A decomposition method for lasso problems with zero-sum constraint

Andrea Cristofari*

*Department of Mathematics “Tullio Levi-Civita”,
University of Padua, Via Trieste, 63, 35121 Padua, Italy
E-mail: andrea.cristofari@unipd.it

Abstract. In this paper, we consider lasso problems with zero-sum constraint, commonly required for the analysis of compositional data in high-dimensional spaces. A novel algorithm is proposed to solve these problems, combining a tailored active-set technique, to identify the zero variables in the optimal solution, with a 2-coordinate descent scheme. At every iteration, the algorithm chooses between two different strategies: the first one requires to compute the whole gradient of the smooth term of the objective function and is more accurate in the active-set estimate, while the second one only uses partial derivatives and is computationally more efficient. Global convergence to optimal solutions is proved and numerical results are provided on synthetic and real datasets, showing the effectiveness of the proposed method. The software is publicly available.


1 Introduction

In this paper, we address the following optimization problem:

\[\min \frac{1}{2} \|Ax - y\|^2 + \lambda \|x\|_1 \]
\[\sum_{i=1}^{n} x_i = 0,\]  

\[\text{(1)}\]

where \(\|\cdot\|\) denotes the Euclidean norm, \(\|\cdot\|_1\) denotes the \(\ell_1\) norm, \(x \in \mathbb{R}^n\) is the variable vector and \(A \in \mathbb{R}^{m \times n}, y \in \mathbb{R}^m, \lambda \in \mathbb{R}\) are given, with \(\lambda \geq 0\).

Problem (1) is an extension of the well known lasso problem [39], imposing the zero-sum constraint \(\sum_{i=1}^{n} x_i = 0\). This constraint is required in some regression models for compositional data, i.e., for data representing percentages, or proportions, of a whole. Applications with data of this type frequently arise in many different fields, such as geology, biology, ecology and economics. For example, in microbiome analysis, datasets are usually normalized and result in compositional data [19, 37].

Let us briefly review the role of the zero-sum constraint in regression analysis for compositional data. Assume we are given a response vector \(y \in \mathbb{R}^m\) and an \(m \times n\) matrix \(Z\) of...
A decomposition method for lasso problems with zero-sum constraint

covariates, where every row of $Z$ is a sample. By definition, compositional data are vectors whose components are non-negative and sum to 1, so we can assume each row of $Z$ belong to the positive simplex $\Delta_n^+ = \{z \in \mathbb{R}^n : \sum_{i=1}^n z_i = 1, z > 0\}$. Due to the constrained form of the sample space, standard statistical tools designed for unconstrained data cannot be applied to build a regression model. To overcome this issue, log-contrast models were proposed in [1, 2], using the following log-ratio transformation of data:

$$W_{ij} = \log \left( \frac{Z_{ij}}{Z_{ir}} \right), \quad i \in \{1, \ldots, m\}, \quad j \in \{1, \ldots, n\} \setminus \{r\},$$

where the $r$th component is referred to as reference component. Denoting by $W_{\setminus r} \in \mathbb{R}^{m \times (n-1)}$ the matrix with entries $W_{ij}$, a linear log-contrast model is then obtained as follows:

$$y = W x_{\setminus r} + \varepsilon,$$

where $x_{\setminus r} = [x_1 \ldots x_{r-1} x_{r+1} \ldots x_n]^T \in \mathbb{R}^{n-1}$ is the vector of regression coefficients and $\varepsilon \in \mathbb{R}^m$ is a vector of independent noise with mean zero and finite variances. Using the definition of $W_{\setminus r}$, we can write

$$y_i = \sum_{j \neq r} x_j \log (Z_{ij}) - \sum_{j \neq r} x_j \log(Z_{ir}) + \varepsilon_i, \quad i = 1, \ldots, m.$$

Hence, introducing $x_r = -\sum_{j \neq r} x_j$, we can express the above model in the following symmetric form:

$$y = Ax + \varepsilon, \quad \text{subject to } \sum_{i=1}^n x_i = 0,$$

where $A \in \mathbb{R}^{m \times n}$ is the matrix with $A_{ij} = \log(Z_{ij})$ in position $(i, j)$. Note that the zero-sum constraint has a central role, since it allows the response $y$ to be expressed as a linear combination of log-ratios.

Starting from (2), an $\ell_1$-regularized least-square formulation with zero-sum constraint was first considered in [28] for variable selection in high-dimensional spaces, leading to the optimization problem (1). The resulting estimator was shown in [28] to be model selection consistent as well as to ensure scale invariance, permutation invariance and selection invariance. Moreover, in [3] it was shown that model (1) is proportional reference point insensitive, allowing to overcome some issues deriving from the choice of a reference point in molecular measurements.

Well established algorithms can be used to solve (1), exploiting the peculiar structure of the problem. In this fashion, an augmented Lagrangian scheme with the subproblem solved by cyclic coordinate descent was proposed in [28], while a coordinate descent strategy based on random selection of variables was proposed in [3]. Moreover, considering more general forms of constrained lasso, an approach based on quadratic programming and an ADMM method were analyzed in [18], a semismooth Newton augmented Lagrangian method was proposed in [15] and path algorithms were designed in [18, 24, 40].

In this paper we propose a decomposition algorithm, named AS-ZSL, to efficiently solve problem (1) in a large-scale setting. The first ingredient of our approach is an active-set technique to identify the zero variables in the final solution, which we expect to be a large number
due to the sparsity-promoting $\ell_1$ regularization. The second ingredient is a 2-coordinate descent scheme to update the variables estimated to be non-zero in the final solution. In more detail, we define two different strategies: the first one uses the whole gradient of the smooth term of the objective function and is more accurate in the identification of the zero variables, while the second one only needs partial derivatives and is computationally more efficient.

To balance computational efficiency and accuracy, in the algorithm we use a simple rule to choose between the two strategies, based on the progress in the objective function. The proposed method is proved to converge to optimal solutions and is shown to perform well in the numerical experiments, compared to other approaches from the literature. The AS-ZSL software is available at https://github.com/acristofari/as-zsl.

The rest of the paper is organized as follows. In Section 2 we give some optimality results for problem (1), in Section 3 we present the AS-ZSL algorithm, in Section 4 we establish global convergence of AS-ZSL to optimal solutions, in Section 5 we show numerical results on synthetic and real datasets, in Section 6 we finally draw some conclusions. Moreover, in Appendix A we report some details on the subproblem we need to solve in the algorithm.

2 Preliminaries and optimality results

In this section, we give some optimality conditions for problem (1). From now on, we indicate by $e$ the vector made of all ones (so that the zero-sum constraint $\sum_{i=1}^n x_i = 0$ can be rewritten as $e^T x = 0$). We denote the Euclidean norm, the $\ell_1$ norm and the sup norm of a vector $v$ by $\|v\|$, $\|v\|_1$ and $\|v\|_\infty$, respectively. The maximum between a vector $v$ and 0, denoted by $\max\{v, 0\}$, is understood as a vector whose $i$th entry is $\max\{v_i, 0\}$. Given a matrix $M$, the element in position $(i, j)$ is indicated with $M_{ij}$, while $M^i$ is the $i$th column of $M$. Given a function $\varphi: \mathbb{R}^n \rightarrow \mathbb{R}$, we denote its gradient by $\nabla \varphi$.

First, we note that an intercept term $x_0 \in \mathbb{R}$ might be included in model (2), resulting in

$$ y = x_0 e + Ax + \varepsilon, \quad \text{subject to } \sum_{i=1}^n x_i = 0, $$

Accordingly, problem (1) would become

$$ \min_{x_0, x} \frac{1}{2} \|x_0 e + Ax - y\|^2 + \lambda \|x\|_1 $$

$$ \sum_{i=1}^n x_i = 0. $$

It is straightforward to verify that any optimal solution $(x_0^*, x^*)$ of the above problem is such that

$$ x_0^* = \frac{e^T(y - Ax^*)}{m}. $$

Therefore, assuming that the vector $y$ and the columns of $A$ have mean zero, it follows that $x_0^* = 0$ and we can omit the intercept $x_0$ from the model without loss of generality.
Now, let us rewrite the objective function of problem (1) as
\[
f(x) = \varphi(x) + \lambda \|x\|_1, \quad \text{where} \quad \varphi(x) = \frac{1}{2} \|Ax - y\|^2.
\]

To derive optimality conditions, let us first rewrite problem (1) as a smooth problem with non-negativity constraints, using a well known variable transformation \([18, 35, 39]\). More specifically, we can introduce the variables \(x_+, x_- \in \mathbb{R}^n\) in order to split \(x\) into positive and negative components, i.e., \(x = x_+ - x_-\), with \(x_+ = \max\{x, 0\}\) and \(x_- = \max\{-x, 0\}\). We obtain the following reformulation:

\[
\begin{align*}
\min & \quad \varphi(x_+ - x_-) + \lambda e^T(x_+ + x_-) \\
e^T(x_+ - x_-) & = 0 \\
x_+ & \geq 0 \\
x_- & \geq 0.
\end{align*}
\]

Note that the number of variables has doubled, i.e., problem (3) has \(2n\) variables. In the next lemma, we state equivalence between problems (1) and (3).

**Lemma 2.1.** Let \(x^*\) be an optimal solution of problem (1). Then, \((x^*_+, x^*_-)\) is an optimal solution of problem (3), where \(x^*_+ = \max\{x^*, 0\}\) and \(x^*_- = \max\{-x^*, 0\}\).

Vice versa, if \((x^*_+, x^*_-)\) is an optimal solution of problem (3), then \(x^* = x^*_+ - x^*_-\) is an optimal solution of problem (1).

**Proof.** Let \(x^*\) be an optimal solution of problem (1), \(x^*_+ = \max\{x^*, 0\}\) and \(x^*_- = \max\{-x^*, 0\}\). To show that \((x^*_+, x^*_-)\) is optimal for problem (3), assume by contradiction that a feasible point \((\bar{x}_+, \bar{x}_-)\) of problem (3) exists such that \(\varphi(\bar{x}_+ - \bar{x}_-) + \lambda e^T(\bar{x}_+ + \bar{x}_-) < \varphi(x^*_+ - x^*_-)+\lambda e^T(x^*_+ + x^*_-)\). Since \(x^* = x^*_+ - x^*_-\) and \(\|x^*\|_1 = e^T(x^*_+ + x^*_-)\), we have

\[
\varphi(\bar{x}_+ - \bar{x}_-) + \lambda e^T(\bar{x}_+ + \bar{x}_-) < \varphi(x^*) + \lambda \|x^*\|_1 = f(x^*).
\]

Define \(\bar{x} = \bar{x}_+ - \bar{x}_-\). Note that \(\|\bar{x}\|_1 \leq e^T(\bar{x}_+ + \bar{x}_-)\). It follows that \(\bar{x}\) is feasible for problem (1) and \(f(\bar{x}) = \varphi(\bar{x}) + \lambda \|\bar{x}\|_1 \leq \varphi(\bar{x}_+ - \bar{x}_-) + \lambda e^T(\bar{x}_+ + \bar{x}_-)\), which, combined with (4), contradicts the fact that \(x^*\) is an optimal solution of problem (1).

