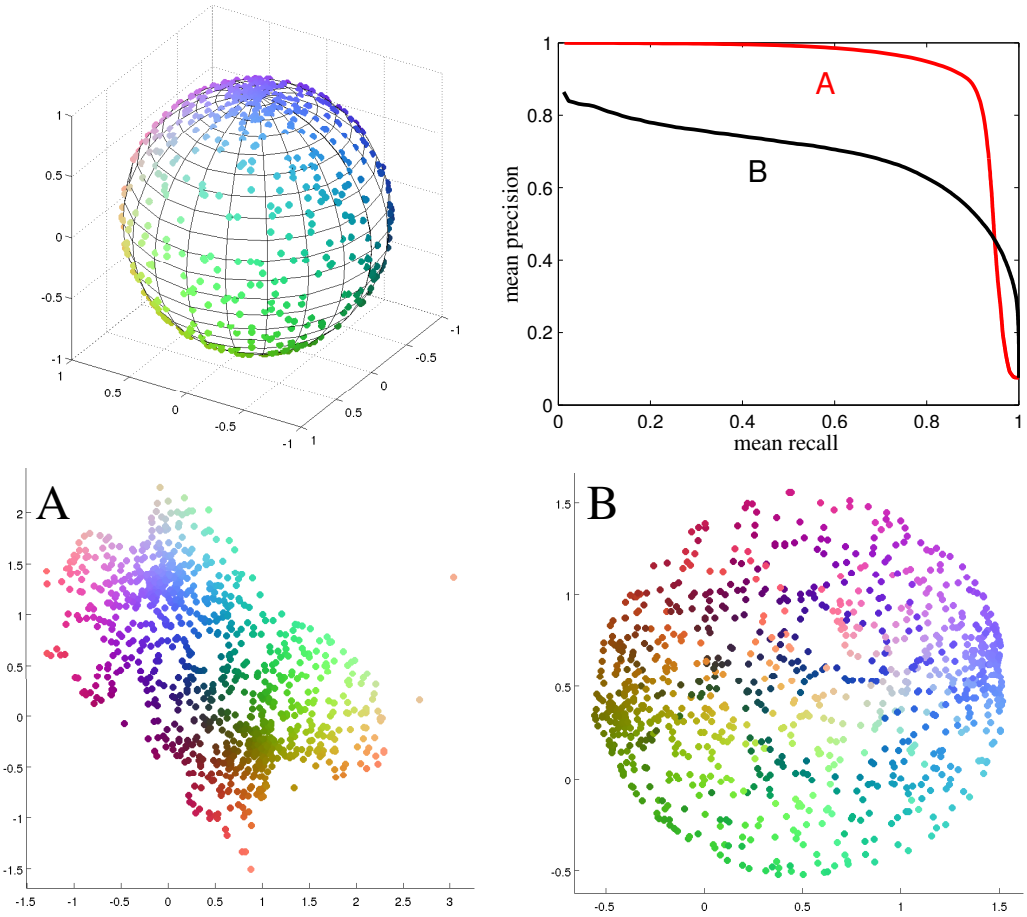


Figure 4: Demonstration of the tradeoff between false positives and misses. **Top left:** A three-dimensional dataset sampled from the surface of a sphere; only the front hemisphere is shown for clarity, although the entire sphere is projected in embeddings **A** and **B**. The color of a glyph encodes its position in the three-dimensional data. **Bottom:** Two embeddings of the dataset. In the embedding **A**, the sphere has been cut open and folded out. This embedding eliminates *false positives*, but there are some *misses* because points on different sides of the tear end up far away from each other. In contrast, the embedding **B** minimizes the number of misses by simply squashing the sphere flat; this results in a large number of false positives because points on opposite sides of the sphere are mapped close to each other. **Top right:** mean precision–mean recall curves with input neighborhood size $r = 75$, as a function of the output neighborhood size k, for the two projections. The embedding **A** has better precision (yielding higher values at the left end of the curve) whereas the embedding **B** has better recall (yielding higher values at the right end of the curve).

4 Model-based straight-line graph drawing: LDA-NeRV

Graphs are an important form of data in many fields. Visualization is often employed to discover or illustrate structural patterns in the data, such as communities in a social network. The simplest and most common way to visualize a graph is to algorithmically generate a two-dimensional plot where nodes are drawn as glyphs and edges are drawn as lines connecting the nodes. The term 'graph layout' in this section refers specifically to straight-line graph drawing. Other ways of generating static graph visualizations exist, and there are also many interactive visualization interfaces; for a review, see (Herman et al., 2000). Nevertheless, many of them use straight-line graph drawing as a basis.

4.1 Traditional graph layout: 'graph neighbor retrieval'

Many graph layout algorithms have been published. Force-based algorithms seem to be the most established and wide-spread class. Going back at least as far as Fruchterman and Reingold (1991), the first algorithms were motivated by a physical analogy. Each edge in the graph is modeled as a mechanical spring with an equilibrium length k. Nodes are steel rings to which the springs attach. Nodes are given (possibly random) starting positions, and then the mechanical system is released and simulated until it finds an equilibrium, which produces the graph layout. Traditional force-based methods were computationally expensive for large graphs, so recently faster approximations, so-called multi-level methods, have been developed (Hadany and Harel, 1999; Walshaw, 2003). Spectral graph layout algorithms based on matrix decompositions are even older than force-based methods (Hall, 1970; Kruskal and Seery, 1980), but have only recently become popular (Civril et al., 2005), apparently largely due to being significantly faster than the fastest force-based methods (Hachul and Juenger, 2007).

Of the authors cited in the last paragraph, only Fruchterman and Reingold explicitly state a goal for graph visualization: "producing a drawing that meets some generally accepted aesthetic criteria", namely avoiding edge crossings and overlapping nodes, making edge lengths uniform, and placing nodes connected by an edge next to each other. The implicit unifying goal behind the aesthetic criteria seems to be to *visualize local graph adjacency* as clearly as possible without making any assumptions about the structure of the graph. In other words, given a node, the user should be able to see which nodes that node connects to; in the spirit of NeRV, we could call this a task of *graph neighbor retrieval*. In a straight-line graph drawing this task amounts to looking at the node on the display and then tracing the outgoing edges. Violating the aesthetic criteria suggested by Fruchterman and Reingold interferes with this visualization task: If the edge being traced by the user crosses many other edges, the user can lose track of the edge. If the connected nodes are next to the node, but not too close, they're easier and faster to find. Shorter edges in general are easier to 'read' correctly.

The other papers seem to implicitly use the criteria put forward by Fruchterman and Reingold, and consequently optimize 'graph neighbor retrieval': they do not state an explicit problem that their algorithm is trying to solve, but they use similar aesthetic criteria and sample visualizations to evaluate the algorithm. Apart from computable aesthetic criteria like the number of edge crossings, quantitative evaluation of algorithms tends to focus on measuring computational speed.

One exception is the LinLog graph drawing algorithm that is specifically designed to highlight graph clusters, that is, subgraphs where nodes within the subgraph are more connected to each other than to nodes outside the subgraph (Noack, 2007). Noack shows that the goal of the LinLog algorithm is different from and in conflict with the goals formulated by Fruchterman and Reingold: optimizing one will result in violating the other.

4.2 LDA-NeRV

Publication 3 proposes a new graph layout algorithm, LDA-NeRV, that uses an explicit model both for what aspect of the data it tries to visualize and for how it is tries to visualize it. The nodes are laid out using t-NeRV, a variant of NeRV also introduced in Publication 2, so the visualization model is the neighbor retrieval task formulated in Section 3.1. Thus the basic idea is to place similar nodes next to each other in the visualization. Defining similarity defines the structure that is being visualized. Informally, LDA-NeRV considers two nodes similar if they have edges going to the same nodes. If we consider a social network where each node is a person, and an edge between two nodes indicates that the persons know each other, this would mean that people who know the same people are considered similar.

