LEARNING WITH CLUSTERING PENALTIES

VINCENT ROULET, FAJWEL FOGEL, ALEXANDRE D’ASPREMONT, AND FRANCIS BACH

ABSTRACT. We study supervised learning problems using clustering penalties to impose structure on either features, tasks or samples, seeking to help both prediction and interpretation. This arises naturally in problems involving dimensionality reduction, transfer learning or regression clustering. We derive a unified optimization formulation handling these three settings and produce algorithms whose core iteration complexity amounts to a k-means clustering step, which can be approximated efficiently. We test the robustness of our methods on artificial data sets as well as real data extracted from movie reviews and a corpus of text documents.

1. INTRODUCTION

To improve generalization, supervised learning problems use regularization penalties on the prediction functions. Statistical guarantees or prior information on the data are often used to design these penalties for the task at hand. Here, we study clustering penalties that seek to group either features, tasks or samples to both improve prediction and provide additional structural insights on the data.

When there exist some groups of highly correlated features for instance, reducing dimensionality by assigning uniform weights inside each distinct groups of features can be beneficial both in terms of prediction and interpretation [5]. This often occurs in text classification for example, where it is natural to group together words having the same meaning for a given task [7, 13].

Similarly, in the classical multi-task setting, clustering predictors of related tasks often improves prediction performance. In computer vision for instance, classifiers associated with different species of cats should be quite similar, but well separated from classifiers associated with cars. This problem has been well studied in the transfer and multi-task learning literature [1, 6, 11] using rank minimization penalties for example. Here, we study the impact of imposing clustering penalties.

Finally, in some settings, it can be valuable to cluster sample points, with each cluster of samples having its own distinct prediction function [10, 16]. This problem has applications in the context of privacy learning, where the group information of an individual may not be revealed because of confidentiality issues.

We present a unified and flexible framework to enforce cluster structure on either features, tasks or samples. We directly formulate supervised learning problems with clustering penalties as an optimization problem on the clustered predictors, where clustering is either enforced through a hard grouping constraint or a soft regularization penalty, and losses are tailored to the application at hand (classification or regression, clustering tasks, features or sample points).

We propose several optimization schemes to solve these problems efficiently. While the original optimization problem is non-convex, we show that the core non-convexity is concentrated in a clustering subproblem akin to k-means, which we solve using kmeans++, a classical approximation technique introduced in [2]. In the particular case of feature clustering for regression, the k-means steps are performed in dimension one, and can therefore be solved exactly by dynamic programming [3, 15].

We also formulate an explicit convex relaxation for the general case which can be solved efficiently using the conditional gradient algorithm [9, 12], where the core inner step also amounts to solving a clustering...
problem. The use of k-means steps makes our approach fast and scalable. We describe experiments on both synthetic and real datasets involving large corpora of text from e.g. movie reviews and 20NewsGroup datasets where our method compares very favorably with standard benchmarks while providing meaningful insights on the data structure.

2. LEARNING & CLUSTERING OF FEATURES, TASKS OR SAMPLES

We assume that we are given \( K \) regression or classification tasks that map \( n \) real observations \( x_i \), lying in a \( d \)-dimensional space, to real or nominal labels \( y_i \).

2.1. General formulation. Given \( n \) sample points represented by the matrix \( X = (x_1, \ldots, x_n)^T \in \mathbb{R}^{n \times d} \) and their real or nominal labels \( y = (y_1, \ldots, y_n) \) depending on the task (classification or regression), we seek to compute linear predictors represented in a matrix \( W \) whose dimensions depend on the setting. Our learning problems with clustering penalties can informally be written

\[
\text{minimize } \text{Loss}(y, X, W) + R(W) + \Omega(W),
\]

in the matrix \( W \) of classifier vectors, where \( \text{Loss}(y, X, W) \) is a learning loss (for simplicity, we consider only square or logistic losses in what follows), \( R(W) \) is a classical regularizer and \( \Omega(W) \) a clustering score to partition features, tasks or sample points into \( Q \) groups.

To define a partition of \( m \) items in \( Q \) groups \( C_1, \ldots, C_Q \) of size \( (s_1, \ldots, s_Q) \), we write \( Z \in \{0, 1\}^{m \times Q} \) the assignment matrix such that \( Z_{iq} = 1 \) if item \( i \) is in group \( C_q \) and \( Z_{iq} = 0 \) otherwise. The clustering score takes two forms which we now describe.

2.1.1. Hard clustering penalty. Here, we impose that all features, tasks or samples in a cluster are represented by a unique predictor to solve the supervised learning problem. In our settings this will constraint the matrix \( W \) to have only \( Q \) different columns. This is captured using an assignment matrix \( Z \) and a matrix \( V \) containing these different predictors, leading to the supervised problem with “hard” clustering penalty

\[
\text{minimize } \text{Loss}(y, X, W) + R(W) \quad \text{subject to } \quad W = VZ^T, \quad Z \in \{0, 1\}^{m \times Q}, \quad Z1 = 1,
\]

where \( m = d, n \) or \( K \) and the dimension of the variables \( W \) and \( V \) depend on the setting. The clustering score \( \Omega \) is then the indicator function of this set of matrices \( W \).

2.1.2. Soft clustering penalty. Here instead, each feature, task or sample has its own predictor but the clustering penalty enforces that these predictors be clustered around a few centroids. In our setting, we thus impose that the matrix of predictors \( W \) be close to an auxiliary matrix \( \tilde{W} \) with only a few different columns, using a penalty \( \Omega \) encouraging clustering. Using \( Z \) and \( V \) variables as before, this leads to the following supervised problem with “soft” clustering penalty

\[
\text{minimize } \text{Loss}(y, X, W) + R(W) + \Omega(W, \tilde{W}) \quad \text{subject to } \quad \tilde{W} = VZ^T, \quad Z \in \{0, 1\}^{m \times Q}, \quad Z1 = 1.
\]

Here again \( m \) and the ambient space dimension in \( W \) depend on the setting.

2.1.3. Non-convexity. Although these formulations are non-convex, we observe that the core non-convexity emerges from a clustering problem on the predictors \( W \), which we can deal with k-means approximations, as detailed in section 3.

We now present in more detail the three key applications associated with each dimension of the data cube: clustering features, tasks or sample points. Our motivation is to highlight their common structure and develop efficient algorithms to solve them.
2.2. Dimensionality reduction: clustering features. Given a regression or classification task, we want to reduce dimensionality by grouping together features which have a similar influence on the output. In practice, this amounts to a loss supervised quantization of the classifier vector. We present the linear regression case [5], which can be extended to classification.

We search \( Q \) groups of features \( C_1, \ldots, C_Q \) such that all predictor coefficients within a group are identical to a scalar \( v_q \). This can be written in terms of “hard” clustering penalty on the prediction vector \( w \) leading to the following problem

\[
\text{minimize } \frac{1}{n} \sum_{i=1}^{n} \text{loss} (y_i, wx_i) + \lambda \|w\|_2^2 \\
\text{s.t. } w = vZ^T, \quad Z \in \{0,1\}^{d \times Q}, \quad Z1 = 1
\]

in the variables \( w \in \mathbb{R}^{1 \times d} \), \( v \in \mathbb{R}^{1 \times Q} \) and \( Z \). Here \( \text{loss}(y_i, wx_i) \) measures the quality of prediction for each sample and we used a \( \ell_2 \) norm regularization, \( R(w) = \lambda \|w\|_2^2 \). This problem imposes that \( w \) has only \( Q \) different coefficient values. The prediction function can then be written \( f(x) = \sum_{q=1}^{Q} v_q \left( \sum_{j \in C_q} x^{(j)} \right) \), quantized over \( Q \) values \( \{v_1, \ldots, v_Q\} \), where \( x^{(j)} \) denotes the \( j \)th coordinate of \( x \).

