


# Kernel conditional clustering and kernel conditional semi-supervised learning

**Journal Article****Author(s):**

He, Xiao; Gumbsch, Thomas; [Roqueiro, Damian Sabas](#) ; Borgwardt, Karsten

**Publication date:**

2020-03

**Permanent link:**

<https://doi.org/10.3929/ethz-b-000390153>

**Rights / license:**

[Creative Commons Attribution 4.0 International](#)

**Originally published in:**

Knowledge and information systems 62(3), <https://doi.org/10.1007/s10115-019-01334-5>



# Kernel conditional clustering and kernel conditional semi-supervised learning

Xiao He<sup>1,2</sup> · Thomas Gumbsch<sup>1,2</sup> · Damian Roqueiro<sup>1,2</sup> · Karsten Borgwardt<sup>1,2</sup>

Received: 21 January 2018 / Revised: 8 January 2019 / Accepted: 19 January 2019 / Published online: 6 June 2019  
© The Author(s) 2019

## Abstract

The results of clustering are often affected by covariates that are independent of the clusters one would like to discover. Traditionally, *alternative clustering* algorithms can be used to solve such clustering problems. However, these suffer from at least one of the following problems: (1) Continuous covariates or nonlinearly separable clusters cannot be handled; (2) assumptions are made about the distribution of the data; (3) one or more hyper-parameters need to be set. The presence of covariates also has an effect in a different type of problem such as semi-supervised learning. To the best of our knowledge, there is no existing method addressing the semi-supervised learning setting in the presence of covariates. Here we propose two novel algorithms, named kernel conditional clustering (KCC) and kernel conditional semi-supervised learning (KCSSL), whose objectives are derived from a kernel-based conditional dependence measure. KCC is parameter-light and makes no assumptions about the cluster structure, the covariates, or the distribution of the data, while KCSSL is fully parameter-free. On both simulated and real-world datasets, the proposed KCC and KCSSL algorithms perform better than state-of-the-art methods. The former detects the ground truth cluster structures more accurately, and the latter makes more accurate predictions.

**Keywords** Conditional clustering · Conditional semi-supervised learning · Conditional dependence measure · Alternative clustering · Label propagation

## 1 Introduction

In many applications, labeling samples by domain experts is extremely expensive, e.g., diagnoses in the biomedical domain. In practice, one may often encounter datasets with only unlabeled samples or perhaps a few labeled ones. On the one hand, clustering techniques are used to uncover the hidden structure in the data by learning the relationship between unlabeled samples. On the other hand, semi-supervised learning methods focus on improving the classification accuracy by learning a relationship between labeled and unlabeled samples

---

✉ Xiao He  
xiao.he@bsse.ethz.ch

<sup>1</sup> Department of Biosystems Science and Engineering, ETH Zurich, Basel, Switzerland

<sup>2</sup> Swiss Institute of Bioinformatics, Basel, Switzerland

[1]. However, a fundamental problem that is intrinsic to both, clustering and semi-supervised learning, is that the relationships one expects to uncover are often driven by the presence and by the effect of covariates associated with the data. The structure these covariates impose on the data is often trivial to find and irrelevant to the interesting structure or relationship one hopes to discover. As an example, if we consider the clustering of text documents, one may discover source-related clusters rather than content-specific clusters. This phenomenon is particularly prominent in high-dimensional settings (e.g., analysis of text documents, images, and genomic data) where covariate-related features are often collected. Classic clustering or semi-supervised learning algorithms do not take into account the above-mentioned covariates, even when it is clear that their presence masks interesting underlying relationships between samples. Our goal is then to correct for covariates in two distinct problem settings: clustering and semi-supervised learning. To that effect, we focus on the problems of *conditional clustering* and *conditional semi-supervised learning*, respectively. Their aim is to maximize the dependence between the data and the clustering/label assignment, conditioned on known covariates.

To the best of our knowledge, there is no literature about the conditional semi-supervised learning problem.

In contrast, the clustering domain has an extensive body of work addressing the conditional clustering problem. The first milestone toward solving this problem is the work by Gondek and Hofmann [2]. Their method is based on maximizing the conditional mutual information. However, it relies on assumptions on the distribution of the data to estimate the mutual information. The research community then focused on *alternative clustering* [3–7]. The aim of alternative clustering is to generate multiple, dissimilar clusterings of a dataset. Most alternative clustering algorithms produce clusterings sequentially, i.e., given a set of clusterings, find a new, dissimilar clustering. When treating known clusterings as categorical covariates, sequential alternative clustering becomes an instance of conditional clustering. Alternative clustering algorithms differ in the way they compute the dissimilarity between the alternative and known clusterings. The work by Bae and Bailey [3] is based on hierarchical clustering with *must-link* and *cannot-link* constraints constructed from a known clustering. Subsequent research performed the clustering on transformed data through an orthogonal projection [4] or an inverse distance metric [5], both with respect to the given clustering. These early methods can only handle categorical covariates because a dependence measure to assess the dissimilarity between the alternative and known clusterings has not been used. Recently, several alternative clustering algorithms that use a kernel-based dependence measure, the Hilbert–Schmidt Independence Criterion (HSIC) [8], have been proposed [9–13]. HSIC enables these methods to also handle continuous covariates. However, the objective functions of these methods are expressed in a regularization framework, where one term is for the clustering quality and the other is for the dissimilarity between clusterings. A hyperparameter is needed to enforce a trade-off between these two terms, something which is difficult to estimate in practice.

In this paper, we propose kernel conditional clustering (KCC) and kernel conditional semi-supervised learning (KCSSL), two de novo algorithms that use an extension of HSIC, known as the Hilbert–Schmidt Conditional Independence Criterion (HSCONIC) [14,15], to solve the conditional clustering problem and the conditional semi-supervised learning problem, respectively. While KCC maximizes the dependence in the reproducing kernel space between the clustering assignment and a subspace projection of the data, KCSSL maximizes the dependence in the reproducing kernel space between original data and the target labels (predicted and known). More importantly, both algorithms perform their respective tasks

while conditioned on observed covariates. The major contributions of KCC and KCSSL are threefold.

1. *Statistically sound* The objective functions of KCC and KCSSL are based on the kernel conditional dependence measure HSCONIC, a nonparametric statistical test. Therefore, the objective functions build on a well-established statistical foundation with a clear interpretation.
2. *Parameter-light* KCC does not require the setting of a hyper-parameter to control the regularization; it only needs the number of clusters as input. This is achieved while making no assumptions on the data distribution, the covariates, or the cluster structure. KCSSL is fully parameter-free without any input hyper-parameter.
3. *Subspace learning* KCC integrates subspace learning into conditional dependence maximization for clustering high-dimensional datasets.

Parts of this paper (kernel conditional clustering) have appeared previously in [16]. In this paper, we provide: (1) an extension of HSCONIC to the problem of conditional semi-supervised learning with a new objective function and optimization algorithm; (2) a theoretical and empirical comparison of KCSSL to state-of-the-art semi-supervised learning methods; (3) a more detailed survey of state-of-the-art methods in both conditional clustering and semi-supervised learning.

The rest of this paper is organized as follows: In Sect. 2, we introduce the kernel dependence measure HSIC and its extension HSCONIC. In Sects. 3 and 4, we present the proposed methods: kernel conditional clustering (KCC) and kernel conditional semi-supervised learning (KCSSL), respectively. Experiments on synthetic and real-world datasets that show the effectiveness of KCC and KCSSL are presented in Sect. 5. Section 6 discusses related work, and Sect. 7 concludes the paper.

## 2 Kernel conditional dependence

We start this section by introducing the kernel dependence measure, Hilbert–Schmidt Independence Criterion (HSIC), followed by its extension, Hilbert–Schmidt Conditional Independence Criterion (HSCONIC). As mentioned earlier, HSCONIC is at the heart of the two methods presented in this paper that address the two disjoint problems of conditional clustering and of conditional semi-supervised learning.

### 2.1 The kernel dependence measure HSIC

HSIC measures the dependence between two variables by mapping them to the reproducing kernel Hilbert space (RKHS). This means that correlations measured in the RKHSs reflect higher-order joint moments between the data distributions in the original space [17].

Given a random variable  $(x, y)$  on  $\mathcal{X} \times \mathcal{Y}$ , and RKHSs  $\mathcal{H}_{\mathcal{X}}$  and  $\mathcal{H}_{\mathcal{Y}}$  on  $\mathcal{X}$  and  $\mathcal{Y}$ , respectively, we define feature representations  $\phi_{\mathcal{X}}(x)$  and  $\phi_{\mathcal{Y}}(y)$  that map  $x$  and  $y$  to the RKHSs  $\mathcal{H}_{\mathcal{X}}$  and  $\mathcal{H}_{\mathcal{Y}}$ , such that the inner product in the RKHS is given by the kernel functions  $k_{\mathcal{X}}$  and  $k_{\mathcal{Y}}$ , e.g.,  $k_{\mathcal{X}}(x, x') = \langle \phi_{\mathcal{X}}(x), \phi_{\mathcal{X}}(x') \rangle$ .

The linear cross-covariance operator [8, 18]  $C_{xy}$  in RKHS is defined as

$$C_{xy} = E_{xy}[(\phi_{\mathcal{X}}(x) - \mu_x) \otimes (\phi_{\mathcal{Y}}(y) - \mu_y)], \quad (1)$$

where  $\mu_x = E_x(\phi_{\mathcal{X}}(x))$ ,  $\mu_y = E_y(\phi_{\mathcal{Y}}(y))$ , and  $\otimes$  is the tensor product. The cross-covariance operator naturally extends the covariance matrix of the Euclidean space to reflect

higher-order correlations between  $x$  and  $y$ . The HSIC measure of dependence between  $x$  and  $y$  is defined as the Hilbert–Schmidt norm (the generalization of the Frobenius norm on matrices [8]) of the cross-covariance operator:

$$\text{HSIC}(x, y) = \|C_{xy}\|_{\text{HS}}^2. \tag{2}$$

Given  $n$  observations  $\{(x_1, y_1), \dots, (x_n, y_n)\}$ , HSIC can be empirically estimated by

$$\text{HSIC}_{\text{emp}}(x, y) = \frac{1}{(n - 1)^2} \text{Tr}(HK_xHK_y), \tag{3}$$

where  $K_x$  and  $K_y$  are the Gram matrices defined on the kernel functions  $k_{\mathcal{X}}$  and  $k_{\mathcal{Y}}$ , which are centered by  $H = I_n - \frac{1}{n}11^T$ .

Theorem 4 in [8] states that  $\text{HSIC}(x, y) = 0$  if and only if  $x$  and  $y$  are independent. Therefore, a large value of  $\text{HSIC}_{\text{emp}}(x, y)$  indicates a strong dependence between  $x$  and  $y$ . In [9–12],  $\text{HSIC}_{\text{emp}}$  is applied to measure the dependence between the clustering and the known covariates.

### 2.2 The kernel conditional dependence measure HSCONIC

HSCONIC measures the dependence between two variables conditioned on observing the third variable in RKHSs through kernel functions.

