Cooperative Learning for Multi-view Analysis

Daisy Yi Ding\(^2\), Balasubramanian Narasimhan\(^{1,2}\), and Robert Tibshirani\(^{1,2}\)

\(^1\)Department of Statistics, Stanford University
\(^2\)Department of Biomedical Data Science, Stanford University

Abstract

We propose a new method for supervised learning with multiple sets of features ("views"). The multi-view problem is especially important in biology and medicine, where "-omics" data such as genomics, proteomics and radiomics are measured on a common set of samples. Cooperative learning combines the usual squared error loss of predictions with an "agreement" penalty to encourage the predictions from different data views to agree. By varying the weight of the agreement penalty, we get a continuum of solutions that include the well-known early and late fusion approaches. Cooperative learning chooses the degree of agreement (or fusion) in an adaptive manner, using a validation set or cross-validation to estimate test set prediction error. One version of our fitting procedure is modular, where one can choose different fitting mechanisms (e.g. lasso, random forests, boosting, neural networks) appropriate for different data views. In the setting of cooperative regularized linear regression, the method combines the lasso penalty with the agreement penalty. The method can be especially powerful when the different data views share some underlying relationship in their signals that can be exploited to strengthen signal, while each view has its idiosyncratic noise that needs to be reduced. We show that cooperative learning achieves higher predictive accuracy on simulated data and real multiomics examples of cancer stage and treatment response prediction. Leveraging aligned signals and allowing flexible fitting mechanisms for different modalities, cooperative learning offers a powerful approach to multiomics data fusion.

1 Introduction

With new technologies in biomedicine, we are able to generate and collect data of various modalities, including genomics, epigenomics, transcriptomics, and proteomics (Figure 1A). Integrating heterogeneous features on a single set of observations provides a unique opportunity to gain a comprehensive understanding of an outcome of interest. It offers the potential for making discoveries that are hidden in data analyses of a single modality and achieving more accurate predictions of the outcome (Kristensen et al. 2014, Ritchie et al. 2015, Gligorijević et al. 2016, Karczewski & Snyder 2018, Ma et al. 2020). While "multi-view data analysis" can mean different things, we use it here in the context of supervised learning, where the goal is to fuse different data views to model an outcome of interest.

To give a concrete example, assume that a researcher wants to predict cancer outcomes from RNA expression and DNA methylation measurements for a set of patients. The researcher suspects that: (1) both data views could potentially have prognostic value; (2) the two views share some underlying relationship with each other, as DNA methylation regulates gene expression and can repress the expression of tumor suppressor genes or promote the expression of oncogenes. Should the researcher use both data views for downstream prediction, or just use one view or the other? If using both views, how can the researcher leverage their underlying relationship in making more accurate prediction? Is there a way to strengthen the shared signals in the two data views while reducing idiosyncratic noise?
Advances in biotechnologies have enabled the collection of a myriad of “-omics” data ranging from genomics to proteomics on a common set of samples. These data capture the molecular variations of human health at multiple levels and can help us understand complex biological systems in a more comprehensive way. Multiomics data fusion offers the potential to improve the predictive accuracy of disease phenotypes and treatment response, thus enabling better diagnostics and therapeutics. However, multi-view analysis of omics data presents computational challenges such as increased dimensionality, noise and complexity.

Commonly-used approaches to the problem can be broadly categorized into early and late fusion. Early fusion begins by transforming all datasets into a single representation, which is then used as the input to a supervised learning model of choice. Late fusion works by developing first-level models from individual data views and then combining the predictions by training a second-level model as the final predictor. Encompassing early and late fusion, cooperative learning combines the usual squared error loss of predictions with an agreement penalty term to encourage the predictions from different data views to align.

There are two broad categories of existing “data fusion methods” for the multi-view problem, as shown in Figure 1B. They differ in the stage at which the “fusion” of predictors takes place, namely early fusion and late fusion. Early fusion works by transforming the multiple data views into a single representation before feeding the aggregated representation into a supervised learning model of choice (Yuan et al. 2014, Gentles et al. 2015, Liu et al. 2016, Perkins et al. 2018, Chaudhary et al. 2018). The simplest approach is to column-wise concatenate the $M$ datasets $X_1, \ldots, X_M$ to obtain a combined matrix $X$, which is then used as the input to a supervised learning model. Another type of approach is to project each high-dimensional dataset into a low-dimensional space using methods such as principal component analysis (PCA) or autoencoders (Wold et al. 1987, Vincent et al. 2010). Then we combine the low-dimensional representations through aggregation and feed the aggregated matrix into a supervised learning model. Early fusion approaches have an important limitation that it does not explicitly leverage the underlying relationship across data views. Late fusion, or “integration” refers to methods where individual models are first built from the distinct data views, and then the predictions of the individual models are combined into the final predictor (Yang et al. 2010, Zhao et al. 2019, Cheerla & Gevaert 2019, Chen, Lu, Wang, Williamson, Rodig, Lindeman & Mahmood 2020, Chabon et al. 2020, Wu et al. 2020).

In this paper, we propose a new method to multi-view data analysis called cooperative learning, a supervised learning approach that fuses the different views in a cooperative way. The method combines the usual squared error loss of predictions with an “agreement” penalty that encourages the predictions from different data views to align. By varying the weight of the agreement penalty, we get a continuum of solutions that include the commonly-used early and late fusion approaches. Our proposal can be especially powerful when the different data views share some underlying relationship that can be leveraged to strengthen signal, while each data view also has its idiosyncratic noise that needs to be reduced.
The rest of the paper is organized as follows. In Section 2, we introduce cooperative learning and characterize its solution. This involves the iterative algorithm for general form of cooperative learning and the explicit closed form solutions for cooperative regularized linear regression. We discuss its relation with early and late fusion, as well as other existing approaches. In Section 3, we extend cooperative learning to settings when we have more than two data views. We demonstrate in Section 4 the effectiveness of cooperative learning in simulation studies, where we compare it to several commonly-used approaches. In Section 5, we apply cooperative learning on some real multiomics datasets and show that it achieves higher predictive accuracy on cancer stage and treatment response prediction. In Section 6, we discuss how cooperative learning can be extended to generalized linear models and Cox proportional hazards models. We outline how the framework can be extended to paired features and interaction models in Section 7. The paper ends with a discussion and an appendix.