2.3. Transfer learning: clustering tasks. Given a set of \( K \) supervised tasks like regression or binary classification, transfer learning aims at jointly solving these tasks, hoping that each task can benefit from the information given by other tasks. For simplicity, we illustrate the case of multi-category classification, which can be extended to the general multitask setting. When performing classification with one-versus-all majority vote, we train one binary classifier for each class vs. all others. Using a regularizing penalty such as the squared \( \ell_2 \) norm, the problem of multitask learning can be cast as

\[
\text{minimize } \frac{1}{n} \sum_{k=1}^{K} \sum_{i=1}^{n} \text{loss} (y_i^k, w_k^T x_i) + \lambda \sum_{i=1}^{K} \|w_k\|_2^2.
\]

in the matrix variable \( W = (w_1, \ldots, w_K) \in \mathbb{R}^{d \times K} \) of classifier vectors (one column per task). We write \( \text{Loss}(y, X, W) \) and \( R(W) \) the first and second term of this problem. Various strategies are used to leverage the information coming from related tasks, such as low rank [1] or structured norm penalties [6] on the matrix of classifiers \( W \). Here we follow the clustered multitask setting introduced in [11]. Namely we add a penalty \( \Omega \) on the classifiers \( (w_1, \ldots, w_K) \) which enforce them to be clustered in \( Q \) groups \( C_1, \ldots, C_Q \) around centroids \( V = (v_1, \ldots, v_Q) \in \mathbb{R}^{d \times Q} \). This penalty can be decomposed in

- A measure of the norm of the barycenter of centers \( \bar{v} = \frac{1}{n} \sum_{q=1}^{Q} s_q v_q \)
  \[ \Omega_{\text{mean}}(V) = \lambda \frac{n}{2} \|\bar{v}\|_2^2 \]

- A measure of the variance between clusters
  \[ \Omega_{\text{between}}(V) = \lambda \frac{2}{Q} \sum_{q=1}^{Q} s_q \|v_q - \bar{v}\|_2^2 \]

- A measure of the variance within clusters
  \[ \Omega_{\text{within}}(W, V) = \lambda \frac{2}{Q} \sum_{q=1}^{Q} \sum_{i \in C_q} \|w_i - v_q\|_2^2 \]

The total penalty \( \Omega(W, V) = \Omega_{\text{mean}}(V) + \Omega_{\text{between}}(V) + \Omega_{\text{within}}(W, V) \) is illustrated in Figure 1.
The projection of a matrix \( V \) is made by finding \( \arg\min_{Z,C} kV - C k^2_F = \arg\min_{QX} q=1 X_i \in C_q k v_i - c_q k^2_2, \)

\[ \text{Figure 1. Decomposed clustering penalty on } m \text{ items.} \]

The clustered multitask learning problem can then be written using the appropriate assignment matrix \( Z \) (see appendix for detailed computations, for simplicity we use here the same notation \( \Omega \)),

\[
\text{minimize } \text{Loss}(y, X, W) + R(W) + \Omega(W, W)
\]

\[
s.t. \ W = V Z^T, \quad Z \in \{0, 1\}^{K \times Q}, \quad Z1 = 1,
\]

in variables \( W \in \mathbb{R}^{d \times K}, \tilde{W} \in \mathbb{R}^{d \times K}, V \in \mathbb{R}^{d \times Q} \) and \( Z \).

2.4. **Regression clustering: clustering sample points.** Imagine for instance that you want to predict the survival rate for a disease which depends on the presence of a given cell but you only have low level information on the patients like their blood pressure or their heart beat. Depending on the presence of the cell, the survival rates according to the low level information will highly change but the presence itself cannot be detected through these low level information. We can use survival rates to simultaneously retrieve the groups of patients and the associated regression functions.

This setting was already studied by Zhang [16] for general predictors, it is also related to subspace clustering [8], however here we already know in which dimension the data points lie. Note also that our setting is different from a mixture of experts model in the sense that the latent cluster assignment variable \( Z \) cannot be deduced from the input features \( X \) and can only be estimated once \( y \) is known (cf. figure 2).

\[ \text{Figure 2. Learning multiple diverse predictors (left), mixture of experts model (right).} \]

More precisely, given a regression or multi-classification task, we want to find \( Q \) groups \( C_1, \ldots, C_Q \) of sample points of size \( (s_1, \ldots, s_Q) \) to maximize the within-group prediction performance. We thus learn \( Q \) predictors \( (v_1, \ldots, v_Q) \), each predictor having low error rate on some group of points. We minimize the loss incurred by the best linear predictor \( v_q \) for each point regularized by a balance penalty, \( i.e. \)

\[
\text{minimize } \frac{1}{n} \sum_{i=1}^n \min_{q \in \{1, \ldots, Q\}} \text{loss}(y_i, v_q^T x_i) + \lambda \sum_{q=1}^Q s_q \|v_q\|_2^2
\]
in the matrix variable $V \in \mathbb{R}^{d \times Q}$ of predictor vectors (one per sample point). Using an appropriate assignment matrix $Z$ this problem can be written

$$\minimize \frac{1}{n} \sum_{i=1}^{n} \text{loss} \left( y_i, \sum_{q=1}^{Q} Z_{iq} v_q^T x_i \right) + \lambda \sum_{q=1}^{Q} s_q \| v_q \|_2^2$$

in variables $V$ and $Z$. Now using the variable $W = VZ^T$ and defining $\text{Loss}(y, X, W) = \frac{1}{n} \sum_{i=1}^{n} \text{loss} \left( y_i, w_i^T x_i \right)$ and $R(W) = \lambda \| W \|_F^2$, it is equivalent to

$$\minimize \text{Loss}(y, X, W) + R(W)$$

s.t. $W = VZ^T$, $Z \in \{0, 1\}^{n \times Q}$, $Z1 = 1$, in variables $W \in \mathbb{R}^{d \times n}$, $V \in \mathbb{R}^{d \times Q}$ and $Z$.

All the problems described above can be formulated in a single unified template. In what follows we will study efficient algorithms to solve them.

3. Algorithms

We now present optimization strategies to solve these supervised clustering problems. We begin by simple greedy procedures, then propose a non-convex projected gradient descent scheme and finally a more refined convex relaxation solved using approximate conditional gradient. This latter is exact in the case of dimensionality reduction.

3.1. Greedy algorithms. A straightforward strategy is to first train predictors as in a classical supervised learning problem, and then cluster predictors together using k-means. The procedure can be repeated in the case of a soft clustering penalty. In the same spirit, when clustering sample points, one can alternate minimization on the predictors of each group and assignment of each point to the group where its loss is smallest. These methods are fast but unstable and highly dependent on the initialization. However, these alternating minimizations can be used to refine the solution of the more robust algorithms proposed below.