Suppose given random variables  $(x, y, z)$  on  $\mathcal{X} \times \mathcal{Y} \times \mathcal{Z}$  with RKHSs  $\mathcal{H}_{\mathcal{X}}, \mathcal{H}_{\mathcal{Y}}, \mathcal{H}_{\mathcal{Z}}$  on  $\mathcal{X}, \mathcal{Y}, \mathcal{Z}$ . The corresponding feature mappings and kernel functions are  $\phi_{\mathcal{X}}, \phi_{\mathcal{Y}}, \phi_{\mathcal{Z}}$  and  $k_{\mathcal{X}}, k_{\mathcal{Y}}, k_{\mathcal{Z}}$ , respectively.

The conditional cross-covariance operator  $C_{xy|z}$  [15] in RKHSs is defined as

$$C_{xy|z} = C_{xy} - C_{xz}C_{zz}^{-1}C_{zy}, \tag{4}$$

where  $C_{**}$  is the cross-covariance operator as defined in (1).  $C_{xy|z}$  can be interpreted as a nonlinear extension of the conditional covariance matrix (section 4.3 in [19]) of Gaussian random variables in RKHSs.

The HSCONIC measure of conditional dependence is defined as the Hilbert–Schmidt norm of the conditional cross-covariance operator:

$$\text{HSCONIC}(x, y|z) = \|C_{\hat{x}\hat{y}|z}\|_{\text{HS}}^2, \tag{5}$$

where  $\hat{x} = (x, z)$  and  $\hat{y} = (y, z)$  are extended variables.

As shown in [15], given  $n$  observations  $\{(x_1, y_1, z_1), \dots, (x_n, y_n, z_n)\}$ , HSCONIC can be empirically estimated by

$$\begin{aligned} \text{HSCONIC}_{\text{emp}}(x, y|z) &= \frac{1}{(n - 1)^2} \text{Tr}(\bar{K}_x\bar{K}_y - 2\bar{K}_xM\bar{K}_y + \bar{K}_xM\bar{K}_yM) \\ M &= \bar{K}_z(\bar{K}_z + \epsilon I_n)^{-2}\bar{K}_z, \end{aligned} \tag{6}$$

where  $\bar{K}_x = HK_xH, \bar{K}_y = HK_yH, \bar{K}_z = HK_zH$ , and  $H$  is the centering matrix described in (3).  $K_x, K_y$ , and  $K_z$  are Gram matrices defined on the kernel functions  $k_{\mathcal{X}}, k_{\mathcal{Y}}$ , and  $k_{\mathcal{Z}}$ , respectively.  $\epsilon$  is a small value to prevent inverting a singular matrix. In this paper we set  $\epsilon$  to  $10^{-8}$ .

Theorem 1 in [15] states that  $\text{HSCONIC}(x, y|z) = 0$  if and only if  $x$  and  $y$  are independent, conditioned on  $z$ . Therefore, a large value of  $\text{HSCONIC}_{\text{emp}}$  indicates a strong dependence between  $x$  and  $y$ , conditioned on  $z$ .

Our contribution is to use  $\text{HSCONIC}_{\text{emp}}$  in two different problems: conditional clustering and conditional semi-supervised learning. The following section focuses on conditional clustering and presents the method KCC in detail, as it was originally proposed in [16]. Section 4 then shifts the focus toward the problem of conditional semi-supervised learning and introduces the method KCSSL.

### 3 Kernel conditional clustering

In conditional clustering, we have a dataset of unlabeled data points and covariates for each of them. Our goal is to cluster the data points, conditioned on the known covariates, while maximizing the dependence between the clustering assignment and the data.

The algorithm kernel conditional clustering (KCC) [16] aims to discover these clustering assignments, conditioned on the covariate information. We first introduce a new objective function for KCC, followed by the details of its implementation regarding the optimization and the initialization of the algorithm. Finally, we contrast KCC with alternative clustering algorithms, in particular in terms of their computational complexity.

#### 3.1 The conditional clustering model

Consider a dataset of  $n$  (unlabeled) data points  $D = \{d_1, \dots, d_n\} \in \mathbb{R}^{n \times p}$  with  $p$  features each and a set of  $c$  known covariates  $C \in \mathbb{R}^{n \times c}$ . The aim of conditional clustering is to find a partition  $U$  on  $D$  such that the dependence between  $U$  and  $D$  is maximized, conditioned on observing  $C$ . We represent  $U \in \mathbb{R}^{n \times k}$  as a clustering indicator matrix, i.e.,  $U(i, j) \in [0, 1]$  and  $\sum_j U(i, j) = 1$ . In other words,  $U$  assigns every data point  $d_i$  to one of the  $k$  clusters. In this paper we assume the number  $k$  of clusters is known beforehand.

Using HSCONIC as the conditional dependence measure and its empirical estimate defined in (6), we can write the conditional clustering optimization objective as

$$\begin{aligned} \max_U \quad & \text{HSCONIC}_{\text{emp}}(D, U|C) \\ \text{s.t.} \quad & U(i, j) \in [0, 1] \quad \text{and} \quad \sum_j U(i, j) = 1. \end{aligned} \quad (7)$$

The binary constraint on the clustering assignment  $U$  results in the optimization being a combinatorial problem of high complexity. Therefore, we use the idea of *Spectral Clustering* [20], where only an orthogonality constraint is imposed on  $U$ . In the second step,  $k$ -means will be performed on the embedding  $U$  to obtain the final clustering labels. The relaxed objective is

$$\begin{aligned} \max_U \quad & \text{HSCONIC}_{\text{emp}}(D, U|C) \\ \text{s.t.} \quad & U^T U = I. \end{aligned} \quad (8)$$

As mentioned in Introduction, the clusterings that are interesting in high-dimensional data are those whose creation is not driven by covariates. However, many features are correlated with the covariates, and therefore, the Gram matrix of data  $D$  using all features will still be affected by the covariates. Niu et al. [11] proposed KDAC to learn a low-dimensional subspace that is less correlated with known covariates. The kernel matrix is then computed in the reduced subspace. Inspired by this approach, we integrate subspace learning into conditional clustering:

$$\begin{aligned} \max_{U,W} \quad & \text{HSCONIC}_{\text{emp}}(DW, U|C) \\ \text{s.t.} \quad & U^T U = I, \quad W^T W = I \end{aligned} \tag{9}$$

In (9), a subspace projection  $W$  and a clustering embedding  $U$  are learned simultaneously, while their dependence is maximized, conditioned on the covariates  $C$ . The dimensionality of the subspace needs to be high enough to capture the cluster structure but low enough to not be affected by noise. We therefore follow the well-established strategy of fixing  $W \in \mathbb{R}^{p \times k}$  with  $k$  dimensions.

Plugging (6) into (9) (and dropping the constant factor) we obtain

$$\begin{aligned} \max_{U,W} \quad & \text{Tr}(\bar{K}_{DW} \bar{K}_U - 2\bar{K}_{DW} M \bar{K}_U + \bar{K}_{DW} M \bar{K}_U M) \\ & M = \bar{K}_C (\bar{K}_C + \epsilon I_n)^{-2} \bar{K}_C \\ \text{s.t.} \quad & U^T U = I, \quad W^T W = I, \end{aligned} \tag{10}$$

where  $K_{DW}(i, j) = K(W^T d_i, W^T d_j)$  and  $K$  is a kernel function.  $\bar{K}_{DW} = H K_{DW} H$  is the centered kernel matrix as in (3).  $K_C$  and  $K_U$  are defined analogously.

To capture nonlinearity, we choose the Gaussian radial basis function kernel (RBF) for  $K_{DW}$  and for  $K_C$ ,

$$K(W^T d_i, W^T d_j) = \exp\left(-\frac{\|W^T d_i - W^T d_j\|^2}{2\sigma^2}\right) \tag{11}$$

To optimize the cluster indicator matrix  $U$ , we follow the procedure in KDAC [11] and use a linear kernel  $K_U = U U^T$ .

### 3.2 Optimization

The objective function in (10) contains two variables: the projection matrix  $W \in \mathbb{R}^{p \times k}$  and the relaxed clustering indicator matrix  $U \in \mathbb{R}^{n \times k}$ .  $M$  is a constant that can be calculated from the known covariates  $C$ . We alternate between optimizing  $U$  and  $W$  to find a local optimum.

#### 3.2.1 Optimizing $U$ with $W$ fixed

We can calculate the kernel matrix  $K_{DW}$  in the projected subspace, which turns the problem into a convex and continuous optimization:

$$\begin{aligned} \max_U \quad & \text{Tr}(U^T L_1 U) \\ & L_1 = (\bar{K}_{DW} - 2\bar{K}_{DW} M + M \bar{K}_{DW} M) \\ \text{s.t.} \quad & U^T U = I \end{aligned} \tag{12}$$

If  $U$  is equal to the eigenvectors corresponding to the largest  $k$  eigenvalues of  $L_1$ , where  $k$  is the number of clusters, the Ky Fan theorem [21] states that this is the global optimum of Eq. (12).

### 3.2.2 Optimizing $W$ with $U$ fixed

We can calculate the linear kernel matrix  $K_U$ . The problem now becomes a non-convex continuous optimization problem, which can be written as

$$\begin{aligned}
 & \max_W \text{Tr}(L_2 K_{DW}) \\
 & L_2 = (\bar{K}_U - 2\bar{K}_U M + M\bar{K}_U M) \\
 & K_{DW}(i, j) = K(W^T d_i, W^T d_j) \\
 & \text{s.t. } W^T W = I
 \end{aligned} \tag{13}$$

This is non-convex because of the orthogonality constraint of the projection  $W$ . Although it has been shown that gradient-based methods may work in some cases [9,22,23], the implementation is still far from robust.

Recently, Wen and Yin have proposed a feasible method for optimization with orthogonality constraints (FOptM) [24]. They construct a homotopy for minimizing the objective function while preserving the orthogonality constraints. The Cayley transform smoothly projects the skew-symmetrized gradient on the constraints, which enables solving the update of the minimization with a curvilinear search algorithm. In other words, the gradient is transformed so that a Crank–Nicolson update rule is guaranteed to (i) converge and (ii) yield a solution satisfying the constraints. Wen and Yin also show experimentally that FOptM is more robust and efficient than its competitors. We use FOptM to solve (13), and we turn the objective into a minimization problem by multiplying by  $-1$ .

Note that FOptM requires the partial derivative of (13) with respect to  $W$ . Inserting the RBF kernel, the partial derivative is

$$\begin{aligned}
 & \frac{\partial(\text{Tr}(L_2 K_{DW}))}{\partial W} \\
 & = \sum_{i,j} L_2(i, j) \frac{\partial K_{DW}(i, j)}{\partial W} \\
 & = \sum_{i,j} -\frac{1}{\sigma_s^2} L_2(i, j) K_{DW}(i, j) (d_i - d_j)(d_i - d_j)^T W \\
 & = \sum_{i,j} -\frac{1}{\sigma_s^2} L_2(i, j) K_{DW}(i, j) (d_i - d_j)(f_i - f_j),
 \end{aligned} \tag{14}$$

where  $f_i = W^T d_i$  is the  $i$ th data point in the projected space and  $\sigma_s$  is the kernel width in the projected space.