Formally, similarity for nodes is defined in terms of SSN-LDA (Zhang et al., 2007), a Bayesian latent-variable model for graphs. SSN-LDA assumes that graphs are generated by a specific stochastic process; given a graph, it uses Bayesian inference to estimate which parameters of the stochastic process are most likely to have produced that graph. The most important parameter is the set of components, or *communities* as Zhang et al. call them. Each component is a probability distribution over the nodes in the graph, where the probability for a node reflects the probability of an edge to that node. Each node in the graph, in turn, is associated with a 'membership distribution', a probability distribution over the components. To generate an edge for a node, the stochastic process first samples a component from the node's membership distribution, and then samples a target for the edge from the distribution of the sampled component. The intuition is that the 'membership distribution' reflects to what extent the node belongs to each of the components. In a social network, the components could be different 'social circles', and a person's membership distribution could be interpreted as indicating how they divide their time and energy between the different circles.

More precisely, then, LDA-NeRV considers nodes with similar estimated membership distributions to be similar, and tries to place those nodes near each other in the visualization, while controlling the trade-off between precision and recall as discussed in Section 3.1. One interpretation of the algorithm is that it embeds

the nodes in a higher-dimensional component space and then visualizes them using dimensionality reduction.

4.3 Differences between graph drawing models in practice

Figures 5 and 6 illustrate the effects of different graph drawing models using two sample graphs, each visualized with LDA-NeRV and three other graph layout algorithms. All drawings were produced with Cytoscape (Shannon et al., 2003) using the node layouts generated by the algorithms. The three other methods are Walshaw's multi-level force-based algorithm (Walshaw, 2003), implemented as a Cytoscape plugin(Salmela et al., 2008); Kruskal and Seery's spectral method (Kruskal and Seery, 1980), later independently rediscovered as SDE by Civril et al. (2005), who kindly provided an implementation; and Noack's Edge-Repulsion Linlog (Noack, 2007, 2010). The first two methods were chosen to have one representative modern algorithms from the two main classes, force-based and spectral methods. LinLog is interesting as the only other model-based graph layout algorithm.

The graph in Figure 5 (Girvan and Newman, 2002) represents football teams and their games: each node is a team, and an edge between two teams indicates that they played each other during a certain period of time. Each team belongs to one of 12 conferences, indicated by the color of the node, and teams mostly played each other. Both LinLog and LDA-NeRV mostly cluster teams by conference. LinLog clusters them because they are clusters in the graph: teams within a conference play each other more than they play other teams. LDA-NeRV clusters them because nodes in a graph cluster have similar edge distributions. Walshaw's algorithm also groups nodes by conference to some extent, but the intra-cluster and inter-cluster distances are so similar that the grouping is only apparent when the nodes are colored by conference. The layout produced by Walshaw's algorithm is the best for seeing exactly which nodes a single node connects to. SDE also tends to place teams from the same conference fairly near each other, but many of the conferences overlap.

Notably a few teams in Figure 5 don't get clustered with teams from the same conference in either the LinLog or the LDA-NeRV visualization. At least the yellow teams seem to break the general rule that teams within a conference mainly play each other, so the LinLog visualization is correct (i.e., consistent with its stated goal) in not clustering them together. The yellow teams don't seem to belong in any of the other clusters either, so they get left out. In the LDA-NeRV visualization, however, each of the yellow nodes belongs to one of the 12 clusters. The LDA-NeRV visualization was produced with the number of components set to 12, and each node has to belong to at least one component (the membership probability distribution has to sum to 1). Here it seems that every node is strongly associated with only one component, which results in 12 tight clusters. If the number of components were set higher, some of the clusters in Figure 5 would most likely split up. This does not necessarily imply that a higher number of components would necessarily be 'correct'; the correct number of components is ambiguous in the same way as the correct number of clusters for K-means is ambiguous. The need to set the number of components is nonenetheless one inconvenience of the LDA-NeRV model as it is

formulated in Publication 3.

In the graph in Figure 6 (Newman, 2006), each noun is either an adjective (blue) or a noun (red), and an edge between nodes indicates that the words appeared next to each other in the Charles Dickens novel David Copperfield. The football graph in Figure 5 was an example of *assortative* structure, where nodes in the same class link to each other. In contrast, the adjective–noun graph is an example of a disassortative graph: nodes in the same class mainly link to nodes in other classes. In English, a noun appearing next to an adjective is far more common than a noun appearing next to a noun or an adjective appearing next to an adjective. This is reflected in the LDA-NeRV visualization, which mostly groups nouns together and adjectives together. (The exceptions in the visualization tend to correspond to true exceptions; for example, there are also nouns that mainly appear next to other nouns.) In this visualization, the number of components was set to 4. Unlike the football graph visualization, this visualization has several nodes outside the four main clusters, which indicates that several nodes belong to more than one component.

LinLog fails to reveal the disassortative structure for the adjective–noun graph in Figure 6, because the algorithm is specifically designed to show assortative structure. Walshaw's algorithm and SDE predictably also do not show the structure. As in the case of the football graph, the force-based algorithm produces the most uniform distribution of nodes and edge lengths, which is one of the goals of traditional graph layout methods.

4.4 Discussion

Most papers on graph layout methods have not explicitly defined what they are trying to visualize; implicitly they have focused on aesthetic features that seem to indicate that they are trying to create visualizations where, given a node, it is easy to find the nodes that connect to that node. At first sight this task of 'graph neighbor retrieval' may seem like a natural goal for graph visualization. A graph, after all, can be defined as a list of nodes and a list of edges between those nodes. A perfect graph neighbor retrieval visualization could be used to reconstruct the graph, which would indicate that the visualization captures the structure of the graph perfectly.

In practice there are two problems with the traditional task of graph neighbor retrieval. The first is that as the number of edges grows, it seems to be very difficult to produce a straight-line graph drawing that would truly perform well in this task. For example, in Figure 5, Walshaw's algorithm manages to produce a visualization where it is possible to retrieve a single node's adjacent nodes. In Figure 6, the situation already looks quite hopeless for most nodes, and this is still a relatively small graph.

The second problem is that even if it were possible to make a visualization that perfectly reveals every node's adjacent nodes even for very large graphs, researchers are often interested in global structural patterns, such as the assortative and disassortative structure found in the football and adjective–noun graphs. Focusing on making every single node's adjacent nodes clearly visible does not necessarily bring out global patterns, and as the visualizations suggest, may even hide them.

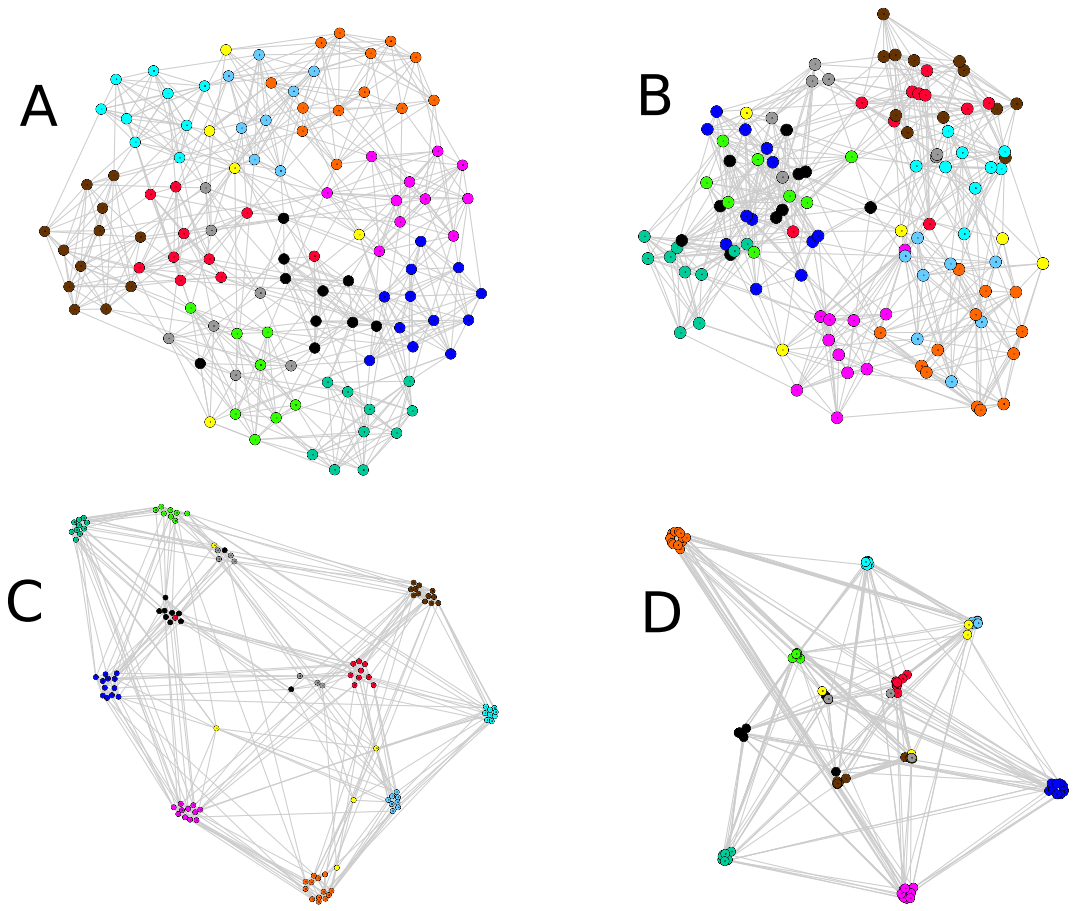


Figure 5: Football graph layouts. (A) Walshaw, (B) SDE, (C) LinLog, (D) LDA-NeRV. Colors correspond to the 12 football conferences.