3.2. Projected gradient descent. We observe that the non-convex set of matrices $\{W, W = VZ^T\}$ is a union of subspaces similar to the set of sparse vectors in compressed sensing. Therefore we can adopt a similar strategy as the Iterative Hard Thresholding algorithm [4], i.e., do a projected gradient on this set. Projecting a matrix $V$ on it means indeed finding

$$\argmin_{Z, V} \| W - VZ^T \|_F^2 = \argmin \sum_{q=1}^{Q} \sum_{i \in C_q} \| w_i - v_q \|_2^2,$$

where the minimum is taken over centroids $v_q$ and partitions $(C_1, \ldots, C_Q)$. Crucially here, this means that the projection problem is a classical clustering problem. It can directly be solved approximately with the k-means++ algorithm which performs alternate minimization on the assignments and the centroids. Although it is a non-convex problem, k-means++ gives approximation bounds on its solution [2].

Writing k-means++$(V, Q)$ the approximate solution of the projection and $\phi$ the objective function, using a backtracking line search for the stepsize $\alpha_t$, the full methods is summarized as Algorithms 1 and 2.
Algorithm 1 Proj. Gradient Descent (HC)

**Input:** $X, y, Q, \epsilon, \lambda$

Initialize $V_0 = 0$

while $|\phi(W_t) - \phi(W_{t-1})| \geq \epsilon$ do

$W_{t+\frac{1}{2}} = W_t - \alpha_t (\nabla \text{Loss}(y, X, W_t) + \nabla \Omega(W_t))$

$[Z_{t+1}, V_{t+1}] = \text{k-means++}(W_{t+\frac{1}{2}}, Q)$

$W_{t+1} = V_{t+1} Z_{t+1}^T$

end while

$Z^*$ and $V^*$ are given through Kmeans++

**Output:** $W^*, Z^*, V^*$

Algorithm 2 Proj. Gradient Descent (SC)

**Input:** $X, y, Q, \epsilon, \lambda, \mu, \lambda_w, \lambda_m, \lambda$

Initialize $W_0 = V_0 = 0$

while $|\phi(W_t, \tilde{W}_t) - \phi(W_{t-1}, \tilde{W}_{t-1})| \geq \epsilon$ do

$W_{t+1} = W_t - \alpha_t (\nabla \text{Loss}(y, X, W_t) + \nabla \Omega(W_t) + \nabla \Omega_C(W_t, \tilde{W}_t))$

$\tilde{W}_{t+\frac{1}{2}} = \tilde{W}_t - \alpha_t \nabla \Omega_C(W_t, \tilde{W}_t)$

$[Z_{t+1}, V_{t+1}] = \text{k-means++}(\tilde{W}_{t+\frac{1}{2}}, Q)$

$\tilde{W}_{t+1} = V_{t+1} Z_{t+1}^T$

end while

$Z^*$ and $V^*$ are given through Kmeans++

**Output:** $W^*, \tilde{W}^*, Z^*, V^*$

3.3. Convex relaxation using conditional gradient algorithm. Another algorithmic approach to the supervised clustering problem is to minimize with respect to the assignment matrix $Z$ using the Frank-Wolfe algorithm (a.k.a. conditional gradient, [9, 12]). Considering the squared loss $\frac{1}{2}(y - f(x))^2$, we use the analytic form of the minimization in $V$ or $(W, V)$ to rewrite the problem. The clustering is then captured in terms of the equivalence matrix $M = Z(Z^T Z)^{-1}Z^T$, which satisfies $M_{ij} = 1/|C_q|$ if item $i$ and $j$ are in the same cluster $q$ and $M_{ij} = 0$ otherwise.

We describe here the simple case of supervised clustering of features in a regression task, introduced in §2.2, where we optimize over the coefficients associated with each feature and the k-means step can be performed exactly using dynamic programming [3, 15]. Detailed computations and explicit procedures for all other settings are given in the appendix. In this setting $W = V Z^T$ and the loss can be written here

$$\text{Loss}(y, X, VZ^T) = \frac{1}{2n} \sum_{i=1}^{n} \left( y_i - \sum_{j=1}^{d} \sum_{q=1}^{Q} Z_{jq} v_q x_i^j \right)^2,$$

where $V = [v_1, \ldots, v_Q] \in \mathbb{R}^{1 \times Q}$. Hence the regularized loss becomes

$$\phi(V, Z) = \frac{1}{2n} \sum_{i=1}^{n} \left( y_i - VZ^T x_i \right)^2 + \frac{\mu}{2} \|VZ^T\|_2^2$$

$$= \frac{1}{2n} \text{Tr}(y^T y) + \frac{1}{2n} \text{Tr}(VZ^T X^T X Z V^T)$$

$$- \frac{1}{n} \text{Tr}(VZ^T X^T y) + \frac{\mu}{2} \text{Tr}(VZ^T Z V^T).$$
Minimizing in $V$ and using the Sherman-Woodbury-Morrison formula we get

$$G(M) := \min_V \phi(V, Z)$$

$$= \frac{1}{2n} y^T (I - XZ(Z^T XZ + \mu n Z^T Z)^{-1} Z^T X) y$$

$$= \frac{1}{2n} \operatorname{Tr} \left( y y^T \left( I + \frac{1}{n \mu} XMX^T \right)^{-1} \right),$$

which is now expressed as a convex function of the equivalence matrix $M = Z(Z^T Z)^{-1} Z^T$ for $Z$ an assignment matrix. These are projection matrices such that $M^2 = M$ with discrete values. We define $\mathcal{M}$ as the set of equivalence matrices of this form and each iteration of the conditional gradient method requires solving an affine minimization subproblem over $\operatorname{hull}(\mathcal{M})$. Crucially here, the set $\mathcal{M}$ being discrete, we have for fixed matrix $M$,

$$\argmin_{N \in \operatorname{hull}(\mathcal{M})} \operatorname{Tr}(N \nabla G(M)) = \argmin_{N \in \mathcal{M}} \operatorname{Tr}(N \nabla G(M)).$$

Therefore we do not need to explicitly define or form $\operatorname{hull}(\mathcal{M})$. Writing $P = -\nabla G(M)$, denoting $H = I + \frac{1}{n \mu} XMX^T$, we get

$$P = \frac{1}{2n^2 \mu} X^T H^{-1} y y^T H^{-1} X$$

which is always semidefinite positive. Writing $P^\frac{1}{2}$ the matrix square root of $P$ we have, using that $I - N$ is a projection matrix for $N \in \mathcal{M}$,

$$\argmin_{N \in \mathcal{M}} \operatorname{Tr}(N \nabla G(M)) = \argmin_{N \in \mathcal{M}} - \operatorname{Tr}(NP^\frac{1}{2} P^\frac{1}{2}^T)$$

$$= \argmin_{N \in \mathcal{M}} \operatorname{Tr}((I - N)P^\frac{1}{2} P^\frac{1}{2}^T)$$

$$= \argmin_{N \in \mathcal{M}} \|P^\frac{1}{2} - NP^\frac{1}{2}\|_F^2$$

$$= \argmin_{Z} \min_{C} \|P^\frac{1}{2} - ZC^T\|_F^2,$$

where we recognize a clustering problem as above. In fact, in this particular case, the k-means subproblem is one-dimensional and can be solved exactly using dynamic programming [3, 15]. In general for the other settings, this step is only approximated by K-means. The entire method is summarized in Algorithm 3 where we use the classical stepsize for conditional gradient $\alpha_k = \frac{1}{k^2}$ and Frank-Wolfe rounding.