### 3.3 The algorithm KCC

We propose a novel algorithm, called kernel conditional clustering (KCC), shown in Algorithm 1, to solve (10). In Lines 2–8, the kernel width  $\sigma_s$  in the projected space and  $W$  are initialized, which will be described in further detail in the next paragraph. With  $\sigma_s$  fixed, we iteratively optimize  $U$  and  $W$  according to (12) and (13) until a local optimum is achieved (Lines 10–16). This is guaranteed to converge, because each step increases the objective function in (10). Finally, we normalize the clustering embedding  $U$  to the row sums



$(u_{ij} = \frac{u_{ij}}{\sqrt{\sum_j u_{ij}^2}})$  and perform  $k$ -means clustering on the normalized  $U$  (Lines 18 and 19). Figure 1 depicts a flowchart of how KCC processes its input to arrive to the final solution  $U$ .

---

**Algorithm 1:** KCC

---

```

1 KernelConditionalClustering ( $D, C, k$ );
   Input : Data  $D$ , Covariates  $C$ , Number of clusters  $k$ 
   Output: Clustering  $U$  and transformation matrix  $W$ 
2  $\sigma_s = \text{MedianDist}(D)$ ;
3 Calculate  $K_U$  as in (11) with  $\sigma_s$  and  $W = I$ ;
4 Calculate  $L_2$  as in (13) with centered  $K_U$ ;
5 while not converge do
6   | Set  $W$  as the solution to (13) by FOptM;
7   |  $\sigma_s = \text{MedianDist}(DW)$  ;
8 end
9
10 while not converge do
11   | Calculate  $K_{DW}$  as in (11);
12   | Calculate  $L_1$  as in (12);
13   | Set  $U$  to the first  $k$  eigenvectors of  $L_1$ ;
14   | Calculate  $L_2$  as in (13) with  $K_U = UU^T$ ;
15   | Set  $W$  as the solution to (13) by FOptM;
16 end
17
18  $U(i, j) = \frac{U(i,j)}{\|U(i,:)\|}$ ;
19  $U = k\text{-means}(U, k)$ ;
```

---

Estimating the parameter  $\sigma$  in the RBF kernel is still an open problem for unsupervised clustering. A heuristic strategy, setting  $\sigma$  to the median of the pairwise distances, has been shown to work well in practice [8,14]. In our case,  $\sigma_s$  is the median of the pairwise distances in the projected space  $DW$ , which depends on the unknown  $W$  in (14). We therefore use an iterative procedure to estimate  $\sigma_s$  and  $W$ , as shown in Lines 5–8 of Algorithm 1. First,  $W$  in (13) is initialized by FOptM with the RBF kernel  $K_D$  (Line 6) setting  $\sigma_s$  from the original data space  $D$ . This can be seen as a dimensionality reduction procedure, ensuring that all the information of the data  $D$  is preserved, apart from those correlated with the covariates  $C$ . Then we calculate the  $\sigma_s$  in the projected subspace (Line 7). We iterate these two steps until convergence. We note that, empirically, this initialization always converges in a few steps.

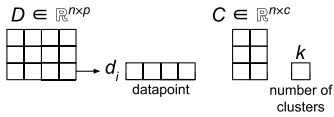
### 3.4 Computational complexity

Assume a dataset with  $n$  samples and  $p$  features. The runtime complexity of a matrix inversion process in KCC is  $O(n^3)$ . The eigenvalue decomposition in KCC costs again  $O(n^3)$ . The runtime of the optimization method with orthogonal constraints (FOptM) is dominated by  $O(n^2 p)$ , where  $O(n^2 p)$  is for calculating the partial derivative in (14). Therefore, the runtime for the KCC is  $O(n^3 t + n^2 pt)$ , where  $t$  is the number of iterations.

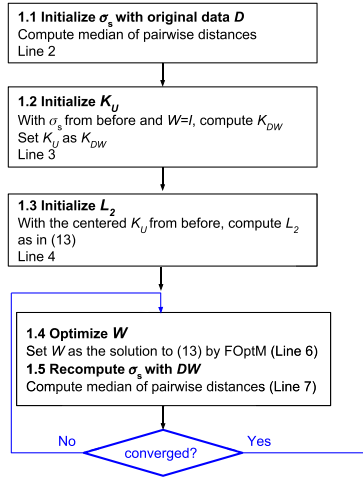
### 3.5 Comparison of KCC to alternative clustering

*Alternative clustering* finds a second clustering that has high quality (dependent on the data  $D$ ) and is dissimilar to the reference clustering  $C$  (independent of  $C$ ). Many alternative

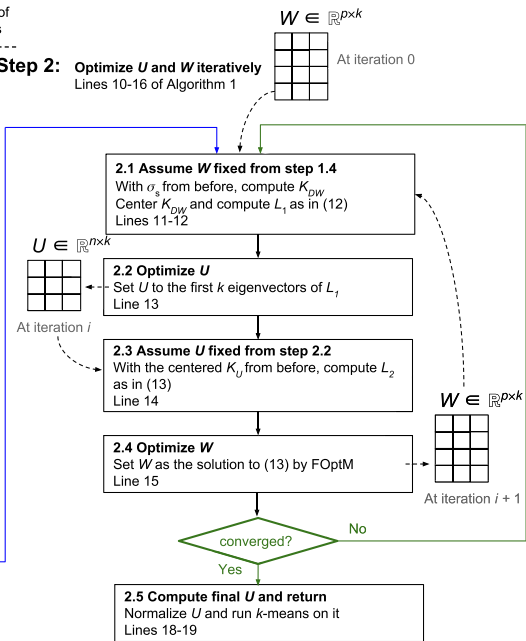
**Input**



**Step 1: Estimate  $\sigma_s$  and  $W$**   
Lines 2-8 of Algorithm 1



**Step 2: Optimize  $U$  and  $W$  iteratively**  
Lines 10-16 of Algorithm 1



**Fig. 1** Flowchart of Algorithm 1 KCC

clustering approaches [7,9–12,25] use a regularized framework to reach this goal:

$$\max_U Q(D, U) - \lambda P(C, U), \tag{15}$$

where  $Q$  measures the clustering quality of  $U$  given  $D$ , and  $P$  measures the dissimilarity between  $C$  and  $U$ .  $\lambda \geq 0$  is a regularization parameter that controls the trade-off between the clustering quality and the dissimilarity function.

Niu et al. proposed KDAC [11] to measure both  $Q$  (clustering quality) and  $P$  (relation between clusterings) by  $HSIC_{emp}$ . It was shown that the spectral clustering used in that paper for  $Q$  can be expressed as  $HSIC_{emp}$ . The objective function of their method KDAC can be written as

$$\begin{aligned} \max_{W, U} HSIC(DW, U) - \lambda HSIC(C, U) \\ \text{s.t. } W^T W = I, U^T U = I. \end{aligned} \tag{16}$$

A large regularization parameter  $\lambda$  will enforce the maximization of the dependence between the projected subspace and the clustering embedding ( $HSIC(DW, U)$ ) and the minimization of the dependence between the clustering and the known covariates ( $HSIC(C, U)$ ). Therefore, this regularization framework can achieve the goal of conditional dependence as a side product. However, such a combination is rather ad hoc and has no clear statistical meaning. In contrast, the proposed method KCC is based on a kernel conditional dependence measure and has a clear statistical interpretation. The hyper-parameter  $\lambda$  in the regularization

framework of (16) is difficult to tune in practice, and note that there is no such hyper-parameter in our proposed method KCC. Compared to KDAC, this is achieved with one extra matrix inversion of the kernel matrix of covariates, which only needs to be performed once in KCC.

### 4 Kernel conditional semi-supervised learning

In this section, we extend the idea of conditional clustering to the problem of conditional semi-supervised learning. As shown in Sect. 3, in the clustering setting we learn a relationship between unlabeled samples. In contrast, a classification setting would require to learn a mapping between a feature space and class labels. In between clustering and classification is semi-supervised learning where there are few labeled samples and a large number of unlabeled ones. In semi-supervised learning, the classification accuracy is improved by learning a relationship between unlabeled and labeled samples. Since the number of labeled samples is usually small, the effect of semi-supervised learning will be strongly affected when there are irrelevant covariates, as it is also the case in clustering.

We here present a novel algorithm named kernel conditional semi-supervised learning (KCSSL) that tackles the semi-supervised learning problem while conditioned on known covariates. We first introduce a new objective function, followed by the details of its implementation regarding the optimization. Finally, we contrast KCSSL to other graph-based semi-supervised learning algorithms [26,27].

#### 4.1 The kernel conditional semi-supervised learning model

Consider a dataset  $D \in \mathbb{R}^{n \times p}$  and a set of  $c$  known covariates  $C \in \mathbb{R}^{n \times c}$ . In the semi-supervised learning setting, there are  $l$  labeled samples and  $u$  unlabeled samples. Denote  $Y_l = [y_1, y_2, \dots, y_l]^T$ , where  $y_i \in \mathbb{R}^k$  is the known indicator for the  $i$ th sample,  $y_i$  is one-hot, and element  $y_{ij} = 1$  means that the  $i$ th sample belongs to the  $j$ th class. The aim of semi-supervised learning is to infer the label  $Y_u \in \mathbb{R}^{u \times k}$  for the  $u$  unlabeled samples, while  $Y_l$  stays fixed. Similar to conditional clustering, we define the conditional semi-supervised learning objective with HSCONIC to achieve this goal as defined in (6):

$$\begin{aligned} \max_{Y_u} \quad & \text{HSCONIC}_{\text{emp}}(D, [Y_l; Y_u] | C) \\ \text{s.t.} \quad & Y_u(i, j) \in [0, 1] \quad \text{and} \quad \sum_j Y_u(i, j) = 1. \end{aligned} \tag{17}$$

In (17), the dependency between the dataset  $D$  and both known and inferred labels  $Y = [Y_l; Y_u]$  is maximized conditioned on the covariates  $C$ .

Due to the combinatorial problem of optimizing  $Y_u$  in (17), we relax it to an orthogonality constraint as in KCC. The relaxed objective is then

$$\begin{aligned} \max_{Y_u} \quad & \text{HSCONIC}_{\text{emp}}(D, [Y_l; Y_u] | C) \\ \text{s.t.} \quad & Y_u^T Y_u = I. \end{aligned} \tag{18}$$

Plugging (6) into the relaxed objective (18) (and dropping the constant factor) we obtain

$$\begin{aligned} \max_{Y_u} \quad & \text{Tr}(\bar{K}_D \bar{K}_Y - 2\bar{K}_D M \bar{K}_Y + \bar{K}_D M \bar{K}_Y M) \\ & M = \bar{K}_C (\bar{K}_C + \epsilon I_n)^{-2} \bar{K}_C \\ \text{s.t.} \quad & Y_u^T Y_u = I, \end{aligned} \tag{19}$$

where  $Y = [Y_l; Y_u]$ ,  $K_*$  is kernel matrix and  $\bar{K}_*$  is the centered kernel matrix, as in KCC.