Now, let \((x^*_+, x^*_-)\) be an optimal solution of problem (3) and \(x^* = x^*_+ - x^*_-\). To show that \(x^*\) is optimal for problem (1), assume by contradiction that a feasible point \(\bar{x}\) of problem (1) exists such that \(f(\bar{x}) < f(x^*) = \varphi(x^*) + \lambda \|x^*\|_1\). Since \(x^* = x^*_+ - x^*_-\) and \(\|x^*\|_1 = e^T(x^*_+ + x^*_-)\), we have

\[
f(\bar{x}) < \varphi(x^*_+ - x^*_-) + \lambda e^T(x^*_+ + x^*_-).
\]

Define \(\bar{x}_+ = \max\{\bar{x}, 0\}\) and \(\bar{x}_- = \max\{-\bar{x}, 0\}\). Note that \(\bar{x} = \bar{x}_+ - \bar{x}_-\) and \(\|\bar{x}\|_1 = e^T(\bar{x}_+ + \bar{x}_-)\). It follows that \((\bar{x}_+, \bar{x}_-)\) is feasible for problem (3) and \(\varphi(\bar{x}_+ - \bar{x}_-) + \lambda e^T(\bar{x}_+ + \bar{x}_-) = \varphi(\bar{x}) + \lambda \|\bar{x}\|_1 = f(\bar{x})\), which, combined with (5), contradicts the fact that \((x^*_+, x^*_-)\) is an optimal solution of problem (3).
Using KKT conditions, a point \((x_+^*, x_-^*)\) is an optimal solution of problem (3) if and only if there exist KKT multipliers \(\sigma_+^*, \sigma_-^* \in \mathbb{R}^n\) and \(\mu^* \in \mathbb{R}\) such that
\[
\nabla_{x_+} \varphi(x_+^* - x_-^*) + \lambda e - \mu^* e - \sigma_+^* = 0,
\n\nabla_{x_-} \varphi(x_-^* - x_-^*) + \lambda e + \mu^* e - \sigma_-^* = 0,
\ne^T (x_+^* - x_-^*) = 0,
\nx_+^* \geq 0, \quad x_-^* \geq 0,
\sigma_+^* \geq 0, \quad \sigma_-^* \geq 0,
(\sigma_+^*)^T x_+^* = 0, \quad (\sigma_-^*)^T x_-^* = 0.
\]

From Lemma 2.1 and the fact that \(\nabla_{x_+} \varphi(x_+ - x_-) = -\nabla_{x_-} \varphi(x_+ - x_-)\), it follows that a point \(x^*\) is an optimal solution of problem (1) if and only if there exist \(\sigma_+^*, \sigma_-^* \in \mathbb{R}^n\) and \(\mu^* \in \mathbb{R}\) such that
\[
\nabla \varphi(x^*) + \lambda e - \mu^* e - \sigma_+^* = 0, \quad (6a)
\n\nabla \varphi(x^*) - \lambda e - \mu^* e + \sigma_-^* = 0, \quad (6b)
\ne^T x^* = 0, \quad (6c)
\sigma_+^* \geq 0, \quad \sigma_-^* \geq 0, \quad (6d)
(\sigma_+^*)_i x_+^* = 0, \quad \forall i: x_+^* > 0, \quad (6e)
(\sigma_-^*)_i x_-^* = 0, \quad \forall i: x_-^* < 0. \quad (6f)
\]

The next theorem provides equivalent optimality conditions for problem (1).

**Theorem 2.2.** Let \(x^*\) be a feasible point of problem (1). The following sentences are equivalent:

(a) \(x^*\) is an optimal solution of problem (1);

(b) A scalar \(\mu^*\) (the same appearing in (6)) exists such that
\[
\begin{align*}
\nabla_i \varphi(x^*) \, - \, \mu^* & \, = \, \lambda, \quad i: x_i^* < 0, \\
\nabla_i \varphi(x^*) \, - \, \mu^* & \, = \, -\lambda, \quad i: x_i^* > 0, \\
|\nabla_i \varphi(x^*) \, - \, \mu^*| & \, \leq \, \lambda, \quad i: x_i^* = 0;
\end{align*}
\]

(c) \(\eta^{\text{min}}(x^*) \geq \eta^{\text{max}}(x^*)\), where
\[
\eta^{\text{min}}(x) = \min_{i=1,\ldots,n} \{ \nabla_i \varphi(x) + (2 \min \{\text{sign}(x_i), 0\} + 1) \lambda \},
\eta^{\text{max}}(x) = \max_{i=1,\ldots,n} \{ \nabla_i \varphi(x) + (2 \max \{\text{sign}(x_i), 0\} - 1) \lambda \}.
\]

**Proof.** We show the following implications.

- (a) \(\Rightarrow\) (b). If \(x^*\) is an optimal solution, then (6) holds. If \(x_i^* < 0\), from (6d)–(6f) we have \((\sigma_+^*)_i = 0\) and, from (6b), it follows that \(\nabla_i \varphi(x^*) - \mu^* = \lambda\). By the same arguments, if \(x_i^* > 0\) we have \((\sigma_-^*)_i = 0\) and \(\nabla_i \varphi(x^*) - \mu^* = -\lambda\). If \(x_i^* = 0\), from (6a), (6b) and (6d) we have \(-\lambda \leq \nabla_i \varphi(x^*) - \mu^* \leq \lambda\).
A decomposition method for lasso problems with zero-sum constraint

- (b) ⇒ (a). If (b) holds, then the KKT system (6) is satisfied by setting \( \sigma^*_+ = \nabla \varphi(x^*) + \lambda e - \mu^* e \) and \( \sigma^- = -\nabla \varphi(x^*) + \lambda e + \mu^* e \), implying that \( x^* \) is an optimal solution.

- (b) ⇒ (c). Let us rewrite \( \eta^{\text{min}}(x^*) \) and \( \eta^{\text{max}}(x^*) \) as follows:

\[
\eta^{\text{min}}(x^*) = \min \{ \min_{i: x_i^* \geq 0} \nabla_i \varphi(x^*) + \lambda, \min_{i: x_i^* < 0} \nabla_i \varphi(x^*) - \lambda \}, \tag{7a}
\]

\[
\eta^{\text{max}}(x^*) = \max \{ \max_{i: x_i^* \leq 0} \nabla_i \varphi(x^*) - \lambda, \max_{i: x_i^* > 0} \nabla_i \varphi(x^*) + \lambda \}. \tag{7b}
\]

If (b) holds, we have

\[
\nabla_i \varphi(x^*) + \lambda \begin{cases}
\geq \nabla_i \varphi(x^*) - \lambda = \mu^*, & i: x_i^* < 0, \\
= \mu^*, & i: x_i^* > 0, \\
\geq \mu^*, & i: x_i^* = 0,
\end{cases}
\]

and

\[
\nabla_i \varphi(x^*) - \lambda \begin{cases}
= \mu^*, & i: x_i^* < 0, \\
\leq \nabla_i \varphi(x^*) + \lambda = \mu^*, & i: x_i^* > 0, \\
\leq \mu^*, & i: x_i^* = 0.
\end{cases}
\]

Therefore, \( \eta^{\text{min}}(x^*) \geq \mu^* \geq \eta^{\text{max}}(x^*) \).

- (c) ⇒ (b). Let us consider \( \eta^{\text{min}}(x^*) \) and \( \eta^{\text{max}}(x^*) \) written as in (7). If (c) holds, let \( \mu^* \) be any number in \( [\eta^{\text{max}}(x^*), \eta^{\text{min}}(x^*)] \). If \( \{i: x_i^* < 0\} \neq \emptyset \), we can write

\[
\min_{i: x_i^* < 0} \nabla_i \varphi(x^*) - \lambda \leq \max_{i: x_i^* < 0} \nabla_i \varphi(x^*) - \lambda \leq \max_{i: x_i^* \leq 0} \nabla_i \varphi(x^*) - \lambda
\]

\[
\leq \eta^{\text{max}}(x^*) \leq \mu^* \leq \eta^{\text{min}}(x^*) \leq \min_{i: x_i^* < 0} \nabla_i \varphi(x^*) - \lambda.
\]

Therefore, all the inequalities in the above chain are actually equalities, implying that

\[
\mu^* = \min_{i: x_i^* < 0} \nabla_i \varphi(x^*) - \lambda = \max_{i: x_i^* < 0} \nabla_i \varphi(x^*) - \lambda.
\]

Namely,

\[
\mu^* = \nabla_i \varphi(x^*) - \lambda, \quad \forall i: x_i^* < 0,
\]

and the first condition of (b) is satisfied. Similarly, if \( \{i: x_i^* > 0\} \neq \emptyset \), we can write

\[
\max_{i: x_i^* > 0} \nabla_i \varphi(x^*) + \lambda \geq \min_{i: x_i^* > 0} \nabla_i \varphi(x^*) + \lambda \geq \min_{i: x_i^* \geq 0} \nabla_i \varphi(x^*) + \lambda
\]

\[
\geq \eta^{\text{min}}(x^*) \geq \mu^* \geq \eta^{\text{max}}(x^*) \geq \max_{i: x_i^* > 0} \nabla_i \varphi(x^*) + \lambda,
\]

implying that

\[
\mu^* = \nabla_i \varphi(x^*) + \lambda, \quad \forall i: x_i^* > 0,
\]

and the second condition of (b) is satisfied. Finally, if \( \{i: x_i^* = 0\} \neq \emptyset \), we can write

\[
\min_{i: x_i^* = 0} \nabla_i \varphi(x^*) + \lambda \geq \min_{i: x_i^* \geq 0} \nabla_i \varphi(x^*) + \lambda \geq \eta^{\text{min}}(x^*) \geq \mu^* \geq \eta^{\text{max}}(x^*)
\]

\[
\geq \max_{i: x_i^* \leq 0} \nabla_i \varphi(x^*) - \lambda \geq \max_{i: x_i^* = 0} \nabla_i \varphi(x^*) - \lambda,
\]

6
implying that

\[ \nabla_i \varphi(x^*) - \lambda \leq \mu^* \leq \nabla_i \varphi(x^*) + \lambda, \quad \forall i: x_i^* = 0, \]

and the third condition of (b) is satisfied.

Now, we derive an expression of the KKT multiplier \( \mu^* \) appearing in Theorem 2.2, which will be useful to define an active-set estimate in the proposed algorithm.

