Using regression makes extraction of shared variation in multiple datasets easy

Jussi Korpela   Andreas Henelius   Lauri Ahonen
Arto Klami   Kai Puolamäki

May 23, 2016

Abstract

In many data analysis tasks it is important to understand the relationships between different datasets. Several methods exist for this task but many of them are limited to two datasets and linear relationships. In this paper, we propose a new efficient algorithm, termed cocoreg, for the extraction of variation common to *all datasets* in a given collection of arbitrary size. cocoreg extends redundancy analysis to more than two datasets, utilizing chains of regression functions to extract the shared variation in the original data space. The algorithm can be used with any linear or non-linear regression function, which makes it robust, straightforward, fast, and easy to implement and use. We empirically demonstrate the efficacy of shared variation extraction using the cocoreg algorithm on five artificial and three real datasets.

1 Introduction

Discovering relationships between different multivariate datasets is useful in many fields, such as in ecology to study the shared variation between different sites (Legendre and Legendre, 1998) or in neuroscience to locate brain areas with correlated activity across subjects (Hasson et al, 2004; Dähne et al, 2014). One recent important application area are the datasets produced by trendy wearable devices. These *quantified self* measurements usually consist of simultaneously recorded physiological data, physical activity and other personal data, possibly from several individuals. One typical question is which signals and individuals are somehow connected and when. Within this setting it is interesting to complement the traditional single-dataset multivariate methods with between-datasets analyses.

In this paper we focus on finding the shared variation between datasets in a *data collection* that consists of several *datasets*. Each dataset is composed of multiple signals, i.e., vectors of data. The signals can be thought of as being composed of two parts: a *shared* part that is similar to all of the other datasets' signals and the remainder referred to as *residual*. What sets cocoreg apart from most other methods is that it extracts only variation that is shared by *all* of the datasets. The output is a set of shared variation projections, one for each dataset. These projections are related to one another but not necessarily identical.

As an example of extracting shared variation, consider the data shown in the left-most column of Figure 1. The plots show the International Monetary Fund (IMF) commodity prices for products in three categories: energy, metals, and meat. The raw price index data is shown on the left. The curves are spiky and fluctuate. It is difficult to directly draw conclusions regarding any shared economic trend of the different commodities in the datasets.

Here we are interested in extracting this shared trend. The middle column shows the *shared variation* for each commodity in the datasets, i.e., what the signals in one dataset have in common with the signals in *all of the other* datasets. The individual variation visible in the commodity price time series is suppressed and the underlying shared variation is highlighted. An increasing trend from 2010 to 2012 is now visible, and is similar in profile for all products, with the exception of the price development of gas (PNGASUS_USD) and lamb (PLAMB_USD). The trend becomes even more clear when viewed using the first principal component (PC) of the shared variation, plotted as a gray line (PC1) in the middle column.

The extraction of the shared variation allows us to better understand the data and draw conclusions, e.g., a recession affecting multiple commodity prices is visible as a price dip around 2009. Finally, the rightmost column shows the *residual variation* for each commodity, i.e., the amount of variation in each of the price curves not explained by all the other commodities. The residual variation here mostly consists of fast price fluctuations unique to a particular commodity, i.e., the spiky part of the price curves, or fluctuations specific to a subset of the datasets.

There exists a wide variety of different methods for investigating shared variation between datasets. However, most of the methods are only applicable to problems with two datasets and do not easily generalize to problems with three or more datasets. Some of the methods are complex and difficult to implement. In general there are two main approaches to the problem of extracting shared variation: (i) to search for maximally dependent projections between the datasets and (ii) factorization of the variance of the data by constructing a generative model. Canonical Correlation Analysis (CCA) (Hotelling, 1936) and Redundancy Analysis (RDA) (Legendre and Legendre, 1998) belong to the first category, but are limited to the case of two datasets. The generative approach, in turn, leads to a natural generalization to multiple datasets (Klami et al, 2015), but requires making explicit assumptions on the nature of the noise and the relationships in the data.

The focus of this paper is a novel algorithm, termed COCOREG (Common Components by Regression). It is designed for the simultaneous analysis of shared variation between multiple datasets. COCOREG makes only few assumptions about the data and can therefore be used in multiple fields. The algorithm is easy to implement using existing robust and mature regression algorithms.

For two datasets, a regression model from one dataset to another naturally extracts the shared variation by modeling their conditional relationship. In this work we use this observation for extracting variation shared by multiple datasets, by constructing *regression chains* of multiple pairwise regressors. A regression chain is a sequence of regressors transferring the signal from one dataset to another via all other sets, eventu-

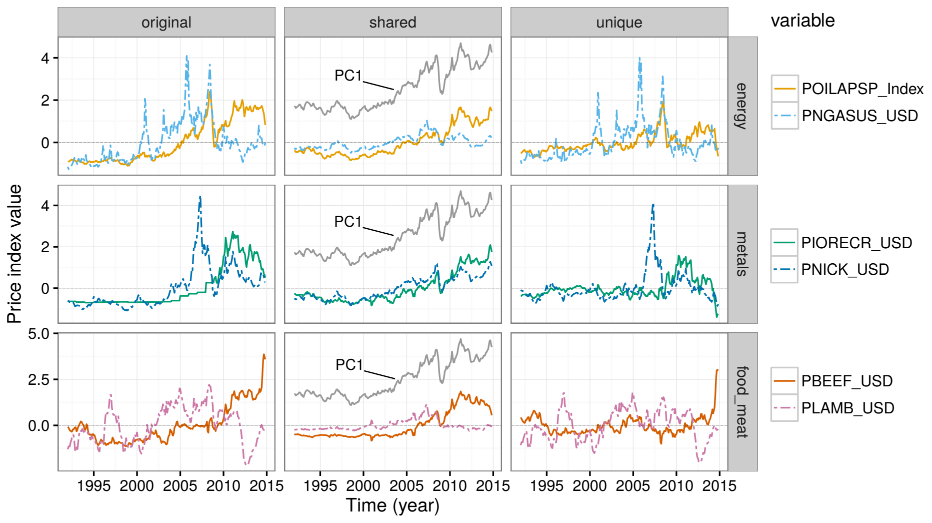


Figure 1: IMF commodity prices for energy, metals and meat. The original prices of the products are shown in the left column, the shared variation for each particular product with the other products in the middle column and the residual variation for each commodity in the right column. The variable PC1 in the middle column (plotted with gray) is the first principal component of the shared variation, scaled to unit variance and shifted up to avoid overplotting.

ally producing a signal containing variation shared by all of the signals. In other words, the regression functions act as filters that only pass shared variation from one dataset to the next.

The cocoreg algorithm estimates the shared variation within the data collection directly, filtering out all variation that is either unique to a single dataset or to a subgroup of datasets. This is in contrast to the generative modeling approach, which solves a computationally more complex task and provides shared variation only as a side result.

Our algorithm has several advantages. The method is extremely flexible and can be easily adapted to different datasets or analysis scenarios simply by selecting a different regressor. Such changes are much more laborious to incorporate into the generative modeling approach. The properties of regression models are well understood and there exists an abundance of powerful regression techniques, which in turn helps in selecting the regressor to use. cocoreg is also easily extendable to non-linear settings by selecting a non-linear regressor. Furthermore, the cocoreg algorithm is fast, making it suitable for exploratory analysis and online applications. The algorithm is simple, intuitive and easy to implement and the output of cocoreg is expressed in the original data space which is often desired for ease of interpretation. If a single time series describing the shared variation is needed, principal component analysis (PCA) can be used as a post-processing step.

Summarizing, the main contributions of this work are

- we present a fast and versatile method for extracting shared variation
- we show the theoretical relation of cocoreg to existing methods
- we show that cocoreg is an extension of RDA to more than two datasets
- we analyze synthetic and real datasets and compare the output of cocoreg to existing methods for finding shared variation

In Section 2 we review related work and discuss the relation of cocoreg to these. In Section 3 we present the cocoreg algorithm and the underlying theory. Section 4 contains empirical experiments in which we demonstrate the utility and properties of cocoreg using both synthetic and real-world datasets. Finally, our conclusions are presented in Section 5.