To capture nonlinearity, we choose the Gaussian RBF kernel for the Gram matrices  $K_D$  and for  $K_C$ . To optimize  $Y_u$ , we use a linear kernel  $K_Y = Y Y^T$ . The problem can be rewritten as

$$\begin{aligned} \max_{Y_u} \quad & \text{Tr}([Y_l; Y_u]^T G [Y_l; Y_u]) \\ \text{s.t.} \quad & Y_u^T Y_u = I, \end{aligned} \tag{20}$$

where  $G = \bar{K}_D - 2\bar{K}_D M + M \bar{K}_D M$ .

### 4.2 Optimization

Without loss of generality we can split  $G$  into blocks as  $G = \begin{bmatrix} G_{ll} & G_{lu} \\ G_{ul} & G_{uu} \end{bmatrix}$  and expand the objective function (19) to

$$\begin{aligned} \text{Tr}([Y_l; Y_u]^T G [Y_l; Y_u]) = & \text{Tr}(Y_l^T G_{ll} Y_l) + \text{Tr}(Y_l^T G_{lu} Y_u) \\ & + \text{Tr}(Y_u^T G_{ul} Y_l) + \text{Tr}(Y_u^T G_{uu} Y_u) \end{aligned} \tag{21}$$

where the first term is constant. Dropping it, (19) can be rewritten as

$$\begin{aligned} \max_{Y_u} \quad & \text{Tr}(Y_l^T G_{lu} Y_u) + \text{Tr}(Y_u^T G_{ul} Y_l) + \text{Tr}(Y_u^T G_{uu} Y_u) \\ \text{s.t.} \quad & Y_u^T Y_u = I \end{aligned} \tag{22}$$

For the non-convex optimization problem with an orthogonality constraint in (22), we again adopt FOptM for optimization.

FOptM requires the analytical form of the derivative of (22) with respect to  $Y_u$ :

$$\frac{\partial(\text{Tr}(Y_l^T G_{lu} Y_u) + \text{Tr}(Y_u^T G_{ul} Y_l) + \text{Tr}(Y_u^T G_{uu} Y_u))}{\partial Y_u} = 2G_{ul} Y_l + 2G_{uu} Y_u \tag{23}$$

The final one-hot-encoded class labels of the unlabeled points are obtained by following the decision function:

$$y_i = \underset{j}{\text{argmax}} Y_u(i, j) \tag{24}$$

### 4.3 The algorithm KCSSL

The algorithm KCSSL is summarized in Algorithm 2. Assume a real-valued dataset  $D$  with  $n$  samples and  $p$  features, covariates  $C$  and labels  $Y_l$ ,  $l < n$ . The first step of KCSSL is to compute  $M$  as in (18) based on the RBF kernel  $K_C$  on  $C$ . Using  $M$  and RBF kernel  $K_D$  on  $D$ , the matrix  $G$  is constructed as in (19). Then, leveraging the partial derivative in (23), FOptM is used to solve the objective in (22). Finally, the class labels  $Y_u$  of the unlabeled samples are obtained with the decision rule stated in (24). Figure 2 depicts a flowchart of the process of KCSSL.

The computational complexity of the matrix inversion is of  $O(n^3)$ . The calculation of the kernel matrix is of  $O(n^2 p)$ . Since there is no dimensionality reduction, one partial derivative calculation is required for solving the objective with FOptM in (23), which takes  $O(n^2)$ . Therefore, the computational complexity of KCSSL is  $O(n^3 + n^2 p)$ .

---

**Algorithm 2: KCSSL**

---

- 1 **KernelConditionalSSL** ( $D, C, Y_l$ );  
    **Input** : Data  $D$ , Covariates  $C$ , Labeled samples  $Y_l$   
    **Output**: Prediction for unlabeled samples  $y_i, l < i \leq n$
  - 2  $\sigma_C = \text{MedianDist}(C)$ ;
  - 3 Calculate  $K_C$  as in (11) with  $\sigma_C$  and  $W = I$ ;
  - 4 Calculate  $M$  as in (19) with  $K_C$ ;
  - 5
  - 6  $\sigma_D = \text{MedianDist}(D)$ ;
  - 7 Calculate  $K_D$  as in (11) with  $\sigma_D$  and  $W = I$ ;
  - 8 Calculate  $G$  as in (20) with  $K_D$  and  $M$ ;
  - 9
  - 10 Set  $Y_u$  as the solution to (22) by FOptM;
  - 11
  - 12  $y_i = \text{argmax}_j Y_u(i, j)$  as in (24);
- 

**4.4 Comparison of KCSSL to graph-based semi-supervised learning methods**

In graph-based semi-supervised learning methods, the relationship between unlabeled and labeled samples is reflected by a graph constructed from the dataset. Graph-based methods differ, among other aspects, by the choice of loss function. For instance, Label Propagation (LP) [26] was the first semi-supervised learning method where the loss is based on the combinatorial graph Laplacian. We proceed to investigate the comparison of KCSSL to LP in detail.

Given the data  $D$  and its adjacency matrix  $W \in \mathbb{R}^{n \times n}$ , the Laplacian matrix of  $D$  is defined as  $L = \text{degree}(W) - W$ , where  $\text{degree}(W)$  is a diagonal matrix with  $\sum_{j=1}^n W(i, j)$  on the  $i$ th position. With labeled and unlabeled samples denoted as  $Y_l$  and  $Y_u$ , the semi-supervised learning problem can be written as

$$\min_{Y_u} \text{Tr}([Y_l; Y_u]^T L [Y_l; Y_u]) \tag{25}$$

Zhu et al. showed in [26] that the optimal solution to (25) is

$$Y_u = -L_{uu}^{-1} L_{ul} Y_l, \tag{26}$$

and  $Y_u$  satisfies that  $Y_u(i, j) \in [0, 1]$  if  $L$  is a Laplacian matrix. Similarly, the labels of unlabeled samples are predicted using (24).

Revisiting the objective function of KCSSL in (22) and dropping the orthogonality constraint  $Y_u^T Y_u = I$ , the optimal solution can be derived by setting its derivative regarding  $Y_u$ , as shown in (23), to 0:

$$Y_u = -G_{uu}^{-1} G_{ul} Y_l, \tag{27}$$

which has the same form of solution as label propagation in (26).

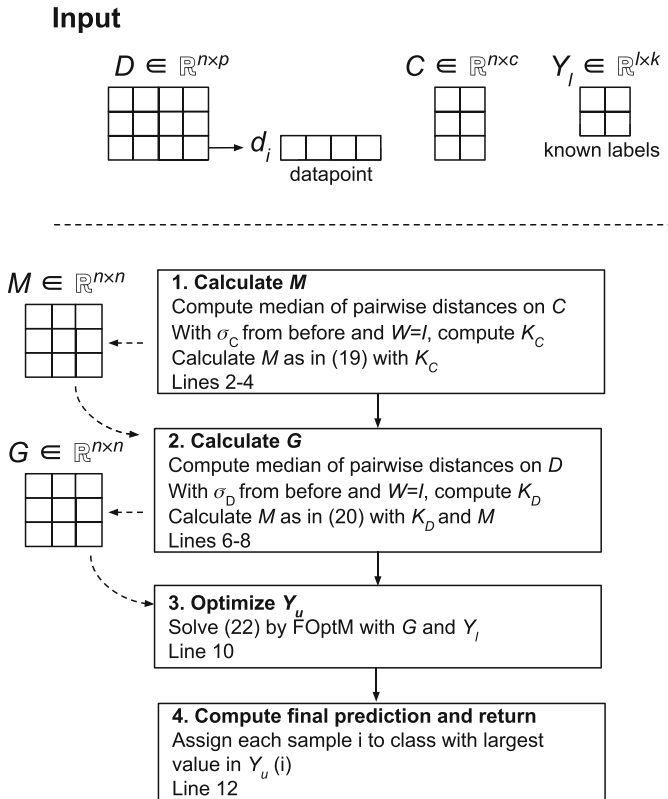


Fig. 2 Flowchart of Algorithm 2 KCSSL

In KCSSL, we form a symmetric matrix  $G$  based on HSCONIC, such that irrelevant covariate information  $C$  is eliminated. However,  $G$  is not a Laplacian matrix and is not positive semi-definite. Thus, the solution to (27) will not be restricted to  $Y_u(i, j) \in [0, 1]$ . To compensate for this, we add the orthogonality constraint on  $Y_u$  and solve it with FOptM. Empirically, and as shown in the next section, KCSSL achieves excellent results in our experiments on semi-supervised learning with covariates.

## 5 Experiments

To demonstrate the effectiveness of our proposed approaches, we report empirical results and compare them to a number of competing methods on both synthetic and real-world datasets. Note that we use the same datasets in two different ways, depending on the type of problem we are considering. For the clustering setting, we provide the methods with the information about the known covariates and we assume that the class labels are missing (although we do know what the real labels are). For semi-supervised learning, we provide the covariates and assume that only a small fraction of the samples have class labels.

## 5.1 Comparison partners and evaluation measures

### 5.1.1 Comparison partners and evaluation measures for clustering

To obtain a baseline of clustering performance, standard  $k$ -means (KM) and kernel  $k$ -means (KKM) are run without considering the known covariates. We then compare KCC to the following 8 state-of-the-art comparison partners:

- *CCIB* [2] is based on the idea of maximizing the conditional mutual information relative to the known covariates. As far as we know, this is the only algorithm in the literature based on a conditional dependence measure.
- *COALA*, from Bae and Bailey [3], is an alternative clustering algorithm that uses *must-link* and *cannot-link* constraints to enforce dissimilarity between clusterings.
- *OC*, orthogonal clustering algorithm 1 from Cui et al. [4], is an alternative clustering algorithm that projects data to a space that is orthogonal to the known clustering.
- *ADFT* from Davidson and Qi [5] also uses *must-link* and *cannot-link* constraints, but it learns an inverse distance metric to discover a different clustering.
- *RPCA* and *RegGB* from Dang and Bailey [10] learn a linear and nonlinear subspace for finding an alternative clustering using PCA and kernel discriminant analysis.
- *MNMF* from Yang and Zhang [12] uses nonnegative matrix factorization for clustering and HSIC as the similarity function and combines them with a regularization parameter.
- *KDAC* from Niu et al. [11] combines spectral clustering, subspace learning, and HSIC in a regularization framework.
- *KCC* is our proposed approach, and it is based on the kernel conditional dependence measure HSCONIC.