**Theorem 2.3.** If \( x^* \neq 0 \) is an optimal solution of problem (1), then the corresponding multiplier \( \mu^* \) appearing in point (b) of Theorem 2.2 is given by

\[
\mu^* = \frac{\sum_{i: x_i^* \neq 0} |x_i^*|^p \left( \nabla_i \varphi(x^*) + \lambda \text{sign}(x_i^*) \right)}{\sum_{i: x_i^* \neq 0} |x_i^*|^p}, \quad p \geq 0.
\]

**Proof.** Consider the KKT multiplier \( \mu^* \) appearing in (6), which is the same appearing in point (b) Theorem 2.2. From (6a) and (6b), we obtain

\[
\sigma^*_+ = \nabla \varphi(x^*) + \lambda e - \mu^* e,
\]
\[
\sigma^*_- = -\nabla \varphi(x^*) + \lambda e + \mu^* e.
\]

Using (6d)–(6f), it follows that \( \mu^* \) is the unique optimal solution of the following strictly convex one-dimensional problem, for all \( p \geq 0 \):

\[
\min_{\mu} \frac{1}{2} \left[ \sum_{i: x_i^* > 0} (\nabla_i \varphi(x^*) + \lambda - \mu)^2 (x_i^*)^p + \sum_{i: x_i^* < 0} (-\nabla_i \varphi(x^*) + \lambda + \mu)^2 (-x_i^*)^p \right].
\]

To compute \( \mu^* \) as the minimizer of the above univariate function, we have to set its derivative to 0, that is,

\[
\mu^* = \frac{\sum_{i: x_i^* > 0} (x_i^*)^p (\nabla_i \varphi(x^*) + \lambda) + \sum_{i: x_i^* < 0} (-x_i^*)^p (\nabla_i \varphi(x^*) - \lambda)}{\sum_{i: x_i^* > 0} (x_i^*)^p + \sum_{i: x_i^* < 0} (-x_i^*)^p},
\]

leading to the expression given in the assertion. \( \square \)

Let us conclude this section by showing a regularity property of the non-zero feasible points of problem (1), which will be useful in the global convergence analysis of the proposed algorithm.

**Proposition 2.4.** Let \( \bar{x} \) be a feasible point of problem (1) such that \( \bar{x}_j \neq 0 \) for some \( j \in \{1, \ldots, n\} \). If there exists \( I \subseteq \{1, \ldots, n\} \) such that

\[
\bar{x} \in \text{Argmin}_{\xi \in \mathbb{R}} f(x + \xi (e_i - e_j)), \quad \forall i \in I,
\]

with \( e_i \) being the \( i \)-th standard basis vector.
then there exists $\bar{\mu} \in \mathbb{R}$ such that

$$
\begin{cases}
\nabla_i \varphi(\bar{x}) - \bar{\mu} = \lambda, & i \in I: \bar{x}_i < 0, \\
\nabla_i \varphi(\bar{x}) - \bar{\mu} = -\lambda, & i \in I: \bar{x}_i > 0, \\
|\nabla_i \varphi(\bar{x}) - \bar{\mu}| \leq \lambda, & i \in I: \bar{x}_i = 0.
\end{cases}
$$

(8)

Proof. For any point $x \in \mathbb{R}^n$ and a (non-zero) direction $d \in \mathbb{R}^n$, let $f'(x; d)$ be the directional derivative of $f$ at $x$ along $d$. We have

$$
f'(x; d) = \lim_{\varepsilon \to 0^+} \frac{f(x + \varepsilon d) - f(x)}{\varepsilon} = \nabla \varphi(x)^T d + \lambda \lim_{\varepsilon \to 0^+} \frac{\|x + \varepsilon d\|_1 - \|x\|_1}{\varepsilon}
$$

(9)

$$
= \nabla \varphi(x)^T d + \lambda \left( \sum_{i=1}^{n} \text{sign}(x_i) d_i + \sum_{i: x_i = 0} |d_i| \right). 
$$

Since $\bar{x} \in \text{Argmin}_{x \in \mathbb{R}} f(x + \xi(e_i - e_j))$ for all $i \in I$ by hypothesis, from the convexity of $f$ we get

$$
f'(\bar{x}; e_i - e_j) \geq 0 \quad \text{and} \quad f'(\bar{x}; e_j - e_i) \geq 0, \quad \forall i \in I. 
$$

(10)

Without loss of generality, we assume that $\bar{x}_j > 0$ (the proof for the case $\bar{x}_j < 0$ is identical, except for minor changes). Using (9), we have

$$
f'(\bar{x}; e_i - e_j) = \begin{cases}
\nabla_i \varphi(\bar{x}) - \nabla_j \varphi(\bar{x}) - 2\lambda, & \text{if } \bar{x}_i < 0, \\
\nabla_i \varphi(\bar{x}) - \nabla_j \varphi(\bar{x}), & \text{if } \bar{x}_i \geq 0;
\end{cases}
$$

(11)

$$
f'(\bar{x}; e_j - e_i) = \begin{cases}
\nabla_j \varphi(\bar{x}) - \nabla_i \varphi(\bar{x}) + 2\lambda, & \text{if } \bar{x}_i \leq 0, \\
\nabla_j \varphi(\bar{x}) - \nabla_i \varphi(\bar{x}), & \text{if } \bar{x}_i > 0.
\end{cases}
$$

(12)

Therefore, in view of (10), for all $i \in I$ such that $\bar{x}_i \neq 0$ we can write

$$
\nabla_i \varphi(\bar{x}) = \begin{cases}
\nabla_j \varphi(\bar{x}) + 2\lambda, & i \in I: \bar{x}_i < 0, \\
\nabla_j \varphi(\bar{x}), & i \in I: \bar{x}_i > 0.
\end{cases}
$$

So, we can set $\bar{\mu} = \nabla_j \varphi(\bar{x}) + \lambda$ and the first two conditions of (8) are satisfied. Moreover, with this choice of $\bar{\mu}$ we have $|\nabla_i \varphi(\bar{x}) - \bar{\mu}| = |\nabla_i \varphi(\bar{x}) - \nabla_j \varphi(\bar{x}) - \lambda|$. So, to show that also the last condition of (8) holds, we have to show that

$$
0 \leq \nabla_i \varphi(\bar{x}) - \nabla_j \varphi(\bar{x}) \leq 2\lambda, \quad i \in I: \bar{x}_i = 0.
$$

This follows from (10), (11) and (12).

\[\square\]

Remark 1. In Proposition 2.4, if $I = \{1, \ldots, n\}$, then $\bar{x}$ is an optimal solution of problem (1), according to point (b) of Theorem 2.2.
3 The algorithm

In this section we propose a decomposition algorithm, named Active-Set Zero-Sum-Lasso (AS-ZSL), to efficiently solve problem (1).

The underlying assumptions motivating our approach are that the optimal solutions are sparse, due to the sparsity-promoting \( \ell_1 \) regularization, and that the problem dimension is large. To face these issues, the proposed method relies on two main ingredients:

(i) an active-set technique to estimate the zero variables in the final solution,

(ii) a 2-coordinate descent scheme to update only the variables estimated to be non-zero in the final solution.

In the field of constrained optimization, several active-set techniques were proposed to identify the active (or binding) constraints, see, e.g., [4, 6, 10, 11, 13, 16, 17, 22, 23, 34, 36]. Active-set strategies were successfully used also to identify the zero variables in \( \ell_1 \)-regularized problems [14, 26, 38, 43, 44] and in \( \ell_1 \)-constrained problems [12]. To introduce our approach, let us first define the following index sets for any optimal solution \( x^* \) of problem (1):

\[
\bar{A}(x^*) = \{ i : x^*_i = 0 \},
\]
\[
\bar{N}(x^*) = \{ 1, \ldots, n \} \setminus \bar{A}(x^*) = \{ i : x^*_i \neq 0 \}.
\]

We say that \( \bar{A}(x^*) \) and \( \bar{N}(x^*) \) represent the active set and the non-active set, respectively, at \( x^* \). In any point \( x^k \) produced by the algorithm, we get estimates of \( \bar{A}(x^*) \) and \( \bar{N}(x^*) \) exploiting point (b) of Theorem 2.2. Namely, we set the vector \( \pi^k \) as an approximation of \( \nabla \varphi(x^k) - \mu^* e \) and define

\[
A^k = \{ i : x^k_i = 0, |\pi^k_i| \leq \lambda \}, \quad \text{(13a)}
\]
\[
N^k = \{ 1, \ldots, n \} \setminus A^k, \quad \text{(13b)}
\]

as estimates of \( \bar{A}(x^*) \) and \( \bar{N}(x^*) \), respectively.

Once \( A^k \) and \( N^k \) have been computed, we want to move only the variables estimated to be non-zero in the final solution, i.e., the variables in \( N^k \), thus working in a lower dimensional space. Moreover, to efficiently address large-scale problems, a 2-coordinate descent scheme is used for the variable update, that is, we move two coordinates at a time (this is the minimum number of variables we can move to maintain feasibility, due to the equality constraint).

Given a feasible point \( x^k \neq 0 \) produced by the algorithm, in the sequel we define two possible strategies to compute \( \pi^k \) in (13) and to update the variables. The first strategy uses the whole gradient \( \nabla \varphi \) and is more accurate in the active-set estimate, while the second strategy only uses partial derivatives and is computationally more efficient. The rationale is trying to balance accuracy and computational efficiency, in order to calculate the whole gradient vector \( \nabla \varphi \) only when needed. In particular, in our algorithm we never compute the matrix \( A^T A \), since this is impractical for large dimensions. So, if the residual is known, \( O(m) \) operations are needed to compute a single partial derivative, while \( O(mn) \) operations are needed to compute \( \nabla \varphi \).
3.1 Strategy MVP

In this first strategy, we need to compute the whole gradient $\nabla \phi(x^k)$. Then, to obtain $\pi^k$ in (13), we use an approximation of the KKT multiplier $\mu^*$ by means of the so called multiplier functions. More precisely, we say that $\mu : \mathbb{R}^n \to \mathbb{R}$ is a multiplier function if it is continuous in any optimal solution $x^*$ and such that $\mu(x^*) = \mu^*$. A class of multiplier functions can be straightforwardly obtained from Theorem 2.3. Namely, we can define

$$
\mu(x) = \frac{\sum_{i=1}^n |x_i|^p (\nabla_i \phi(x) + \lambda \text{sign}(x_i))}{\sum_{i=1}^n |x_i|^p}, \quad p > 0.
$$

Once $\nabla \phi(x^k)$ and $\mu(x^k)$ have been computed, we can set

$$
\pi^k = \nabla \phi(x^k) - \mu(x^k)e.
$$

With this choice of $\pi^k$ we have the following identification property, ensuring that, if we are sufficiently close to an optimal solution $x^*$, then $i \in A^k \Rightarrow x_i^* = 0$, while the inverse implication (i.e., $x_i^* = 0 \Rightarrow i \in A^k$) holds if $|\nabla_i \phi(x^*) - \mu^*| < \lambda$.

**Proposition 3.1.** Let $A^k$ and $N^k$ be defined as in (13), with $\pi^k$ computed as in (15) and $\mu(x^k)$ computed as in (14). Then, for any optimal solution $x^*$ of problem (1), there exists a neighborhood $B(x^*)$ such that

$$
\bar{A}^+(x^*) \subseteq A^k \subseteq \bar{A}(x^*), \quad \forall x^k \in B(x^*),
$$

where $\bar{A}^+(x^*) = \{i : x_i^* = 0, |\nabla_i \phi(x^*) - \mu^*| < \lambda\}$ and $\mu^*$ is the KKT multiplier.