2 Cooperative learning

2.1 Cooperative learning with two data views

We begin with a simple form of our proposal for the population (random variable) setting. Let \( X \in \mathcal{R}^{n \times p_x} \), \( Z \in \mathcal{R}^{n \times p_z} \) — representing two data views — and \( y \in \mathcal{R}^n \) be a real-valued random variable (the target). Fixing the hyperparameter \( \rho \geq 0 \), we propose to minimize the population quantity:

\[
\min \mathbb{E}\left[ \frac{1}{2}(y - f_X(X) - f_Z(Z))^2 + \frac{\rho}{2}(f_X(X) - f_Z(Z))^2 \right].
\]

(1)

The first term above is the usual prediction error, while the second term is an “agreement” penalty, encouraging the predictions from different views to agree. This penalty term is related to “contrastive learning” (Chen, Kornblith, Norouzi & Hinton 2020, Khosla et al. 2020, Chen & He 2021, Jaiswal et al. 2021), which we discuss in more detail in Section 2.6. For more than two views, this generalizes easily, with details given below in Section 3.

The solution to (1) has fixed points:

\[
\begin{align*}
    f_X(X) &= \mathbb{E}\left[ \frac{y}{1 + \rho} - \frac{(1 - \rho)f_Z(Z)}{(1 + \rho)} | X \right], \\
    f_Z(Z) &= \mathbb{E}\left[ \frac{y}{1 + \rho} - \frac{(1 - \rho)f_X(X)}{(1 + \rho)} | Z \right].
\end{align*}
\]

(2)

We can optimize the objective by repeatedly updating the fit for each data view in turn, holding the other view fixed. When updating a function, this approach allows us to apply the fitting method for that data view to a penalty-adjusted “partial residual”.

The following relationships to early and late fusion can be seen immediately:

- If \( \rho = 0 \), from (1) we see that cooperative learning chooses a functional form for \( f_X \) and \( f_Z \) and fits them together. If these functions are additive (for example, linear) then it yields a simple form of early fusion, where we simply use the combined set of features in a supervised learning procedure.
• If $\rho = 1$, then from (2) we see that the solutions are the average of the marginal fits for $X$ and $Z$. This is a simple form of late fusion. This also holds when $X$ and $Z$ are independent, so that $E(f_Z(Z)|X)$ does not depend on $X$, similarly for $f_X(X)$.

We explore the relation of cooperative learning to early/late fusion in more detail in Section 2.4 in the setting of regularized linear regression.

Note that this fitting procedure is modular, so that we can choose a fitting mechanism appropriate for each data view. Specifically:

• For quantitative features like gene expression, copy number variation, or methylation: regularized regression (lasso, elastic net), a generalized additive model, boosting, random forests, or neural networks.
• For images: a convolutional neural network.
• For time series data: an auto-regressive model or a recurrent neural network.

Remark A. The iteration (2) looks much like the backfitting algorithm for generalized additive models (Hastie & Tibshirani 1990). In that setting, each of $f_X$ and $f_Z$ are typically functions of one-dimensional features $X$ and $Z$, and the backfitting algorithm iterations correspond to (2) with $\rho = 0$. In the additive model setting, backfitting is a special case of the Gauss-Seidel algorithm (Hastie & Tibshirani 1990).

In cooperative learning, each of $X, Z$ are views with multiple features; we could use an additive model for each view, i.e. $f_X(X) = \sum_j g_j(x_j)$, $f_Z(Z) = \sum_j h_j(z_j)$. Then each of the iterations in (2) could be solved using a backfitting algorithm, leading to a nested procedure.

2.2 Cooperative regularized linear regression

We make our proposal more concrete in the setting of cooperative regularized linear regression. Consider feature matrices $X \in \mathbb{R}^{n \times p_x}$, $Z \in \mathbb{R}^{n \times p_z}$, and our target $y \in \mathbb{R}^n$. We assume that the columns of $X$ and $Z$ have been standardized, and $y$ has mean 0 (hence we can omit the intercept below). For a fixed value of the hyperparameter $\rho \geq 0$, we want to find $\theta_x \in \mathbb{R}^{p_x}$ and $\theta_z \in \mathbb{R}^{p_z}$ that minimize:

$$J(\theta_x, \theta_z) = \frac{1}{2}||y - X\theta_x - Z\theta_z||^2 + \frac{\rho}{2}||(X\theta_x - Z\theta_z)||^2 + \lambda_x P_x(\theta_x) + \lambda_z P_z(\theta_z),$$

where $\rho$ is the hyperparameter that controls the relative importance of the agreement term $||(X\theta_x - Z\theta_z)||^2$ in the objective, and $P_x$ and $P_z$ are penalty functions. Most commonly, we use $\ell_1$ penalties, giving the objective function:

$$J(\theta_x, \theta_z) = \frac{1}{2}||y - X\theta_x - Z\theta_z||^2 + \frac{\rho}{2}||(X\theta_x - Z\theta_z)||^2 + \lambda_x ||\theta_x||_1 + \lambda_z ||\theta_z||_1.$$

Note that when $\rho = 0$, this reduces to early fusion, where we simply concatenate the columns of $X$ and $Z$ and apply lasso. Furthermore, in Section 2.4 we show that $\rho = 1$ yields a late fusion estimate.

In our experiments, we standardize the features and simply take $\lambda_x = \lambda_z = \lambda$. We have found that generally there is often no advantage to allowing different $\lambda$ values for different views. But for completeness, in Appendix
Algorithm 1 Direct algorithm for cooperative regularized regression.

Input: \( X \in \mathbb{R}^{n \times p_x} \) and \( Z \in \mathbb{R}^{n \times p_z} \), the response \( y \in \mathbb{R}^n \), and a grid of hyperparameter values \((\rho_{\text{min}}, \ldots, \rho_{\text{max}})\).

\[
\text{for } \rho \leftarrow \rho_{\text{min}}, \ldots, \rho_{\text{max}} \text{ do}
\]
\[
\hat{X} = \left( \begin{array}{cc} X & Z \\ -\sqrt{\rho}X & \sqrt{\rho}Z \end{array} \right), \hat{y} = \left( \begin{array}{c} y \\ 0 \end{array} \right).
\]
\[
\text{Solve Lasso}(\hat{X}, \hat{y}, \lambda) \text{ over a decreasing grid of } \lambda \text{ values.}
\]
\[
\text{end}
\]

Select the optimal value of \( \rho^* \) based on the CV error and get the final fit.

Section 9.1, we outline an adaptive strategy for optimizing over \( \lambda_x \) and \( \lambda_z \). We call this adaptive cooperative learning in our simulation and real data studies.