**Algorithm 3** Conditional gradient on the equivalence matrix

**Input:** $X, y, Q, \epsilon$

Initialize $M_0 \in \mathcal{M}$

while $|G(M_k) - G(M_{k-1})| \geq \epsilon$ do

- Compute the matrix square root $P^\frac{1}{2}$ of $-\nabla G(M_k)$
- Get oracle $\Delta_k = \text{k-means++}(P^\frac{1}{2}, Q)$

$M_{k+1} = M_k + \alpha_k (\Delta_k - M_k)$

end while

Use Frank Wolfe rounding to get a solution $M^* \in \mathcal{M}$

$Z^*$ is given by k-means

$V^*$ is given by the analytic solution of the minimization for $Z^*$ fixed

**Output:** $V^*, Z^*$

7
3.4. **Complexity.** The core complexity of Algorithms 1 and 2 is concentrated in the inner k-means sub-problem, which standard alternating minimization approximates at cost $O(tQS)$, where $t$ is the number of alternating steps, $Q$ is the number of clusters, and $S$ is the product of the dimensions of $V$. Using a proper conditioning of the gradient, the number of iterations before convergence is typically below 100, which makes Algorithms 1 and 2 both fast and scalable. For Algorithm 3, we also need to compute a matrix square root of the gradient at each iteration, which can slow down computations for large datasets. The choice of the number of clusters can be done given an a priori on the problem (e.g., if we know the number of hidden groups in the sample points), or cross-validation, idem for the other regularization parameters.

4. **Numerical Experiments**

We now test our methods first on artificial datasets to state their robustness to noisy data in the case of clustering samples or to the number samples in the dimensionality reduction case. Then we compare them for real data extracted from the 20NewsGroup dataset for multitask learning and from movie reviews in the dimensionality reduction case.

4.1. **Synthetic dataset.**

4.1.1. **Clustering penalty on sample points.** We generate $n$ data points $(x_i, y_i)$ for $i = 1, \ldots, n$ with $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$, divided in two clusters corresponding to regression tasks with weight vectors $v_1$ and $v_2$. Regression labels for points $x_i$ in cluster $q$ are given by $y_i = v_q^T x_i + \eta$, where $\eta \sim \mathcal{N}(0, \sigma^2)$. We test the robustness of the algorithms to the addition of noisy dimensions by completing $x_i$ with $d_n$ dimensions of noise $\eta_d \sim \mathcal{N}(0, \sigma_d)$. We use the loss defined above to compare the algorithms for testing.

The results are reported in Table 1 where the intrinsic dimension is 10 and “oracle” refers to the least-squares fit given the true assignments, which can be seen as the best achievable error rate. On the algorithmic side, AM refers to alternate minimization, PG refers to projected gradient, CG refers to conditional gradient on $M$ and RC to regression clustering as proposed by Zhang [16]. We implemented it with $q = 2$ i.e. using the Harmonic K-means formulation. PG, CG and RC were followed by AM refinement. 200 points were used for training, 200 for testing and the regularization parameter was set to $\lambda = 10^{-2}$ for all experiments. Noise on labels and added dimensions are $\sigma_y = \sigma_d = 2 \times 10^{-1}$. Results were averaged over 100 experiments with figures after the ± sign correspond to one standard deviation.

|               | $d_n = 0$ | $d_n = 100$ |
|---------------|-----------|-------------|
| Oracle        | $4.2 \pm 0.5$ | $29.5 \pm 7.0$ |
| AM            | 67.4 ± 102.9 | 518.7 ± 322.6 |
| PG            | 50.0 ± 52.1 | $\textbf{486.1} \pm 302.5$ |
| CG            | 64.7 ± 77.5 | 515.5 ± 334.4 |
| RC            | $\textbf{43.0} \pm 52.7$ | 639.6 ± 405.3 |
|               | $d_n = 300$ | $d_n = 600$ |
| Oracle        | $17.3 \pm 6.2$ | $\textbf{46.6} \pm 17.0$ |
| AM            | 190.2 ± 154.3 | 257.3 ± 160.2 |
| PG            | $\textbf{129.6} \pm 106.8$ | $\textbf{154.3} \pm 114.0$ |
| CG            | 139.3 ± 129.3 | 173.7 ± 136.6 |
| RC            | 168.6 ± 141.7 | 199.2 ± 131.8 |

**Table 1.** Test of the robustness of algorithms to additional dimension of noise in the case of regression clustering. PG refers to projected gradient, CG refers to conditional gradient on $M$ and RC to regression clustering.

We observe that PG achieves the best results when adding noise. CG gives good results, improving on the naive alternating minimization scheme and RC. However in the absence of noise RC gives the best results.
In table 2 we compare the CPU time needed for RC and our algorithm PG. Ø means that the algorithm didn’t converge after more than 6 hours. We observe that our algorithm can deal with large data in reasonable time in comparison to RC.

|   | PG | RC |
|---|----|----|
| $d = 5, n = 50$ | 0.12 | 0.05 |
| $d = 50, n = 500$ | 1.88 | 7.23 |
| $d = 200, n = 2000$ | 119 | 705 |
| $d = 500, n = 5000$ | 2480 | Ø |

Table 2. CPU time in seconds to solve cluster samples, versus problem size. PG refers to projected gradient and RC to regression clustering.

4.1.2. Clustering penalty on features. We generate $n$ data points $(x_i, y_i)$ for $i = 1, \ldots, n$ with $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$. Regression weights $w$ have only 10 different values $v_q$ for $q = 1, \ldots, 10$, uniformly distributed around 0. Regression labels are given by $y_i = w^T x_i + \eta$, where $\eta \sim \mathcal{N}(0, \sigma^2)$. We test the robustness of the algorithms to the number of learning examples, i.e. the size of the training set $n$, and measure the $l_2$ norm of the difference between the true vector of weights and the estimated ones.

In Table 3, we compare the proposed algorithms to OSCAR [5], Ridge, Lasso and Ridge followed by k-means on the weights (using associated centroids as predictors). “Oracle” refers to the mean square fit given the true assignments of features and can be seen as the best achievable error rate. Here too, PG refers to projected gradient (initialized with the solution of Ridge followed by k-means), CG refers to conditional gradient on $M$ followed by PG. Noise on labels is set to $\sigma = 10^{-1}$. Algorithm parameters were all cross-validated using a logarithmic grid. Results (multiplied by 100 to shorten the table) were averaged over 30 experiments and figures after the ± sign correspond to one standard deviation.

|   | n=50 | n=75 | n=100 |
|---|------|------|-------|
| Oracle | 3.3±1.2 | 2.6±0.9 | 2.2±0.5 |
| Ridge | 69.9±0.5 | 58.0±0.8 | 33.0±5.8 |
| Ridge+Kmeans | 72.1±1.6 | 58.6±2.7 | 46.9±20.6 |
| Lasso | 91.2±0.6 | 81.9±4.2 | 56.1±30.5 |
| OSCAR | 80.3±7.5 | 65.9±1.4 | 32.8±4.2 |
| PG | 72.9±2.7 | 58.0±2.7 | 41.6±12.1 |
| CG+PG | 73.1±2.5 | 55.9±3.5 | 43.5±7.9 |

|   | n=150 | n=200 |
|---|------|------|
| Oracle | 1.8±0.4 | 1.5±0.4 |
| Ridge | 14.0±1.6 | 10.1±0.9 |
| Ridge+Kmeans | 7.7±4.1 | 2.4±1.0 |
| Lasso | 13.9±1.7 | 10.0±1.0 |
| OSCAR | 14.0±1.9 | 9.9±0.9 |
| PG | 1.9±1.0 | 1.5±0.4 |
| CG+PG | 4.6±7.4 | 1.5±0.4 |

Table 3. Test of the robustness of algorithms to the number of data points in dimensionality reduction with artificial data. PG refers to projected gradient, CG refers to conditional gradient on $M$.