Publication 3 proposes a two-stage solution for graph visualization: first define and model the structure of interest as explicitly as possible; then produce an optimal visualization based on a model of how the user uses the visualization. In LDA-NeRV SSN-LDA models the graph structure; because the structure can be understood and visualized in terms of node similarity, NeRV can be used for the visualization. A different model of graph structure might benefit from a different visualization model.

Of course this solution can be applied to visualizing other kinds of data. In the case of NeRV, the model of structure is encoded in the form of the 'input probability distribution' p_{j_i}, which postulates that local neighborhood structure is important (Section 3.1). Although the model superficially focuses on local features of single data points, in this case optimizing the local structure does bring out certain kinds of global features in practice, as we will see in Section 5.

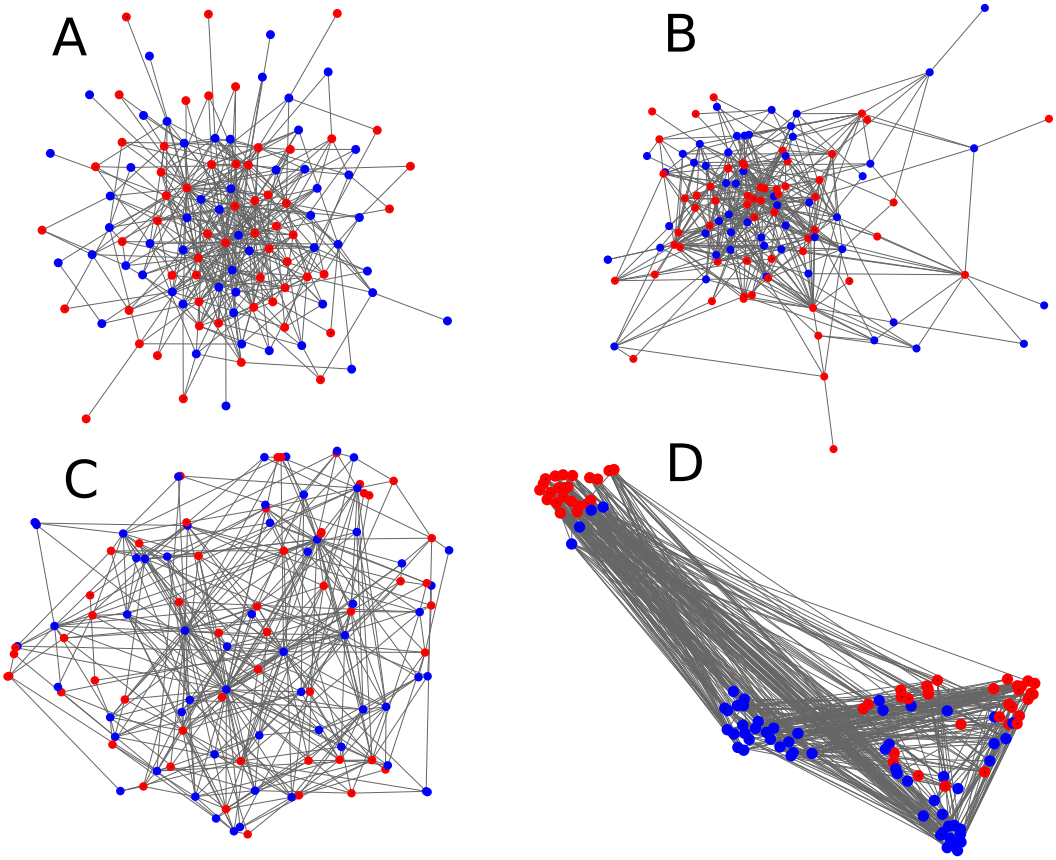


Figure 6: Adjective-noun graph layouts. (A) Walshaw, (B) SDE, (C) LinLog, (D) LDA-NeRV. Colors: blue nodes are adjectives, red nodes are nouns.

5 What exploratory data visualization can tell us about voxel selection

When working with high-dimensional data, creative use of exploratory visualization can be an invaluable tool for gaining intuition and generating hypotheses to guide further research. In Section 2.4.3 we saw that a small set of randomly selected voxels performed surprisingly well in the two-class classification task: averaged over subjects, the classification rate was clearly above chance, and only 12 percentage points below the classification rate of the voxels selected using SCCA.

In this section, I will try to shed some light on these observations by using NeRV from Section 3 and Principal Component Analysis (PCA) (Hotelling, 1933) to visually explore the different voxel sets. As is typical for exploratory visualization, the results not only give us some tentative answers to the questions we set out with, but also raise questions we didn't think of asking before we saw the visualizations.

5.1 Visualization by dimensionality reduction in fMRI: Representational Similarity Analysis

The idea of visualizing fMRI data using unsupervised dimensionality reduction is not new as such. According to Kriegeskorte et al. (2008), the use of multi-dimensional scaling (MDS; Kruskal, 1964) for visualizing fMRI data was first suggested by Edelman et al. (1998). More recently, Kriegeskorte et al. have introduced the Representational Similarity Analysis (RSA; Kriegeskorte et al., 2008) framework, where visualization using MDS plays an important part.

Kriegeskorte et al. proposed RSA as one solution to the challenge of quantitatively relating data from the three main branches of neuroscience: brain-activity measurement, behavioral measurement, and computational modeling. Even within a single branch, there is no obvious general way to relate data from different sources; for instance, how does one quantitatively compare microelectrode recordings from a single neuron with fMRI measurements of the BOLD response in the entire brain? One might simultaneously also want to relate the two to the predictions of computational models of some aspect of the cognition being studied.

RSA addresses this problem by shifting the focus from the data vectors to relationships between the data vectors. For each data set, we compute a pairwise dissimilarity matrix — called a *representational dissimilarity matrix* or RDM in the context of RSA — exactly as we would as the first step for computing NeRV (Section 3.1.1). Instead of comparing the samples from different data modalities or models directly, we can compare their RDMs. RDMs serve as a kind of universal data format. Conceptually the idea is very similar to multi-view kernel methods like Kernel CCA (Lai and Fyfe, 2000), where instead of relating two different data sets directly, we compute a kernel matrix for each data set, and relate the kernel matrices.

As one part of the RSA analysis, Kriegeskorte et al. advocate visualizing RDMs using MDS; others have also applied it successfully (Cichy et al., 2014). Kriegesko-

rte et al. motivate the use of MDS by noting that "similar entities will be placed together, dissimilar entities apart" (Kriegeskorte et al., 2008, p. 11). As a disadvantage of MDS visualizations they mention the unpredictable distortion of distances and the lack of error bars or other statistical indicators. We can see NeRV visualizations like the ones in this section as a refinement of the basic visual RSA analysis: compared with MDS, we have some control over how distances are distorted by by controlling the precision–recall trade-off as described in Section 3.

5.2 The data

In Section 2.4.3 three different subsets of fMRI voxels were compared:

1. the entire brain;
2. the voxels selected as relevant by our variable selection algorithm ('stable voxels');
3. and a randomized subset of voxels (obtained by 'shuffling' the relevant voxels; see Section 2.4.3 for details).