With a common \( \lambda \) the objective becomes
\[
J(\theta_x, \theta_z) = \frac{1}{2} ||y - X\theta_x - Z\theta_z||^2 + \frac{\rho}{2} ||(X\theta_x - Z\theta_z)||^2 + \lambda(||\theta_x||_1 + ||\theta_z||_1),
\]
and we can compute a regularization path of solutions indexed by \( \lambda \).

Problem (5) is convex, and the solution can be computed as follows. Letting
\[
\hat{X} = \left( \begin{array}{cc} X & Z \\ -\sqrt{\rho}X & \sqrt{\rho}Z \end{array} \right), \hat{y} = \left( \begin{array}{c} y \\ 0 \end{array} \right), \hat{\beta} = \left( \begin{array}{c} \theta_x \\ \theta_z \end{array} \right),
\]
then the equivalent problem to (5) is
\[
J(\theta_x, \theta_z) = \frac{1}{2} ||\hat{y} - \hat{X}\hat{\beta}||^2 + \lambda(||\theta_x||_1 + ||\theta_z||_1).
\]
This is a form of the lasso, and can be computed, for example by the glmnet package (Friedman et al. 2010).

This new problem has \( 2n \) observations and \( p_x + p_z \) features.

Let Lasso\((X, y, \lambda)\) denote the generic problem:
\[
\min_\beta \frac{1}{2} ||y - X\beta||^2 + \lambda \|eta\|_1.
\]

We outline the direct algorithm for cooperative regularized regression in Algorithm 1.

\textbf{Remark B.} We note that for cross-validation (CV) we do not form folds from the rows of \( \hat{X} \), but instead form folds from the rows of \( X \) and \( Z \) and then construct the corresponding \( \hat{X} \).

\textbf{Remark C.} We can add \( \ell_2 \) penalties to the objective in (5), replacing \( \lambda(||\theta_x||_1 + ||\theta_z||_1) \) by the elastic net form
\[
\lambda \left[ (1 - \alpha)(||\theta_x||_1 + ||\theta_z||_1) + \alpha(||\theta_x||_2^2/2 + ||\theta_z||_2^2/2) \right].
\]
This leads to elastic net fitting, in place of the lasso, in the last step of the algorithm. This option will be included in our publicly available software implementation of cooperative learning.

We show here an illustrative simulation study of cooperative learning in the regression setting in Figure 2A. We will discuss more comprehensive studies and simulation details in Section 4.1. In Figure 2A, the left
and middle panels correspond to the settings where the two data views $X$ and $Z$ are correlated, while in the right panel $X$ and $Z$ are uncorrelated. We see that when the data views are correlated, cooperative learning offers significant performance gains over the early and late fusion methods, by encouraging the predictions from different views to agree. When the data views are uncorrelated and only one view $X$ contains signal as in the right panel, early and late fusion methods hurt performance as compared to the separate model fit on only $X$, while adaptive cooperative learning is able to perform on par with the separate model.

![Figure 2: An illustrative simulation study of cooperative learning in the regression setting, and sparsity of the solution.](image)

Cooperative learning achieves superior prediction accuracy on a test set when the data views $X$ and $Z$ are correlated. The y-axis shows the mean squared error (MSE) on a test set. The methods in comparison from left to right in each panel correspond to (1) Separate $X$: lasso applied on the data view $X$ only; (2) Separate $Z$: lasso applied on the data view $Z$ only; (3) Early fusion: lasso applied on the concatenated data views of $X$ and $Z$; (4) Late fusion: separate lasso models are fit on $X$ and $Z$ independently and the predictors are then combined through linear least squares; (5) Coop: cooperative learning as outlined in Algorithm 1; (6) Adap Coop: adaptive cooperative learning as outlined in Algorithm 4. Note that the test MSE in each panel is of a different scale because we experiment with simulating the data of different signal-to-noise ratios (SNR). We conducted each simulation experiment 10 times. (B) The number of non-zero coefficients as a function of the $\ell_1$ norm of the solution with different values of the weight on the agreement penalty term $\rho$: the solution becomes less sparse as $\rho$ increases.

### 2.3 One-at-a-time algorithm for cooperative regularized linear regression

As an alternative, one can optimize by iteratively optimizing over $\theta_x$ and $\theta_z$, fixing one and optimizing over the other. The updates are as follows:

\[
\hat{\theta}_x = \text{Lasso}(X, y_x^*, \lambda_x), \text{ where } y_x^* = \frac{y}{1 + \rho} - \frac{(1 - \rho)Z\theta_z}{(1 + \rho)}
\]

\[
\hat{\theta}_z = \text{Lasso}(Z, y_z^*, \lambda_z), \text{ where } y_z^* = \frac{y}{1 + \rho} - \frac{(1 - \rho)X\theta_x}{(1 + \rho)}.
\]

(10)

This is analogous to the general iterative procedure in [2]. It is summarized in Algorithm [2].
Algorithm 2 One-at-a-time algorithm for cooperative regularized regression.

**Input:** \( X \in \mathbb{R}^{n \times p_x} \) and \( Z \in \mathbb{R}^{n \times p_z} \), the response \( y \in \mathbb{R}^n \), and a grid of hyperparameter values \((\rho_{\text{min}}, \ldots, \rho_{\text{max}})\).

Fix the lasso penalty weights \( \lambda_x \) and \( \lambda_z \), for \( \rho \leftarrow \rho_{\text{min}}, \ldots, \rho_{\text{max}} \) do

for \( k \leftarrow 0, 1, 2, \ldots \) until convergence do

1. Set \( y^*_{x} = \frac{y}{1+\rho} - \frac{(1-\rho)Z \theta_z}{(1+\rho)} \). Solve Lasso(\( X, y^*_{x}, \lambda_x \)) and update \( \theta_x^{(k+1)} \) to be the solution.

2. Set \( y^*_{z} = \frac{y}{1+\rho} - \frac{(1-\rho)X \theta_x}{(1+\rho)} \). Solve Lasso(\( Z, y^*_{z}, \lambda_z \)) and update \( \theta_z^{(k+1)} \) to be the solution.

end

end

Select the optimal value of \( \rho^* \) based on the sum of the CV errors and get the final fit.

By iterating back and forth between the two lasso problems, we can find the optimal solution to (4). When both \( X \) and \( Z \) have full column rank, the problem (4) is strictly convex and each iteration decreases the overall objective value. Therefore, the one-at-a-time procedure is guaranteed to converge. In general, it can be shown to converge to some stationary point, using results such as those in Tibshirani (2017). This algorithm uses fixed values for \( \lambda_x, \lambda_z \); we need to run the algorithm over a grid of such values, or use cross-validation to choose \( \lambda_x, \lambda_z \) within each iteration.