2 Related work

Since the introduction of CCA (Hotelling, 1936), numerous methods have been proposed for extracting shared variation in datasets. Here we briefly discuss how cocoreg is positioned amongst the most closely related methods.

The problem of decomposing the variation in a dataset into shared and residual variation can in general be solved in two different ways. One approach is to search for maximally dependent projections of the datasets to extract shared variation, explicitly discarding everything else. This is typically done by maximizing some dependency criterion like correlation or various approximations of mutual information. The other approach is to factorize the whole data variance, typically by constructing a generative model that assumes each dataset is an additive composition of shared and residual variation.

The primary method for the former approach is CCA that extracts correlating linear components. This approach is also easy to generalize for non-linear projections by building on, e.g., kernel-based representations (Hardoon et al, 2004) or neural networks (Hsieh, 2000; Andrew et al, 2013). Irrespective of the functional form, the basic optimization problem is still to maximize the dependency of the projections. This approach, however, is difficult to generalize for multiple datasets because the natural optimization criteria of correlation and mutual information are bivariate.

Various multi-set generalizations of CCA have still been presented (Kettenring, 1971; Tenenhaus, 2011), but they retain only a subset of the basic properties of CCA depending on the generalization. As these methods optimize sums of bivariate dependency criteria, the result is a weighted mixture of shared features, some of which might not be common to all of the datasets. This is in contrast to cocoreg, which finds the ”smallest common denominator“ for the datasets.

Another solution would be to optimize with respect to multi-variate extensions of mutual information (Fisher and Darrell, 2003) or recent multivariate extensions of correlation (Nguyen et al, 2014) but these are computationally heavy due to working in (often discretized) multidimensional spaces. See also Hwang et al (2013) for a multiset approach that resembles factor analysis.

The second approach of learning the full factorization requires explicitly assuming a generative process for the data. For example, Klami et al (2013) implemented a Bayesian version of CCA by assuming linear decomposition of variation to latent components, combined with additive Gaussian noise. This Gaussian latent variable model can be seen as a generative version of the traditional CCA with equivalence of the models holding for multivariate Gaussian data.

These kinds of models are easy to generalize to multiple datasets by extending the generative description to simply produce more sets. For multiple datasets they typically assume a richer factorization including not only shared variation, but also terms that describe joint variation between any subsets of the datasets, resulting in the method of group factor analysis (GFA) (Klami et al, 2015). This approach, however, needs to make explicit assumptions on the nature of the noise and does not generalize easily to non-linear mappings. An example of this is the manifold relevance determination (MRD) method by Damianou et al (2012) which allows for nonlinear dependence structures but is computationally heavy. From the perspective of extracting only the shared variation, these type of methods solve a more complex task of fully factorizing the variation, and providing shared variation as a side-result. Typically it pays off to directly solve the problem at hand instead of solving a more difficult one.

There are also methods, such as simultaneous component analysis (SCA) (Timmerman and Kiers, 2003), that divide datasets into components by explicitly assuming some correlation structure for the components. These methods help in exploring possible shared covariance structures but do not solve the shared variation extraction problem defined later in Section 3.1. Their application is also restricted to data collections in which all datasets have exactly the same variables.

cocoreg combines the advantages of both main approaches described above, providing natural support for multiple datasets and easy extensions for nonlinear projections. In terms of modeling principle, it is somewhat of an outlier; it does not explicitly optimize a dependency measure, nor does it specify a generative process. Instead, it operates solely based on chains of bivariate regression functions, which is the main reason for its flexibility. The same principle has earlier been used in RDA (Legendre and Legendre, 1998). RDA is limited to only two datasets and linear regressors, and it can be viewed as a special case of cocoreg. From this perspective, cocoreg extends RDA to multiple datasets and nonlinear regressors.

3 Methods

3.1 Definitions

The following briefly describes the basic notation and definitions used in this paper.

Definition *Data collection, dataset and signal* Let the *data collection* $\mathbf{D} \in \mathbb{R}^{N \times D}$ be a data matrix with *D signals* each having N samples. Each column hence represents a signal denoted as $\mathbf{s}_i = \mathbf{D}[\cdot, i]$. We assume that it is meaningful to compare the signals sample-by-sample. We define a disjoint partition of the columns of the data matrix into K submatrices $\mathbf{D}_k \in \mathbb{R}^{N \times d_k} \subseteq \mathbf{D}$. We refer to these partitions $\mathbf{D}_k$ as the *datasets* in the data collection $\mathbf{D}$.

The notation $\mathbf{D}_{Sk}^{\{k,j\}}$ is used for variation in $\mathbf{D}_k$ that is shared between datasets $\mathbf{D}_k$ and $\mathbf{D}_j$. In general, the (true) shared variation projection is marked $\mathbf{D}_S^{\{\text{all}\}}$and an estimate of it as $\hat{\mathbf{D}}_S^{\{\text{all}\}}$.

The data collections in this paper consist of real-valued time series. The method presented here does not explicitly take the time series nature of the signal into account and hence generalizes to other types of data as well. In the following the signals $\mathbf{s}_i$ are assumed to have zero mean and we use $[N]$ to denote the set $\{1, \ldots, N\}$.

3.2 Identification of shared variation

By shared variation we mean variance that is shared by two or more datasets. In the case of two datasets we would like to divide the datasets into two parts:

$$\mathbf{D}_1 = \mathbf{D}_{S1}^{\{1,2\}} + \mathbf{D}_{S1}^{\{1\}} \tag{1}$$

$$\mathbf{D}_2 = \mathbf{D}_{S2}^{\{1,2\}} + \mathbf{D}_{S2}^{\{2\}} \tag{2}$$

where $\mathbf{D}_{S1}^{\{1,2\}}$ is the variation in $\mathbf{D}_1$ that is shared by datasets 1 and 2 and $\mathbf{D}_{S1}^{\{1\}}$ is the residual variation (and respectively for dataset 2). A problem with this construct is that the partition is not unique. Variance may be freely shifted between the shared and residual components without any change to the covariance matrix of the data collection $\mathbf{D}$.

One way to see this is to form a generative model for the data. Let us assume that the datasets have been generated by three component vectors $\mathbf{z}_1^t \in \mathbb{R}^{d_1}$, $\mathbf{z}_2^t \in \mathbb{R}^{d_2}$, and

$\mathbf{z}_{12}^t \in \mathbb{R}^{\max(d_1,d_2)}$, where each of the components are independent random variables with zero mean and unit variance. More specifically, the generating mechanism is

$$\mathbf{D}_1[t,\cdot] = \mathbf{x}_1^t = \mathbf{A}_{12}\mathbf{z}_{12}^t + \mathbf{A}_1\mathbf{z}_1^t \tag{3}$$
$$\mathbf{D}_2[t,\cdot] = \mathbf{x}_2^t = \mathbf{A}_{21}\mathbf{z}_{12}^t + \mathbf{A}_2\mathbf{z}_2^t \tag{4}$$

where $\mathbf{A}_{12} \in \mathbb{R}^{d_1\times\max(d_1,d_2)}$, $\mathbf{A}_{21} \in \mathbb{R}^{d_2\times\max(d_1,d_2)}$, $\mathbf{A}_1 \in \mathbb{R}^{d_1\times d_1}$, and $\mathbf{A}_2 \in \mathbb{R}^{d_2\times d_2}$ are given real matrices, and $t \in [N]$ is the time index. In this case the "shared component", for example, of dataset 2 is—at least intuitively—given by $\mathbf{D}_2^{\{1,2\}}[t,\cdot] \simeq \mathbf{A}_{21}\mathbf{z}_{12}^t$ and the "residual component" by $\mathbf{D}_2^{\{2\}}[t,\cdot] \simeq \mathbf{A}_2\mathbf{z}_2^t$.