**Proof.** The inclusion $A^k \subseteq \bar{A}(x^*)$ is trivial, since $x_i^* \neq 0$ implies that, for all $x^k$ in a neighborhood of $x^*$, we have $x_i^k \neq 0$ and then $i \notin A^k$. The inclusion $\bar{A}^+(x^*) \subseteq A^k$ follows from the continuity of $\nabla \phi(x)$ and the continuity of $\mu(x)$, recalling that $\mu(x^*) = \mu^*$. \qed

To update the variables in $N^k$, we use a 2-coordinate descent scheme based on the so called maximal violating pair (MVP), i.e., we move the two variables that most violate an optimality measure. In the literature, similar choices were considered for singly linearly constrained problems when the objective function is smooth, using proper optimality measures (see, e.g., [5, 25, 27, 31, 32]). In our case, using point (c) of Theorem 2.2, a maximal violating pair in $N^k$ is defined as any pair $(i, j)$ such that

$$
\begin{align*}
\bar{i} &\in \text{Argmin}_{i \in N^k} \{ \nabla_i \phi(x^k) + (2 \min \{ \text{sign}(x_i^k), 0 \} + 1) \lambda \}, \\
\bar{j} &\in \text{Argmax}_{j \in N^k} \{ \nabla_j \phi(x^k) + (2 \max \{ \text{sign}(x_j^k), 0 \} - 1) \lambda \}.
\end{align*}
$$

The next feasible point $x^{k+1}$ is then obtained by minimizing $f$ with respect to $x_i$ and $x_j$, keeping all the other variables fixed in $x^k$, that is,

$$
x^{k+1} \in \text{Argmin}\{ f(x) : e^T x = 0, x_h = x_h^k, h \in \{1, \ldots, n\} \setminus \{i, j\} \}.
$$
Andrea Cristofari

Equivalently,

\[ x^{k+1} = x^k + \xi^*(e_i - e_j), \]
\[ \xi^* \in \text{Argmin}_{\xi \in \mathbb{R}} \{f(x^k + \xi(e_i - e_j))\}. \tag{17} \]

3.2 Strategy AC2CD

This second strategy does not require to compute the whole gradient \( \nabla \varphi \). In (13) we simply set \( \pi^k = \pi^{k-1} \) and then we update the variables in \( N^k \) by means of an almost cyclic rule. In particular, we extend a 2-coordinate descent method, named AC2CD, proposed in [8] and further analyzed in [9]. Considering singly linearly constrained problems with lower and upper bound on the variables, this method chooses two variables at a time, such that one of them must be “sufficiently far” from the lower and the upper bound in some points produced by the algorithm, while the other one is picked cyclically. In order to adapt this approach to our setting, we first have to choose a variable index \( j(k) \) such that

\[ |x^k_{j(k)}| \geq \tau \|x^k\|_{\infty}, \quad j(k) \in N^k, \tag{18} \]

where \( \tau \in (0, 1] \) is a fixed parameter. Namely, we require \( x^k_{j(k)} \) to be “sufficiently different” from zero. Then, we start a cycle of inner iterations where we update two variables at a time. In particular, first we set \( z^{k,1} = x^k \) and choose a permutation \( p^k_1, \ldots, p^k_{|N^k|} \) of \( N^k \), then we select one index \( p^k_i \) at a time in a cyclic fashion, with \( i = 1, \ldots, |N^k| \), and consider the index pair \( (p^k_i, j(k)) \). We compute \( z^{k,i+1} \) by minimizing \( f(z) \) with respect to \( z_{p^k_i} \) and \( z_{j(k)} \), keeping all the other variables fixed in \( z^{k,i} \). Namely,

\[ z^{k,i+1} \in \text{Argmin}\{f(z): e^T z = 0, z_h = z^{k,i}_h, h \in \{1, \ldots, n\} \setminus \{p^k_i, j(k)\}\}. \]

Equivalently,

\[ z^{k,i+1} = z^{k,i} + \xi^*(e_{p^k_i} - e_{j(k)}), \]
\[ \xi^* \in \text{Argmin}_{\xi \in \mathbb{R}} \{f(z^{k,i} + \xi(e_{p^k_i} - e_{j(k)}))\}. \tag{19} \]

After producing the points \( z^{k,1}, z^{k,2}, \ldots, z^{k,|N^k|+1} \), we set the next feasible point \( x^{k+1} = z^{k,|N^k|+1} \).

3.3 Choosing between the two strategies

We have seen that, one the one hand, Strategy AC2CD does not need the expensive computation of the whole gradient \( \nabla \varphi \), but, on the other hand, we expect the active-set estimate used in Strategy MVP to be more accurate in a neighborhood of an optimal solution, according to Proposition 3.1. In order to balance computational efficiency and accuracy, we want to use Strategy MVP only when we judge it is worthwhile to compute a new gradient and a new \( \pi^k \) in (13). This occurs when we observe no sufficient progress in the objective function. In particular, given a parameter \( \theta \in (0, 1] \), at iteration \( k \) we use Strategy MVP if
A decomposition method for lasso problems with zero-sum constraint

both \( f(x^{k-1}) - f(x^k) \leq \theta \max\{f(x^{k-1}), 1\} \) and Strategy MVP was not used in \( x^{k-1} \), otherwise we use Strategy AC2CD. The scheme of the proposed \( \text{AS-ZSL} \) method is reported in Algorithm 1.

**Algorithm 1** Active-Set Zero-Sum-Lasso (\( \text{AS-ZSL} \))

0 Given \( \theta \in (0, 1] \) and \( \tau \in (0, 1] \), set \( x^0 = 0 \) and \( k = 0 \)

1 If \( x^0 \) is not optimal, compute \( x^1 \) such that \( f(x^1) < f(x^0) \) and set \( k = 1 \), else STOP

2 While \( x^k \) is not optimal

3 If \( \frac{f(x^{k-1}) - f(x^k)}{\max\{f(x^{k-1}), 1\}} \leq \theta \) and Strategy MVP was not used for \( x^{k-1} \)

   \( \text{Strategy MVP} \)

   4 Compute \( A^k \) and \( N^k \) as in (13), with \( \pi^k \) computed as in (15)

   5 Compute a maximal violating pair \((i, j)\) as in (16)

   6 Compute \( x^{k+1} \) as in (17)

   Else

   \( \text{Strategy AC2CD} \)

   8 Compute \( A^k \) and \( N^k \) as in (13), with \( \pi^k = \pi^{k-1} \)

   9 Choose a variable index \( j(k) \) satisfying (18)

   10 Set \( z^{k,1} = x^k \)

   11 Choose a permutation \( \{p_1^k, \ldots, p_{|N^k|}^k\} \) of \( N^k \)

   For \( i = 1, \ldots, |N^k| \)

   12 Compute \( z^{k,i+1} \) as in (19)

   End for

   13 Set \( x^{k+1} = z^{k,|N^k|+1} \)

   End if

17 Set \( k = k + 1 \)

18 End while

**Remark 2.** Both Strategy MVP and Strategy AC2CD require to solve a subproblem for every variable update, given in (17) and (19), respectively. We see that each subproblem consists in an exact minimization of \( f \) over a direction of the form \( \pm(e_i - e_j) \). A finite procedure for this minimization is described in Appendix A.

4 Convergence analysis

In this section, we show the convergence of \( \text{AS-ZSL} \) to optimal solutions, using some results on the proposed active-set estimate and standard arguments on block coordinate descent.
First, we note that most results stated in the above sections require \( x^k \neq 0 \). Indeed, AS-ZSL ensures that \( x^k \neq 0 \), \( \forall k \geq 1 \), since \( x^0 = 0 \) and
\[
f(x^{k+1}) \leq f(x^k) < f(x^0) = f(0), \quad \forall k \geq 1.
\]
(20)
As a consequence, for all \( k \geq 1 \),
- the multiplier function \( \mu(x) \) given in (14) is well defined at \( x^k \);
- \( \mathcal{N}_k \neq \emptyset \), according to (13).

Since \( f \) is continuous and coercive, also the following result follows from (20).

**Lemma 4.1.** Let \( \{x^k\} \) be an infinite sequence of points produced by AS-ZSL. Then,

(i) \( \lim_{k \to \infty} f(x^k) = f^* \in \mathbb{R} \), with \( f^* > 0 \);

(ii) every subsequence \( \{x^k\}_{K \subseteq \mathbb{N}} \) has limit points, each of them being feasible and different from zero.

Moreover, for any iteration \( k \) where Strategy AC2CD is used, from the instructions of the algorithm we have
\[
f(x^{k+1}) = f(z^{k,|\mathcal{N}_k|+1}) \leq f(z^{k,|\mathcal{N}_k|}) \leq \ldots \leq f(z^{k,1}) = f(x^k).
\]
So, also the following result follows from (20) and the fact that \( f \) is continuous and coercive.

**Lemma 4.2.** Let \( \{x^k\}_{K \subseteq \mathbb{N}} \) be an infinite subsequence of points produced by AS-ZSL such that Strategy AC2CD is used for all \( k \in K \). Then, for every fixed \( i \in \{1, \ldots, \min_k |N^k| + 1\} \),

(i) \( \lim_{k \to \infty} f(z^{k,i}) = \lim_{k \to \infty} f(x^k) = f^* \in \mathbb{R} \), with \( f^* > 0 \);

(ii) the subsequence \( \{z^{k,i}\}_{K} \) has limit points, each of them being feasible and different from zero.

Now, we show that AS-ZSL cannot select Strategy MVP or Strategy AC2CD for an arbitrarily large number of consecutive iterations. In particular, provided \( \{x^k\} \) is infinite, eventually the algorithm alternates between the two strategies.

**Proposition 4.3.** Let \( \{x^k\} \) be an infinite sequence of points produced by AS-ZSL. Then, there exists \( k \) such that Strategy MVP is used for \( k = \bar{k}, \bar{k} + 1, \bar{k} + 2, \bar{k} + 3, \bar{k} + 5, \ldots \) and Strategy AC2CD is used for \( k = \bar{k} + 1, \bar{k} + 3, \bar{k} + 5, \ldots \).

**Proof.** From point (i) of Lemma 4.1 and (20), the sequence \( \{f(x^k)\} \) converges to a value \( f^* \) such that \( f(x^k) \geq f^* \) for all \( k \geq 0 \). It follows that
\[
\lim_{k \to \infty} \frac{f(x^{k+1}) - f(x^k)}{\theta \max\{f(x^{k+1}), 1\}} \leq \lim_{k \to \infty} \frac{f(x^{k+1}) - f(x^k)}{\theta \max\{f^*, 1\}} = 0.
\]
Therefore, according to the test at line 3 of Algorithm 1, the algorithm alternates between Strategy MVP and Strategy AC2CD infinitely. \( \square \)
4.1 Global convergence to optimal solutions

Without loss of generality, for every index pair \((i, j)\) selected by Strategy MVP or Strategy AC2CD, now we require \(f\) to be strictly convex along the directions \(\pm (e_i - e_j)\). In Appendix A.1, we show how this can be easily guaranteed. Essentially, we just have to remove variables from the problem when we find identical columns in the matrix \(A\). Using this non-restrictive requirement, we can prove that \(\|x^{k+1} - x^k\| \to 0\).