With just two views, there seems to be no advantage to this approach over the direct solution given in Algorithm 1. However, for a larger number of views, there can be a computational advantage, which we will discuss in Section 3.

2.4 Relation to early/late fusion

From the objective functions (3) and (4), when the weight on the agreement term \( \rho \) is set to 0, cooperative learning (regression) reduces to early fusion: we simply concatenate the columns of different views and apply lasso or another regularized regression method.

Next we discuss the relation of cooperative learning to late fusion. Since \( X \) and \( Z \) have centered columns, from (25) we obtain

\[
\hat{\theta}_x = \left( \begin{array}{ccc}
2N & 0 & 0 \\
0 & X^T X(1+\rho) & X^T Z(1-\rho) \\
0 & Z^T X(1-\rho) & Z^T Z(1+\rho)
\end{array} \right)^{-1} \left( \begin{array}{c}
1^T y \\
X^T y \\
Z^T y
\end{array} \right).
\]

Assuming \( X \) and \( Z \) have full rank, and omitting the \( \ell_1 \) penalties, we obtain the least squares estimates

\[
\begin{pmatrix}
\hat{\theta}_x \\
\hat{\theta}_z
\end{pmatrix} = \left( \begin{array}{ccc}
2N & 0 & 0 \\
0 & X^T X(1+\rho) & X^T Z(1-\rho) \\
0 & Z^T X(1-\rho) & Z^T Z(1+\rho)
\end{array} \right)^{-1} \left( \begin{array}{c}
1^T y \\
X^T y \\
Z^T y
\end{array} \right).
\]

If \( X^T Z = 0 \) (uncorrelated features between the views) or \( \rho = 1 \), this reduces to the average of the least squares estimates for each block. The above relation also holds when we include the \( \ell_1 \) penalties.

This calculation suggests a scaling of the final estimates by \((1+\rho)\), but we have not found this to be helpful in our simulation studies. It also suggests that restricting \( \rho \) to be in \([0,1]\) would be natural. However, we have
found that values larger than 1 can sometimes yield lower prediction error (see the simulation studies in Section 4.1).

2.5 Sparsity of the solution and degrees of freedom

In the lasso setting, we have seen the cooperative learning estimator can be re-expressed as a lasso with modified features and responses as in (26). Thus, the degrees of freedom of the fit is simply the number of non-zero coefficients in the solution, see e.g. (Tibshirani & Taylor 2011).

In Figure 2, we explore how the sparsity of the solution depends on the agreement hyperparameter ρ. We did 100 simulations of Gaussian data with n = 100 and p = 20 in each of two views, with all coefficients equal to 2.0. The standard deviation of the errors was chosen so that the SNR was about 2. The figure shows the number of non-zero coefficients as a function of the overall ℓ_1 of the solutions, for different values of ρ. Note that the lasso parameter λ is varying along the horizontal axis; we chose to plot against the ℓ_1 norm, a more meaningful quantity. We see that the solutions become less sparse as ρ increases, much like the behavior that one sees in the elastic net.

2.6 Relation to existing approaches

We have mentioned the close connection of cooperative learning to early and late fusion: setting ρ = 0 or 1 gives a version of each of these, respectively. There are many variations of late fusion, including the use of stacked generalization to combine the predictions at the last stage (Garcia-Ceja et al. 2018).

Cooperative learning is also related to collaborative regression (Gross & Tibshirani 2015). This method uses an objective function of the form

\[
\begin{align*}
\frac{b_{xy}}{2}||y - X\theta_x||^2 + \frac{b_{zy}}{2}||y - Z\theta_z||^2 + \frac{b_{xz}}{2}||X\theta_x - Z\theta_z||^2.
\end{align*}
\]

With ℓ_1 penalties added, this is proposed as a method for sparse supervised canonical correlation analysis. It is different from cooperative learning in an important way: here X and Z are not fit jointly to the target. The authors state that collaborative regression is not well suited to the prediction task. We note that if \(b_{xy} = b_{zy} = b_{xz} = 1\), each of \(\hat{\theta}_x, \hat{\theta}_z\) are the one-half of the least squares (LS) estimates on X, Z respectively. Hence the overall prediction \(\hat{y}\) is the average of the individual LS predictions. This late fusion estimate is the same as that obtained from cooperative learning with ρ = 1. In addition, a related framework based on optimizing measures of agreement between data views was also proposed in Sindhwani et al. (2005), but it is different from cooperative learning in the sense that the data views are not used jointly to model the target.

Cooperative learning also has connections with contrastive learning (Becker & Hinton 1992, Chen, Kornblith, Norouzi & Hinton 2020, Khosla et al. 2020, Chen & He 2021, Jaiswal et al. 2021). This method is an unsupervised learning technique first proposed for learning visual representations. Without the supervision of y, it learns representations of images by maximizing agreement between differently augmented “views” of the same data example. While both contrastive learning and cooperative learning have a term in the objective that encourages agreement between correlated views, our method combines the agreement term with the usual prediction error loss and is thus supervised.

We next discuss the relation of cooperative learning to a recently proposed method for multi-view analysis called sparse integrative discriminant analysis (SIDA) (Safo et al. 2021). This method aims to identify variables that are associated across views while also able to optimally separate data points into different classes.
Specifically, it combines canonical correlation analysis and linear discriminate analysis by solving the following optimization problem. Let \( X_k = (x_{1k}, \ldots, x_{nk,k})^T \in \mathbb{R}^{n_k \times p} \), \( x_k \in \mathbb{R}^p \) be the data matrix for class \( k \), where \( k = 1, \ldots, K \), and \( n_k \) is the number of samples in class \( k \). Then, the mean vector for class \( k \) is \( \mu_k = \frac{1}{n_k} \sum_{i=1}^{n_k} x_{ik} \); the common variance matrix for all class is \( S_w = \sum_{k=1}^{K} \sum_{i=1}^{n} (x_{ik} - \mu_k)(x_{ik} - \mu_k)^T \); the between class covariance matrix is \( S_b = \sum_{k=1}^{K} n_k(\mu_k - \hat{\mu})(\mu_k - \hat{\mu})^T \), where \( \hat{\mu} = \frac{1}{\pi} \sum_{k=1}^{K} n_k \mu_k \) is the combined class mean vector. Assume that we have two data views \( X \in \mathbb{R}^{n \times p_x} \) and \( Z \in \mathbb{R}^{n \times p_z} \) with centered columns, we want to find \( A = [a_1, \ldots, a_K] \) and \( B = [b_1, \ldots, b_{K-1}] \) such that