All clustering methods require the number of clusters  $k$  to be predetermined, which we set to be the true number of clusters. We fix a Gaussian RBF kernel for the kernel-based methods (KKM, RegGB, KDAC, and KCC) and use one-hot encodings to represent categorical covariates. For the subspace methods (RPCA, RegGB, KDAC, and KCC), the dimensionality in the projected subspace is set to the number of clusters  $k$ . The parameter  $\sigma$  in the RBF kernel is determined using the median of the pairwise distances (a common heuristic, as shown in [8]). For KDAC and KCC,  $\sigma_s$  for the RBF kernel in the projected subspace is determined as explained in Sect. 3.3. The  $\epsilon$  in (10) is set to  $10^{-8}$ , and the convergence threshold is set to  $10^{-3}$  of the relative improvement. For a fair comparison of the methods, with and without hyper-parameters, we cluster according to many different parameter choices from an allowed range ( $[10^{-3}, 10^3]$ ). We take a majority vote on the best performing parameter from each dataset. This is the parameter we chose for comparison. For KDAC, the regularization parameter  $\lambda = 0.01$  was determined by applying this procedure. For CCIB and MNMF, the best performing  $\lambda$  turns out to be  $\lambda = 0.1$ . RegGB requires the number of nearest neighbors, which is  $n = 10$ . For those methods which use  $k$ -means, we ran  $k$ -means 100 times and report the clustering with the minimum objective value. For a fair comparison with the other methods, we also report the one with the best  $k$ -means objective.

The evaluation of the methods is based on four measures. On the one hand, the clustering result is compared to the ground truth labels in terms of accuracy (ACC) and normalized mutual information ( $\text{NMI}_g$ ), where a larger value means better performance. On the other hand, we measure the dependence between the clustering and the known categorical covariates by NMI (denoted by  $\text{NMI}_c$ ) and the Jaccard index (JI). The goal is to minimize the dependence, i.e., low values correspond to better performance.

### 5.1.2 Comparison partners and evaluation measures for semi-supervised learning

For semi-supervised learning, we compare KCSSL to the following state-of-the-art semi-supervised learning comparison partners:

- *LP* (Label Propagation) from Zhu et al. [26], the semi-supervised learning method based on graph Laplacian, which is conceptually closest to KCSSL.
- *SP* (spread propagation) from Zhou et al. [27], which is similar to label propagation but uses a normalized graph Laplacian and soft clamping across the labels.
- *ST* (self-training) is a naive semi-supervised learning framework applicable for any classifier. In our case, we use logistic regression as the base classifier.
- *KCSSL* is the proposed semi-supervised learning method that is based on the kernel conditional dependence measure HSCONIC.

Semi-supervised learning methods are implemented as follows. Similar to clustering, we use a Gaussian RBF kernel for the kernel-based methods (*LP*, *SP*, and *KCSSL*) and one-hot encodings. The parameter  $\sigma$  in the RBF kernel is determined using the median of the pairwise distances. For *SP*,  $\alpha \in [0, 1]$  specifies the relative amount of information that an instance should adopt from its neighbors as opposed to its initial label. We evaluate  $\alpha = 0.01, \dots, 0.9$  and report the results for the majority vote on the best performing  $\alpha$  across all datasets. For *ST*, logistic regression is used as a base classifier because (i) the majority of the datasets contain multiple class labels and logistic regression can handle multi-class scenarios easily, and (ii) the convergence of *ST* in combination with logistic regression is theoretically guaranteed [28].

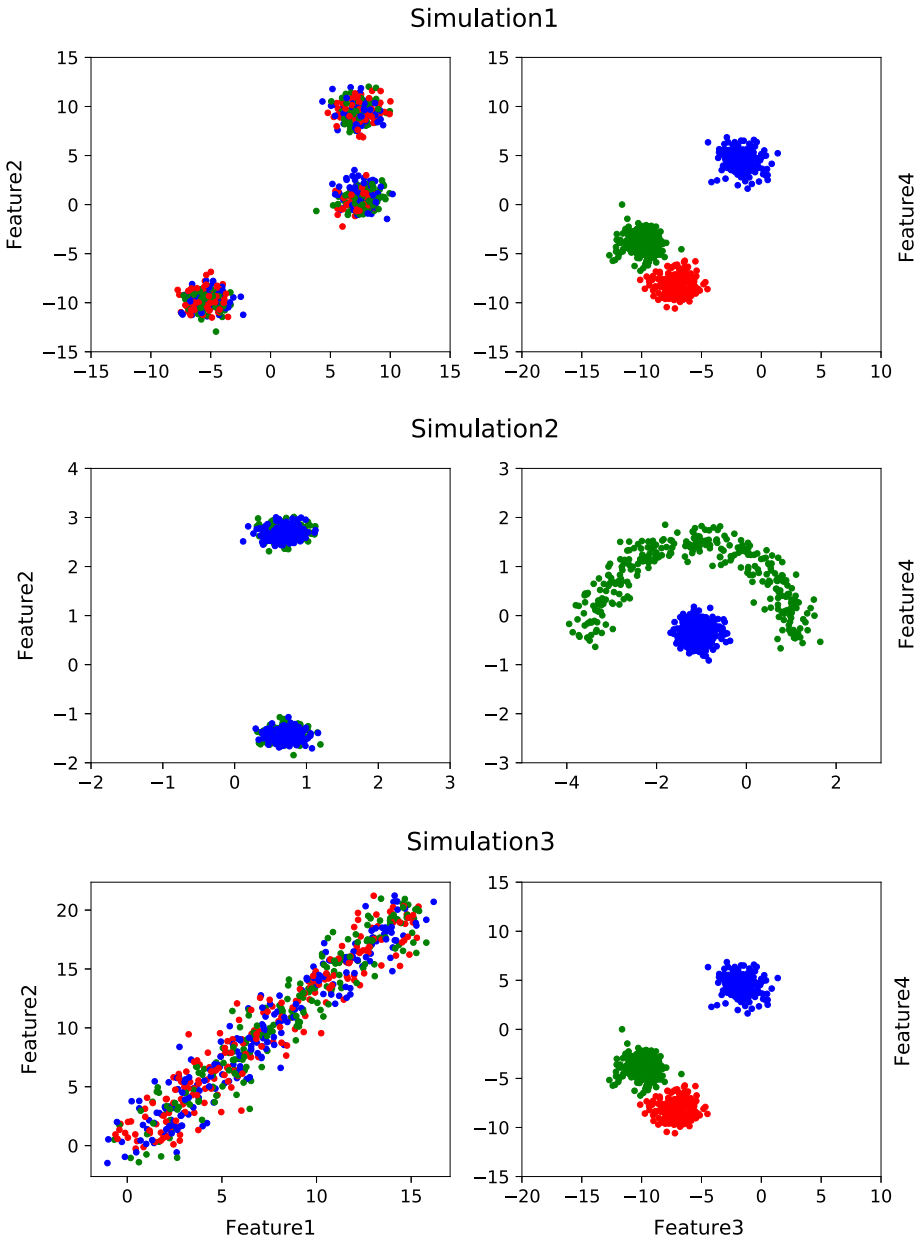
The evaluation of semi-supervised learning is measured using prediction accuracy (*ACC*) on the unlabeled data points, where a larger value corresponds to better performance. This means that we are evaluating a *transductive* setting where no test points are held back.

## 5.2 Datasets

We evaluate the performance of all the above methods on three simulation datasets and on five real-world datasets.

- *Simulation datasets* The three simulation datasets (*Simu1*, *Simu2*, and *Simu3*) are shown in Fig. 3. All three consist of a four-dimensional space which can be separated into two subspaces: *Feature1* and *Feature2* show a different structure than *Feature3* and *Feature4*. The difference between the simulations can be summarized as follows: In *Simu1*, the clusters are linearly separable. In *Simu2*, one cluster is of non-normal shape. In *Simu3*, *Feature1* and *Feature2* are linear transformations of a Gaussian continuous variable. *Feature3* and *Feature4* show the clusters one would like to discover (indicated also by color), while the cluster labels (*Simu1* and *Simu2*) and the Gaussian variable (*Simu3*) that generate *Feature1* and *Feature2* are taken as the covariates.
- *Face and Face4* *Face* is the CMUFace dataset (from the UCI repository [29]). It contains 624 face images of 20 people in all combinations of orientation (straight, left, right, up), facial expression (neutral, happy, sad, angry), and eyewear (wearing sunglasses or not). The two dominating signals are the identity of the person and their pose. The image resolution is  $32 \times 30$ , which means that each sample has 960 features on a continuous scale. *Face* contains all 624 images. *Face4* enforces an additional symmetry in the data: Since the number of people is much larger than the number of poses, we follow the idea of [10,12] and extract the subsamples from the first 4 people (*an2i*, *at33*, *boland*, and *bpm*).





**Fig. 3** All three simulation datasets consist of 4 features. Two features express the covariate (LHS plots); two features exhibit the clustering which is to be discovered (RHS plots). Simu1 has three Gaussian clusters in both views. In the second view of Simu2, a non-normal separation into two clusters is synthesized. The covariate of Simu3 is continuous; the clustering is the same as that of Simu1

**Table 1** Datasets

Name	Samples	Features	Covariates	Clusters
Simu1	600	4	3	3
Simu2	600	4	2	2
Simu3	600	4	Continuous	3
Face	624	960	20	4
Face4	128	960	4	4
Multi	135	5565	2	4
DLBCL	321	661	2	3
WebKB	1041	456	4	4

Therefore, there are the same number of people and poses. For both Face and Face4, we use the person's identity as the known covariate and discover the clustering according to the pose.

- *Multi and DLBCL* The gene expression datasets are generated from two different generations of micro-array platforms [30], which we treat as covariates. Multi contains gene expressions from 4 tissue types (breast, lung, prostate, and colon). The DLBCL dataset contains gene expressions from three subtypes of lymphoma cancer (diffuse large B cell lymphoma, DLBCL). The goal is to discover clusters according to tissue and cancer types, conditioned on the generation of micro-array platform.
- *WebKB dataset* The CMU World Wide Knowledge Base collected in 1997 is a dataset containing web pages from computer science departments of 4 universities in the USA: Cornell University, the University of Texas at Austin, the University of Washington, and the University of Wisconsin at Madison [31]. The pages are additionally labeled as being from 4 topical areas: course, faculty, project, and student. We follow the preprocessing strategy of [4,11], where we remove the rare words (frequency smaller than 5%) and use term frequency–inverse document frequency as the feature values, retaining a dataset with 1041 samples and 456 features. In the experiments, the university is given as the covariate and the clustering of the topics is treated as the signal.

Table 1 shows the summary statistics of these datasets. The number of categories in the categorical covariates and the number of clusters in the ground truth are in the last two columns. The original source of all the real datasets and the scripts that generated the ones used in the paper as well as an implementation of KCC can be found online.<sup>1</sup>

The same datasets are used for the evaluation of clustering and of semi-supervised learning algorithms. For semi-supervised learning, each dataset is randomly split such that 10% of samples are considered labeled and the remaining 90% samples are assumed unlabeled. We create 100 random splits of each dataset and report the average performance and standard deviation.