**Proposition 4.4.** Let \(\{x^k\}_{K_1 \subset \mathbb{N}}\) and \(\{x^k\}_{K_2 \subset \mathbb{N}}\) be two infinite subsequences of points produced by AS-ZSL, such that Strategy MVP is used for all \(k \in K_1\) and Strategy AC2CD is used for all \(k \in K_2\). Then,

\[
\begin{align*}
\lim_{k \to \infty} \{x^k\}_{K_1} & \to x^*, \\
\lim_{k \to \infty} \{x^k\}_{K_2} & \to x^*.
\end{align*}
\]

**Proof.** By contradiction, assume that (21a) does not hold. It follows that there exists a subsequence \(\{x^k\}_{K_3 \subset K_1}\) such that \(\liminf_{k \to \infty, k \in K_3} \|x^{k+1} - x^k\| > 0\). By point (ii) of Lemma 4.1, we can assume that both \(\{x^k\}_{K_3}\) and \(\{x^{k+1}\}_{K_3}\) converge to feasible points (passing into a further subsequence if necessary), so that

\[
\lim_{k \to \infty, k \in K_3} x^k = x' \neq x'' = \lim_{k \to \infty, k \in K_3} x^{k+1}.
\]  

(22)

Since the set of variable indices is finite, we can also assume that the maximal violating pair \((i, j)\) is the same for all \(k \in K_3\) (passing again into a further subsequence if necessary). Using (17), (22) and the continuity of \(f\), we obtain

\[x'' = x' + \xi^*(e_i - e_j), \quad \text{where} \quad \xi^* \in \text{Argmin}_{\xi \in \mathbb{R}} \{f(x' + \xi(e_i - e_j))\}.
\]

Namely, \(x'\) and \(x''\) belong to the line \(\{x' + \xi(e_i - e_j), \xi \in \mathbb{R}\}\). As said at the beginning of this subsection, without loss of generality here we require \(f\) to be strictly convex along \(\pm (e_i - e_j)\) (in Appendix A.1 it is shown how this can be easily guaranteed). Since \(x' \neq x''\), it follows that

\[
f\left(\frac{x' + x''}{2}\right) < \frac{1}{2} f(x') + \frac{1}{2} f(x'').
\]

(23)

Moreover, using again (17) we can write

\[
f(x^{k+1}) \leq f\left(x^k + \frac{1}{2}(x^{k+1} - x^k)\right) = f\left(\frac{x^k + x^{k+1}}{2}\right) \leq \frac{f(x^k)}{2} + \frac{f(x^{k+1})}{2},
\]

where the last inequality follows from the convexity of \(f\). Since \(f(x^{k+1}) \leq f(x^k)\), we obtain

\[
f(x^{k+1}) \leq f\left(\frac{x^k + x^{k+1}}{2}\right) \leq f(x^k).
\]

(24)
By (22) and point (i) of Lemma 4.1, using the continuity of \( f \) we have that
\[
f(x') = \lim_{k \to \infty} f(x^k) = \lim_{k \to \infty} f(x^{k+1}) = f(x'').
\]
Therefore, taking the limits in (24), we get
\[
f\left(\frac{x' + x''}{2}\right) = f(x') = f(x''),
\]
that is,
\[
f\left(\frac{x' + x''}{2}\right) = \frac{1}{2} f(x') + \frac{1}{2} f(x''),
\]
contradicting (23).

To show (21b), from the instructions of the algorithm we can write
\[
x^{k+1} - x^k = \sum_{i=1}^{|N_k|} (z^{k,i+1} - z^{k,i}) \quad \text{for all } k \in K_2,
\]
implying that
\[
\|x^{k+1} - x^k\| \leq \sum_{i=1}^{|N_k|} \|z^{k,i+1} - z^{k,i}\|, \quad \forall k \in K_2.
\]
Arguing by contradiction, assume that
\[
\liminf_{k \to \infty, k \in K} \sum_{i=1}^{|N_k|} \|z^{k,i+1} - z^{k,i}\| > 0.
\]
It follows that there exist an index \( \bar{i} \in \{1, \ldots, n\} \) and a subsequence \( \{z^{k,\bar{i}}\}_{K_4 \subseteq K_2} \) such that \( \liminf_{k \to \infty, k \in K_4} \|z^{k,\bar{i}+1} - z^{k,\bar{i}}\| > 0 \) for all \( k \in K_4 \). By point (ii) of Lemma 4.2, we can assume that both \( \{z^{k,\bar{i}}\}_K \) and \( \{z^{k,\bar{i}+1}\}_K \) converge to feasible points (passing into a further subsequence if necessary). Thus, we get a contradiction by the same arguments used to prove (21a).

**Proposition 4.5.** Let \( \{x^k\} \) be an infinite sequence of points produced by AS-ZSL. Then,
\[
\lim_{k \to \infty} \|x^{k+1} - x^k\| = 0.
\]

**Proof.** From Proposition 4.3, two infinite subsequences \( \{x^k\}_{K_1 \subseteq N} \) and \( \{x^k\}_{K_2 \subseteq N} \) exist such that Strategy MVP is used for all \( k \in K_1 \) and Strategy AC2CD is used for all \( k \in K_2 \), with \( K_1 \cup K_2 = N \). Then, the desired result follows from Proposition 4.4.

We are finally ready to show global convergence of AS-ZSL to optimal solutions.

**Theorem 4.6.** Let \( \{x^k\} \) be an infinite sequence of points produced by AS-ZSL. Then, \( \{x^k\} \) has limit points and every limit point is an optimal solution of problem (1).

**Proof.** From point (ii) of Lemma 4.1, \( \{x^k\} \) has limit points, each of them being feasible and different from zero. Let \( x^* \) be any of these limit points and let \( \{x^k\}_{K \subseteq N} \) be a subsequence converging to \( x^* \). In view of Proposition 4.5, \( x^* \) is also a limit point of \( \{x^{k+1}\}_K \) and of \( \{x^{k-1}\}_K \). Namely,
\[
\lim_{k \to \infty, k \in K} x^k = \lim_{k \to \infty} x^{k+1} = \lim_{k \to \infty} x^{k-1} = x^* \neq 0.
\]
Using Proposition 4.3, without loss of generality we can assume that Strategy AC2CD is used for all \( k \in K \). (In particular, if Strategy AC2CD is used for infinitely many \( k \in K \), we can simply discard from \( \{x^k\}_K \) the indices \( k \) where Strategy AC2CD is not used. On the contrary, if \( \bar{k} \) exists such that Strategy MVP is used for all \( k \geq \bar{k}, k \in K \), then we can consider the subsequence \( \{x^{k+1}\}_K \) instead. This subsequence still converges to \( x^* \) by (26) and, for all sufficiently large \( k \in K \), Strategy AC2CD is used for all \( k+1 \) in view of Proposition 4.3.)

Since the set of variable indices is finite, for all \( k \in K \) we can assume that \( A^k, N^k, j(k) \) and \( p^k_i \) are the same, that is,

\[
A^k = A, \quad N^k = N, \quad j(k) = j, \quad p^k_i = p_i, \quad \forall i \in \{1, \ldots, |N| + 1\}
\]

(passing into a further subsequence if necessary) and, by point (ii) of Lemma 4.2, that

\[
\lim_{k \to \infty} z^{k,i} = \bar{z}^i, \quad \forall i \in \{1, \ldots, |N| + 1\} \tag{27}
\]

(passing again into a further subsequence if necessary).

By continuity of \( f \), we can take the limits in (19) and, using (27), for all \( i \in \{1, \ldots, |N|\} \) we have

\[
\bar{z}^{i+1} = \bar{z}^i + \xi^*(e_{p_i} - e_j), \quad \xi^* \in \text{Argmin}_{\xi \in \mathbb{R}} \{f(\bar{z}^i + \xi(e_{p_i} - e_j))\}. \tag{28}
\]

Now, we show that

\[
\bar{z}^i = x^*, \quad \forall i \in \{1, \ldots, |N| + 1\}. \tag{29}
\]

From the instructions of the algorithm we have \( z^{k,1} = x^k \), so we can write \( \|z^{k,i} - x^k\| \leq \sum_{h=1}^{i-1} \|z^{k,h+1} - z^{k,h}\| \) for all \( i \in \{1, \ldots, |N| + 1\} \). Using (21b), it follows that

\[
\lim_{k \to \infty} \|z^{k,i} - x^k\| = 0, \quad \forall i \in \{1, \ldots, |N| + 1\}. \tag{30}
\]

Since \( \|z^{k,i} - x^*\| \leq \|z^{k,i} - x^k\| + \|x^k - x^*\| \), from (26) and (30) we get

\[
\lim_{k \to \infty} \|z^{k,i} - x^*\| = 0.
\]

Using (27), we thus obtain (29).

Therefore, from (28) and (29) it follows that

\[
x^* \in \text{Argmin}_{\xi \in \mathbb{R}} f(x^* + \xi(e_{i} - e_j)), \quad \forall i \in \{1, \ldots, N\}. \tag{31}
\]

Using (26), a real number \( \eta > 0 \) exists such that \( \|x^k\|_\infty \geq \eta/\tau \) for all sufficiently large \( k \in K \), where \( \tau \in (0, 1] \) is the parameter used in AS-ZSL such that \( |x^k_j| \geq \tau\|x^k\|_\infty \) (see line 9 of Algorithm 1). Consequently, for all sufficiently large \( k \in K \), we have \( |x^k_j| \geq \eta \), and then,

\[
x^*_j \neq 0. \tag{32}
\]
So, using (31), (32) and Proposition 2.4, there exists $\mu^* \in \mathbb{R}$ such that
\[
\begin{align*}
\nabla_i \varphi(x^*) - \mu^* &= \lambda, \quad i \in \mathcal{N}: x^*_i < 0, \\
\nabla_i \varphi(x^*) - \mu^* &= -\lambda, \quad i \in \mathcal{N}: x^*_i > 0, \\
|\nabla_i \varphi(x^*) - \mu^*| &\leq \lambda, \quad i \in \mathcal{N}: x^*_i = 0.
\end{align*}
\]
(33)

Now, from the active-set estimate (13a) we observe that $i \in \mathcal{A} \Rightarrow x^k_i = 0$ for all $k \in K$. Since $\{x^k\}_K \rightarrow x^*$ from (26), it follows that
\[
\mathcal{A} \subseteq \{i: x^*_i = 0\}.
\]
(34)

Taking into account (33) and (34), according to point (b) of Theorem 2.2, we thus have to show that
\[
|\nabla_i \varphi(x^*) - \mu^*| \leq \lambda, \quad \forall i \in \mathcal{A},
\]
in order to prove that $x^*$ is optimal and conclude the proof. Note that, from (33) and (34), we can write $|x^*_i|^p(\nabla_i \varphi(x^*) + \lambda \text{sign}(x^*_i)) = |x^*_i|^p\mu^*$, with $p \geq 0$, for all $i = 1, \ldots, n$. This implies that
\[
\sum_{i=1}^n |x^*_i|^p(\nabla_i \varphi(x^*) + \lambda \text{sign}(x^*_i)) = \mu^* \sum_{i=1}^n |x^*_i|^p, \quad p \geq 0.
\]

So, using the definition of the multiplier function $\mu(x)$ given in (14), we get
\[
\mu^* = \mu(x^*).
\]
(36)

Moreover, Proposition 4.3 ensures that, for sufficiently large $k \in K$, Strategy MVP is used at $x^{k-1}$, implying that $\pi^k = \pi^{k-1} = \nabla \varphi(x^{k-1}) - \mu(x^{k-1})e$ (see lines 8 and 4 of Algorithm 1). Therefore, from the active-set estimate (13a), for all sufficiently large $k \in K$ we can write
\[
\mathcal{A} = \{i: x^k_i = 0, |\nabla_i \varphi(x^{k-1}) - \mu(x^{k-1})| \leq \lambda\}.
\]

Since $\{x^k\}_K$ and $\{x^{k-1}\}_K$ converge to $x^*$ from (26), using the continuity of $\nabla \varphi$, the continuity of the multiplier function $\mu(x)$ and (36), we can take the limits for $k \rightarrow \infty$, $k \in K$, and we finally get (35).