\[
\min_{\rho} \rho \cdot \text{tr}(A^T S_{xz}^2 A + B^T S_{zz}^2 B) + (1 - \rho) \cdot \text{tr}(A^T S_{xz} S_{zz} B^T S_{xz}^T A)
\]

subject to \( \text{tr}(A^T S_{wz} A)/(K - 1) = 1 \) and \( \text{tr}(B^T S_{zz} B)/(K - 1) = 1 \),

where \( S_{xz} \in \mathbb{R}^{p_x \times p_z} \) is the sample cross-covariance matrix between \( X \) and \( Z \). Here, \( \text{tr}(\cdot) \) is the trace function, and \( \rho \) is the parameter that controls the relative importance of the “separation” term and the “association” terms in the objective. While SIDA also considers the association across data views by choosing vectors that are associated and able to separate data points into classes, it solves the problem in a “backward” manner, that is the features are modeled as a function of the outcome. Cooperative learning, in contrast, solves the problem in a “forward” manner \((Y \sim X, Z)\), which is more suitable for prediction.

We also note the connection between cooperative learning (regression) with the standardized group lasso \cite{Simon:2012}. This method is a variation of the group lasso \cite{Yuan:2006}, and uses

\[
\|X\theta_x\|_2 + \|Z\theta_z\|_2
\]

as the penalty term, rather than the sum of squared two norms. It encourages group-level sparsity by eliminating entire blocks of features at a time. In the group lasso, each block is a group of features, and we do not expect each block to be predictive on its own. This is different from cooperative learning, where each feature block is a data view and we generally do not want to eliminate an entire view for prediction. In addition, the standardized group lasso does not have an agreement penalty. One could in fact add the standardized group lasso penalty \cite{Simon:2012} to the cooperative learning objective, which would allow elimination of entire data views.

## 3 Cooperative learning with more than two data views

When we have more than two views of the data, \( X_1 \in \mathbb{R}^{n \times p_1}, X_2 \in \mathbb{R}^{n \times p_2}, \ldots, X_M \in \mathbb{R}^{n \times p_M} \), the population quantity that we want to minimize becomes

\[
\min \mathbb{E} \left[ \frac{1}{2} (y - \sum_{m=1}^{M} f_{X_m}(X_m))^2 + \frac{\rho}{2} \sum_{m\neq m'} (f_{X_m}(X_m) - f_{X_{m'}}(X_{m'}))^2 \right].
\]

As with two views, this can be optimized with an iterative algorithm that updates each \( f_{X_m}(X_m) \) as follows:

\[
f_{X_m}(X_m) = \mathbb{E} \left[ \frac{y}{1 + (M - 1)\rho} - \frac{(1 - \rho) \sum_{m' \neq m} f_{X_{m'}}(X_{m'})}{1 + (M - 1)\rho} | X_m \right].
\]

As in the two-view setup above, the fitter \( \mathbb{E}(\cdot|X_m) \) can be tailored to the data type of each view.

For regularized linear regression with more than two views, the objective becomes

\[
J(\theta_1, \theta_2, \ldots, \theta_M) = \frac{1}{2} ||y - \sum_{m=1}^{M} X_m \theta_m||^2 + \frac{\rho}{2} \sum_{m\neq m'} \|((X_m \theta_m - X_{m'} \theta_{m'})^2 + \sum_{m=1}^{M} \lambda_m \|\theta_m\|_1.
\]
This is again a convex problem. The optimal solution can be found by forming augmented data matrices as before in (25) and (26).

Let

\[
\tilde{X} = \begin{pmatrix}
X_1 & X_2 & \ldots & X_{M-1} & X_M \\
-\sqrt{\rho}X_1 & \sqrt{\rho}X_2 & \ldots & 0 & 0 \\
-\sqrt{\rho}X_1 & 0 & \ldots & \sqrt{\rho}X_{M-1} & 0 \\
0 & -\sqrt{\rho}X_2 & \ldots & 0 & \sqrt{\rho}X_M \\
0 & 0 & \ldots & 0 & \sqrt{\rho}X_M \\
\ldots & \ldots & \ldots & \ldots & \ldots \\
0 & 0 & \ldots & -\sqrt{\rho}X_{M-1} & \sqrt{\rho}X_M
\end{pmatrix},
\tilde{y} = \begin{pmatrix} y \\ 0 \\ 0 \\ 0 \\ 0 \\ \ldots \\ 0 \end{pmatrix}, \tilde{\beta} = \begin{pmatrix} \theta_1 \\ \theta_2 \\ \ldots \\ \theta_{M-1} \\ \theta_M \end{pmatrix},
\]

(20)

then the equivalent problem to (19) becomes

\[
\frac{1}{2}||\tilde{y} - \tilde{X}\tilde{\beta}||^2 + \sum_{m=1}^{M} \lambda_m ||\theta_m||_1.
\] (21)

With \(M\) views, the augmented matrix in (20) has \(n + \binom{M}{2} \cdot n\) rows, which could be computationally challenging to solve.

Alternatively, the optimal solution \(\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_M\) has fixed points

\[
\hat{\theta}_m = \text{Lasso}(X, y_m^*, \lambda_m), \text{ where } y_m^* = \left[ \frac{y}{1 + (M-1)\rho} - \frac{(1 - \rho) \sum_{m' \neq m} X_{m'} \theta_{m'}}{1 + (M-1)\rho} \right].
\] (22)

This leads to an iterative algorithm, where we successively solve each subproblem, until convergence. For a large number of views, this can be a more efficient procedure than the direct approach in (21) above.