The correlation matrices are given by

$$\begin{aligned}
\mathbf{R}_{11} &= E\left[\mathbf{x}_1^t\mathbf{x}_1^{tT}\right] = \mathbf{A}_{12}\mathbf{A}_{12}^T + \mathbf{A}_1\mathbf{A}_1^T, && (5)\\
\mathbf{R}_{22} &= E\left[\mathbf{x}_2^t\mathbf{x}_2^{tT}\right] = \mathbf{A}_{21} + \mathbf{A}_{21}^T + \mathbf{A}_2\mathbf{A}_2^T, && (6)\\
\mathbf{R}_{12} &= E\left[\mathbf{x}_1^t\mathbf{x}_2^{tT}\right] = \mathbf{A}_{12}\mathbf{A}_{21}^T, && (7)\\
\mathbf{R}_{21} &= \mathbf{R}_{12}^T = \mathbf{A}_{21}\mathbf{A}_{12}^T. && (8)
\end{aligned}$$

It is easy to see that we can, e.g., set $\mathbf{A}_1 = 0$ and still be able to set the other matrices $\mathbf{A}_{12}, \mathbf{A}_{21}$ and $\mathbf{A}_2$ such that the covariance matrices remain intact. This implies that we cannot specify the shared variation component without further restrictions. We will next briefly compare two established ways of defining shared variation in a two dataset case.

3.3 RDA and CCA for two datasets (K = 2)

CCA and RDA are both established methods that characterize shared variation between two datasets using linear projections. The following defines RDA and CCA in the context of this paper and highlights some differences between them.

In RDA the shared variation components of datasets 1 and 2 are:

$$\hat{\mathbf{D}}_{S1}^{\{1,2\}} = \mathbf{B}_{12}\mathbf{D}_2 = \mathbf{R}_{12}\mathbf{R}_{22}^{-1}\mathbf{D}_2, \tag{9}$$
$$\hat{\mathbf{D}}_{S2}^{\{1,2\}} = \mathbf{B}_{21}\mathbf{D}_1 = \mathbf{R}_{21}\mathbf{R}_{11}^{-1}\mathbf{D}_1 \tag{10}$$

where $\mathbf{B}_{ij}$ are set of linear ordinary least squares (OLS) regression coefficients when explaining $\mathbf{D}_i$ with $\mathbf{D}_j$. Hence RDA defines the shared variation of, e.g., dataset 1 to be that part of the signal that can be explained using dataset 2.

In CCA the goal is to find (orthogonal) directions in each of the datasets such that the projections of the data onto these directions are maximally correlated. That is, CCA finds vectors $\mathbf{w}_1$ and $\mathbf{w}_2$ such that

$$\arg\max_{\mathbf{w}_1,\mathbf{w}_2} \mathbf{w}_1^T\mathbf{D}_1^T\mathbf{D}_2\mathbf{w}_2 \simeq \arg\max_{\mathbf{w}_1,\mathbf{w}_2} \mathbf{w}_1^T\mathbf{R}_{12}\mathbf{w}_2 \tag{11}$$

reaches its maximum. Once the vectors are found (see, e.g., Müller (1982)), variation in their direction is removed from the data and the process is repeated until

$d_{CCA} = \max(d_1, d_2)$ orthogonal directions have been found. The output of this process is the original dataset expressed in a new basis. For the dataset whose dimensionality is smaller, this new basis is complete and expresses all variation in the data. For the other dataset some variation might be left unexplained. Hence CCA does not find a single shared variation projection of the data, but rather a set of orthogonal directions which sequentially maximize the correlation between the datasets. In a linear OLS setting maximizing correlation is equivalent to finding the OLS optimal fit.

3.4 Shared variation extraction problem

Following the example of RDA and CCA we adopt the following definition of shared variation:

Definition *Shared variation* The variation that dataset k shares with dataset a is defined as $\mathbf{D}_{Sb}^{\{a,b\}} = f_{a \to b}(\mathbf{D}_a)$, where $f_{a \mapsto b} : \mathbb{R}^{N \times d_a} \mapsto \mathbb{R}^{N \times d_b}$ is some regression model that estimates $\mathbf{D}_b$ using $\mathbf{D}_a$.

The main problem addressed in this paper can be formulated as:

Problem *Shared variation extraction* Given a data collection $\mathbf{D}$ with K datasets, extract from each of the datasets $\mathbf{D}_k, k \in [K]$, the variation shared by all datasets, i.e., $\mathbf{D}_{Sk}^{\{[K]\}}$.

We solve this problem using definition 3.4. This leads a solution that can be seen as a generalization of the RDA method for data collections with more than two datasets ($K > 2$). Next, we will explain how we estimate the shared variation of more than two datasets.

3.5 Regression chains

The goal of cocoreg is to extract variation shared by multiple datasets without explicitly optimizing complex criteria formulated over multiple datasets. We do this by creating a chain of pairwise regression models. To extract the variation that a dataset $\mathbf{D}_k$ shares with all other datasets, we model it as the output of a chain of regressors going through all other datasets. Since we use every dataset in the chain, only variation shared by all datasets will be carried forward. Next, we will describe the concept of *regression chain* in more detail.

The regression chain that finds the shared variation for $\mathbf{D}_k$ starts from some dataset $\mathbf{D}_i$ and always ends in $\mathbf{D}_k$, visiting each of the other $K-2$ datasets $\mathbf{D}_{[K]\setminus\{i,k\}}$ exactly once. Let l be a random permutation of the integers $[K] \setminus \{i, k\}$. To extract the shared variation in $\mathbf{D}_k$ we can now define the regression chain from $\mathbf{D}_i$ to $\mathbf{D}_k$ as

$$\mathbf{D}_{Sk}^{\{[K]\}} = f_{l_{K-2} \to k} \circ f_{l_{K-3} \to l_{K-2}} \circ \ldots \circ f_{l_1 \to l_2} \circ f_{i \to l_1}(\mathbf{D}_i), \tag{12}$$

where $\circ$ denotes function composition.

In the regression chain, we hence start from the dataset $\mathbf{D}_i$ and use this to model the dataset $\mathbf{D}_{l_1}$ using $f_{i \mapsto l_1}$, which gives us the estimate $\mathbf{D}_{l_1}^*$ of $\mathbf{D}_{l_1}$. Next, we use $\mathbf{D}_{l_1}^*$ to

$$\begin{tikzcd}\mathbf{D}_i \arrow[r, "f_{i\mapsto l_1}"] & \mathbf{D}_{l_1} \arrow[r, "f_{l_1\mapsto l_2}"] & \mathbf{D}_{l_2} \arrow[r] & \cdots \arrow[r] & \mathbf{D}_{l_{K-3}} \arrow[r, "f_{l_{K-3}\mapsto l_{K-2}}"] & \mathbf{D}_{l_{K-2}} \arrow[r, "f_{l_{K-2}\mapsto l_k}"] & \mathbf{D}_k\end{tikzcd}$$

Figure 2: The regression chain of Equation (12).

estimate $\mathbf{D}_{l_2}$ using $f_{l_1 \mapsto l_2}$ and so on for each link in the regression chain. This is shown in Figure 2.

The regression function f in Equation (12) is general, and can be any function that allows a mapping from one multidimensional dataset to another, such as ordinary linear regression, regression using support vector machines, random forests etc.

We next present two theorems that describe the behavior of the regression chain at two extreme cases. Let $f_{i \mapsto j}$ be a *consistent regression function* if it outputs a constant value in case the distributions of the datasets $\mathbf{D}_i$ and $\mathbf{D}_j$ are independent, i.e., $\Pr(\mathbf{D}_j \mid \mathbf{D}_i) = \Pr(\mathbf{D}_j)$.

Theorem Full loss of shared variation. For a consistent regression function the regression chain of Equation (12) outputs a constant value if the distribution of any of the datasets is independent of the other datasets.

Theorem Full transfer of shared variation. If all of the regression functions in the the regression chain of Equation (12) have zero error, then the output of the chain also has zero error.

Proof. The proofs of Theorems 3.5 and 3.5 follow directly from Equation (12) and the definition of the consistent regression function. □ □

In other words, if at least one of the datasets is totally unrelated to the rest of the datasets, the chain breaks and outputs a constant value (Theorem 3.5). On the other hand, if all of the datasets contain the same variation and this is captured by the regression function, the shared variation will be equal to the datasets themselves (Theorem 3.5). Intuitively, the chain of regressors suppresses the part of the signal that is not shared by all of the datasets and maintains the shared part.