5 Numerical results

In this section, we report the numerical results obtained on synthetic and real datasets with compositional data. We implemented AS-ZSL in C++, using a MEX file to call the algorithm from Matlab. The AS-ZSL software is available at https://github.com/acristofari/as-zsl.

In our experiments, we use $p = 1$ for the multiplier functions defined in (14) and $\tau = 1$. Moreover, $\theta$ is set to $10^{-2}$ at the beginning of the algorithm and is gradually decreased to $10^{-6}$. All tests were run on an Intel(R) Core(TM) i7-9700 with 16 GB RAM memory.

We compared AS-ZSL with the following algorithms, with all parameters set to their default values:
• compCL [28], which solves (1) by the method of multiplier minimizing the augmented Lagrangian function by cyclic coordinate descent. The code was written in C++ and called from R. It was downloaded from https://cran.r-project.org/package=Compack as part of the R package Compack.

• QP [18], which solves the quadratic reformulation (3). Since \( n > m \), a ridge term \( 10^{-4}\|x\|^2 \) was added to the original objective function, as suggested in [18]. The code was downloaded from https://github.com/Hua-Zhou/SparseReg, it builds the problem via Matlab and uses the Gurobi Optimizer (version 9.5) [21] for the minimization.

• zeroSum [3], which uses an extension of the random coordinate descent method to solve (1). The code was written in C++ and called from R. It was downloaded from https://github.com/rehbergT/zeroSum as part of the R package zeroSum.

We observe that both compCL and zeroSum use a (block) coordinate descent approach and were specifically designed for regression problems with zero-sum constraint, while QP uses the default algorithm implemented in Gurobi for quadratic programs, i.e., the barrier algorithm.

The results obtained from the comparisons are described in Subsection 5.1 and 5.2. Finally, in Subsection 5.3 we show how a warm start strategy can be used in AS-ZSL to solve a sequence of problems with decreasing regularization parameters.

### 5.1 Synthetic datasets

In the first experiments, we generated some synthetic datasets for log-contrast model as suggested in [28], using the function \texttt{comp.Model} implemented in the Compack package. More specifically, a matrix \( M \in \mathbb{R}^{m \times n} \) was first generated from a multivariate normal distribution \( N(\theta, \Sigma) \). To model the presence of five major components in the composition, the vector \( \theta \) has all zeros except for \( \theta_i = \log(0.5n) \), \( i = 1, \ldots, 5 \). The matrix \( \Sigma \) has \( 0.5^{i-j} \) in position \((i, j)\). Then, the log-contrast model (2) was obtained by setting the matrix \( A \) such that \( A_{ij} = \log(Z_{ij}) \) in position \((i, j)\), with

\[
Z_{ij} = \frac{e^{M_{ij}}}{\sum_{h=1}^n e^{M_{ih}}}, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n,
\]

the vector of regression coefficients \( x = (1, -0.8, 0.6, 0, 0, -1.5, -0.5, 1.2, 0, \ldots, 0)^T \) and the noise terms in \( \varepsilon \) were generated from a normal distribution \( N(0, 0.5^2) \). We see that \( x \) has six non-zero coefficients, with three of them being among the five major components.

We considered problems with dimensions \( m = 2000 \) and \( n \in \{m, 2m, 5m\} \). For every pair \((m, n)\) we generated 10 different datasets and, for each of them, we used \( \lambda \in \{\lambda_1, \ldots, \lambda_5\} \) such that \( \lambda_1, \ldots, \lambda_5 \) are logarithmically equally spaced between \( 10^{\lambda_1} \) and \( 10^{\lambda_5} \), where

\[
\lambda_1 = 0.95\lambda_{\text{max}}, \quad \lambda_5 = 10^{-3}\lambda_{\text{max}}
\]
Table 1: Results on synthetic datasets for log-contrast models with six non-zero regression coefficients. The final objective value is indicated by $f^*$, while the CPU time in seconds is indicated by $\text{time}$.

<table>
<thead>
<tr>
<th></th>
<th>$\lambda_1$</th>
<th>$\lambda_2$</th>
<th>$\lambda_3$</th>
<th>$\lambda_4$</th>
<th>$\lambda_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$f^*$</td>
<td>time</td>
<td>$f^*$</td>
<td>time</td>
<td>$f^*$</td>
</tr>
<tr>
<td>$m = 2000, n = 2000$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AS-ZSL</td>
<td>4.70e+04</td>
<td>0.02</td>
<td>1.61e+04</td>
<td>0.03</td>
<td>4.71e+03</td>
</tr>
<tr>
<td>compCL</td>
<td>4.70e+04</td>
<td>2.10</td>
<td>1.61e+04</td>
<td>1.44</td>
<td>4.71e+03</td>
</tr>
<tr>
<td>QP</td>
<td>4.70e+04</td>
<td>7.38</td>
<td>1.61e+04</td>
<td>7.06</td>
<td>4.71e+03</td>
</tr>
<tr>
<td>zeroSum</td>
<td>4.71e+04</td>
<td>2.53</td>
<td>4.71e+04</td>
<td>2.54</td>
<td>2.79e+04</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>$\lambda_1$</th>
<th>$\lambda_2$</th>
<th>$\lambda_3$</th>
<th>$\lambda_4$</th>
<th>$\lambda_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$f^*$</td>
<td>time</td>
<td>$f^*$</td>
<td>time</td>
<td>$f^*$</td>
</tr>
<tr>
<td>$m = 2000, n = 4000$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AS-ZSL</td>
<td>5.41e+04</td>
<td>0.04</td>
<td>1.84e+04</td>
<td>0.09</td>
<td>5.19e+03</td>
</tr>
<tr>
<td>compCL</td>
<td>5.41e+04</td>
<td>4.30</td>
<td>1.84e+04</td>
<td>2.77</td>
<td>5.19e+03</td>
</tr>
<tr>
<td>QP</td>
<td>5.41e+04</td>
<td>104.77</td>
<td>1.84e+04</td>
<td>96.41</td>
<td>5.19e+03</td>
</tr>
<tr>
<td>zeroSum</td>
<td>5.42e+04</td>
<td>9.97</td>
<td>5.42e+04</td>
<td>9.93</td>
<td>3.85e+04</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>$\lambda_1$</th>
<th>$\lambda_2$</th>
<th>$\lambda_3$</th>
<th>$\lambda_4$</th>
<th>$\lambda_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$f^*$</td>
<td>time</td>
<td>$f^*$</td>
<td>time</td>
<td>$f^*$</td>
</tr>
<tr>
<td>$m = 2000, n = 10000$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AS-ZSL</td>
<td>6.50e+04</td>
<td>0.10</td>
<td>2.18e+04</td>
<td>0.19</td>
<td>5.81e+03</td>
</tr>
<tr>
<td>compCL</td>
<td>6.50e+04</td>
<td>11.30</td>
<td>2.18e+04</td>
<td>6.86</td>
<td>5.81e+03</td>
</tr>
<tr>
<td>QP</td>
<td>6.50e+04</td>
<td>3239.67</td>
<td>2.18e+04</td>
<td>2537.57</td>
<td>5.81e+03</td>
</tr>
<tr>
<td>zeroSum</td>
<td>6.51e+04</td>
<td>61.90</td>
<td>6.51e+04</td>
<td>61.60</td>
<td>6.45e+04</td>
</tr>
</tbody>
</table>

and $\lambda_{\max}$ is such that $x^* = 0$ is an optimal solution of problem (1) if and only if $\lambda \geq \lambda_{\max}$. We can easily compute $\lambda_{\max}$ from point (c) of Theorem 2.2 (it also follows from Corollary 1 of [24]):

$$\lambda_{\max} = \max_{j = 1, \ldots, n} [(A^T)y]_j - \min_{i = 1, \ldots, n} [(A^T)y]_i$$

In Table 1, we show the results obtained with the different problem dimensions. In particular, for each considered $\lambda$, we report the average values over the 10 runs in terms of final objective value and CPU time. We see that AS-ZSL always took less 2 seconds on average to solve all the considered problems, being much faster than the other methods and also achieving the lowest objective function value. In particular, AS-ZSL is one or two orders of magnitude faster than the other coordinate descent based methods, i.e., compCL and zeroSum.

Next, we used the same datasets described above, but with the vector of regression coefficients $x$ containing 5% of randomly chosen non-zero entries, which were generated from a uniform distribution in $(-1, 1)$.

The results are shown in Table 2. Also in this case, we see that AS-ZSL achieves the lowest objective function value and is the fastest method, except for the largest problems.
A decomposition method for lasso problems with zero-sum constraint

Table 2: Results on synthetic datasets for log-contrast models with 5% of non-zero regression coefficients. The final objective value is indicated by $f^*$, while the CPU time in seconds is indicated by $time$.