We observe that both PG and CG give significantly better results than other methods and even reach the performance of the Oracle for $n > d$, while for $n \leq d$ results are in the same range. Note that contrary
to OSCAR, Lasso and Ridge, dimensionality reduction is guaranteed in our case (the number of clusters is fixed).

4.2. Real data.

4.2.1. Topics classification on 20NewsGroup dataset. We now perform classification of documents in topics, i.e., multitask learning. We used the publicly available 20NewsGroup dataset which contains 2800 documents that are classified in 20 topics. Some of the topics are very closely related to each other (e.g. pc.hardware and mac.hardware), while others are highly unrelated (e.g. misc.forsale and soc.religion.christian). Using this prior we try to benefit from the classification of some topics with the use of similar topics. We use a dictionary of 5000 words selected by TF-IDF coefficients and take the word frequencies as covariates for each document.

In Table 4, we compare our approach to other classical regularizations such as the Frobenius norm and the trace norm, as implemented by Ciliberto et al. [6], using either a ridge or a logistic loss (Log). The algorithm proposed by Jacob et al. [11] was too slow on this large dataset to compare with. We initialize it by the solution given by the logistic loss. All algorithms were 5-fold cross-validated on 80% of the data then tested on the remaining 20%. The number of clusters was set to 5, as suggested by the names of newsgroups. Figures after the ± sign correspond to one standard deviation when varying the training and test sets. Table 4 shows how clustering regularization can help prediction on topics.

| Method       | Frobenius Log | Trace Ridge | CG Ridge | PG Log |
|--------------|---------------|-------------|----------|--------|
|              | 5.8±2.0       | 4.3±0.8     | 4.9±0.8  | 2.6±1.5|

Table 4. 100 × mean absolute errors for predicting topics on 20NewsGroup dataset, comparing classical regularizers (Frobenius and Trace) with our algorithms. PG refers to projected gradient, CG refers to conditional gradient.

4.2.2. Predicting ratings from reviews using groups of words. We now perform “sentiment” analysis of newspaper movie reviews. We use the publicly available dataset introduced by Pang and Lee [14] which contains movie reviews paired with star ratings. We treat it as a regression problem, taking log-responses for y in (0, 1) and word frequencies as covariates. With a 5000 term vocabulary chosen by TF-IDF coefficients, the corpus contains 5006 documents and 1.6M words. We evaluate our algorithms for regression with clustered features against standard regression approaches: Lasso, Ridge, and Ridge followed by k-means on predictors (denoted by RK). All algorithms were 5-fold cross-validated on 80% of the dataset and then tested on the remaining 20%. The number of clusters was arbitrarily set to 10, though we did not notice significant changes when varying it. Results are reported in Table 5, figures after the ± sign correspond to one standard deviation when varying the training and test sets. While all methods including ours give 10% mean absolute errors on the test set (up to 0.5% accuracy), our approach has the additional benefit of providing clusters of words which have a similar influence. The clusters with highest absolute weights are also the ones with smallest number of words, which confirms the intuition that only a few words are very discriminative. We illustrate this in Table 6, picking randomly words of the first and last clusters sorted by weight.

5. DISCUSSION

We have developed a unified framework for supervised clustering over tasks, features or samples and provided two robust algorithms for solving the associated optimization problems. Results on synthetic and realistic text datasets suggest that our method is competitive against standard regression and classification methods, while having the additional benefit of providing clusters of tasks, features or samples.
| Ridge  | RK    | Lasso | PG    | CG+PG |
|--------|-------|-------|-------|-------|
| 9.9±0.1| 10.0±0.1| 9.8±0.1| 9.7±0.1| 9.8±0.1 |

Table 5. $100 \times$ mean absolute errors for predicting movie ratings associated with reviews. We evaluate supervised clustering on features using regression losses (PG and OM+PG) against standard regression approaches: Lasso, Ridge, and Ridge followed by k-means on predictors. RK refers to k-means on predictors, PG to projected gradient, CG to conditional gradient on $M$ and RC to regression clustering.

Table 6. Clustering of words on movie reviews. First and last word clusters using clustering penalty on features (corresponding to words here).

| Cluster 1 | Cluster 10 |
|-----------|------------|
| (negative) | (positive) |
| size 47 | size 87 |
| stupidity, shred, inept, clue, ludicrous, tedious, remotely, crass, tolerable, bland, devoid, mess, idea, lifeless, abysmal, badly, bearable, poorly, care, interesting, problem, worst, ridiculous, flat, dull, awful, suppose, bad, predictable, ludicrous | perfect, fine, brilliant, delightful, strongly, easy, rare, wonderfully, perfectly, goodspirited, masterpiece, strong, potent, memorable, unique, send, small, funniest, life, poignancy, imaginative, simplicity, confidence, delightfully, delight, touching, nuanced, study, oscar, classification, masterfully, intense, surprisingly, triumph, flawless, move, power, accomplishment, keen, love, finest, crisp, superb, marvelous |

REFERENCES

[1] Argyriou, A., Evgeniou, T., and Pontil, M. (2008). Convex multi-task feature learning. Machine Learning, 73(3):243–272.
[2] Arthur, D. and Vassilvitskii, S. (2007). k-means++: The advantages of careful seeding. In Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, pages 1027–1035. Society for Industrial and Applied Mathematics.
[3] Bellman, R. (1973). A note on cluster analysis and dynamic programming. Mathematical Biosciences, 18(3):311–312.
[4] Blumensath, T. and Davies, M. E. (2009). Iterative hard thresholding for compressed sensing. Applied and Computational Harmonic Analysis, 27(3):265–274.
[5] Bondell, H. D. and Reich, B. J. (2008). Simultaneous regression shrinkage, variable selection, and supervised clustering of predictors with oscar. Biometrics, 64(1):115–123.
[6] Ciliberto, C., Mroueh, Y., Poggio, T., and Rosasco, L. (2015). Convex learning of multiple tasks and their structure. In Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015, pages 1548–1557.
[7] Dhillon, I. S., Mallela, S., and Kumar, R. (2003). A divisive information theoretic feature clustering algorithm for text classification. The Journal of Machine Learning Research, 3:1265–1287.
[8] Elhamifar, E. and Vidal, R. (2009). Sparse subspace clustering. In CVPR.
[9] Frank, M. and Wolfe, P. (1956). An algorithm for quadratic programming. Naval research logistics quarterly, 3(1-2):95–110.
[10] Guzman-Rivera, A., Kohli, P., Batra, D., and Rutenbar, R. (2014). Efficiently enforcing diversity in multi-output structured prediction. In Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, pages 284–292.
[11] Jacob, L., Vert, J.-P., and Bach, F. (2009). Clustered multi-task learning: A convex formulation. In *Advances in Neural Information Processing Systems 21*, pages 745–752.

[12] Jaggi, M. (2013). Revisiting frank-wolfe: Projection-free sparse convex optimization. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 427–435.

[13] Jiang, J.-Y., Liou, R.-J., and Lee, S.-J. (2011). A fuzzy self-constructing feature clustering algorithm for text classification. *Knowledge and Data Engineering, IEEE Transactions on*, 23(3):335–349.

[14] Pang, B. and Lee, L. (2005). Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 115–124. Association for Computational Linguistics.

[15] Wang, H. and Song, M. (2011). Ckmeans. 1d. dp: optimal k-means clustering in one dimension by dynamic programming. *The R Journal*, 3(2):29–33.