Equation (12) defines a COCOREG projection using a single regression chain. However, there are $K - 1$ ways to choose the starting dataset $\mathbf{D}_i$ and $(K - 2)!$ possible permutations of the intermediary datasets between $\mathbf{D}_i$ and $\mathbf{D}_k$, giving $(K - 1)!$ possible regression chains differing only by the order in which the datasets are used. A natural option is to weigh all regression chains equally and use the average over all possible chains as the final $\mathbf{D}_k^{\text{shared}}$. We will refer to this as the *full* COCOREG algorithm and abbreviate it as CCR.

Due to the large number of chains the computational load of CCR grows fast with the number of datasets K. To reduce the burden, we can sample only a subset of all possible regression chains. In this sampling-based variant of COCOREG, which we refer to as *sampling* COCOREG and abbreviate as CCRS, one chain is randomly chosen for each starting dataset $\mathbf{D}_i$. In the experiments below we study the chain variability and show that CCR and CCRS produce practically the same result.

3.6 A regression chain example

To demonstrate the concepts explained above, we will next compute the output of cocoreg in the case of three datasets ($K = 3$) with one signal in each dataset ($d_1 = d_2 = d_3 = 1$). All signals are considered to be centered. Using a generative model similar to Equation 4 we get for the OLS regression coefficients:

$$B_{ij} = \frac{\sigma_i}{\sigma_j} r_{ij}, \tag{13}$$

where B_{ij} is the regression coefficient $j \to i$, σ_k is the standard deviation of the k:th dataset and r_{ij} the Pearson correlation coefficient between datasets i and j. For, e.g., dataset 3 there are two possible paths and the average of these would be

$$\begin{aligned} \hat{x}_3 &= \frac{1}{2} \left(B_{32} B_{21} x_1 + B_{31} B_{12} x_2 \right) &(14) \\ &= \frac{1}{2} \left(r_{23} r_{12} \frac{\sigma_3}{\sigma_1} x_1 + r_{13} r_{12} \frac{\sigma_3}{\sigma_2} x_2 \right). &(15) \end{aligned}$$

We see that the result is a weighted average where datasets are both scaled to match in variance and also weighted according to the product of correlation coefficients along the path. Note that this weighting obeys the properties of the above theorems. In the case of Theorem 3.5 if dataset k is independent of the others we have $r_{ik} = 0$ for all i and since at least one r_{ik} is present in every path we get $\hat{x}_3 = 0$ (the mean). For Theorem 3.5 it holds that all $r_{ji} = 1$ which is equivalent to $x_i = \alpha x_j = \frac{\sigma_i}{\sigma_j} x_j$ and therefore $\hat{x}_3 = x_3$.

3.7 The cocoreg algorithm

The cocoreg algorithm that solves the *shared variation extraction* problem (Problem 3.4) using regression chains for a data collection with K datasets is as follows:

1. Center each column of $\mathbf{D}$ (zero mean).
2. Create the $K^2 - K$ mappings $f_{\mathbf{D}_i \mapsto \mathbf{D}_j}$ between the K datasets.
3. Form the regression chains (all datasets): $K(K-1)!$ for ccr and $K(K-1)$ for ccrs.
4. Using the chains, estimate $\mathbf{D}_i^{\text{shared}}$ in each of the K datasets.

The output of cocoreg is in the same space as original data. A single shared variation projection can be created by applying PCA to the cocoreg projections. The first principal component from this analysis can be used as a signal describing the average shared variation in the data collection. To make sure that all datasets contribute equally, the cocoreg projections are scaled to unit variance prior to the PCA computation.

The cocoreg algorithm is available in CRAN as the R package `cocoreg`.

3.8 Computational complexity of the cocoreg algorithm

The computational complexity of the cocoreg algorithm depends on the regression method used. We here denote the time complexity of training the regression function by T_{train} and the time complexity of applying the regression function by T_{apply}.

In both versions of the cocoreg algorithm, $K^2 - K$ regression functions must be trained, which gives a complexity of $O\left(T_{\text{train}}\left(K^2 - K\right)\right)$.

In the full version of cocoreg there are $(K-2)!$ possible chains between the datasets $\mathbf{D}_i$ and $\mathbf{D}_k$, $K-1$ datasets to start from and K datasets to end in. This means that the regression function is applied $K!$ times. The time complexity of ccr is thus

$$O\left(T_{\text{train}}\left(K^2 - K\right) + T_{\text{apply}}K!\right).$$

For ccrs only one chain per starting dataset is used, resulting in the evaluation of $K(K-1)$ regression models. This results in a total complexity of

$$O\left(\left(T_{\text{train}} + T_{\text{apply}}\right)\left(K^2 - K\right)\right).$$

For linear regression, the training time complexity $T_{\text{train}} = O(C^2N)$ where C is the number of regressors and N is the number of observations. The time complexity of applying the linear regression function is $T_{\text{apply}} = O(CN)$. A conservative estimate is obtained when $C = \max_k (d_k) = d_{\max}$. The time complexity for the cocoreg algorithm using linear regression is thus $O(K^2 d_{\max}^2 N + d_{\max} NK!) = O(d_{\max} NK!)$ for ccr and $O(d_{\max}^2 NK^2)$ for ccrs.

One noteworthy observation is that K is typically fairly small, in our applications at most six. Hence even the full version is often computationally feasible despite the $K!$ term, and the sampling version is only slower than the base regression models by a small factor K^2. Applications with very large K would require different modeling tools, but the value of extracting variation shared by *all* datasets may in any case be questionable for such applications.

4 Experiments

4.1 Experimental Setup

We run cocoreg in R (R Core Team, 2014) with three alternative regression functions: Ordinary Least-Squares linear regression (OLS) from the package `stats` (R Core Team, 2014), Random Forests (RF) from the `randomForest` package (Liaw and Wiener, 2002) and Support Vector Machines (SVM) from the `e1071` package (Meyer et al, 2014). The OLS versions of cocoreg are named ccr (full paths) and ccrs (path sampling), the RF version is named ccr-rf and the SVM ccr-svm. All regressors were used at their default settings.

We compare cocoreg against two alternatives representing the tools typically used for finding shared variation. The primary comparison method is gfa as implemented in the R-package `CCAGFA` (Klami et al, 2015; Virtanen et al, 2012), chosen to represent methods that factorize data variation. The output of gfa is a set of latent variables and

Data collection	**N**	K	d_k **[min, max]**
synth-*	100	4	[3, 3]
synth-nl	100	2	[1, 1]
commodity-prices	275	6	[4, 20]
commodity-prices-small	275	3	[2, 2]
WHO-risk	30	3	[3, 3]
ERP	384	4	[6, 6]

Table 1: Data collections and their properties. N is the number of observations, K is the number of datasets in the collection and d_k are the dimensions of the kth dataset.

associated weights to reconstruct the original signal. A projection similar to cocoreg output can be created by reconstructing data using only those latent variables that are active in all datasets.

We additionally compare against the Regularized Generalized Canonical Correlation Analysis (rgcca) (Tenenhaus, 2011), which generalizes canonical correlation to three or more datasets and hence represents a method that finds shared variation using maximally dependent projections. rgcca finds the projections by maximizing the average correlation over the whole data collection. This criterion differs from that used by cocoreg, since the averaging process includes also variation that is shared only by a subset of the datasets. Hence rgcca should not be considered as a direct replacement to cocoreg.

The output of rgcca contains several orthogonal projections per dataset ordered by strength of correlation. We only use the first rgcca projection, i.e., the one associated with the highest average correlation. To be able to compare the outputs of rgcca and cocoreg we an OLS regression from the first rgcca projection back onto the original variables to show what rgcca has picked up as shared.

These methods are compared using seven artificial and three real data collections. The data collections are described in detail below.

4.2 Data collections

The basic properties of the data collections used in the experiments are presented in Table 1. Each data collection is divided into K datasets, and the datasets in turn consist of d_k signals.