<table>
<thead>
<tr>
<th>$m = 2000, n = 2000$</th>
<th>$\lambda_1$</th>
<th>$\lambda_2$</th>
<th>$\lambda_3$</th>
<th>$\lambda_4$</th>
<th>$\lambda_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$f^*$</td>
<td>$time$</td>
<td>$f^*$</td>
<td>$time$</td>
<td>$f^*$</td>
</tr>
<tr>
<td>AS-ZSL</td>
<td>3.46e+04</td>
<td>0.03</td>
<td>1.85e+04</td>
<td>0.12</td>
<td>6.03e+03</td>
</tr>
<tr>
<td>compCL</td>
<td>3.46e+04</td>
<td>4.18</td>
<td>1.94e+04</td>
<td>10.21</td>
<td>8.39e+03</td>
</tr>
<tr>
<td>QP</td>
<td>3.46e+04</td>
<td>6.85</td>
<td>1.85e+04</td>
<td>6.74</td>
<td>6.03e+03</td>
</tr>
<tr>
<td>zeroSum</td>
<td>3.46e+04</td>
<td>2.43</td>
<td>2.02e+04</td>
<td>3.08</td>
<td>7.71e+03</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$m = 2000, n = 4000$</th>
<th>$\lambda_1$</th>
<th>$\lambda_2$</th>
<th>$\lambda_3$</th>
<th>$\lambda_4$</th>
<th>$\lambda_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$f^*$</td>
<td>$time$</td>
<td>$f^*$</td>
<td>$time$</td>
<td>$f^*$</td>
</tr>
<tr>
<td>AS-ZSL</td>
<td>8.52e+04</td>
<td>0.05</td>
<td>5.36e+04</td>
<td>0.31</td>
<td>3.22e+04</td>
</tr>
<tr>
<td>compCL</td>
<td>8.52e+04</td>
<td>17.26</td>
<td>5.51e+04</td>
<td>19.75</td>
<td>3.50e+04</td>
</tr>
<tr>
<td>QP</td>
<td>8.52e+04</td>
<td>96.86</td>
<td>5.36e+04</td>
<td>103.67</td>
<td>3.22e+04</td>
</tr>
<tr>
<td>zeroSum</td>
<td>8.52e+04</td>
<td>9.78</td>
<td>6.75e+04</td>
<td>12.09</td>
<td>4.45e+04</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$m = 2000, n = 10000$</th>
<th>$\lambda_1$</th>
<th>$\lambda_2$</th>
<th>$\lambda_3$</th>
<th>$\lambda_4$</th>
<th>$\lambda_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$f^*$</td>
<td>$time$</td>
<td>$f^*$</td>
<td>$time$</td>
<td>$f^*$</td>
</tr>
<tr>
<td>AS-ZSL</td>
<td>1.73e+05</td>
<td>0.15</td>
<td>1.18e+05</td>
<td>0.91</td>
<td>4.85e+04</td>
</tr>
<tr>
<td>compCL</td>
<td>1.73e+05</td>
<td>34.30</td>
<td>1.21e+05</td>
<td>46.58</td>
<td>5.41e+04</td>
</tr>
<tr>
<td>QP</td>
<td>1.73e+05</td>
<td>2622.73</td>
<td>1.18e+05</td>
<td>2890.54</td>
<td>4.85e+04</td>
</tr>
<tr>
<td>zeroSum</td>
<td>1.73e+05</td>
<td>61.94</td>
<td>1.23e+05</td>
<td>92.17</td>
<td>5.47e+04</td>
</tr>
</tbody>
</table>

with $\lambda_5$, where compCL took less time, but it returned a higher objective function value.

5.2 Real datasets

Now we show the results obtained on real microbiome data, which are generally regarded as compositional in the literature (see, e.g., [7, 19, 30, 37]). We used three datasets considered in [33], downloaded from https://github.com/nphdang/DeepCoDA and containing data from [42] suitably adjusted to deal with log-contrast model by proper replacement of the zero features (which are not allowed in such models). The considered datasets are for binary classification (then, responses are in $\{0, 1\}$) and are described in Table 3.

For each dataset, first we applied a log transformation and then we chose $\lambda$ by a 5-fold

Table 3: Microbiome datasets from [33, 42].

<table>
<thead>
<tr>
<th>Dataset</th>
<th>$m$</th>
<th>$n$</th>
<th>Class 1</th>
<th>Class 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2070</td>
<td>3090</td>
<td>Gastro</td>
<td>Oral</td>
</tr>
<tr>
<td>2</td>
<td>404</td>
<td>3090</td>
<td>Stool</td>
<td>Tongue</td>
</tr>
<tr>
<td>3</td>
<td>408</td>
<td>3090</td>
<td>Subgingival</td>
<td>Supragingival</td>
</tr>
</tbody>
</table>
cross validation. The final results are shown in Table 4. We see that AS-ZSL took less than 1 second on all the problems, still being the fastest method and achieving the lowest objective function value.

Table 4: Results on microbiome datasets. The final objective value is indicated by $f^*$, while the CPU time in seconds is indicated by $time$.

<table>
<thead>
<tr>
<th>Dataset 1</th>
<th>Dataset 2</th>
<th>Dataset 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f^*$</td>
<td>time</td>
<td>$f^*$</td>
</tr>
<tr>
<td>AS-ZSL</td>
<td>3.38e+01</td>
<td>0.48</td>
</tr>
<tr>
<td>compCL</td>
<td>3.38e+01</td>
<td>13.86</td>
</tr>
<tr>
<td>QP</td>
<td>3.38e+01</td>
<td>95.63</td>
</tr>
<tr>
<td>zeroSum</td>
<td>2.56e+02</td>
<td>7.27</td>
</tr>
</tbody>
</table>

Table 5: Comparison of CPU time required by AS-ZSL with and without warm start over 10 values of $\lambda$. All results are in seconds.

<table>
<thead>
<tr>
<th>$\lambda_1$</th>
<th>$\lambda_2$</th>
<th>$\lambda_3$</th>
<th>$\lambda_4$</th>
<th>$\lambda_5$</th>
<th>$\lambda_6$</th>
<th>$\lambda_7$</th>
<th>$\lambda_8$</th>
<th>$\lambda_9$</th>
<th>$\lambda_{10}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>time with warm start</td>
<td>0.14</td>
<td>0.46</td>
<td>0.99</td>
<td>1.77</td>
<td>2.95</td>
<td>4.87</td>
<td>7.83</td>
<td>12.34</td>
<td>15.60</td>
</tr>
<tr>
<td>time without warm start</td>
<td>0.14</td>
<td>0.44</td>
<td>0.80</td>
<td>1.29</td>
<td>2.79</td>
<td>5.24</td>
<td>10.94</td>
<td>24.69</td>
<td>52.16</td>
</tr>
<tr>
<td>cumulative time with warm start</td>
<td>0.14</td>
<td>0.60</td>
<td>1.59</td>
<td>3.36</td>
<td>6.31</td>
<td>11.18</td>
<td>19.01</td>
<td>31.35</td>
<td>46.95</td>
</tr>
<tr>
<td>cumulative time without warm start</td>
<td>0.14</td>
<td>0.58</td>
<td>1.38</td>
<td>2.67</td>
<td>5.46</td>
<td>10.69</td>
<td>21.63</td>
<td>46.32</td>
<td>98.48</td>
</tr>
</tbody>
</table>

5.3 Optimizing over a grid of regularization parameters

In the previous experiments, we have shown the performances of AS-ZSL on several instances of problems (1) for a specific value of the regularization parameter $\lambda$. Now, we want to analyze a simple warm start strategy for AS-ZSL to solve a sequence of problems with decreasing regularization parameters. This can be useful in practice when a suitable value of $\lambda$ is not known in advance and a parameter selection procedure must be carried out.

We generated 10 synthetic datasets as explained in Subsection 5.1, with $m = 2000$, $n = 10000$ and 5% of non-zero entries in the vector of regression coefficients. For each dataset, we considered 10 different parameters $\lambda_1, \ldots, \lambda_{10}$ logarithmically equally spaced between $10^{\lambda_1}$ and $10^{\lambda_{10}}$, where $\lambda_1 = 0.95\lambda_{\text{max}}$, $\lambda_{10} = 10^{-3}\lambda_{\text{max}}$ and $\lambda_{\text{max}}$ was computed as explained in Subsection 5.1.

For $\lambda = \lambda_i$, $i = 2, \ldots, 10$, the warm start strategy simply consists in setting the starting point of AS-ZSL as the optimal solution computed with $\lambda = \lambda_{i-1}$.

In Table 5, we report the average results obtained with and without warm start. From the third and the fourth row of the table, we observe that the warm start strategy allowed us to complete the whole optimization process in 66 seconds, while it took more than 200 seconds without warm start. From the second and the third row of the table, we also note that running AS-ZSL with one small $\lambda$ took more than running AS-ZSL several times using decreasing regularization parameters with warm start. For example, AS-ZSL took 104 seconds for $\lambda = \lambda_{10}$ without warm start, but it took a total of 66 seconds when it was run ten times.
with \( \lambda = \lambda_1, \ldots, \lambda_{10} \) using the warm start strategy. This suggests that the warm start strategy might be used to further speed up the algorithm when problem (1) must be solved with small values of \( \lambda \).

6 Conclusions

In this paper we proposed AS-ZSL, a 2-coordinate descent method with active-set estimate to solve lasso problems with zero-sum constraint. At every iteration, AS-ZSL chooses between two strategies: Strategy MVP uses the whole gradient of the smooth term of the objective function and is more accurate in the active-set estimate, while Strategy AC2CD only needs partial derivatives and is computationally more efficient. A suitable test is used to choose between the two strategies, considering the progress in the objective function. A theoretical analysis was carried out, showing global convergence of AS-ZSL to optimal solutions. We performed numerical experiments on synthetic and real datasets, showing the effectiveness of the proposed method compared to other algorithms from the literature. We finally outlined a warm start strategy for AS-ZSL to solve a sequence of problems with decreasing regularization parameters.

Appendix A. The subproblem

According to (17) and (19), every variable update in AS-ZSL requires the resolution of a subproblem, both when using Strategy MVP and when using Strategy AC2CD. As highlighted in Remark 2, each subproblem consists in the minimization of \( f \) along a direction of the form \( \pm (e_i - e_j) \). In the next proposition, we give an equivalent expression of \( f \) along \( \pm (e_i - e_j) \) as an univariate function, which will be useful to compute the minimizer.

Proposition A.1. Let \( \bar{x} \) be any feasible point of problem (1). For any \( i \neq j \), we have

\[
  f(\bar{x} + \xi(e_i - e_j)) = f^{i,j}_\bar{x}(\bar{x} + \xi), \quad \forall \xi \in \mathbb{R},
\]

where \( f^{i,j}_\bar{x} : \mathbb{R} \to \mathbb{R} \) is the function defined as follows:

\[
  f^{i,j}_\bar{x}(u) = \frac{1}{2} \alpha u^2 - \beta u + \lambda (|u| + |u - \bar{x}_i - \bar{x}_j|) + c,
\]

with

\[
  \alpha = \|A^i - A^j\|^2, \\
  \beta = \alpha \bar{x}_i - \nabla_i \varphi(\bar{x}) + \nabla_j \varphi(\bar{x}), \\
  c = \frac{1}{2} \|y - A \bar{x} + (A^i - A^j)\bar{x}_i\|^2 + \lambda \sum_{t \neq i,j} |\bar{x}_t|.
\]