4 Simulation studies

4.1 Simulation studies on cooperative regularized linear regression

Here we compare cooperative learning in the regression setting with early and late fusion methods in simulations. The set up is as follows. Given values for parameters \(n, p_x, p_z, p_r, s_u, t, \beta_x, \beta_z, \sigma\), we generate data according to the following procedure:

1. \(x_j \in \mathbb{R}^n\) distributed i.i.d. MVN(0, \(I_n\)) for \(j = 1, 2, \ldots, p_x\)
2. \(z_j \in \mathbb{R}^n\) distributed i.i.d. MVN(0, \(I_n\)) for \(j = 1, 2, \ldots, p_z\)
3. For \(i = 1, 2, \ldots, p_r\) (\(p_r < p_x\) and \(p_r < p_z\), where \(p_r\) corresponds to the number of relevant features):
   (a) \(u_i \in \mathbb{R}^n\) distributed i.i.d. MVN(0, \(s_u^2 I_n\))
   (b) \(x_i = x_i + t \ast u_i\)
   (c) \(z_i = z_i + t \ast u_i\)
4. \( X = [x_1, x_2, \ldots, x_p], \quad Z = [z_1, z_2, \ldots, z_p] \)

5. \( y = X\beta_x + Z\beta_z + \epsilon \) where \( \epsilon \in \mathbb{R}^n \) distributed i.i.d. MVN\((0, \sigma^2 I_n)\)

To introduce sparsity, we set \( \beta_x = [b_x, \ldots, b_x, 0, \ldots, 0] \) and \( \beta_z = [b_z, \ldots, b_z, 0, \ldots, 0] \), where \( b_x \) and \( b_z \) are the effect sizes of the relevant features in \( X \) and \( Z \), respectively. Data sets are simulated with different levels of correlation between the two data views \( X \) and \( Z \), different contributions of \( X \) and \( Z \) to the signal, and different signal-to-noise ratios (SNR). We consider the settings of both small \( p \) and large \( p \) regimes, and of both low and high SNR ratios. We use 10-fold cross-validation (CV) to select the optimal values of hyperparameters. We compare the following methods:

- Separate \( X \) and separate \( Z \): The standard lasso is applied on the separate data views of \( X \) and \( Z \) with 10-fold CV.
- Early fusion: The standard lasso is applied on the concatenated data views of \( X \) and \( Z \) with 10-fold CV. Note that this is equivalent to cooperative learning with \( \rho = 0 \).
- Late fusion: Separate lasso models are first fitted on \( X \) and \( Z \) independently with 10-fold CV, and the two resulting predictors are then combined through linear least squares for the final prediction.
- Cooperative learning (regression) and adaptive cooperative learning.

We evaluated the performance based on the mean-squared error (MSE) on a test set. We conducted each simulation experiment 10 times.

Overall, the simulation results can be summarized as follows:

- Cooperative learning performs the best in terms of test MSE across the range of SNR and correlation settings. It is most helpful when the data views are correlated and both contain signal (as in Figure 3A and Figure 3B). When the correlation between data views is higher, higher values of \( \rho \) are more likely to be selected.
- When only one data view contains signal but the two data views are correlated (as in Figure 3C), cooperative learning also gives significant performance gain, likely achieved through the agreement penalty term. In contrast, early and late fusion could hurt performance as compared to the separate models.
- Lastly, when only one view contains signal and the views are not correlated (as in Figure 3D), cooperative learning is outperformed by the separate model fit on the view containing the signal, but adaptive cooperative learning is able to perform on par with the separate model. And it again outperforms early and late fusion.
- Moreover, we also find that cooperative learning tends to yield a less sparse model, as expected from the results of Section 2.5.

See Section 9.3 in the Appendix for more comprehensive results across a wider range of simulation settings.

4.2 Simulation studies on cooperative learning with imaging and “omics” data

Here we extend the simulation studies for cooperative learning to the setting where we have two data views of more distinct data modalities, such as imaging and omics data (e.g. transcriptomics and proteomics). We tailor
Figure 3: Simulation studies on cooperative regularized linear regression. (A) Simulation results when X and Z have a medium level of correlation ($t = 2, s_u = 1$); X and Z both contain signal ($b_x = b_z = 2$), $n = 200, p = 1000$, SNR = 1.8. The first panel shows MSE on a test test; the second panel shows the MSE difference on the test set relative to early fusion; the third panel shows the number of features selected; the fourth panel shows the $\rho$ values selected by CV in cooperative learning. Here “Coop” refers to cooperative learning outlined in Algorithm 1 and “Adap Coop” refers to adaptive cooperative learning outlined in Algorithm 4. (B) Simulation results when X and Z have a high level of correlation ($t = 6, s_u = 1$); X and Z both contain signal ($b_x = b_z = 2$), $n = 200, p = 1000$, SNR = 1.0. (C) Simulation results when X and Z have a high level of correlation ($t = 6, s_u = 1$); only X contains signal ($b_x = 2, b_z = 0$), $n = 200, p = 1000$, SNR = 0.5. (D) Simulation results when X and Z have no correlation ($t = 0, s_u = 1$); only X contains signal ($b_x = 2, b_z = 0$), $n = 200, p = 1000$, SNR = 2.5.
the fitter suitable to each view, i.e. convolutional neural networks (CNN) for images and lasso for omics. In the study, we simulate the “omics” data \((X)\) and the “imaging” data \((Z)\) data such that they share some common factors. These factors are also used to generate the signal in the response \(y\). We use a factor model to generate the data, as it is a natural way to create correlations between \(X, Z\), and \(y\).

Assume that our task is to use the omics and imaging data to predict if a patient has a certain disease. We use CNN for modeling the imaging data and lasso for the omics data, and optimize the objective for the general form of cooperative learning as in (1) with the iterative algorithm outlined in (2). We consider both low and high SNR settings. In Appendix Section 9.2, we outline the full details of the simulation procedure.

Figure 4 shows some examples of the synthetic images generated for this study.

![Generated images for “healthy” and “disease” samples.](image)

We compare cooperative learning to the following methods: (1) Only images: a simple one-layer CNN with max pooling and ReLU activation is applied on the imaging data only; (2) Only omics: the standard lasso is applied on the omics data only; (3) Late fusion: separate models (CNN and lasso) are first fit on the imaging and omics data, respectively, and the resulting predictors are then combined through linear least squares using a validation set. We evaluated the performance based on the misclassification error on a test set, as well as the difference in misclassification error relative to late fusion. We conducted each simulation experiment 10 times.

The results are shown in Figure 5. We find that (1) late fusion achieves a lower misclassification error on the test set than the separate models; (2) cooperative learning outperforms late fusion and achieves the lowest test error by encouraging the predictions from the two views to agree; (3) cooperative learning is especially helpful when the SNR is low, while its benefit is less pronounced when the SNR is higher. The last observation makes sense, because when the SNR is lower the marginal benefit of leveraging the other view(s) in strengthening signal becomes larger.
Figure 5: Simulation studies on cooperative learning with imaging and “omics” data. Panel (A) corresponds to the relatively low SNR setting (SNR = 1) and panel (B) to the higher SNR setting (SNR = 6). For each setting, the left panel shows the misclassification error on the test set for CNN on only images, lasso on only omics, late fusion, and cooperative learning; the right panel shows the difference in misclassification error relative to late fusion. Here “Coop” refers to cooperative learning. For both settings, the range of $\rho$ values for cooperative learning to select from is (0,20). The average $\rho$ selected in the low SNR setting is 6.8 and in the high SNR setting is 8.0.