[16] Zhang, B. (2003). Regression clustering. In *ICDM*, pages 451–. IEEE Computer Society.
6. APPENDIX

6.1. NOTATIONS. For any integer \( p \), we let \( \mathbf{1}_p \in \mathbb{R}^p \) be the vector whose coordinates are all ones, \( I_p \) the identity matrix in \( \mathbb{R}^{p \times p} \), \( \Gamma_p = \mathbf{1}_p \mathbf{1}_p^T/p \) and \( \Pi_p = I_p - \Gamma_p \) the corresponding centering matrix.

For all settings input samples are represented by the matrix \( X = (x_1, \ldots, x_n)^T \in \mathbb{R}^{n \times d} \). For regression problems their respective labels are represented by a vector \( y = (y_1, \ldots, y_n) \in \mathbb{R}^n \). For classification problems in \( K \) classes, we denote \( y_k = (y^1_k, \ldots, y^K_k) \) the vector of binary labels corresponding to the class \( k \in [1, K] \) and \( Y = (y^1, \ldots, y^K) \in \mathbb{R}^{n \times K} \). Here we denote simply \( L \) the loss in each setting, \( i.e. \ L(W) = \text{Loss}(y, X, W) \).

To define the clustering of \( m \) items (here features, tasks or samples) in \( Q \) groups, \( C_1, \ldots, C_Q \) of size \( s = (s_1, \ldots, s_Q) \) we use the assignment matrix \( Z \in \{0, 1\}^{m \times Q} \) such that \( Z_{iq} = 1 \) if item \( i \) is in group \( C_q \) and \( Z_{iq} = 0 \) otherwise. This matrix verifies \( Z1_Q = 1_m \) and \( Z^T 1_m = s \).

The ambient dimension of the predictors is denoted \( \delta \) such that \( W = (w_1, \ldots, w_m) \in \mathbb{R}^{\delta \times m} \). Centroids of clusters are represented in a matrix \( V = (v_1, \ldots, v_Q) \in \mathbb{R}^{\delta \times Q} \). For example in the case of dimensionality reduction \( m = d \) and \( \delta = 1 \), in the case of transfer learning \( m = K \) and \( \delta = d \) and in the case of regression clustering \( m = n \) and \( \delta = d \).

6.2. Decomposed soft clustering penalty. In paragraph 2.3. we defined a soft regularizing penalty in terms of centroids of clusters \( V \). We explicit it here in terms of the corresponding assignment matrix \( Z \) to obtain the general formulation with \( W = VZ^T \).

We therefore develop each term of the penalty, denoting \( \bar{v} = \frac{1}{m} \sum_{q=1}^{Q} s_q v_q \), we have

\[
\Omega_{\text{mean}}(V, Z) = \frac{\lambda_M}{2} m \frac{||\bar{v}||_2^2}{2} = \frac{\lambda_M}{2} \text{Tr}(VZ^T(I - \Pi)ZV^T),
\]

\[
\Omega_{\text{between}}(V, Z) = \frac{\lambda_B}{2} \sum_{q=1}^{Q} s_q ||v_q - \bar{v}||_2^2 = \frac{\lambda_B}{2} \text{Tr}(VZ^T \Pi ZV^T),
\]

\[
\Omega_{\text{within}}(W, V, Z) = \frac{\lambda_W}{2} \sum_{q=1}^{Q} \sum_{i \in C_q} ||w_i - v_q||_2^2 = \frac{\lambda_W}{2} ||W - VZ^T||_F^2.
\]

Using \( \bar{W} = VZ^T \) the total penalty can then be written

\[
\Omega(W, \bar{W}) = \frac{\lambda_M}{2} \text{Tr}(\bar{W}(I - \Pi)\bar{W}^T) + \frac{\lambda_B}{2} \text{Tr}(\bar{W}\Pi \bar{W}^T) + \frac{\lambda_W}{2} ||W - \bar{W}||_F^2.
\]

6.3. Detailed conditional gradients for features, tasks or samples clustering penalties. We give here detailed computations of the minimisation in \( V \) or \( (V, W) \) to obtain the function \( G \) that we will use for conditional gradient in all settings. We always suppose \( Z^T Z \) invertible \( i.e. \forall q, s_q \neq 0 \) (there is no empty cluster).

6.3.1. Simplification for Soft Clustering Problems. For supervised problem with a soft clustering penalty, the minimisation in the variable \( V \) for \( W, Z \) fixed can be made without assumptions on the specific loss \( L \) as it appears only in the clustering penalty. We denote by \( \lambda_W, \lambda_B, \lambda_M \) the weights associated respectively to the within, between and mean penalties. Minimization in \( V \) is then given by
\[ H(W, Z) := \min_{V \in \mathbb{R}^{d \times q}} \frac{\lambda_W}{2} \|W - VZ^T\|_F^2 + \frac{\lambda_B}{2} \text{Tr}(VZ^T \Pi_m ZV^T) + \frac{\lambda_M}{2} \text{Tr}(VZ^T \Gamma_m ZV^T) \]
\[ = \min_{V \in \mathbb{R}^{d \times q}} \frac{\lambda_W}{2} \|W\|_F^2 + \frac{1}{2} \text{Tr} \left( V \left( (\lambda_W + \lambda_B)Z^T Z + (\lambda_M - \lambda_B)Z^T \Gamma_m Z \right) V^T \right) - \lambda_W \text{Tr}(V^T W Z) \]
\[ = \frac{\lambda_W}{2} \text{Tr}(WW^T) - \frac{\lambda^2}{2} \text{Tr} \left( WZ \left( (\lambda_W + \lambda_B)ZZ^T + (\lambda_M - \lambda_B)Z^T \Gamma_m Z \right)^{-1} Z^TW \right). \]

Recall that the vector containing the sizes of the cluster verifies \( s = Z^T 1_m \). We denote by \( s^{\frac{1}{2}} \) the vector whose coordinates are \( \sqrt{s_q} \), and by \( s^{-\frac{1}{2}} \) the vector whose coordinates are \( \frac{1}{\sqrt{s_q}} \). The inversion in the precedent formula can be written

\[ J^{-1} = (\lambda_W + \lambda_B)ZZ^T + (\lambda_M - \lambda_B)Z^T \Gamma_m Z \]
\[ = \left( \lambda_W + \lambda_B \right) \text{diag}(s) + (\lambda_M - \lambda_B) \frac{s^{\frac{1}{2}} s^{\frac{1}{2}}^T}{m} \]
\[ = \frac{1}{\lambda_W + \lambda_B} \text{diag}(s^{\frac{1}{2}}) \left( I_Q + \frac{\lambda_M - \lambda_B}{\lambda_W + \lambda_B} \frac{s^{\frac{1}{2}} s^{\frac{1}{2}}^T}{m} \right) \text{diag}(s^{\frac{1}{2}}) \]
\[ = \frac{1}{\lambda_W + \lambda_B} (Z^T Z)^{-1} - \frac{\lambda_M - \lambda_B}{(\lambda_W + \lambda_M)(\lambda_W + \lambda_B) m} \frac{1_Q 1_Q^T}{m} \]
\[ = \frac{1}{\lambda_W + \lambda_B} (Z^T Z)^{-1} - \frac{\lambda_M - \lambda_B}{(\lambda_W + \lambda_M)(\lambda_W + \lambda_B) m} \frac{1_Q 1_Q^T}{m} + \frac{1}{\lambda_W + \lambda_M} \frac{1_Q 1_Q^T}{m}, \]

where we used that \( p = \frac{s^{\frac{1}{2}} s^{\frac{1}{2}}^T}{m} = \frac{s^{\frac{1}{2}} s^{\frac{1}{2}}^T}{\|s^{\frac{1}{2}}\|^2} \) is a projector and therefore \((I + \alpha p)^{-1} = I - \frac{\alpha}{\alpha + 1} p\), added to the fact that \( \text{diag}(s) = Z^T Z \).