Synthetic data The linear synthetic data collections were created by combining sinusoidal waves, a linear trend and independent Gaussian noise. Figure 3 shows the data collection synth-base. For this data collection the high frequency sinusoid is missing from the fourth dataset ($\mathbf{D}_4$) while other components are present in all datasets. The observed data is a sum of “shared-by-all” , “shared-by-some” and independent Gaussian noise.

The data collections synth-uds, synth-uvar and synth-puvar are derived from synth-base. In synth-uds the fourth dataset is replaced by random Gaussian noise, thus rendering it unrelated to the other datasets. In synth-uvar the first variable of the fourth dataset is replaced by random Gaussian noise. In synth-puvar the last

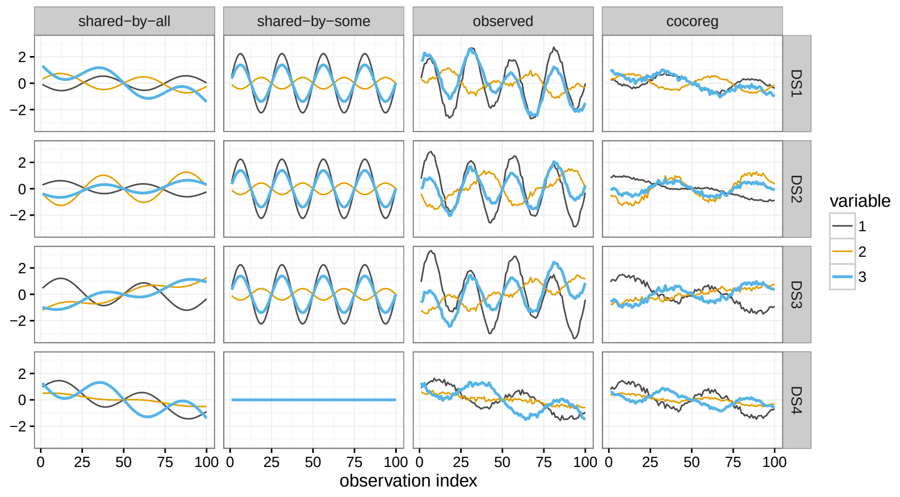


Figure 3: **Structure of the synthetic data** Shown are from left to right: variation shared by all datasets, variation shared by a subset of the datasets, observed data **D**, and the cocoreg projections. Note how the higher frequency sine wave is not common to all of the datasets and that cocoreg projection resembles the "shared-by-all" column as desired.

50 observations of variables 1 and 2 from the fourth dataset are replaced by Gaussian noise.

The non-linear dataset `synth-nl` contains two datasets: a linear trend as the first dataset and piecewise linear transformation of this trend as the second dataset. Hence there is shared variation, but that can only be revealed by a non-linear mapping.

Commodity prices data This data consists of monthly commodity prices[1] as listed by the IMF Primary Commodity Prices web site.[2] The dataset contains 63 different price indices both at the industry level (e.g., "energy") as well as at the product level (e.g., "oil"). We divided the price indices into six groups according to industry: food, food_meat, beverages, energy, metals and other_materials. Each group forms one dataset with price index time series as the signals. Only months without missing data were accepted for analysis yielding $N = 275$ observations.

WHO health risk factors data This data collection consists of health risk factor time series from different countries from the last three decades[3]. Mean trend estimates standardized by age were included for blood cholesterol, glucose and pressure as well as body mass index spanning the years 1980–2009. Additionally, the alcohol con-

[1]`http://www.imf.org/external/np/res/commod/External_Data.xls`
[2]`http://www.imf.org/external/np/res/commod/index.aspx`
[3]`http://apps.who.int/gho/data/node.main.A867?lang=en`

sumption per capita between the years 1990–2012 was included. These variables were averaged within region and only years with complete observations were included in analysis. Each region (Europe, America, Africa) formed one dataset with the risk factor time series as the signals.

ERP data Electroencephalographic (EEG) data from the HeadIT repository[4] was used to investigate the efficacy of cocoreg in event-related potential (ERP) analysis. The data is from an auditory oddball paradigm epoched into segments starting 100 ms prior to the deviant stimuli and ending 1500 ms after the stimuli. Four subjects were included in the analysis. The final data collection consists of electrode wise ERP data from randomly selected six electrodes, 67–76 trials per electrode depending on the subject. Each subject forms one dataset with the electrodes as the signals.

4.3 Basic concepts

It is important for the interpretation of the cocoreg output that all variables in all datasets have been centered prior to analysis forcing the mode of the amplitude distribution to zero. As a consequence, if there is no shared variation between two variables, the output of their (linear) regression is the mode of the dependent variable, i.e., zero. Hence, if the output is zero for many successive samples there is no shared variation, otherwise there is. This is clearly visible above in Figure 3, where the higher frequency sinusoidal component visible in datasets D1-D3 is close to zero in the cocoreg projection. The lower frequency sine and the linear trend are shared between all datasets and hence remain visible. Note also how positive and negative correlation are both valid forms of shared variation and are not suppressed.

Figure 4 illustrates what happens if one dataset is completely independent of the others. In this case no part of the variation is shared between *all* datasets and $\mathbf{D}_S^{\{\text{all}\}}$ is zero for all signals, in all datasets. This is again an example of the situation described earlier in Theorem 3.5: one dataset blocks the flow of information forcing the linear regressor to a constant output.

Figure 5 illustrates that if only a single variable in one of the datasets is unrelated to the other datasets (thin black signal in $\mathbf{D}_4$), then only the shared variation of that variable becomes zero.

Figure 6 shows observed data and shared variation for two datasets from the `commodity-prices` data collection. The general rising trend and the exact location of the 2008 financial crisis are clearer in the shared variation projection. The cocoreg projection of the price of lamb meat is a flat line near zero meaning that lamb price does not share much with the other price indices.

Figure 7 shows the ERP data. Here cocoreg is able to extract weak but consistent shared variation that is almost completely hidden in the original data. The amplitude of the cocoreg projection is very small compared to original data, which reflects the fact that the shared part accounts for only a small fraction of the total variation. The cocoreg projection reveals a consistent response to the auditory stimulus around 200 ms

[4]`http://headit-beta.ucsd.edu/studies/9d557882-a236-11e2-9420-0050563f2612`

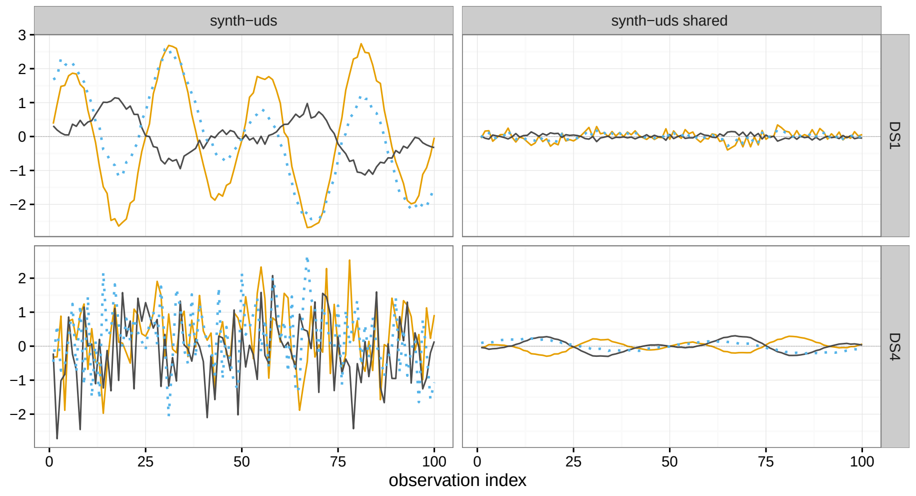


Figure 4: **Unrelated datasets** Original (left) and shared variation (right) for `synth-uds`, datasets $\mathbf{D}_1$ and $\mathbf{D}_4$. Note how shared variation projection (right column) is zero for all signals in both datasets.