## 5.3 Results

### 5.3.1 Clustering results

The results from the clustering experiments are displayed in Table 2. In each subtable, the rows correspond to the datasets and the columns to the comparison partners, with the last column showing the proposed approach, KCC.

<sup>1</sup> <https://github.com/BorgwardtLab/Kernel-Conditional-Clustering>.

Table 2(a), (b) is external performance measures, i.e., with respect to the ground truth labels, and reports the accuracy (ACC) and the normalized mutual information ( $NMI_g$ ), respectively. For any given method and dataset, the closer the score is to 1, the better the performance is. Table 2(c), (d) reports the dependence between the clustering and the known categorical covariates by NMI (denoted by  $NMI_c$ ) and the Jaccard index (JI), respectively. For these measures, the lower the score, the better the performance.

For each dataset, the score of the best performing method is highlighted in bold (row-wise). If there is a tie, or if the score attained by a method is very close to the best score ( $\leq 0.01$ ), then more than one score will be highlighted in bold.

For the three simulated datasets, we first validate that neither of the  $k$ -means algorithms (KM and KKM) can discover the additional clustering structure shown in Fig. 3 (right-hand side). Among the comparison partners for conditional clustering, many methods work for the Gaussian case (Simu1), including OC, RPCA, KDAC, and KCC. As expected, the linear methods (OC and RPCA) fail in the nonlinearly separable case (Simu2). We note that the proposed method KCC, in conjunction with KDAC, gives almost optimal results for both simulations. For Simu3 with continuous covariates, COALA, ADFT, and OC cannot be used directly. KCC and CCIB discover the ground truth clustering perfectly. KDAC failed on Simu3 with  $\lambda = 0.01$ , but worked with  $\lambda = 1$ . Later, we will compare KCC with KDAC with different values of  $\lambda$ .

On the Face, Face4, and WebKB datasets, regarding the external performance (ACC and  $NMI_g$ ), KCC dominates the list of comparison partners. For WebKB, OC and RPCA give similarly good results. The performances of CCIB and RegGB are not stable on these datasets. Multi and DLBCL are datasets from a clinical setting where the platform information is given as a covariate, and the resulting clustering is evaluated against the biological type of the sample. MNMF is the best performer on the DLBCL datasets, while OC and KDAC perform the best for Multi, when considering  $NMI_g$  and ACC, respectively. Nevertheless, KCC is always among the top three methods in these two datasets.

The last row in Table 2(a)–(d) shows the average rank of all methods across the datasets. For each dataset (row-wise) we rank the eleven methods, based on their scores, from 1 through 11. The method with the best score is ranked first, i.e.,  $r = 1$ , the next best score gets a rank  $r = 2$ , all the way to the worst performing method ( $r = 11$ ). In case of ties, more than one method receives the same rank  $r$ , and the subsequent value of  $r$  is incremented accordingly. For example, in Table 2(a) for the Simu1 dataset, four methods share the rank  $r = 1$ ; therefore, the next performer (CCIB) receives a rank  $r = 5$ . The average rank per method (column-wise) is then computed by summing up all the ranks for the method and dividing by seven (i.e., the number of datasets).

The average ranks in Table 2(a), (b) show the average performance of all methods with respect to ACC and  $NMI_g$ , respectively. KCC ranked in the first place for both measures, followed by KDAC and OC. The next methods are RPCA, CCIB, MNMF, and RegGB. The performance of these methods is not stable across all tested datasets. COALA and ADFT perform the worst on the tested datasets. One possible reason is that they are not designed for high-dimensional data because they rely on global pairwise distances.

Regarding the similarity of the results to the given covariate (described by  $NMI_c$  and JI), KCC, KDAC, and MNMF perform equally well on all the datasets. The absolute differences between  $NMI_c$  and JI among these methods are minor. In addition, we observe that most comparison partners can discover a clustering that is dissimilar to the covariates, yet the ground truth labels are generally not recovered well.

To summarize, the proposed algorithm KCC gives on average the best clustering performance in terms of recovering the ground truth and in terms of dissimilarity to the covariates,

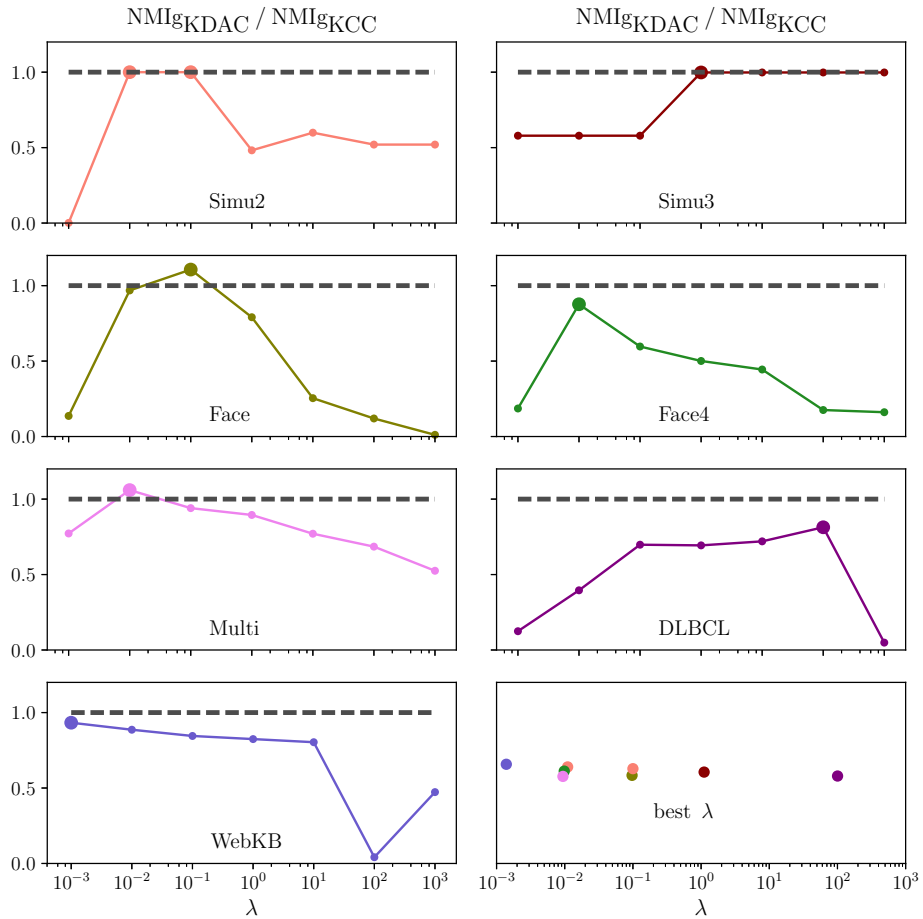
**Table 2** The results of the experiments are displayed in tables (a)–(d)

Dataset or method	KM	KKM	CCIB	COALA	ADFT	OC	RPCA	RegGB	MNMF	KDAC	KCC
<i>(a) ACC</i>											
Simu1	0.573	0.573	0.683	0.565	0.493	<b>0.993</b>	<b>0.993</b>	0.667	0.663	<b>0.993</b>	<b>0.993</b>
Simu2	0.505	0.505	0.710	0.701	0.710	0.725	0.710	0.557	0.831	<b>1</b>	<b>1</b>
Simu3	0.668	0.675	<b>1</b>	–	–	–	0.438	0.818	0.873	0.675	<b>1</b>
Face	0.358	0.466	0.653	0.246	0.356	0.738	0.717	0.373	0.647	<b>0.788</b>	<b>0.796</b>
Face4	0.375	0.375	0.632	0.453	0.391	0.640	0.648	0.554	0.578	0.812	<b>0.851</b>
Multi	0.474	0.592	0.644	0.288	0.355	0.755	0.688	0.644	0.562	<b>0.837</b>	0.807
DLBCL	0.492	0.512	0.467	0.429	0.501	0.467	0.470	0.613	<b>0.688</b>	0.448	0.595
WebKB	0.491	0.518	0.528	0.317	0.336	<b>0.654</b>	<b>0.645</b>	0.622	0.483	0.573	<b>0.644</b>
∅ Rank	8.375	7.25	4.875	9.50	8.625	4.125	4.25	5.57	5.375	3.375	<b>1.625</b>
<i>(b) NMI<sub>g</sub></i>											
Simu1	0.387	0.387	0.580	0.371	0.493	<b>0.966</b>	<b>0.966</b>	0.367	0.270	<b>0.966</b>	<b>0.966</b>
Simu2	0	0	0.258	0.238	0.251	0.258	0.250	0.060	0.455	<b>1</b>	<b>1</b>
Simu3	0.553	0.579	<b>1</b>	–	–	–	0.047	0.695	0.615	0.579	<b>1</b>
Face	0.070	0.152	0.424	0.258	0.065	0.483	0.483	0.058	0.361	0.518	<b>0.526</b>
Face4	0.126	0.126	0.394	0.174	0.149	0.602	0.448	0.515	0.375	0.604	<b>0.689</b>
Multi	0.070	0.152	0.434	0.041	0.113	<b>0.671</b>	0.533	0.381	0.375	0.656	0.619
DLBCL	0.113	0.118	0.092	0.022	0.077	0.123	0.128	0.171	<b>0.377</b>	0.089	0.225
WebKB	0.276	0.225	0.314	0.024	0.048	<b>0.384</b>	0.368	0.302	0.200	0.343	<b>0.387</b>
∅ Rank	8.375	7.75	4.875	9.25	8.625	3.375	4.5	6.5	6.0	3.25	<b>1.375</b>

Table 2 continued

Dataset or method	KM	KKM	CCIB	COALA	ADFT	OC	RPCA	RegGB	MNMF	KDAC	KCC
<i>(c) NMI<sub>c</sub></i>											
Simu1	0.590	0.590	0.387	0.589	0.677	<b>0.001</b>	<b>0.001</b>	0.441	<b>0.004</b>	<b>0.001</b>	<b>0.001</b>
Simu2	1	1	<b>0.002</b>	<b>0.002</b>	<b>0.003</b>	<b>0.001</b>	<b>0.002</b>	0.105	<b>0.004</b>	<b>0.002</b>	<b>0.002</b>
Face	0.503	0.356	<b>0.017</b>	0.238	0.361	0.021	0.043	0.468	<b>0.026</b>	<b>0.016</b>	<b>0.018</b>
Face4	0.781	0.781	<b>0.019</b>	0.301	0.548	0.090	0.151	0.195	<b>0.015</b>	<b>0.012</b>	0.024
Multi	0.651	0.637	0.064	0.135	0.294	0.038	0.122	0.080	<b>0.018</b>	<b>0.016</b>	0.034
DLBCL	0.814	0.832	0.326	0.959	0.782	0.273	0.273	0.107	<b>0.008</b>	<b>0.010</b>	0.046
WebKB	0.412	0.338	0.063	0.020	0.393	<b>0.009</b>	<b>0.010</b>	0.039	0.012	<b>0.001</b>	<b>0.010</b>
∅ Rank	10.14	9.42	4.71	7.14	9.00	3.14	4.28	7.14	4.00	<b>1.28</b>	2.71
<i>(d) JI</i>											
Simu1	0.514	0.514	0.366	0.512	0.619	<b>0.200</b>	<b>0.200</b>	0.430	0.216	<b>0.200</b>	<b>0.200</b>
Simu2	1	1	0.401	<b>0.333</b>	0.404	0.392	0.401	0.476	0.359	<b>0.333</b>	<b>0.333</b>
Face	0.144	0.109	<b>0.048</b>	0.066	0.144	<b>0.047</b>	0.051	0.120	<b>0.045</b>	<b>0.046</b>	<b>0.046</b>
Face4	0.655	0.655	<b>0.159</b>	0.284	0.424	0.183	0.201	0.218	<b>0.149</b>	<b>0.149</b>	<b>0.153</b>
Multi	0.588	0.423	0.307	0.651	0.331	0.257	0.321	0.283	<b>0.226</b>	<b>0.226</b>	<b>0.232</b>
DLBCL	0.810	0.810	0.513	0.982	0.795	0.499	0.499	0.322	<b>0.256</b>	<b>0.263</b>	<b>0.266</b>
WebKB	0.342	0.285	<b>0.152</b>	0.370	0.359	<b>0.156</b>	<b>0.158</b>	0.206	<b>0.151</b>	<b>0.147</b>	<b>0.157</b>
∅ Rank	9.42	8.85	5.28	8.14	9.14	4.00	5.28	6.85	2.14	<b>1.28</b>	2.57