Now introducing the equivalence matrix \( M = Z (Z^T Z)^{-1} Z^T \), and using that \( Z \frac{1_Q 1_Q^T}{m} Z^T = \Gamma_m \), we finally obtain

\[ H(W, Z) = \frac{\lambda_W}{2} \text{Tr}(WW^T) - \frac{\lambda^2}{2} \text{Tr} \left( W \left( \frac{1}{\lambda_W + \lambda_B} (M - \Gamma_m) + \frac{1}{\lambda_W + \lambda_M} \Gamma_m \right) W^T \right) \]
\[ = \frac{\lambda_W}{2} \text{Tr} \left( W \left( I_m - \alpha (M - \Gamma_m) - \beta \Gamma_m \right) W^T \right), \]

where \( \alpha = \frac{\lambda_W}{\lambda_W + \lambda_B} \) and \( \beta = \frac{\lambda_W}{\lambda_W + \lambda_M} \).

Denoting \( \bar{P} = I_m - \alpha (M - \Gamma_m) - \beta \Gamma_m \), one can also use Kronecker’s formula:

\[ H(W, Z) = \frac{\lambda_W}{2} \text{Vec}(W)^T (P \otimes I_{\delta}) \text{Vec}(W). \]

6.3.2. Learning with clustered tasks. We derive here the computation of \( G \) when clustering tasks. We restrict ourselves to the case of multi categorical classification setting formulated as a multitask problem such that each task shares the same input data. Given \( K \) classes, we denote by \( W = (w_1, \ldots, w_K) \in \mathbb{R}^{d \times K} \).
the matrix of linear predictors. Using a squared loss $l(f(x), y) = \frac{1}{2}(y - f(x))^2$ the empirical loss is given by

$$L(W) = \frac{1}{2n} \sum_{i=1}^{n} \sum_{k=1}^{K} (y_i^k - w_k^T x_i)^2.$$  

Using Kronecker’s product formula, we get

$$L(W) = \frac{1}{2n} \sum_{k=1}^{K} w_k^T X^T X w_k - \frac{1}{n} \sum_{k=1}^{K} w_k^T X^T y^k + \frac{1}{2n} \|y\|_2^2$$

$$= \frac{1}{2n} \sum_{k=1}^{K} e_k^T W^T X^T X W e_k - \frac{1}{n} \text{Tr}(W X^T Y) + \frac{1}{2n} \|y\|_2^2$$

$$= \frac{1}{2n} \text{Vec}(W)^T (I_K \otimes X^T X) \text{Vec}(W) - \frac{1}{n} \text{Vec}(W)^T \text{Vec}(X^T Y) + \frac{1}{2n} \|y\|_2^2.$$  

Using the expression found by minimizing in $V$ the regularization penalty, we get an expression for $G :$

$$G(M) = \min_W L(W) + H(W, Z)$$

$$= \min_W \frac{1}{2n} \text{Vec}(W)^T (I_K \otimes X^T X + P \otimes I_d) \text{Vec}(W) - \frac{1}{n} \text{Vec}(W)^T \text{Vec}(X^T Y) + \frac{1}{2n} \|y\|_2^2$$

$$= \frac{1}{2n} \text{Vec}(X^T Y)^T (I_K \otimes X^T X + \lambda W n P \otimes I_d)^{-1} \text{Vec}(X^T Y) + \frac{1}{2n} \|y\|_2^2$$

Denote $(v_1, ..., v_d) \in \mathbb{R}^{d \times d}, (\lambda_1, ..., \lambda_d) \in \mathbb{R}^d$ and $(u_1, ..., u_S) \in \mathbb{R}^{S \times S}$, $(\mu_1, ..., \mu_S) \in \mathbb{R}^S$ the eigenvectors and corresponding eigenvalues respectively of matrices $X^T X$ and $P = (I_K - \alpha (M - \Gamma_K) + \beta \Gamma_K)$. The eigenvectors and corresponding eigenvalues of $I_K \otimes X^T X + \lambda W n P \otimes I_d$ are $(u_i \otimes v_j)_{i \in [1,n]}$ and $(\lambda W n \mu_i + \lambda_j)_{j \in [1,d]}$. The inversion in the expression of $G$ is then given by

$$J^{-1} = (I_K \otimes X^T X + \lambda W n P \otimes I_d)^{-1} = \sum_{i=1}^{n} \sum_{j=1}^{d} \frac{1}{\lambda W n \mu_i + \lambda_j} u_i u_i^T \otimes v_j v_j^T.$$  

We then note that the set of eigenvectors of $P$ can be decomposed into three sets. Indeed the matrices $I_K - M, M - \Gamma_K$ and $\Gamma_K$ are orthogonal projectors on orthogonal subspaces spanning the entire space. Denote by $I_W, I_B, I_M$ the sets of eigenvectors corresponding respectively to $I_K - M, M - \Gamma_K$ and $\Gamma_K$, their corresponding eigenvalues in $P$ can easily be computed and we obtain

$$P = I_K - M + (1 - \alpha)(M - \Gamma_K) + (1 - \beta)\Gamma_K$$

$$= \sum_{i \in I_W} u_i u_i^T + (1 - \alpha) \sum_{i \in I_W} u_i u_i^T + (1 - \beta) \sum_{i \in I_M} u_i u_i^T.$$  

15
This decomposition can be used for the inversion

\[
J^{-1} = \sum_{i \in I_W} \sum_{j=1}^{d} \frac{1}{\lambda_i n + \lambda_j} u_i u_i^T \otimes v_j v_j^T \\
+ \sum_{i \in I_B} \sum_{j=1}^{d} \frac{1}{\lambda_i n(1 - \alpha) + \lambda_j} u_i u_i^T \otimes v_j v_j^T \\
+ \sum_{i \in I_B} \sum_{j=1}^{d} \frac{1}{\lambda_i n(1 - \beta) + \lambda_j} u_i u_i^T \otimes v_j v_j^T \\
= (I_K - M) \otimes (X^T X + n\lambda_W I_d)^{-1} + (M - \Gamma_K) \otimes (X^T X + n\lambda_W (1 - \alpha) I_d)^{-1} \\
+ \Gamma_K \otimes (X^T X + n\lambda_W (1 - \beta) I_d)^{-1}.
\]

Finally \( G \) can be simplified using properties of the Kronecker product

\[
G(M) = -\frac{1}{2n} \text{Vec}(X^T Y)^T ((I_K - M) \otimes (X^T X + n\lambda_W I_d)^{-1}) \text{Vec}(X^T Y) \\
- \frac{1}{2n} \text{Vec}(X^T Y)^T ((M - \Gamma_K) \otimes (X^T X + n\lambda_W (1 - \alpha) I_d)^{-1}) \text{Vec}(X^T Y) \\
- \frac{1}{2n} \text{Vec}(X^T Y)^T (\Gamma_K \otimes (X^T X + n\lambda_W (1 - \beta) I_d)^{-1}) \text{Vec}(X^T Y) + \frac{1}{2n} \|Y\|_F^2 \\
= -\frac{1}{2n} \text{Tr}(Y^T X (X^T X + n\lambda_W I_d)^{-1} X^T Y (I_K - M)) \\
- \frac{1}{2n} \text{Tr}(Y^T X (X^T X + n\lambda_W (1 - \alpha) I_d)^{-1} X^T Y (M - \Gamma_K)) \\
- \frac{1}{2n} \text{Tr}(Y^T X (X^T X + n\lambda_W (1 - \beta) I_d)^{-1} X^T Y \Gamma_K) + \frac{1}{2n} \|Y\|_F^2.
\]