We can interpret each of these voxel subsets as a model of the brain that generates its own data set: the fMRI measurements restricted to just that particular subset of voxels. We can compute an RDM for each voxel set and visualize the RDMs.

The voxel selection was done separately for each subject, so each visualization also corresponds to a single subject. To guard against drawing conclusions based on outliers, I visualized all the subjects, but in the interest of clarity and brevity I will only show and discuss visualizations for one subject. This is one of the limitations of exploratory visualization: the effectiveness of discussing visualizations in text scales poorly with the number of visualizations.

I chose subject #6 because they were one of the subjects for whom the SVM classification rate for stable and random voxels was fairly close; as I mentioned in Section 2.4.3, there was notable individual variability here. For subject #6, the classification rate for the entire brain was 83.78%. The classification rate for stable voxels (stability threshold 0.4, yielding 535 stable voxels out of 219727) was 81.08%, and the classification rate for the corresponding random voxel subset was 75.81%. To see where the stable voxels are located in the brain, consult Figure 1.

The random voxel subset classification rate quoted above is an average over several different random subsets. There is no reasonable way of averaging the NeRV visualizations shown below, so I will instead show visualizations of several random sets.

5.3 Visual analysis using NeRV and PCA

The left column of Figure 7 shows NeRV visualizations of the RDMs corresponding to the full brain, the stable voxels and one random voxel set. NeRV was computed using $\lambda = 0.9$, emphasizing precision (see Sections 3.1.1 and 3.1.2). The dissimilarity measure used for the RDMs was $(1 - r_p(i, j)$, where $r_p(i, j)$ is the Pearson correlation

of fMRI volumes i and j, computed over the voxel subset in question. This is one of the dissimilarity measures suggested in (Kriegeskorte et al., 2008) for fMRI data. For comparison I have also included PCA visualizations, shown in the right column of Figure 7. The PCA visualizations were generated for each data set by projecting it onto its first two principal components. PCA is computed directly for the data matrix without first computing an RDM, so no distance measure is explicitly involved; nevertheless, PCA can be interpreted as the linear projection that minimizes the reconstruction error as measured by Euclidean distance (see, e.g., Shawe-Taylor and Cristianini, 2004).

5.3.1 Overview based on static plots

Looking at the NeRV visualization for the full brain in the upper left of Figure 7, we can immediately see that the 'happy' and 'sad' classes are very cleanly separated. More precisely, we can enclose all the samples from one class using a small number (relative to the number of samples) of convex polygons without including a single sample from the other class, as illustrated in Figure 8.

As discussed in Section 3.1, the NeRV algorithm is completely unsupervised and merely tries to represent pairwise relationships between neighbors faithfully; it does not use the class labels, and does not perform any kind of clustering. Thus the fact that a two-dimensional NeRV projection of a 219727-dimensional data set achieves perfect separation between the classes strongly suggests that a good supervised classification method should perform very well. One might even wonder why the SVM classification rate wasn't even higher than 83.78%, but it is important to keep in mind that the SVM used was a linear classifier, whereas NeRV is a nonlinear projection method.

Comparing the NeRV visualizations of the stable voxels and one random voxel set in Figure 7 with the full-brain visualization, we see that the separation between the classes deteriorates slightly for the stable voxels, and more noticeably for the random voxel set, but overall it remains nearly as good as for the full brain. There are some differences between visualizations of different random voxel sets (Figure 9), but the class separation is consistently good.

Overall, the visualizations would lead us to expect the classification results that we witnessed in Section 2.4.3. It seems that there is something in our fMRI data that separates happy music from sad music quite clearly, and that distinguishing quality is measurable everywhere in the brain.

The PCA visualizations in the right column of Figure 7 do not separate the classes particularly well. The data also looks less structured in general; I will discuss this further below.

5.3.2 Detailed analysis using interactive neighbor retrieval

Apart from the obvious class separation discussed above, Figure 7 also hints at more subtle structure in the data:

- There seems to be a slight asymmetry between the classes: in the full-brain and

stable-voxel visualizations, the samples in the sad class seem to be relatively tightly clustered, whereas the happy samples are more evenly dispersed.

- The full-brain and stable-voxel visualizations contain many formations where multiple samples from the same class are stringed together, as if tracing out a curve. These strings might be consecutive samples from the same song. This would not be surprising, as the BOLD response in two consecutive scans are naturally correlated.

- On the other hand, not all clusters take the form of a chain. In particular, all three NeRV visualizations feature one conspicuous cluster of sad samples on the right side, far away from the other sad samples, and surrounded by happy samples. If samples from some songs form a chain and samples from some songs clump together, the samples that clump together might indicate a song with a stronger reaction (e.g., recognition), because it would suggest that the correlation between samples in the song in general is stronger than the correlation between consecutive samples.

- The random-voxel visualization seems to make the distance from a sample to its nearest neigbor more uniform across samples. Happy and sad are closer together, and the sad samples seem somewhat less tightly clustered.

All the conjectures above were formulated based on the static plots in Figure 7. To verify them requires the ability to interact with the visualization and the data. At the very least we need to be able to determine which sample in the data set a certain glyph in the visualization corresponds to — that is, we need to *retrieve data points based on the visualizations*, precisely the task for which NeRV was formulated. Ideally we would have an interface that would look roughly like Figure 7, but allow us to select glyphs in any of the individual scatter plots. Once the user selects a set of glyphs, the interface would both list the corresponding samples in the data set and highlight the corresponding glyphs in the other scatter plots.

For the purpose of this experiment, I implemented a makeshift version of such an interface by loading a visualization I wished to interact with into Cytoscape (Shannon et al., 2003) as a layout of a graph that has 150 nodes and no edges. Below I will attempt to report my findings from interactively exploring the visualizations in Figure 7. Figure 10 contains the same visualizations shown in Figure 7, but annotated to highlight features referred to in the text.

5.3.3 The conspicuous sad cluster

I began by investigating the conspicuous cluster of sad samples marked with an ellipse in the all-voxel NeRV visualization in Figure 10. I discovered that this cluster consists of 15 consecutive samples — all the samples from one of the sad songs — and that it is indeed clearly visible in every visualization (see the ellipses in the other visualizations). In the random voxel NeRV visualization the formation is very close to neighboring happy samples, but still separate and tightly clustered.

It is particularly noteworthy that this cluster is visible and clearly separate from the rest of the data even in the PCA visualizations, where all the other samples form one large diffuse cloud. The axes correspond to the directions in which the variance in the data is largest, so it seems that the subject's BOLD response for this particular song is somehow markedly different from everything else. The NeRV projections also suggest this, but because they were computed using $\lambda = 0.9$ to emphasize precision, the visualization might allow some samples that are actually close to each other in the data to become separated in the visualization, as with the cut-open sphere in Section 3.1, so one must be more careful about making conclusions based on points being separated in the visualization than one would for a visualization that emphasizes recall.

In this case the NeRV and PCA visualizations together offer sufficient evidence that we can be relatively certain that this cluster of samples is somehow unique, but determining the cause requires further research. Perhaps the subject had a unique reaction to this particular song, or perhaps it was just a scanner artefact that happened to be active during that song. It could even be both: perhaps the subject having a unique reaction to the song and consequently slightly moved their head while the song was playing.

5.3.4 Another notable cluster

In addition to the 'elliptic' cluster discussed in the previous section, the stable-voxel NeRV visualization has another notable tight cluster of sad samples, enclosed by a black rectangle in Figure 10. Again the black rectangles elsewhere in Figure 10 mark the areas where the samples in this cluster landed in the other visualizations.

The samples in this cluster are also from a single song, but this time it does not include every sample from that song; some of the remaining samples are in fact quite far. Another difference from the 'elliptic' cluster is that these samples do not form an obvious cluster in every visualization; this is one example of structure that the stable voxels capture better than the random voxels. In the random-voxel NeRV and all-voxel PCA visualizations the samples are still quite neatly together, but distances within the cluster and between the cluster and its neighbors are similar. In the random-voxel PCA visualization the cluster is merged with other samples to the extent that there is no point in drawing a rectangle: to include all the samples, it would need to cover most of the data.

5.3.5 One song, two strings

I examined several string formations in the all-voxel and stable-voxel NeRV visualizations, and they did generally represent consecutive or nearly consecutive samples from one song, as conjectured.