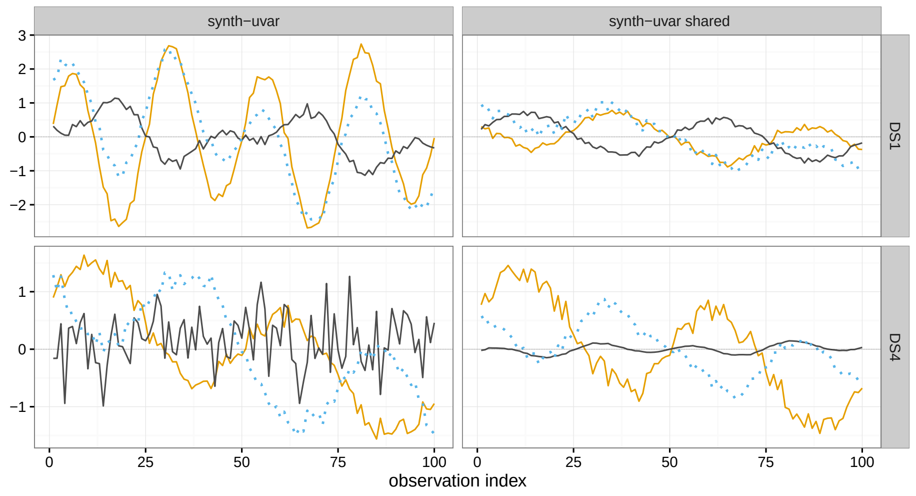


Figure 5: **Related datasets** Original (left) and shared variation (right) for `synth-uvar`, datasets $\mathbf{D}_1$ and $\mathbf{D}_4$. Note how shared variation projection is close to zero for the unrelated signal in $\mathbf{D}_4$ (thin black curve, bottom row). The higher frequency sinusoid in $\mathbf{D}_1$ is attenuated as the corresponding component is not shared by all datasets (top row).

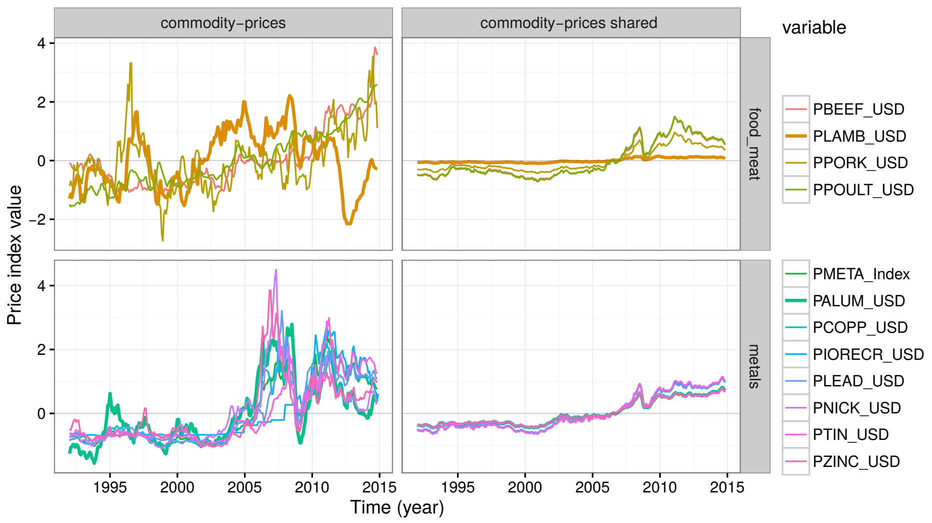
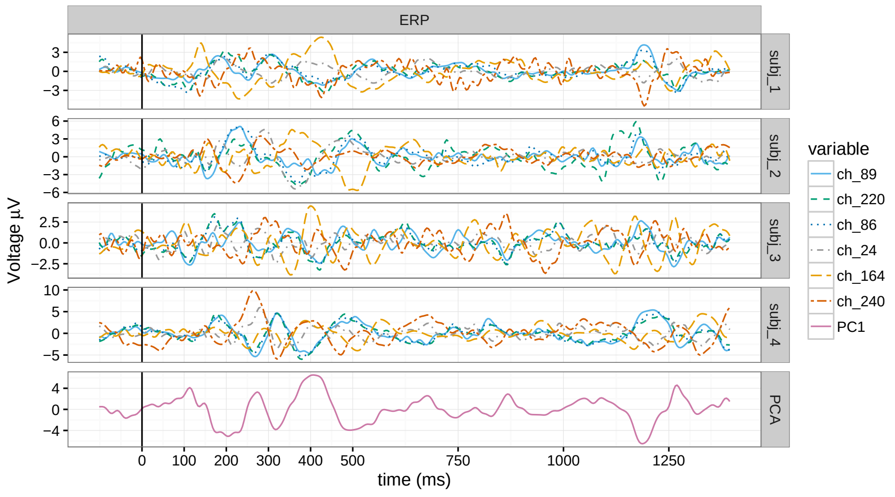


Figure 6: Original (left) and shared variation (right) data from the `commodity-prices` data collection. Only datasets with meat (top) and metal prices (bottom) are shown. The general trend is similar in both datasets. The price of lamb (PLAMB_USD, top row, bold orange) has least in common with the other industries.

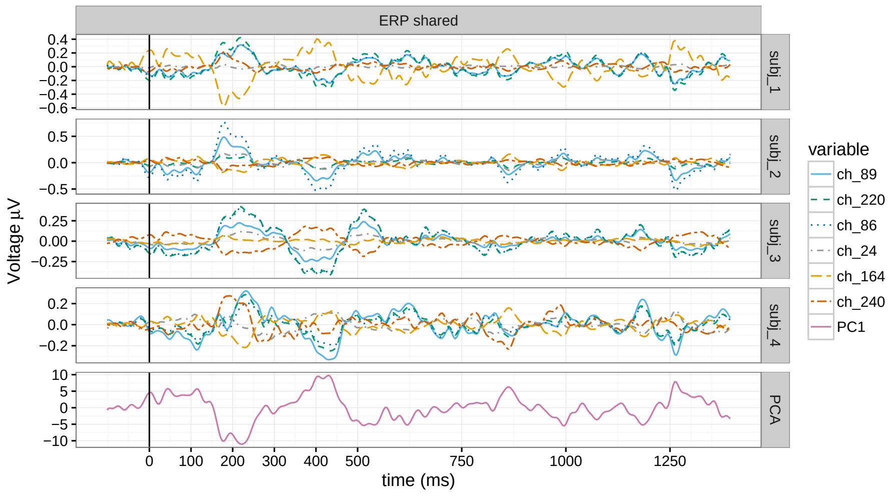
and 400 ms, as identified, e.g., from the first principal component of the cocoReg projections. Shared variation in channels 86, 89 and 220 (shades of blue and green) behaves consistently for all subjects, implying that the response must be due to the stimulus paradigm. The locations on the scalp of these channels are also relatively close together supporting the assumption that the induced activity is also localized. In sum, cocoReg is able to extract weak but consistent activity from a collection of signals by exploiting the fact that all signals are responses to the same stimulus. We found that the results are meaningful for ERP experts and help them to interpret the signals.

4.4 Chain variability

As explained above in Section 3.5, the shared variation is computed using several regression chains. When there are $K \geq 4$ datasets in a data collection there are multiple possible chains starting at dataset $\mathbf{D}_i$ and ending at dataset $\mathbf{D}_k$. We have observed that in many data collections the cocoReg projections are more similar to each other, when the projections originate from the same dataset, i.e., when they share the same starting dataset. For a target dataset $\mathbf{D}_k$, let a *chain group* be the set of all chains originating from the same dataset $\mathbf{D}_i$. Our observations suggest that the variability within chain groups is smaller than the variability between chain groups. This empirical observation is tested by computing the sum-of-squares variability of the chains separately for each dataset, variable and time point. As shown in Table 2 the relative proportion of variability between chain groups is largest for the simplest datasets. For more complex



(a) original



(b) shared

Figure 7: Original (a) and shared variation (b) for ERP data using ccr. Vertical lines mark the stimulus onset. The last panel (row) in each figure contains the first principal component of the data in the other panels.

datasets the proportion of within-chain group variability increases.