The first two tables, (a) and (b), show the performance of all comparison partners (columns) on all datasets (rows) with respect to ACC and NMI<sub>g</sub>, respectively. These measures are obtained by comparing the results to the ground truth clustering labels. The last two tables, (c) and (d), show the scores for NMI<sub>c</sub> and JI which compare the clustering results to the given covariates. For (a) and (b), a higher score corresponds to a better performance, whereas for (c) and (d), a lower score corresponds to a better performance. “-” indicates methods that cannot be directly applied to Simu3. The last row in each table shows the average rank of all methods for a given metric. For each dataset (row), the score of the best performing method is highlighted in bold. In case of a tie, or if the score of a method is very close to the best score ( $\leq 0.01$ ), more than one score will be highlighted in bold



**Fig. 4** A comparison of the performance of the proposed algorithm KCC (black line) and its best performing comparison partner KDAC. KDAC requires the setting of a regularization parameter  $\lambda$ . For all datasets, several choices of  $\lambda$  ( $x$ -axis) are evaluated in terms of the KCC-normalized  $NMI_g$  score ( $y$ -axis). Three observations follow: (1) Non-optimal choices of  $\lambda$  lead to a low  $NMI_g$  score for KDAC. (2) The optimal  $\lambda$  (in bold) varies by multiple orders of magnitude between all datasets (summary in bottom right panel). (3) Among all datasets and all parameter choices, KCC outperforms KDAC in almost all instances

closely followed by KDAC. This can not only be seen by looking at the average ranking, but also when inspecting the results independently for each dataset. In Sect. 5.3.2 we investigate the difference between KCC and KDAC—the top two performers—more closely.

### 5.3.2 Comparison of KCC to KDAC

From the results of the experiments in Table 2, we observe that our proposed algorithm KCC gives on average the best performance in terms of recovering the ground truth and difference from a given covariate. Another method that gives on average high-quality results is KDAC, which requires the setting of a regularization parameter  $\lambda$ . We take a closer look at the difference in performance in terms of  $NMI_g$  and ACC. In Fig. 4, the KCC-normalized  $NMI_g$  score for KDAC is shown (for ACC we have similar results), varying the regularization

**Table 3** The results of the semi-supervised learning experiments are displayed

Dataset or method	LP	SP	ST	KCSSL
<i>ACC</i>				
Simu1	0.355 ± 0.07	0.601 ± 0.14	0.967 ± 0.02	<b>0.993 ± 0.01</b>
Simu2	0.507 ± 0.06	0.722 ± 0.18	0.868 ± 0.01	<b>0.985 ± 0.02</b>
Simu3	0.353 ± 0.08	0.663 ± 0.13	0.972 ± 0.01	<b>0.988 ± 0.01</b>
Face	0.251 ± 0.01	0.345 ± 0.02	0.655 ± 0.05	<b>0.811 ± 0.02</b>
Face4	0.248 ± 0.01	0.317 ± 0.03	0.445 ± 0.09	<b>0.581 ± 0.15</b>
Multi	0.247 ± 0.04	0.311 ± 0.09	0.579 ± 0.11	<b>0.735 ± 0.18</b>
DLBCL	0.369 ± 0.05	0.388 ± 0.05	–	<b>0.656 ± 0.07</b>
WebKB	0.534 ± 0.01	0.534 ± 0.01	<b>0.674 ± 0.03</b>	<b>0.668 ± 0.01</b>

We report mean and standard error of accuracy among 100 random splits of the data into 10% labeled and 90% unlabeled. Please note that the datasets are balanced. The best results are shown in bold. If the score of a method is very close to the best score ( $\leq 0.01$ ), then more than one score will be highlighted in bold

parameter  $\lambda$  on the  $x$ -axis. In these figures,  $\text{NMI}_g^{\text{KDAC}}/\text{NMI}_g^{\text{KCC}} = 1$  means that KCC and KDAC attain the same  $\text{NMI}_g$  score. If  $\text{NMI}_g^{\text{KDAC}}/\text{NMI}_g^{\text{KCC}} \leq 1$ , KCC outperforms KDAC, and similarly the other way around. The bottom right panel of Fig. 4 shows the optimal choice of  $\lambda$  for each dataset.

It is important to note the following:

1. Non-optimal choices of  $\lambda$  lead to a low  $\text{NMI}_g$  score for KDAC.
2. The optimal  $\lambda$  (in bold) varies by several orders of magnitude among all datasets. In the bottom right panel, which shows the best performing  $\lambda$  for each dataset, all orders of magnitude are present. (Values are in log scale.)
3. Among all datasets and all parameter choices, KCC outperforms KDAC in almost all instances. Among the 35 explored settings, there are only two where KDAC has a better  $\text{NMI}_g$  score than KCC.

We therefore conclude that unless there is prior knowledge about the regularization, KCC should be chosen over KDAC.

### 5.3.3 Semi-supervised learning results

In Table 3, the results from the semi-supervised learning experiments are shown. The rows correspond to the datasets, and the columns to the comparison partners, with the last column showing the proposed approach, KCSSL. We report the mean and standard error of the classification accuracy of the unlabeled samples with respect to different splittings of the dataset. As it was the case of the clustering results, the datasets are balanced and the best results are highlighted in bold.

With the exception of the WebKB dataset, KCSSL yields the best classification accuracy. This is expected because ours is the only method correcting for covariates. We conclude that to tackle a semi-supervised learning problem in the presence of covariates, KCSSL should be the method of choice.

## 6 Related work

### 6.1 Conditional clustering

The first approach to conditional clustering is found in the work of Gondek and Hofmann [2]. There, the authors present a general framework for exploring non-redundant clustering apart from known background information, which can be in the form of both categorical or continuous covariates. Their method, CCIB, is based on maximizing the conditional mutual information. However, to estimate the mutual information, CCIB needs to make assumptions about the distribution of the data.

Apart from CCIB, the conditional clustering problem can be approached and solved by alternative clustering methods. Alternative clustering approaches can be classified into two categories: simultaneous [7,9,32] and sequential. Sequential approaches solve an instance (categorical covariates) of the conditional clustering problem, while simultaneous approaches solve a slightly different problem. Early sequential works, i.e., COALA [3], OC [4], and ADFT [5], use constraints or orthogonal projections to enforce dissimilarity between clusterings. However, these methods have two major shortcomings: They (i) can only handle categorical covariates and (ii) cannot discover nonlinearly separable clusters.

Recently, sequential alternative clustering approaches have been proposed, such as the optimization in a regularization framework that combines a clustering quality and a dissimilarity function. For instance, mutual information [33] and entropy [25] have been proposed to be used as such functions. Other methods use the kernel dependence measure HSIC as a dissimilarity function [10–13]. Yet, all these methods require the setting of a hyper-parameter  $\lambda$ , which is difficult to estimate in practice.

The proposed KCC approach [16] addresses all the drawbacks of the existing methods mentioned above and outperforms them in the conducted experiments.

### 6.2 Semi-supervised learning

In the semi-supervised learning domain, there is a distinction between generative methods and graph-based methods [34]. In the former, a model is imposed on the data where parameters are fitted using labeled and unlabeled data. In the latter, a graph is generated from all the data which are then partitioned optimizing a loss function.

In this paper we mainly focus on graph-based methods and do not consider generative methods. However, an expectation–maximization approach to solving generative methods can be seen as an instance of (soft) *self-training* (ST): First, a classifier is trained on the labeled samples. Then, high-confidence unlabeled points are added to the training set. Both steps are repeated until no unlabeled samples are left. The convergence of ST depends highly on the base classifier and the quality of the data [28,35], yet it has been successfully implemented in areas such as natural language processing [36] or image recognition [37].

Graph-based algorithms are traditionally formulated in a regularization framework trading off correct classification of labeled points with smoothness on the whole graph. The first graph-based algorithm uses mincuts [38] to partition the data. Later, Zhu et al. [39] proposed Label Propagation (LP) which propagates node information to neighbors in a graph. The authors relaxed the discrete case to Gaussian random fields and harmonic functions based on the graph combinatorial Laplacian in [26], which can also be seen as a general case of mincut. In parallel, inspired by the work on spreading activation networks [40], Zhou et al. [27]



proposed spread propagation (SP) which is similar to LP but achieves lower classification error for fewer labeled points using the normalized Laplacian.

Our method KCSSL can be seen as in the line of research of graph-based methods. As shown in Sect. 4.4, obtaining the solution in KCSSL when no covariates are present is of the same form as obtaining the fixed point in LP.

Extensive research has undergone to learn a suitable graph embedding, also to improve the classification of graph-based semi-supervised learning methods (see Section 2.3 in [41]). Most of the methods were evaluated in the context of natural language processing.

Another popular approach to solve the semi-supervised learning problem is to entertain a transductive support vector machine (TSVM) (also known as semi-supervised SVM [42]) which can be viewed as an SVM with an additional term maximizing the margin also with respect to the unlabeled data. However, since finding a good approximation is still ongoing research [43], TSVMs' performance may be limited under some circumstances [44] and available software packages do not include multi-class classification options. We do not compare to TSVMs.

Most importantly, all of the mentioned algorithms, including the recent advances, do not correct for covariates, which results in inferior classification accuracy compared to KCSSL (see Sect. 5.3).