There were two interesting exceptions, however, marked by green rounded rectangles in the all-voxel NeRV visualization in Figure 10. Together these two strings form one complete song, but the string in the upper rectangle contains all the odd-numbered samples in sequence, whereas the lower rectangle contains all the even-numbered samples in sequence.

This suggests that there is some kind of periodic structure, with a period of about four seconds, in the BOLD response for this specific song. The phenomenon could naturally be an irrelevant artefact, but it nevertheless warrants closer investigation.

This structure is only present in the all-voxel NeRV visualization, which is why the green rectangles are missing from the others. In the stable-voxel visualization, for example, the samples from this song are split into two groups, but not according to the parity of the sample index as in the all-voxel visualization. The two groups are also quite far from each other in the stable-voxel plot. Whatever the periodic structure is, it does not seem to be present in the stable voxels (or the random voxels), or at least it is weaker than the correlations between the samples in the song in general.

5.4 Discussion

The results from the exploratory data analysis both support and elaborate the classification results Section 2.4. It is clear based on the NeRV visualizations that we should expect good performance from a support vector machine trained on the random voxels. The distances between samples from different classes seem to shrink in the random-voxel visualizations, which might account for the lower classification performance.

The visualizations also suggest that there is subtle structure in the BOLD measurements that the stable voxels captured but the random voxels did not. The all-voxel and stable-voxel NeRV visualizations contain clusters and other patterns that would be clearly visible even if we omitted the class information by replacing the red and blue glyphs in Figure 7 with black dots, whereas the random-voxel visualizations would appear far noisier.

One may of course ask to what extent we should trust conjectures based on visualizations. After all, a low-dimensional visualization of a high-dimensional data set is necessarily distorted in some way, and the human analyst may be biased. The most conservative answer is that we should not and need not trust visualizations: to verify a conjecture inspired by a visualization, we can always devise a specific, separate experiment that is as rigorous as we like.

In practice, however, we may sometimes be able to make some conclusions based on a visualization alone. I would argue that the NeRV 'neighborhood retrieval' task formulation was a natural fit for the actual analysis tasks that I performed above. Particularly in the interactive analysis I was literally looking at points and retrieving their neighbors from the display. Hence knowing that the visualizations should have high precision helped me judge whether I should trust a conjecture or not. For example, when I saw a tight cluster in a visualization, I could be relatively confident that it corresponds to a tight cluster in the high-dimensional data (in terms of the distance measure used, correlation). I was more careful to draw conclusions about two points being on the opposite sides of the visualization. Even if I still want to carry out a separate quantitative experiment to be certain, knowing that the algorithm generally can (or cannot) be trusted to visualize a specific property (e.g., tight clusters) reliably will at least help with prioritizing which conjectures to

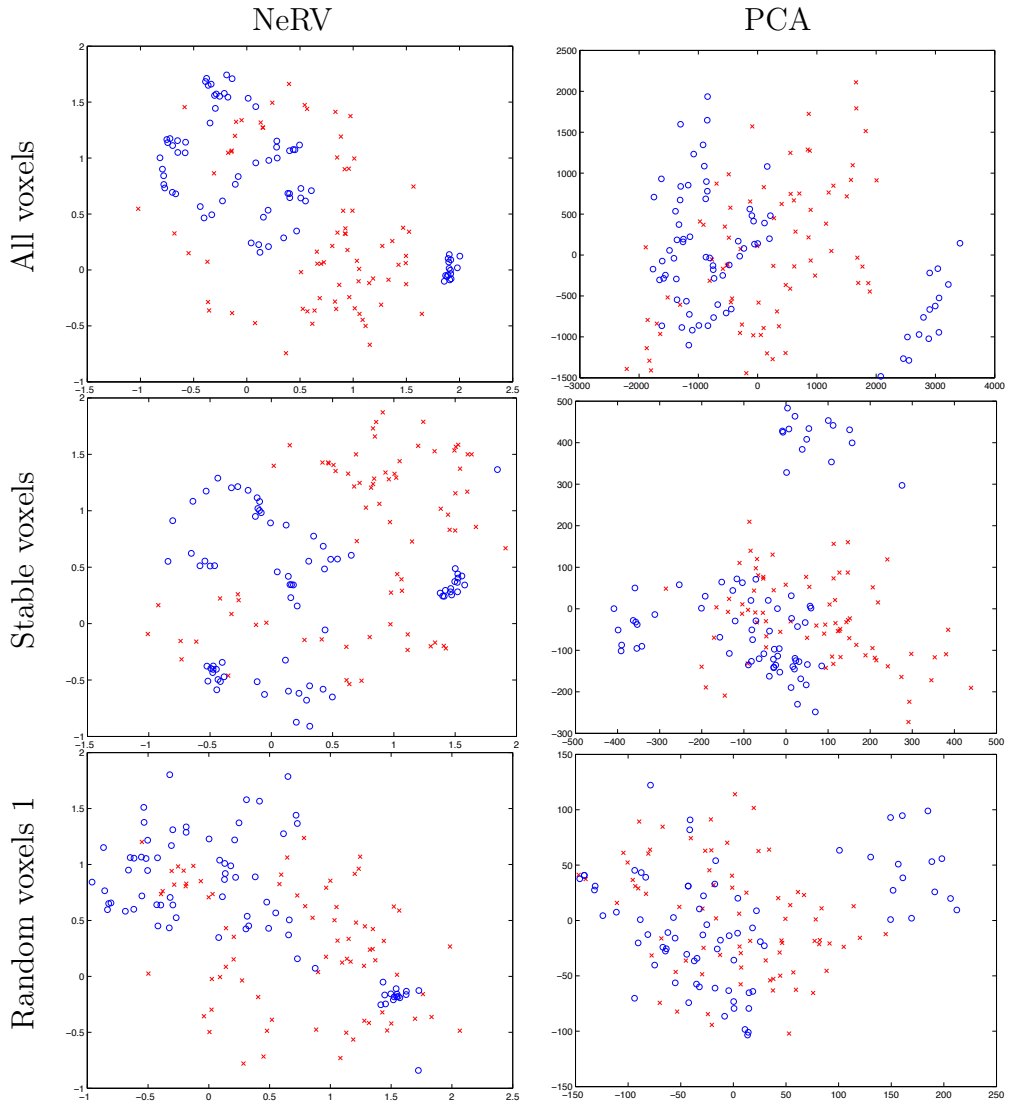


Figure 7: NeRV visualizations (first column) and PCA visualizations (second column) for subject #6. The NeRV visualizations were computed using (1 - Pearson's R) as the dissimilarity measure. Blue circles represent samples from sad songs; red crosses represent samples from happy songs. The black ellipses, the black rectangles, and the green rounded rectangles have been added to highlight features discussed in Section 5.3.2.