Proof. First, let us write the function \( f \) as follows:

\[
  f(x) = \frac{1}{2} \sum_{h=1}^{m} \left( A_{hi} x_i + A_{hj} x_j + \sum_{t \neq i,j} A_{ht} x_t - y_h \right)^2 + \lambda \left( |x_i| + |x_j| + \sum_{t \neq i,j} |x_t| \right).
\]
Now, choose any $\xi \in \mathbb{R}$ and let $u = \bar{x} + \xi(e_i - e_j)$. To prove the desired result, we have to show that
\[
f(u) = f^i_j(u_i) = \frac{1}{2} \alpha u_i^2 - \beta^T u_i + \lambda(|u_i| + |u_i - \bar{x}_i - \bar{x}_j|) + c. \tag{37}
\]
Clearly, $u_t = \bar{x}_t$ for all $t \neq i, j$. Since $\epsilon^T \bar{x} = 0$ from the feasibility of $\bar{x}$, it follows that $\epsilon^T u = 0$ and
\[
u_j = -\sum^n_{t \neq i, j} u_t - u_i = -\sum^n_{t \neq i, j} \bar{x}_t - u_i = \bar{x}_i + \bar{x}_j - u_i.
\]
Then,
\[
f(u) = \frac{1}{2} \sum^m_{h=1} \left( (A_{hi} - A_{hj}) u_i + \sum_{t \neq i, j} (A_{ht} - A_{hj}) \bar{x}_t - y_h \right)^2
\]
\[+ \lambda \left( |u_i| + |u_i - \bar{x}_i - \bar{x}_j| + \sum_{t \neq i, j} |\bar{x}_t| \right).\]

Now, let us define the vector $\rho \in \mathbb{R}^m$ as
\[
\rho_h = y_h - \sum_{t \neq i, j} (A_{ht} - A_{hj}) \bar{x}_t, \quad h = 1, \ldots, m,
\]
so that
\[
f(u) = \frac{1}{2} \alpha u_i^2 - \rho^T (A^i - A^j) u_i + \frac{1}{2} \lambda \left( |u_i| + |u_i - \bar{x}_i - \bar{x}_j| + \sum_{t \neq i, j} |\bar{x}_t| \right) + c.
\]
For all $h = 1, \ldots, m$, we can write
\[
\rho_h = \sum_{t \neq i, j} A_{ht} \bar{x}_t + A_{hj} \sum_{t \neq i, j} \bar{x}_t = y_h - (A\bar{x})_h + A_{hi} \bar{x}_i + A_{hj} \bar{x}_j + A_{hj} \sum_{t \neq i, j} \bar{x}_t,
\]
\[
= y_h - (A\bar{x})_h + (A_{hi} - A_{hj}) \bar{x}_i,
\]
where the last inequality follows from the fact that $\bar{x}_j = -\sum_{t \neq i, j} \bar{x}_t - \bar{x}_i$, since $\epsilon^T \bar{x} = 0$. Therefore,
\[
\rho = y - A\bar{x} + (A^i - A^j) \bar{x}_i \tag{38}
\]
and we obtain
\[
f(u) = \frac{1}{2} \alpha u_i^2 - \rho^T (A^i - A^j) u_i + \lambda \left( |u_i| + |u_i - \bar{x}_i - \bar{x}_j| \right) + c.
\]
Finally, since $(A\bar{x} - y)^T A^i = \nabla_i \varphi(\bar{x})$ and $(A\bar{x} - y)^T A^j = \nabla_j \varphi(\bar{x})$, from (38) it follows that
\[
\rho^T (A^i - A^j) = \|A^i - A^j\|^2 \bar{x}_i - \nabla_i \varphi(\bar{x}) + \nabla_j \varphi(\bar{x}) = \alpha \bar{x}_i - \nabla_i \varphi(\bar{x}) + \nabla_j \varphi(\bar{x}) = \beta,
\]
thus proving (37).
Appendix A.1. Ensuring strict convexity

In the convergence analysis of AS-ZSL, we require $f$ to be strictly convex along $\pm(e_i - e_j)$ for every index pair $(i, j)$ selected by Strategy MVP or Strategy AC2CD (see Subsection 4.1). Using Proposition A.1, it is now easy to show that this requirement is satisfied if and only if $A^i \neq A^j$.

**Corollary A.2.** Let $\bar{x}$ be a feasible point for problem (1). For any $i \neq j$, we have that $f$ is strictly convex over the line $\{\bar{x} + \xi(e_i - e_j)\}$, $\xi \in \mathbb{R}$ if and only if $A^i \neq A^j$.

**Proof.** The result follows from Proposition A.1, observing that $f_{\bar{x}j}$ is strictly convex if and only if $A^i \neq A^j$ (from the expression of $\alpha$) and that any pair of distinct points $x'$, $x''$ over the line $\{\bar{x} + \xi(e_i - e_j)\}$, $\xi \in \mathbb{R}$ can be expressed as $x' = \bar{x} + \xi'(e_i - e_j)$ and $x'' = \bar{x} + \xi''(e_i - e_j)$, for some $\xi' \neq \xi''$.

When $A^i = A^j$, in the next proposition we show the variable $x_i$ can be safely removed from the problem.

**Proposition A.3.** Assume that $A^i = A^j$ for some $i \neq j$. If $x^*$ is an optimal solution of $\min\{f(x): e^T x = 0, x_i = 0\}$, then $x^*$ is an optimal solution of problem (1) as well.

**Proof.** By contradiction, assume that $x^*$ is not an optimal solution of problem (1). Then, there exists a feasible point $x'$ for problem (1) such that $f(x') < f(x^*)$. Now, let us define $x''$ as follows:

$$
x''_h = \begin{cases} 
  x'_h, & \text{if } h \in \{1, \ldots, n\} \setminus \{i, j\}, \\
  0, & \text{if } h = i, \\
  x'_i + x'_j, & \text{if } h = j.
\end{cases}
$$

Clearly, $e^T x'' = e^T x' = 0$. Since $A^i = A^j$, we have $Ax'' = Ax'$ and, using the triangular inequality, $\|x''\|_1 \leq \|x'\|_1$. It follows that $f(x'') \leq f(x') < f(x^*)$, contradicting the fact that $x^*$ is an optimal solution of $\min\{f(x): e^T x = 0, x_i = 0\}$.

In conclusion, Proposition A.3 and Corollary A.2 suggest a simple procedure to ensure that, after a finite number of iterations, $f$ is strictly convex along $\pm(e_i - e_j)$ for every index pair $(i, j)$ selected by Strategy MVP or Strategy AC2CD. Namely, if we find two identical columns $A^i$ and $A^j$, we can simply fix $x_i = 0$ and remove this variable from problem (1) (together with the column $A^i$). We note that checking if two columns $A^i$ and $A^j$ are identical does not require additional computational burden because, as explained below, $\|A^i - A^j\|$ must be computed anyway (in order to calculate the coefficients $\alpha$ and $\beta$ appearing in Proposition A.1).

Appendix A.2. Computing the optimal solution

Given a feasible point $\bar{x}$ of problem (1) and an index pair $(i, j)$, with $i \neq j$, now we show how to compute

$$
\hat{x} = \bar{x} + \xi^*(e_i - e_j),
\xi^* \in \underset{\xi \in \mathbb{R}}{\text{Argmin}}\{f(\bar{x} + \xi(e_i - e_j))\}.
$$
According to (17) and (19), a computation of this form is needed for the variable update both when using Strategy MVP and when using Strategy AC2CD. Note that $\hat{x}$ can be equivalently obtained as an optimal solution of the following problem:

$$\min f(x)$$
$$x \in \{\bar{x} + \xi (e_i - e_j), \xi \in \mathbb{R}\}.$$ (39)

So, in view of Proposition A.1, we can calculate

$$u^* \in \text{Argmin } f_{x}^{i,j}(u)$$

and set

$$\hat{x}_h = \begin{cases} u^*, & \text{if } h = i, \\ \bar{x}_h, & \text{if } h \neq i, j, \\ -\sum_{t \neq j} \hat{x}_t, & \text{if } h = j. \end{cases}$$

To compute $u^*$, let us first recall that, from Proposition A.1, we have

$$f_{x}^{i,j}(u) = \frac{1}{2} \alpha u^2 - \beta u + \lambda(|u| + |u - \bar{x}_i - \bar{x}_j|) + c,$$ (40)

where $\alpha = \|A^i - A^j\|^2$, $\beta = \alpha \bar{x}_i - \nabla_i \varphi(\bar{x}) + \nabla_j \varphi(\bar{x})$ and $c$ is a constant. Since $f_{x}^{i,j}$ is coercive, then it has a minimizer. If $\alpha = 0$ (i.e., if $A^i = A^j$), we explained above that we can simply fix $x_i = 0$ and remove this variable from the problem. So, here we only focus on $\alpha > 0$. In this case, observe that $f_{x}^{i,j}$ is strictly convex, has a unique minimizer $u^*$ and we can write

$$f_{x}^{i,j}(u) = \begin{cases} \frac{1}{2} \alpha u^2 - \beta u + 2\lambda u - \lambda(\bar{x}_i + \bar{x}_j) + c, & \text{if } u \geq 0 \text{ and } u \geq \bar{x}_i - \bar{x}_j, \\ \frac{1}{2} \alpha u^2 - \beta u - 2\lambda u + \lambda(\bar{x}_i + \bar{x}_j) + c, & \text{if } u \leq 0 \text{ and } u \leq \bar{x}_i - \bar{x}_j, \\ \frac{1}{2} \alpha u^2 - \beta u + \lambda(\bar{x}_i + \bar{x}_j) + c, & \text{if } u \geq 0 \text{ and } u < \bar{x}_i - \bar{x}_j, \\ \frac{1}{2} \alpha u^2 - \beta u - \lambda(\bar{x}_i + \bar{x}_j) + c, & \text{if } u \leq 0 \text{ and } u > \bar{x}_i - \bar{x}_j. \end{cases}$$

Hence, we can first seek a stationary point of $f_{x}^{i,j}$ where the function is differentiable, i.e., in $\mathbb{R} \setminus \{0, \bar{x}_i + \bar{x}_j\}$. If such a stationary point exists, then it will be the desired minimizer $u^*$. Otherwise, $u^*$ will be a point of non-differentiability, that is, either 0 or $\bar{x}_i + \bar{x}_j$. This procedure is reported in Algorithm 2.

Note that, in addition to $\nabla_i \varphi(\bar{x})$ and $\nabla_j \varphi(\bar{x})$, in Algorithm 2 we only have to compute $\|A^i - A^j\|^2$ to get $\alpha$ and $\beta$, with a cost of $O(m)$ operations.
A decomposition method for lasso problems with zero-sum constraint

Algorithm 2 to compute the minimizer $u^*$ of (40)

seek a stationary point

0 Set $\bar{u} = \frac{\beta - 2\lambda}{\alpha}$

1 If $\bar{u} > \max\{\bar{x}_i + \bar{x}_j, 0\}$
2 Set $u^* = \bar{u}$ and EXIT (stationary point found)

3 Else
4 Set $\bar{u} = \frac{\beta + 2\lambda}{\alpha}$
5 If $\bar{u} < \min\{\bar{x}_i + \bar{x}_j, 0\}$
6 Set $u^* = \bar{u}$ and EXIT (stationary point found)

7 Else
8 Set $\bar{u} = \frac{\beta}{\alpha}$
9 If $\bar{u}(\bar{u} - \bar{x}_i - \bar{x}_j) < 0$
10 Set $u^* = \bar{u}$ and EXIT (stationary point found)

11 End if
12 End if
13 End if

stationary point not found, the minimizer is a point of non-differentiability

14 If $f_{i,j}(0) \leq f_{i,j}^{\bar{x}_i + \bar{x}_j}$, then set $u^* = 0$
15 Else, set $u^* = \bar{x}_i + \bar{x}_j$

References