5 Real data examples

5.1 Multiomics studies on cancer stage prediction

We use publicly available data on kidney renal clear cell carcinoma from The Cancer Genome Atlas (TCGA) (Tomczak et al. 2015). The data set contains a cohort of 245 kidney renal clear cell carcinoma patients. We obtained RNA sequencing data for 19,533 genes and microRNA (miRNA) expression data for 732 miRNAs. We applied cooperative learning (regression) to predict the stage of cancer using RNA and miRNA expression data. We split the dataset of 245 patients in a 70/30 ratio into training and test sets of 170 and 75 patients, respectively. Both the RNA expression and miRNA measurements were screened by their variance across the subjects. We conducted the same set of experiment across 100 different random splits of the training and test sets.

| Methods                | Test MSE | Relative to Early Fusion | Support |
|------------------------|----------|--------------------------|---------|
|                        | Mean     | Std                      | Mean    | Std | Mean |
| Separate RNA           | 1.233    | 0.013                    | 0.005   | 0.009 | 22   |
| Separate miRNA         | 1.265    | 0.012                    | 0.037   | 0.009 | 23   |
| Early fusion           | 1.228    | 0.013                    | 0 | 0 | 41   |
| Late fusion            | 1.288    | 0.015                    | 0.060   | 0.011 | 28   |
| Cooperative learning   | **1.210**| **0.012**                | **-0.017** | **0.006** | **79** |
| Adaptive cooperative learning | 1.215 | 0.012                   | -0.013   | 0.007 | 68   |

Table 1: Results for the TCGA kidney cancer data set. The first two columns in the table show the mean and standard deviation (std) of MSE on the test set across different splits of the training and test sets; the third and fourth column show the MSE difference relative to early fusion; the last column shows the average number of features selected. The methods include (1) separate RNA: the standard lasso is applied on the RNA expression data only; (2) separate miRNA: the standard lasso is applied on the miRNA data only; (3) early fusion: the standard lasso is applied on the concatenated data of RNA and miRNA data; (4) late fusion: separate lasso models are first fit on RNA and miRNA independently and the predictors are then combined through linear least squares; (5) cooperative learning (Algorithm 1); (6) adaptive cooperative learning (Algorithm 4). The range of $\rho$ values is (0,9). The average of the selected $\rho$ values is 3.0 for cooperative learning and 2.1 for adaptive cooperative learning.

The results are shown in Table 1. Early fusion gives some performance gain over the models fit on separate
Table 2: Results for the treatment response data set. The first two columns in the table show the mean and standard deviation (std) of MSE on the test set across different splits of the training and test sets; the third and fourth column show the MSE difference relative to early fusion; the last column shows the average number of features selected. The methods include (1) separate gene expression: the standard lasso is applied on the gene expression data only; (2) separate CNV: the standard lasso is applied on the CNV data only; (3) early fusion: the standard lasso is applied on the concatenated data of gene expression and CNV data; (4) late fusion: separate lasso models are first fit on gene expression and CNV independently and the predictors are then combined through linear least squares; (5) cooperative learning (Algorithm 1); (6) adaptive cooperative learning (Algorithm 4). The range of \( \rho \) values is \((0, 9)\). The average of the selected \( \rho \) values is 2.8 for cooperative learning and 1.8 for adaptive cooperative learning.

| Methods                     | Test MSE  | Relative to Early Fusion | Support |
|-----------------------------|-----------|--------------------------|---------|
|                             | Mean  | Std  | Mean  | Std  | Mean  |
| Separate gene expression    | 1.878 | 0.187 | 0.027 | 0.027 | 14   |
| Separate CNV                | 1.804 | 0.177 | -0.048 | 0.078 | 4    |
| Early fusion                | 1.852 | 0.191 | 0 | 0 | 13   |
| Late fusion                 | 5.842 | 3.719 | 3.990 | 3.760 | 13   |
| Cooperative learning        | 1.735 | 0.172 | -0.117 | 0.061 | 30   |
| Adaptive cooperative learning | 1.914 | 0.205 | 0.063 | 0.078 | 26   |

Data views only. Cooperative learning outperforms both early and late fusion, achieving the lowest MSE on the test set. In addition, cooperative learning tends to select more number of features, which has the potential to enable new biological discoveries. In this case study, the features selected by early fusion is a subset of the features selected by cooperative learning. For example, cooperative learning identifies genes such as “IL6” for prediction, which is not identified by early fusion or the other methods. IL6, involved in immune-related signaling, has been shown to be an important factor for the development and spread of kidney renal clear cell carcinoma (The Cancer Genome Atlas Research Network 2013, Kamińska et al. 2015, Fu et al. 2015, Gudbrandsdottir et al. 2021).

5.2 Multiomics studies on treatment response prediction

Finally, we applied cooperative learning (regression) to a data set from a breast cancer treatment response study, as described in (Telli et al. 2015). The data were collected from a Phase 2 clinical trial of standard chemotherapy plus a PARP inhibitor in the neoadjuvant treatment of triple negative breast cancer. Details of the trial can be found with the identifier NCT00813956 at ClinicalTrials.gov. It involves 74 patients who underwent the treatment and had their measurements taken on gene expression and copy number variation (CNV) before the treatment. Six months after the treatment, each patient was evaluated with the Residual Cancer Burden (RCB) index, which assesses the pathologic response. The RCB index is a composite score of several metrics on the tumor, including primary tumor bed area, overall percentage cellularity, percentage of disease that is in situ, diameter of largest auxiliary metastasis, etc.

The goal of the analysis is to use the gene expression and CNV data to predict the treatment response, i.e. the RCB index. We split the data set of 74 patients into training and test sets of 50 and 24 patients, respectively. Both the gene expression and CNV measurements were screened by their variance across the subjects. We conducted the same set of experiments across 10 different random splits of the training and test sets.

The results are shown in Table 2. Neither early fusion nor late fusion gives any performance gain over lasso fit on the CNV data only. The performance of late fusion is especially poor in this case study, because the method splits out a validation set to combine the separate predictors through linear least squares, while the number of patients in this data set is small and does not support it to perform well. Cooperative learning achieves the lowest MSE on the test set. However, we also want to recognize that the null model, which simply
uses the mean of the data in the training set as the predictor for the test set, also performs very competitively, with a mean MSE of 1.697 and standard deviation of 0.173. This indicates that the signal is very low in the data set, which is often the case in multiomics studies. Cooperative learning is the only method that could perform on par. In addition, it identifies genes such as “NOTCH1”, “HSPG2”, “AKT” and “PRARγ”. These genes are not identified by the other methods but have been shown to be potential therapeutic targets for triple negative breast cancer (Yuan et al. 2015, Kalscheuer et al. 2019, Giuli et al. 2019, Lin et al. 2019, Augimeri et al. 2020, Miao et al. 2020, Martorana et al. 2021).