Therefore \( G \) is a linear function of \( M \) whose gradient is given by

\[
\nabla G(M) = \frac{1}{2n} Y^T X ((X^T X + n\lambda_W I_d)^{-1} - (X^T X + n\lambda_W (1 - \alpha) I_d)^{-1}) X^T Y.
\]

As \( 1 \geq \alpha \geq 0 \), we get that \( -\nabla G(M) \geq 0 \). For a fixed \( M, W \) is given using precedent computations and Kronecker’s formula by

\[
W_M = (X^T X + n\lambda_W I_d)^{-1} X^T Y (I_K - M) \\
+ (X^T X + n\lambda_W (1 - \alpha) I_d)^{-1} X^T Y (M - \Gamma_K) \\
+ (X^T X + n\lambda_W (1 - \beta) I_d)^{-1} X^T Y \Gamma_K.
\]

6.3.3. Learning with clustered features. For more generality we cast the problem of learning clustered features in the multiclass learning setting. We restrict here to the soft supervised clustering problem as the hard one has already be done. We keep notations introduced in section 6.3.2 though we now have
Learning multiple predictors. In the setting of learning multiple predictors, we denote by \( C_q \) the set of points whose best linear predictor is \( w_q \), having therefore \( C_1, \ldots, C_Q \) a partition of \([1, n]\). We let as above \( s_q \) be the cardinal of the set \( C_q \), \( X_q \in \mathbb{R}^{s_q \times d} \) is the matrix whose columns are the points belonging to the cluster \( q \), and \( y_q \) is the column vector of labels corresponding to cluster \( q \). We use a squared loss \( l(f(x), y) = \frac{1}{2}(y - f(x))^2 \).

For \( C_1, \ldots, C_Q \) fixed (or equivalently \( Z \) fixed), we can compute \( G \) using the Sherman-Woodbury-Morrison formula as

\[
G(M) = \min_{W} \frac{1}{2n} \sum_{k=1}^{K} \text{Tr} \left( W^T (X^T X + \lambda W n P) \right) - \frac{1}{n} \text{Tr}(W X^T Y) + \frac{1}{2n} \|Y\|_F^2
\]

Adding the regularization penalty and minimizing in \( V \) we obtain

\[
G(M) = \min_{W} \frac{1}{2n} \sum_{k=1}^{K} \text{Tr} \left( W^T (X^T X + \lambda W n P) \right) - \frac{1}{n} \text{Tr}(W X^T Y) + \frac{1}{2n} \|Y\|_F^2
\]

As detailed before, the inverse of \( P \) can be found analytically by observing that it is composed of orthogonal projectors on orthogonal subspaces spanning the entire space. Hence we have \( P^{-1} = I_d - M + \frac{1}{1-\alpha} (M - \Gamma_d) + \frac{1}{1-\beta} \Gamma_d \). We now get

\[
G(M) = \frac{1}{2n} \text{Tr} \left( Y Y^T (I_n + \frac{1}{\lambda W n} X P^{-1} X^T)^{-1} \right)
\]

Note that we still have \(-\nabla G(M) \succeq 0\). For a fixed \( M \), \( W \) is given analytically by

\[
W_M = Y^T X \left( X^T X + \lambda W n (I_d - \alpha(M - \Gamma_d) - \beta \Gamma_d) \right)^{-1}.
\]

6.3.4. Learning multiple predictors. In the setting of learning multiple predictors, we denote by \( \Gamma \) the set of points whose best linear predictor is \( w \), having therefore \( \Gamma_1, \ldots, \Gamma_Q \) a partition of \([1, n]\). We let as above \( s \) be the cardinal of the set \( \Gamma_q \), \( X_q \in \mathbb{R}^{s \times d} \) is the matrix whose columns are the points belonging to the cluster \( q \), and \( y_q \) is the column vector of labels corresponding to cluster \( q \). We use a squared loss \( l(f(x), y) = \frac{1}{2}(y - f(x))^2 \).

For \( \Gamma_1, \ldots, \Gamma_Q \) fixed (or equivalently \( Z \) fixed), we can compute \( G \) using the Sherman-Woodbury-Morrison formula as

\[
G(M) = \min_{v_1, \ldots, v_Q} \frac{1}{2n} \sum_{q=1}^{Q} \sum_{i \in \Gamma_q} (y_i - v_i^T x_i)^2 + \frac{\mu}{2} \sum_{q=1}^{Q} s_q \|v_q\|_2^2
\]

\[
= \sum_{q=1}^{Q} \frac{1}{2n} y_q^T \left( \frac{1}{s_q \mu n} X_q X_q^T + I_n \right)^{-1} y_q.
\]
We define $E \in \mathbb{R}^{n \times n}$ the permutation matrix permuting order of points such that

$$E y = \begin{pmatrix} y c_1 \\ \vdots \\ y c_Q \end{pmatrix}$$

$$EX = \begin{pmatrix} X_1 \\ \vdots \\ X_Q \end{pmatrix}.$$ 

We denote for $q \in [1, Q]$, $R_q = \sum_{p=1}^{Q} \sum_{s_p \geq i \leq \sum_{p=1}^{Q} s_p} e_i e_i^T$ orthogonal projectors on the ordered sets of points belonging to cluster $q$ such that

$$\begin{pmatrix} X_1 X_1^T & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & X_Q X_Q^T \end{pmatrix} = \text{diag}(EXX^T E^T) = \sum_{q=1}^{Q} R_q EX^T E^T R_q.$$ 

Thus we get

$$G(M) = \frac{1}{2n} y^T E^T \left( \sum_{q=1}^{Q} \frac{1}{s_q \mu n} R_q EX X^T E^T R_q + I_n \right)^{-1} E y$$

$$= \frac{1}{2n} y^T \left( \sum_{q=1}^{Q} \frac{1}{s_q \mu n} E^T R_q EX X^T E^T R_q E + I_n \right)^{-1} y.$$ 

Then we notice that $E^T R_q E = \text{diag}(Z_q)$ i.e. the projector on the set of points belonging to cluster $q$, and that $\sum_{q=1}^{Q} \frac{1}{s_q} \text{diag}(Z_Q) XX^T \text{diag}(Z_q) = M \circ XX^T$, where $\circ$ denotes the Hadamard product. Hence we finally get

$$G(M) = \frac{1}{2n} y^T \left( \frac{1}{\mu n} XX^T \circ M + I_n \right)^{-1} y.$$ 

Its gradient is given by

$$2n \nabla G(M) = -\frac{1}{\mu n} XX^T \circ \left( I_n + \frac{1}{\mu n} XX^T \circ M \right)^{-1} y y^T \left( I_n + \frac{1}{\mu n} XX^T \circ M \right)^{-1},$$

for which we have $-\nabla G(M) \succeq 0$. For a fixed $Z$, denoting by $X_q = Z_q^T X$ the set of points belonging to cluster $q$, the linear predictors $v_q$ for each cluster of points are given by

$$v_q = (n \mu s_q I_d + X_q^T X_q)^{-1} X_q^T y_q.$$