These results suggests that if computation of all possible chains is computationally unfeasible due to their large number, a good choice is to sample all chain groups uniformly in order to capture most of the chain variability. This is also the sampling strategy used by cocoreg. To test the validity of the sampling approach, we compared the variances explained (R^2), see Table 2. The results show, that the difference between ccr and ccrs is in all cases negligible, less than 1%.

Table 2: Comparison of ccr and ccrs for datasets with $K > 3$. On the left, the relative proportion of variability between chain groups, where chains from $\mathbf{D}_i \rightarrow \mathbf{D}_k$ form the i:th chain group. Shown are the mean sum-of-squares variabilities for data collections. For simpler datasets the total chain variability consists of mainly between-group variability, whereas for the more complex ones also chains within each chain group vary considerably. On the right, the mean difference in variance explained is shown in per mille units. The confidence interval (mean ± se) is computed over all variables of a data collection. The difference in R^2 -values is negligible for all practical purposes (largest difference in R^2 below 1%).

Data collection	Proportion of variability between chain groups in %	mean($R^2_{\text{CCR}} - R^2_{\text{CCRS}}$) in ‰
`synth-base`	96	-0.64 [-1.34, 0.06]
`synth-uds`	69	0.17 [-0.22, 0.56]
`synth-uvar`	93	0.23 [-0.27, 0.73]
`synth-puvar`	94	-0.21 [-0.89, 0.47]
`commodity-prices`	58	-1.62 [-2.87, -0.37]
`ERP`	54	-0.66 [-1.90, 0.59]

4.5 Comparison to similar methods

In this experiment the estimated shared variation $\hat{\mathbf{D}}_S^{\{all\}}$computed using the cocoreg algorithm is compared to the respective projection of gfa and rgcca. We use synthetic data, for which the true shared variation $\mathbf{D}_S^{\{all\}}$is known. For gfa we reconstruct the data using only those latent components that are active in all datasets (function `CCAGFA::GFAtrim`) and use this as $\hat{\mathbf{D}}_S^{\{all\}}$. For rgcca we use the first projection of each dataset, projected back to the original data space using OLS regression.

Figure 8 shows the mean rmse error between $\hat{\mathbf{D}}_S^{\{all\}}$and $\mathbf{D}_S^{\{all\}}$for several datasets and methods. For the linear data collections (four leftmost columns) ccr and gfa show lowest reconstruction error. gfa has an edge when datasets share information but it fails when one of the datasets is independent of the others. This has to do with the selection of the active components. If a small contribution is considered enough to make the component active, many components will be selected and the absence of shared variation may go unnoticed. cocoreg lacks this selection step; variation that survives the regression chain is shared by all – no further decisions needed. The non-linear regressors perform poorly as they are incompatible with the assumption of linear rela-

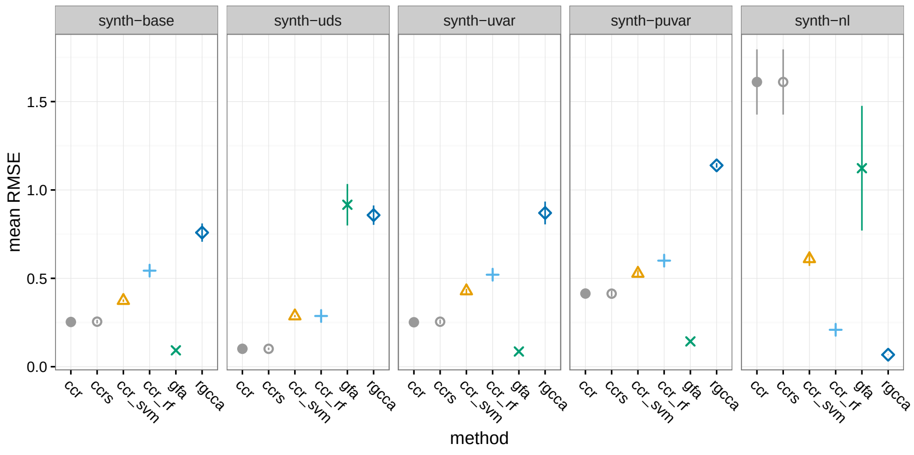


Figure 8: Mean RMSE error between true and estimated shared variation for different synthetic data collections (columns) and methods. The reported mean RMSE is the average over signal RMSE values within each data collection. The vertical lines indicate the standard error of mean over five repetitions.

tionships between datasets. Large error for RGCCA underlines the fact that this method is not suitable for finding the smallest common denominator of the datasets.

For the non-linear data collection `synth-nl` the non-linear COCOREG methods CCR-SVM and CCR-RF outperform GFA and the linear versions of COCOREG. RGCCA has the lowest error which, however, is just an artifact: since there is only one signal in each dataset the output of RGCCA is almost the same as input, which in this case that happens to be the correct solution. The main purpose of `synth-nl` is to show that GFA cannot model non-linear relationships between datasets, not even in a simple case.

To sum up, we were able to demonstrate that CCRS performs comparably to GFA for most of our synthetic datasets, it outperforms GFA for non-linear data, and it outperforms the RGCCA method for all data collections. This happens because RGCCA models also variation that is specific for a subset of the datasets (not shared by all).

4.6 Computation times

Typical running times for the different algorithms and datasets are shown in Table 3. The experiments were run using unoptimized R-code on a quad-core 2.6 GHz Intel i5 CPU with 8 Gb RAM.

The important observation is that CCRS, the fastest COCOREG variant, is computationally very efficient, taking less than half a second for all collections. The full CCR variant computing all the chains is also still very fast despite the $O(K!)$ complexity since K is small, and even the non-linear variants can be computed efficiently for most data sets, often faster than the linear comparison method of GFA. The running times of CCR-SVM and CCR-RF depend on the parameters and implementation of the SVM and RF models. RGCCA is included in the table for completeness although it does not extract shared variation in the same sense as COCOREG.

Table 3: Wall-clock running times in seconds. It can be seen that COCOREG is fast and with linear regressors faster than GFA. The use of non-linear regressors naturally increases the computation time which still remains acceptable.

	method					
dataset	CCR	CCRS	CCR-SVM	CCR-RF	GFA	RGCCA
`synth-base`	0.1	0.1	0.7	3.7	10.1	0.2
`synth-uds`	0.1	0.1	0.6	3.3	2.9	0.1
`synth-uvar`	0.2	0.1	0.5	3.4	20.5	0.2
`synth-puvar`	0.1	0.1	0.6	2.9	54.4	0.1
`synth-nl`	0.0	0.0	0.0	0.2	84.7	0.0
`commodity-prices-small`	0.1	0.1	0.3	3.2	30.8	0.1
`WHO-risk`	0.0	0.0	0.2	0.4	335.2	0.0
`commodity-prices`	7.1	0.4	187.0	523.5	62.9	0.4
`ERP`	0.3	0.3	7.5	43.0	20.4	0.3

5 Discussion

In this work we present the COCOREG algorithm for extracting shared variation between datasets. The COCOREG algorithm can be viewed as an extension of RDA to the case with more than two datasets. The number of datasets is unlimited and they can be multivariate. The only assumption of the COCOREG algorithm is that it is meaningful to compare the *n*th observation across all datasets. If the signals in the datasets are time series this follows naturally if all signals are sampled at the same time scale. The COCOREG algorithm finds the shared variation by applying independently learned regression models in a chain-like fashion over all datasets. Since the chains always include all datasets only the variation shared by all datasets is transferred through. The extracted shared variation is conveniently in the same data space as the original data which makes the interpretation straightforward.

A strength of the COCOREG algorithm is that it works with all types of regression. The regression function can be freely chosen so that it is suited for the structure of the data being investigated. In this paper we primarily focus on the use of ordinary least squares linear regressors since they are computationally efficient and well-studied. It is also possible to use regression functions such as, e.g., random forests and support vector machines that can capture nonlinear relationships in the data. Even though the main effects in the real data sets studied here were linear, we demonstrated on artificial data that random forest regressors outperform linear ones when the data has clear nonlinearities.

The main principle in the COCOREG algorithm is the use of regression chains. The algorithm can either use all possible regression chains or a sampled subset of them.

The experiments in this paper demonstrate that sampling of the chains reduces computational load without significantly changing the result. In general, the computational complexity of the cocoreg algorithm depends on the regression functions used.