## 7 Conclusion

In this paper, we have presented two approaches to (i) conditional clustering and (ii) conditional semi-supervised learning. Both of the proposed methods are able to take into account known covariates, which would otherwise mask the existence of the underlying relationships of interest. For the well-studied case of alternative clustering, our method kernel conditional clustering (KCC) outperforms competing alternatives on synthetic and real-world data while simplifying the usage due to the need for fewer parameters. For the same datasets, in the case of semi-supervised learning, our method kernel conditional semi-supervised learning (KCSSL) outperforms competing alternatives, because it is the only method correcting for known covariates. Moreover, KCSSL is fully parameter-free. In general, we advise the adoption of KCC and KCSSL, which perform the best in our experiments.

KCC can achieve both clustering quality and dissimilarity requirements in one objective without an additional hyper-parameter. By using a kernel measure, KCC does not make any assumption about the data distribution, it can also handle continuous covariates, and it can discover nonlinearly separable clusters. Furthermore, to handle high-dimensional data, KCC integrates clustering with subspace learning, where the dependence between the clustering and a low-dimensional subspace is maximized, conditioned on the observed covariates.

Kernel conditional clustering and kernel conditional semi-supervised learning will serve as methods for a wide range of machine learning applications where known effects of covariates must be corrected for. For instance, future research can apply our method to a wider range of settings, such as sub-phenotype discovery in medical datasets. Another open challenge is to estimate the number of clusters for KCC, which we will explore in our future work. In short, we see exciting challenges for future work, but also great potential for the methods presented here to help in the discovery of structures conditionally independent from known covariates.

**Acknowledgements** The authors thank Prof. Zaiwen Wen and Prof. Wotao Yin for sharing the implementation of FOptM. This work has been supported by the SNSF Starting Grant "Significant Pattern Mining."

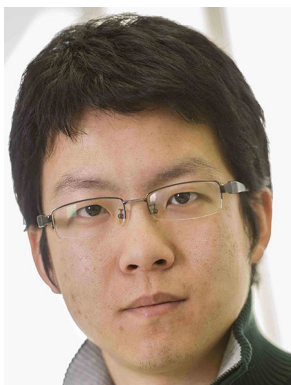
**Open Access** This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

## References

1. Aggarwal CC (2015) Data mining: the textbook. Springer, Berlin
2. Gondek D, Hofmann T (2004) Non-redundant data clustering. In: IEEE ICDM 2004, pp 75–82
3. Bae E, Bailey J (2006) Coala: a novel approach for the extraction of an alternate clustering of high quality and high dissimilarity. In: IEEE ICDM 2006, pp 53–62
4. Cui Y, Fern XZ, Dy JG (2007) Non-redundant multi-view clustering via orthogonalization. In: IEEE ICDM 2007, pp 133–142
5. Davidson I, Qi Z (2008) Finding alternative clusterings using constraints. In: IEEE ICDM 2008, pp 773–778
6. Qi Z, Davidson I (2009) A principled and flexible framework for finding alternative clusterings. In: KDD. ACM, pp 717–726
7. Dang XH, Bailey J (2010) Generation of alternative clusterings using the cami approach. In: Proceedings of the 2010 SIAM international conference on data mining. SIAM, pp 118–129
8. Gretton A, Bousquet O, Smola A, Schölkopf B (2005) Measuring statistical dependence with Hilbert–Schmidt norms. In: ALT. Springer, pp 63–77
9. Niu D, Dy JG, Jordan MI (2010) Multiple non-redundant spectral clustering views. In: ICML, pp 831–838
10. Dang XH, Bailey J (2014) Generating multiple alternative clusterings via globally optimal subspaces. *Data Min Knowl Discov* 28(3):569–592
11. Niu D, Dy JG, Jordan MI (2014) Iterative discovery of multiple alternative clustering views. *IEEE Trans Pattern Anal Mach Intell* 36(7):1340–1353
12. Yang S, Zhang L (2017) Non-redundant multiple clustering by nonnegative matrix factorization. *Mach Learn* 106(5):695–712
13. He X, Li L, Roqueiro D, Borgwardt K (2017) Multi-view spectral clustering on conflicting views. In: Joint European conference on machine learning and knowledge discovery in databases. Springer, pp 826–842
14. Fukumizu K, Gretton A, Sun X, Schölkopf B (2007) Kernel measures of conditional dependence. In: NIPS, vol 20, pp 489–496
15. Sun X, Janzing D, Schölkopf B, Fukumizu K (2007) A kernel-based causal learning algorithm. In: ICML. ACM, pp 855–862
16. He X, Gumbsch T, Roqueiro D, Borgwardt K (Nov 2017) Kernel conditional clustering. In: IEEE international conference on data mining (ICDM), pp 157–166
17. Bach FR, Jordan MI (2002) Kernel independent component analysis. *J Mach Learn Res* 3(Jul):1–48
18. Baker CR (1973) Joint measures and cross-covariance operators. *Trans Am Math Soc* 186:273–289
19. Murphy KP (2012) Machine learning: a probabilistic perspective. MIT Press, Cambridge
20. Ng AY, Jordan MI, Weiss Y et al (2001) On spectral clustering: analysis and an algorithm. In: NIPS, vol 14, no 2, pp 849–856
21. Bhatia R (1997) Matrix analysis. Springer, New York
22. Fukumizu K, Bach FR, Jordan MI (2009) Kernel dimension reduction in regression. *Ann Stat* 37(4):1871–1905
23. Wang M, Sha F, Jordan MI (2010) Unsupervised kernel dimension reduction. In: NIPS, pp 2379–2387
24. Wen Z, Yin W (2013) A feasible method for optimization with orthogonality constraints. *Math Program* 142(1–2):397–434
25. Vinh NX, Epps J (2010) Mincentropy: a novel information theoretic approach for the generation of alternative clusterings. In: IEEE ICDM 2010, pp 521–530
26. Zhu X, Ghahramani Z, Lafferty JD (2003) Semi-supervised learning using Gaussian fields and harmonic functions. In: Proceedings of the 20th international conference on machine learning (ICML-03), pp 912–919
27. Zhou D, Bousquet O, Lal TN, Weston J, Schölkopf B (2004) Learning with local and global consistency. In: Advances in neural information processing systems, pp 321–328
28. Culp M, Michailidis G (2008) An iterative algorithm for extending learners to a semi-supervised setting. *J Comput Graph Stat* 17(3):545–571
29. Blake CL (1998) UCI repository of machine learning databases. University of California, Irvine. <http://www.ics.uci.edu/~mllearn/MLRepository.html>

30. Hoshida Y, Brunet J-P, Tamayo P, Golub TR, Mesirov JP (2007) Subclass mapping: identifying common subtypes in independent disease data sets. *PLoS ONE* 2(11):e1195
31. Craven M, Slattery S (2001) Relational learning with statistical predicate invention: better models for hypertext. *Mach Learn* 43(1):97–119
32. Jain P, Meka R, Dhillon IS (2008) Simultaneous unsupervised learning of disparate clusterings. *Stat Anal Data Min* 1(3):195–210
33. Dang X-H, Bailey J (2010) A hierarchical information theoretic technique for the discovery of non linear alternative clusterings. In: *KDD*. ACM, pp 573–582
34. Zhu X (2005) Semi-supervised learning literature survey. Computer Sciences, University of Wisconsin-Madison, Technical report 1530
35. Haffari GR, Sarkar A (2012) Analysis of semi-supervised learning with the Yarowsky algorithm. arXiv preprint [arXiv:1206.5240](https://arxiv.org/abs/1206.5240)
36. Yarowsky D (1995) Unsupervised word sense disambiguation rivaling supervised methods. In: *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics, pp 189–196
37. Rosenberg C, Hebert M, Schneiderman H (2005) Semi-supervised self-training of object detection models. In: *Proceedings of the seventh IEEE workshops on application of computer vision (WACV/MOTION'05)*, vol 1. IEEE Computer Society, Washington, DC, USA, pp 29–36. <https://doi.org/10.1109/ACVMOT.2005.107>
38. Blum A, Chawla S (2001) Learning from labeled and unlabeled data using graph mincuts. In: Brodley CE, Danyluk AP (eds) *Proceedings of the eighteenth international conference on machine learning (ICML '01)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp 19–26
39. Zhu X, Ghahramani Z (2002) Learning from labeled and unlabeled data with label propagation. CMU CALD tech report CMU-CALD-02-107
40. Shrager J, Hogg T, Huberman BA (1987) Observation of phase transitions in spreading activation networks. *Science* 236(4805):1092–1094
41. Yang Z, Cohen WW, Salakhutdinov R (2016) Revisiting semi-supervised learning with graph embeddings. arXiv preprint [arXiv:1603.08861](https://arxiv.org/abs/1603.08861)
42. Bennett KP, Demiriz A (1999) Semi-supervised support vector machines. In: *Advances in neural information processing systems*, pp 368–374
43. Gieseke F, Airola A, Pahikkala T, Kramer O (2014) Fast and simple gradient-based optimization for semi-supervised support vector machines. *Neurocomputing* 123:23–32
44. Zhang T, Oles F (2000) The value of unlabeled data for classification problems. In: Langley P (ed) *Proceedings of the seventeenth international conference on machine learning*, pp 1191–1198

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



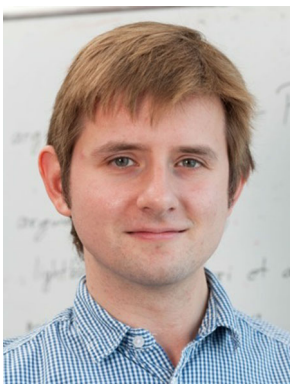
**Xiao He** is a postdoctoral researcher at the Machine Learning and Computational Biology Lab at ETH Zürich. He received the B.Sc. and M.Sc. degrees in electronic engineering from Xidian University, China, in 2007 and 2010, respectively, and a Ph.D. degree from the Institute of Computer Science, University of Munich, Germany. His research interests lie in data mining, machine learning and optimization, especially clustering, kernel methods, graph mining, and their application in biomedical problem.



**Thomas Gumbsch** received his B.Sc. and M.Sc. degrees in Physics from ETH Zürich. He is currently a Ph.D. student in the Machine Learning and Computational Biology Lab at ETH Zürich.



**Damian Roqueiro** is a Senior Researcher at the Machine Learning and Computational Biology Lab at ETH Zürich. His undergraduate work was in Information Systems Engineering, followed by an M.Sc. degree in Computer Science and a Ph.D. degree in Bioinformatics. His research interests center around the development of probabilistic models and machine learning methods to better understand the association between specific diseases and the genetic markup of individuals afflicted by those diseases.



**Karsten Borgwardt** is Full Professor of Data Mining at ETH Zürich, in the Department of Biosystems located in Basel. His work won several awards, including the NIPS 2009 Outstanding Student Paper Award, the Krupp Award for Young Professors in 2013, and a Starting Grant 2014 from the ERC-Backup Scheme of the Swiss National Science Foundation. From 2013 to 2016, he headed the Marie Curie Initial Training Network for “Machine Learning for Personalized Medicine” with 12 partner laboratories in 8 countries ([www.mlpm.eu](http://www.mlpm.eu)). The business magazine “Capital” listed him as one of the “Top 40 under 40” in Science from Germany in 2014, 2015, and 2016.