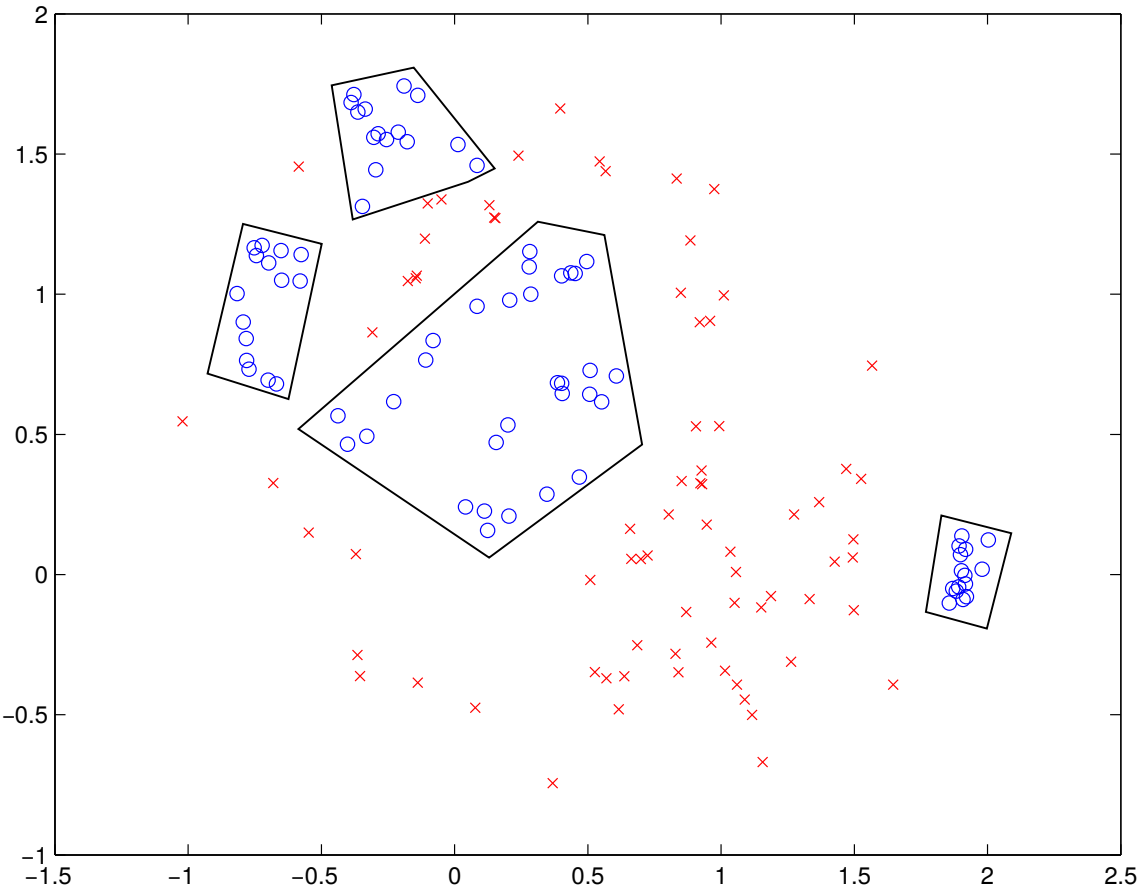


Figure 8: The full-brain NeRV visualization from Figure 7 with polygons added to illustrate the separability of the two classes.

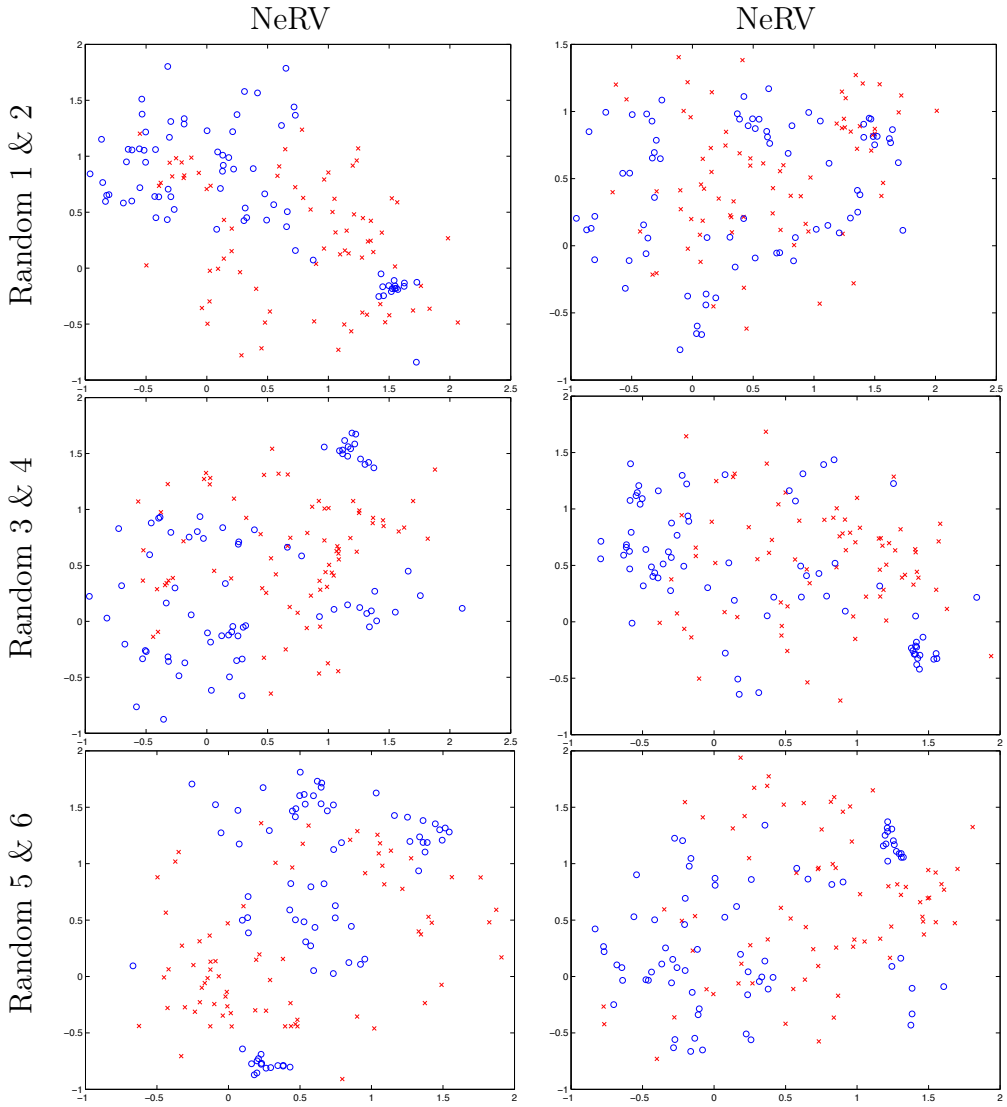


Figure 9: NeRV visualizations of six different random voxel subsets for subject #6 using (1 - Pearson's R) as the dissimilarity measure. Blue circles represent samples from sad songs; red crosses represent samples from happy songs.

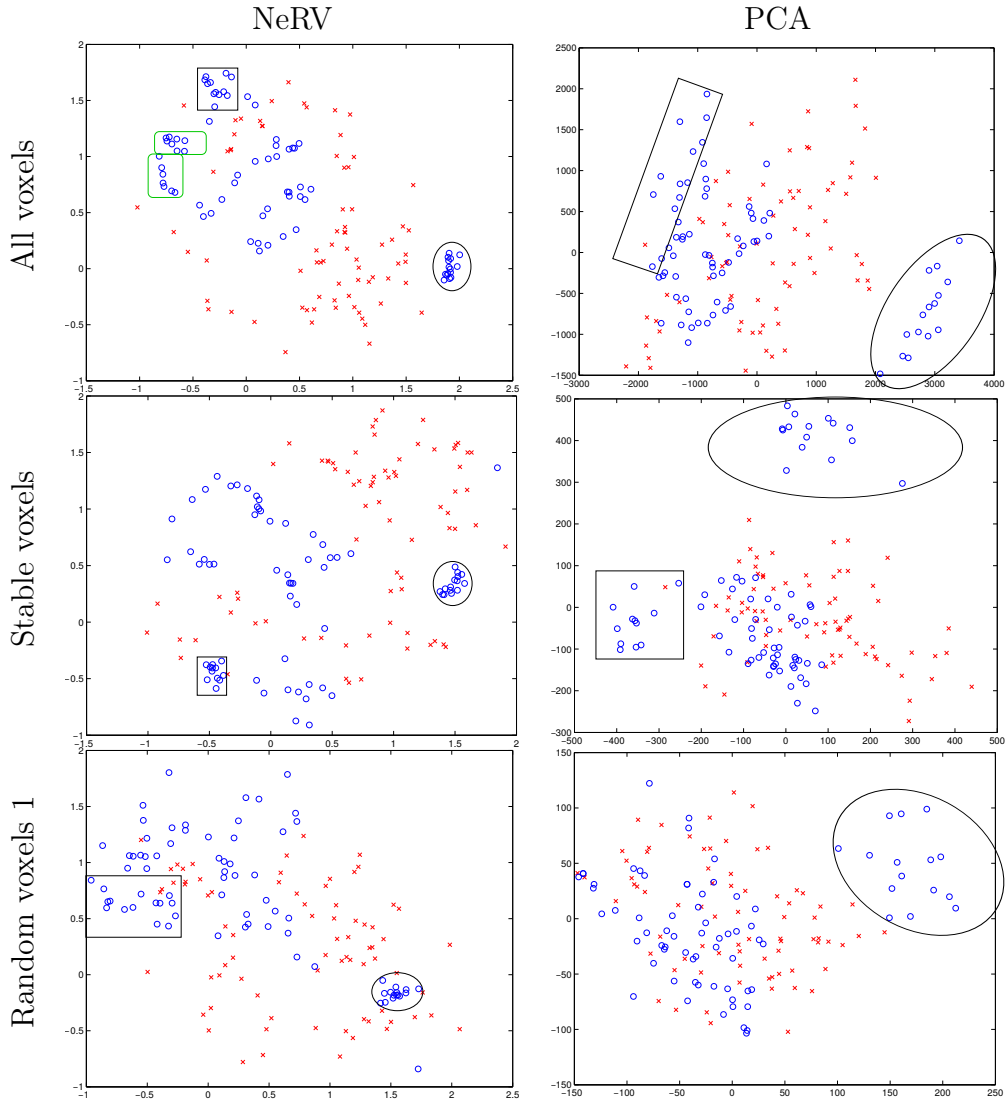


Figure 10: The same visualizations shown in Figure 7, but with annotations (the black ellipses, the black rectangles, and the green rounded rectangles) added to highlight features discussed in Section 5.3.2.