While methods for computing the shared variation between two datasets are abundant, there are clearly fewer methods that generalize to problems with multiple datasets. One such method is group factor analysis (gfa) (Klami et al, 2015), which we used as the primary comparison method. It produces a detailed generative model for the data, but is slow to compute and assumes the datasets to be multivariate Gaussians. Accordingly gfa is best suited for situations where the data is roughly Gaussian, remains fixed and the computation time is not an issue. The advantages of cocoreg are the opposite: it is most useful in exploratory or online settings where computation time is crucial. cocoreg is also easier to adapt to the specific needs of the data, such as nonlinearities or frequent outliers, by simply changing the type of regressor used.

Using synthetic data we showed that cocoreg is able to find shared variation as accurately as gfa, even though cocoreg does not make as strong assumptions on the data. The experiments also show that a simple nonlinearity in the data is enough to tip the scale in favor of cocoreg (non-linear version).

cocoreg is designed to find only variation common to all datasets, which is desirable in many applications. This helps to reduce overfitting as well, because we do not try to model variance shared by arbitrary subsets of datasets. Regularized regressors can also be used to further reduce overfitting.

The fact that cocoreg outputs the shared variation in the original data space can be an advantage or a limitation, depending on the application. If a compressed view of shared variation is needed, PCA can be applied as a post-processing step to the cocoreg output.

One should bear in mind that the choice of regressor affects the cocoreg output like the choice of model affects any modeling outcome. Selecting a good regressor to use with cocoreg essentially boils down to the well-known bias–variance dilemma of modeling (see, e.g., ch. 7.3 in Hastie et al (2003)). A good regressor should have both low bias (i.e., fit well to training data) and low variance (i.e., generalize well to unseen data). To give a simple example, consider a case with two univariate datasets where $\mathbf{D}_1$ would be uniformly distributed and $\mathbf{D}_2 = \mathbf{D}_1^3 + \epsilon$, where ϵ represents a small random measurement error. On the one hand, when modeling $\mathbf{D}_2$ using $\mathbf{D}_1$ a linear regressor would not fit well as it cannot model the nonlinear relationship between the variables, therefore leading to underfitting and an incomplete reconstruction of the shared variation. On the other hand, a polynomial of high degree would model the noise as well which would cause overfitting. Naturally here the smallest reconstruction error would be provided by a properly fitted third order polynomial.

In general the regressor should be flexible enough to learn the essential features of the data but also simple enough to avoid overfitting. The problem of bias is mostly a matter of domain knowledge, i.e., one should use regressors that are able to model the structures of interest. The problem of overfitting can be addressed using standard procedures such as studying generalization error with a separate validation dataset, cross-validation procedures or by using robust models that penalize for model complexity (chapters 7.2, 7.10, and 3.4 in Hastie et al (2003)). For example simple linear regression is known to be sensitive to outliers and therefore robust regression methods such as the least absolute shrinkage and selection operator (LASSO) regression (Tibshirani (1996)) should be used if outliers are present. A conservative choice is to favor simpler models over more complex ones to avoid making false discoveries. The upside is that regression models are well studied thus increasing the chances of finding an appropriate model to a wide variety of problems.

It should also be noticed that the cocoreg algorithm assumes the relation between datasets to remain fixed through time. To give an example, if two datasets start by having a perfect positive correlation but this relation suddenly turns into a perfect negative one, the output of cocoreg might indicate no shared variation at all. This is unintuitive as the correlation has been perfect the whole time. Problems of this kind can be remedied by applying cocoreg to shorter time windows for which the assumption of a fixed relation approximately holds.

Summarizing, we have presented a novel algorithm for extracting shared variation between multiple datasets. Our approach is generic and allows one to use different kinds of regressors and the method can thus be tuned to fit the properties of the data. When used with linear regressors the method scales better than alternative approaches. The cocoreg algorithm is available as the R package `cocoreg` in CRAN.

Possible future work includes extending cocoreg to clustering, such that datasets or signals with a similar shared variation are placed in clusters. The solution would require multiple runs of cocoreg which would still be fast at least using linear regressors. Another avenue for further development is to make the algorithm capable of extracting shared variation even if the datasets are shifted with respect to each other in time.

6 Acknowledgements

This work was partly supported by the Revolution of Knowledge Work project, funded by Tekes (grants 40228/13 and 5159/31/2014), and in part by Academy of Finland (Finnish Centre of Excellence in Computational Research COIN, 251170; 26696).

References

Andrew G, Arora R, Bilmes J, Livescu K (2013) Deep canonical correlation analysis. In: Proceedings of the 30th International Conference on Machine Learning, vol 28, pp 1247–1255

Dähne S, Nikulin VV, Ramírez D, Schreier PJ, Müller KR, Haufe S (2014) Finding brain oscillations with power dependencies in neuroimaging data. NeuroImage 96:334–348

Damianou A, Ek C, Titsias MK, Lawrence ND (2012) Manifold relevance determination. In: Proceedings of the 29th International Conference on Machine Learning, pp 145–152

Fisher J, Darrell T (2003) Speaker association with signal-level audiovisual fusion. IEEE Transactions on Multimedia 6(3):406–413

Hardoon D, Szedmak S, Shawe-Taylor J (2004) Canonical correlation analysis: An overview with application to learning methods. Neural Computation 16(12):2639–2664, DOI 10.1162/0899766042321814

Hasson U, Nir Y, Levy I, Fuhrmann G, Malach R (2004) Intersubject synchronization of cortical activity during natural vision. Science 303(5664):1634–1640

Hastie T, Tibshirani R, Friedman J (2003) The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer

Hotelling H (1936) Relations between two sets of variates. Biometrika 28:321–377

Hsieh WW (2000) Nonlinear canonical correlation analysis by neural networks. Neural Networks 13:1095–1105

Hwang H, Jung K, Takane Y, Woodward TS (2013) A unified approach to multiple-set canonical correlation analysis and principal components analysis. The British journal of mathematical and statistical psychology 66(2):308–21, DOI 10.1111/j.2044-8317.2012.02052.x

Kettenring J (1971) Canonical analysis of several sets of variables. Biometrika 58:433–451

Klami A, Virtanen S, Kaski S (2013) Bayesian Canonical Correlation Analysis. Journal of Machine Learning Research 14:965–1003

Klami A, Virtanen S, Leppäho E (2015) Group Factor Analysis. IEEE Transactions on Neural Networks and Learning Systems 26(9):2136–2147, DOI 10.1109/TNNLS.2014.2376974

Legendre P, Legendre L (1998) Numerical Ecology, 2nd edn. Elsevier

Liaw A, Wiener M (2002) Classification and regression by randomforest. R News 2(3):18–22, URL `https://cran.r-project.org/package=randomForest`

Meyer D, Dimitriadou E, Hornik K, Weingessel A, Leisch F (2014) e1071: Misc Functions of the Department of Statistics (e1071), Technische Universität Wien. URL `http://cran.r-project.org/package=e1071`

Müller KE (1982) Understanding Canonical Correlation through the General Linear Model and Principal Components. The American Statistician 36(4):342–354, DOI 10.1080/00031305.1982.10483045

Nguyen HV, Müller E, Vreeken J, Efros P, Böhm K (2014) Multivariate maximal correlation analysis. In: Proceedings of the 31st International Conference on Machine Learning, pp 775–783

R Core Team (2014) R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, URL `http://www.R-project.org/`

Tenenhaus A (2011) Regularized Generalized Canonical Correlation Analysis and PLS Path Modeling. Psychometrika 76(2):257–284

Tibshirani R (1996) Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society Series B (Methodological) 58(1):267–288

Timmerman ME, Kiers HaL (2003) Four simultaneous component models for the analysis of multivariate time series from more than one subject to model intraindividual and interindividual differences. Psychometrika 68(1):105–121, DOI 10.1007/BF02296656

Virtanen S, Klami A, Khan SA, Kaski S (2012) CCAGFA: Bayesian Canonical Correlation Analysis and Group Factor Analysis. URL `http://cran.r-project.org/package=CCAGFA`