explore first.

The PCA visualizations on their own were generally less informative than the NeRV visualizations: they simply did not show most of the structure that gave rise to the conjectures investigated in Section 5.3.2. On the other hand, PCA did offer important supporting evidence in some cases, most notably the 'elliptic cluster' in Section 5.3.3. The lesson here is that visualizing the same data using several different algorithms at the same time can often give extra information at almost no additional cost. We need to make trade-offs when we are creating a single visualization, but when we are performing interactive real-world data analysis we can study several complementary visualizations in parallel, as I did here with NeRV and PCA. In that vein, having additional NeRV visualizations generated with $\lambda = 0.1$ to emphasize recall would have been particularly useful to complement the precision-focused $\lambda = 0.9$ NeRV visualizations, but in the interest of readability I did not want to increase the number of figures in the text further.

6 Conclusions

Each of the publications included in this thesis contributes a new algorithm for an established problem, but more importantly, each of them tries to argue for a paradigm that is somehow different from the one shared by existing algorithms. Publication 1 champions the use of rich, high-dimensional stimulus features for voxel selection, in lieu of or in addition to the categorical features associated with traditional fMRI experimental designs (block design, event design) and traditional analysis methods (e.g., SPM). Publication 2 asserts that visualization of high-dimensional data, as well as quantitative evaluation of visualizations, should be based on an explicit model of how the user uses the visualization. Publication 3 states that the same should apply to graph visualization, and that it is also important to have an explicit model for the structure that one wants to visualize.

Because the new algorithms break with established paradigms, they actually solve different problems than existing algorithms, which makes comparison and hence validation difficult. Publication 2 validates NeRV mainly by introducing and theoretically justifying quantitative quality measures for the new visualization paradigm, which are then used to compare NeRV with existing methods. One of my intentions with the case study in Section 5 was to complement the solid theoretical justifications with empirical evidence both of the algorithm's usefulness in practical data analysis and of the relevance of the 'visual neighbor retrieval' task formulation.

The classification experiment in Publication 1 shows that the voxel selection method successfully recovers something that is relevant to the experience of listening to music, but it is difficult to assess how strong the results are. The good performance of the random voxels suggests that this particular classification task is weak for evaluating a voxel selection algorithm, because sufficient information for discriminating between the classes can be found in large portions of the brain. This might be a neuroscientifically interesting result in its own right. I would also be interested in knowing how well random voxels generally perform in SVM classification

tasks; this does not seem to have been widely studied.

The exploratory data analysis in Section 5 reveals that although the random voxels have surprisingly good class separation in terms of 'happy' and 'sad', the stable voxels and the full brain contain more subtle structure that is not evident in the random voxels. This suggests that the difference between stable and random voxels might be larger in some other task. Further experiments are required to determine whether the proposed voxel selection method is truly effective, or merely somewhat better than random.

The subtle structure found in the stable-voxel NeRV visualizations also gave rise to some interesting conjectures and insights about the fMRI data being visualized, which gives some evidence for the value of exploratory data analysis in general and RSA-style (Kriegeskorte et al., 2008) fMRI analysis in particular. NeRV seems to be a useful visualization method for RSA analysis. Kriegeskorte et al. originally suggest MDS (Kruskal, 1964), which I have not tried, but the experiments in Publication 2 show that NeRV is generally superior to MDS in terms of the neighbor retrieval quality measures. Because neighbor retrieval turned out to be a useful task formulation for the RSA analysis — the conjectures in Section 5 were mainly based on visually retrieving neighbors from the visualizations — it seems possible that NeRV would perform better than MDS in practice, although further experiments would be required to establish that.

In Publication 3, LDA-NeRV is mainly validated by showing visualizations of graphs that contain the kind of structure that LDA-NeRV is designed to visualize. Publication 3 also features a KNN classification task that I did not discuss in Section 4. The visualizations are enough to establish that LDA-NeRV reveals certain kinds of graph structure much better than existing graph drawing algorithms, but they do not indicate whether the algorithm could be even better.

More important than the algorithms themselves, however, are the ideas behind them. Regardless of whether the combination of SCCA and stability selection suggested in Publication 1 is a good algorithm for selecting voxels, I believe that there is value in using rich stimulus features for selecting variables and analyzing fMRI data in general. Regardless of whether NeRV is a good algorithm for visualizing high-dimensional data, I believe that there is value in explicitly modeling how people use visualizations, and optimizing the visualization for that. Regardless of whether LDA-NeRV is a good algorithm for visualizing graphs, I believe that there is value in explicitly modelling the structure that one wants to visualize.

Regarding visualization, it seems to me that many traditional dimensionality reduction methods like PCA and MDS(Kruskal, 1964) are based on the idea of minimizing some kind of very general reconstruction error that makes as few assumptions about the structure of the data as possible. The same could be said of traditional graph visualization methods, which, I argued in Section 4.1), seem to try to produce a visualization from which the user could reconstruct the graph. I would argue that, for data visualization, an algorithm that visualizes a specific kind of relevant structure reliably and ignores every other aspect of the data is more useful than an algorithm that tries to do everything. For example, by studying a LinLog(Noack, 2007) visualization of a graph, an analyst can quickly tell with

high confidence whether the graph has assortative clusters or not; knowing that the graph does not have them can be just as informative as knowing that it does. Traditional force-based graph visualization methods cannot offer similar assurances because their optimization criteria are difficult to interpret visually. I feel that this is a strong argument for adopting model-based visualization methods in exploratory data analysis.

References

Alluri, V., Toiviainen, P., Jääskeläinen, I. P., Glerean, E., Sams, M., and Brattico, E. (2012). Large-scale brain networks emerge from dynamic processing of musical timbre, key and rhythm. *NeuroImage*, 59(4):3677–3689.

Amaro, Jr, E. and Barker, G. J. (2006). Study design in fmri: Basic principles. *Brain and Cognition*, 60(3):220–232.

Ashby, F. (2011). *Statistical Analysis of FMRI Data*. MIT Press.

Bishop, C. M. (2007). *Pattern Recognition and Machine Learning*. Springer, 1 edition.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.

Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Cichy, R. M., Pantazis, D., and Oliva, A. (2014). Resolving human object recognition in space and time. *Nature Neuroscience*, 3:455–462.

Civril, A., Magdon-ismail, M., and Bocek-rivele, E. (2005). Sde: Graph drawing using spectral distance embedding. In *The Proceedings of the 13th International Symposium on Graph Drawing*, pages 512–513. Springer.

Cristianini, N. and Shawe-Taylor, J. (2000). *An introduction to support Vector Machines: and other kernel-based learning methods*. Cambridge University Press, New York, NY, USA.

Davis, T. and Poldrack, R. A. (2013). Measuring neural representations with fmri: practices and pitfalls. *Annals of the New York Academy of Sciences*, 1296(1):108–134.

Edelman, S., Grill-Spector, K., Kushnir, T., and Malach, R. (1998). Toward direct visualization of the internal shape representation space by fMRI. *Psychobiology*, 26(4):309–321.

Friston, K. (1995). Functional and effective connectivity in neuroimaging: A synthesis. *Human Brain Mapping*, 2:56–78.

Friston, K. J., Holmes, A. P., Worsley, K. J., Poline, J.-P., Frith, C. D., and Frackowiak, R. S. (1994). Statistical parametric maps in functional imaging: a general linear approach. *Human brain mapping*, 2(4):189–210.

Fruchterman, T. M. J. and Reingold, E. M. (1991). Graph drawing by force-directed placement. *Software — Practice and Experience*, 21(11):1129–1164.

Girvan, M. and Newman, M. E. J. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences USA*, 99(12):7821–7826.

Grant, M. and Boyd, S. (2008). Graph implementations for nonsmooth convex programs. In Blondel, V., Boyd, S., and Kimura, H., editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited. http://stanford.edu/~boyd/graph_dcp.html.

Grant, M. and Boyd, S. (2014). CVX: Matlab software for disciplined convex programming, version 2.1. http://cvxr.com/cvx.

Hachul, S. and Juenger, M. (2007). Large-graph layout algorithms at work: An experimental study. *Journal of Graph Algorithms and Applications*, 11(2):345–369.

Hadany, R. and Harel, D. (1999). A multi-scale algorithm for drawing graphs nicely. In *WG '99: Proceedings of the 25th Int. Workshop on Graph-Theoretic Concepts in Compute Science*, pages 262–277. Springer.

Hall, K. M. (1970). An r-dimensional quadratic placement algorithm. *Management Science*, 17(3):219–229.

Hardoon, D. R. and Shawe-Taylor, J. (2011). Sparse canonical correlation analysis. *Machine Learning*, 83(3):331–353.

Herman, I., Society, I. C., Melançon, G., and Marshall, M. S. (2000). Graph visualization and navigation in information visualization: a survey. *IEEE Transactions on Visualization and Computer Graphics*, 6:24–43.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441, 498–520.

Hotelling, H. (1936). Relations Between Two Sets of Variates. *Biometrika*, 28(3/4):321–377.

Kriegeskorte, N., Goebel, R., and Bandettini, P. (2006). Information-based functional brain mapping. *Proceedings of the National Academy of Sciences of the United States of America*, 103(10):3863–3868.

Kriegeskorte, N., Mur, M., and Bandettini, P. (2008). Representational similarity analysis–connecting the branches of systems neuroscience. *Frontiers in systems neuroscience*, 2.

Kruskal, J. B. (1964). Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrica*, 29(1):1–26.

Kruskal, J. B. and Seery, J. B. (1980). Designing network diagrams. In *Proc. First General Conference on Social Graphics*, pages 22–50.

Lai, P. L. and Fyfe, C. (2000). Kernel and nonlinear canonical correlation analysis. *Int. J. Neural Syst.*, 10(5):365–377.

Lartillot, O., Toiviainen, P., and Eerola, T. (2008). A Matlab Toolbox for Music Information Retrieval. *Data Analysis, Machine Learning and Applications*, pages 261–268.

Logothetis, N. K. (2008). What we can do and what we cannot do with fmri. *Nature*, 453:869–878.

Mahmoudi, A., Takerkart, S., Regragui, F., Boussaoud, D., and Brovelli, A. (2012). Multivoxel pattern analysis for fmri data: A review. *Comp. Math. Methods in Medicine*, 2012.

Martino, F. D., Valente, G., Staeren, N., Ashburner, J., Goebel, R., and Formisano, E. (2008). Combining multivariate voxel selection and support vector machines for mapping and classification of fmri spatial patterns. *NeuroImage*, 43(1):44–58.

Meinshausen, N. and Bühlmann, P. (2010). Stability selection. *Journal of the Royal Statistical Society, Series B*, 72:417–473.

Mitterschiffthaler, M. T., Fu, C. H., Dalton, J. A., Andrew, C. M., and Williams, S. C. (2007). A functional MRI study of happy and sad affective states induced by classical music. *Human brain mapping*, 28(11):1150–1162.

Newman, M. E. J. (2006). Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, 74(3):036104.

Noack, A. (2007). Energy models for graph clustering. *Journal of Graph Algorithms and Applications*, 11(2):453–480.

Noack, A. (2010). Linloglayout. http://code.google.com/p/linloglayout/.

Norman, K. A., Polyn, S. M., Detre, G. J., and Haxby, J. V. (2006). Beyond mind-reading: multi-voxel pattern analysis of fmri data. *Trends in Cognitive Science*, 10:424–430.

Salmela, P., Nevalainen, O. S., and Aittokallio, T. (2008). A multilevel graph layout algorithm for cytoscape bioinformatics software platform. Technical Report 861, Turku Centre for Computer Science.

Shannon, P., Markiel, A., Ozier, O., Baliga, N. S., Wang, J. T., Ramage, D., Amin, N., Schwikowski, B., and Ideker, T. (2003). Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Research*, 13(11):2498–2504.

Shawe-Taylor, J. and Cristianini, N. (2004). *Kernel Methods for Pattern Analysis*. Cambridge University Press.

Tenenbaum, J. B., Silva, V. D., and Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*.

Uurtio, V. (2015). personal communication.

Uurtio, V., Bomberg, M., Nybo, K., Itävaara, M., and Rousu, J. (2015). Canonical correlation methods for exploring microbe-environment interactions in deep subsurface. In Japkowicz, N. and Matwin, S., editors, *Discovery Science: 18th International Conference, DS 2015*. Springer.

Varoquaux, G., Gramfort, A., and Thirion, B. (2012). Small-sample brain mapping: sparse recovery on spatially correlated designs with randomization and clustering. In John, L. and Joelle, P., editors, *ICML*. icml.cc / Omnipress.

Venna, J. (2007). *Dimensionality Reduction for Visual Exploration of Similarity Structures*. PhD thesis, Helsinki University of Technology, Espoo, Finland.

Walshaw, C. (2003). A multilevel algorithm for force-directed graph drawing. *Journal of Graph Algorithms and Applications*, 7(3):253–285.

Ware, C. (2004). *Information Visualization: Perception for design*. Elsevier, 2 edition.

Yamashita, O., aki Sato, M., Yoshioka, T., Tong, F., and Kamitani, Y. (2008). Sparse estimation automatically selects voxels relevant for the decoding of fmri activity patterns. *NeuroImage*, 42(4):1414 – 1429.

Zhang, H., Qiu, B., Giles, C. L., Foley, H. C., and Yen, J. (2007). An LDA-based community structure discovery approach for large-scale social networks. In *Intelligence and Security Informatics (ISI) 2007*, pages 200–207. IEEE.

A An alternative MATLAB implementation of SCCA using the CVX package

```
%Input parameters are the same as for David Hardoon's SCCA2.m.
%Before you run this, make sure CVX is installed (unpack CVX files
%somewhere) and setup (call cvx_setup in MATLAB in the CVX folder)
%and that you're using the SeDuMi solver (call 'cvx_solver sedumi'
%after you've installed and setup cvx). The default solver SDPT3 seems
%to run into numerical problems with the SCCA optimization problem.
%The numerical problems seemed to disappear if I set mu and gamma right,
%but SeDuMi seems to be able to handle mu and gamma as they are computed
%in David's code.
function [w, e, correlation, optval, beta] = scca_cvx_singleprog_tau(X, ...
                                              K, seed_index, sk)

primal_dim = size(X,1);
N_samples = size(X,2);
tau = 0.5;

%This is how mu and gamma are set in David's SCCA2.m
Ij = eye(size(K,2));
Ij(seed_index,seed_index) = 0;
c = X*K(:,seed_index);
KK = K'*K;
d1 = 2*tau*(1-tau)*c;
mu = sk*mean(abs(d1));
gamma = mean(abs(2*(1-tau)^2*Ij*KK(:,seed_index)));
beta = 1

cvx_begin
  variable w(primal_dim)
  variable e(N_samples)
  minimize(square_pos(norm(tau * X'*w - (1-tau)*K*e)) + mu*norm(w,1) ...
            + gamma*norm(e,1))
  e >= 0 %Remove this line to remove the non-negativity constraint
  e(seed_index) == 1
  norm(e,Inf) <= 1
cvx_end

optval = cvx_optval;
correlation = corr(X'*w, K*e);
```