6 Cooperative generalized linear models and Cox regression

Here we describe how cooperative learning can be extended to generalized linear models (GLMs) (Nelder & Wedderburn 1972) and Cox proportional hazards models (Cox 1972).

Consider a GLM, consisting of 3 components: (1) a linear predictor: \( \eta = X\beta \); (2) a link function \( g \) such that \( E(Y|X) = g^{-1}(\eta) \); (3) a variance function as a function of the mean: \( V = V(E(Y|X)) \). For cooperative GLMs, we have the linear predictor as \( \eta = X\theta_x + Z\theta_z \), and an additional agreement penalty term \( \rho \|(X\theta_x - Z\theta_z)\|^2 \) with the following objective to be minimized:

\[
J(\theta_x, \theta_z) = \ell(X\theta_x + Z\theta_z, y) + \frac{\rho}{2}\|(X\theta_x - Z\theta_z)\|^2 + \lambda_x\|\theta_x\|_1 + \lambda_z\|\theta_z\|_1,
\]

(23)

where \( \ell \) is the negative log likelihood of the data. For Cox proportional hazards models, \( \ell \) becomes the negative log partial likelihood of the data.

We make the usual quadratic approximation to (23), reducing the minimization problem to a weighted least squares (WLS) problem, which yields

\[
\min \frac{1}{2} \left\| W(z - X\theta_x - Z\theta_z) \right\|^2 + \rho\|(X\theta_x - Z\theta_z)\|^2 + \lambda_x\|\theta_x\|_1 + \lambda_z\|\theta_z\|_1.
\]

(24)

Here \( z \) is the adjusted dependent variable and \( W \) is the diagonal weight matrix, both of which are functions of \( \theta_x \) and \( \theta_z \).

This leads to an iteratively reweighted least squares (IRLS) algorithm:

- Outer loop: Update the quadratic approximation using the current parameter \( \hat{\theta}_x \) and \( \hat{\theta}_z \), i.e. update the working response \( z \) and the weight matrix \( W \).
- Inner loop: Letting

\[
\tilde{X} = \begin{pmatrix} W^{1/2}X & W^{1/2}Z \\ -\sqrt{p}X & \sqrt{p}Z \end{pmatrix}, \tilde{z} = \begin{pmatrix} W^{1/2}z \\ 0 \end{pmatrix}, \tilde{\beta} = \begin{pmatrix} \theta_x \\ \theta_z \end{pmatrix},
\]

(25)

solve the following problem

\[
J(\theta_x, \theta_z) = \frac{1}{2}\left\| \tilde{z} - \tilde{X}\tilde{\beta} \right\|^2 + \lambda_x\|\theta_x\|_1 + \lambda_z\|\theta_z\|_1,
\]

(26)

which is equivalent to (24).

16
7 Some extensions

7.1 Paired features from different views

One can extend cooperative learning to the setting where a feature in one view is naturally paired with a feature in another view. For example, if the \( j \)th column \( X_j \) of \( X \) is the gene expression for gene \( j \), while \( Z_k \) is the expression of the protein \( k \) for which gene \( j \) codes. In that setup, we would like to encourage agreement between \( X_j \theta_{xj} \) and \( Z_k \theta_{zk} \). This pairing need not exist for all features, but can occur for a subset of features.

Looking back at our objective function (4) for two views in the linear case, we add to this objective a pairwise agreement penalty of the form

\[
\rho_2 \sum_{j,k \in P} (X_j \theta_{xj} - Z_k \theta_{zk})^2
\]

(27)

where \( P \) is the set of indices of the paired features.

This additional penalty can be handled easily in the optimization framework of Section 2. For the direct algorithm (Algorithm 1), we simply add a new row to \( \tilde{X} \) and \( \tilde{y} \) for each pairwise constraint, while the one-at-a-time algorithm (Algorithm 2) can be similarly modified. We plan to explore this extension in future work.

7.2 Modeling interactions between views

In our general algorithm (1), we can capture interactions between features in the same view, by using random forests or boosting for the learners \( f_X \) and \( f_Z \). However, this framework does not allow for interactions between features in different views. Here is an objective function to facilitate such interactions:

\[
\min E \left[ \frac{1}{2} (y - f_X(X) - f_Z(Z) - f_{XZ}(X,Z))^2 + \frac{\rho}{2} (f_X(X) - f_Z(Z))^2 + \frac{\rho}{2(1 - \rho)} f_{XZ}^2(X,Z) \right],
\]

(28)

where \( f_{XZ}(X,Z) \) is a joint function of \( X \) and \( Z \), including for example, interactions between the features in each view.

The solution to (28) has fixed points:

\[
\begin{align*}
    f_X(X) &= E \left[ \frac{y}{1 + \rho} - \frac{(1 - \rho) f_Z(Z)}{1 + \rho} - \frac{f_{XZ}(X,Z)}{1 + \rho} \right] | X \\
    f_Z(Z) &= E \left[ \frac{y}{1 + \rho} - \frac{(1 - \rho) f_X(X)}{1 + \rho} - \frac{f_{XZ}(X,Z)}{1 + \rho} \right] | Z \\
    f_{XZ}(X,Z) &= E \left[ (1 - \rho)(y - f_X(X) - f_Z(Z)) | X, Z \right].
\end{align*}
\]

(29)

When \( \rho = 0 \), from (28) the solution reduces to the additive model \( f_X(X) + f_Z(Z) + f_{XZ}(X,Z) \). As \( \rho \to 1 \), the joint term \( f_{XY} \to 0 \) and we again get the late fusion estimate as the average of the marginal predictions \( f_X(X) \) and \( f_Z(Z) \). To implement this in practice, we simply insert random forest or boosting learners for \( f_X, f_Z \) and \( f_{XZ} \). The first two use only features from \( X \) and \( Z \), while the last uses features from both.
8 Discussion

In this paper, we have introduced a new method called cooperative learning for supervised learning with multiple set of features, or “data views”. The method encourages the predictions from different views to align through an agreement penalty. By varying the weight of the agreement penalty in the objective, we obtain a spectrum of solutions that include the commonly-used early and late fusion methods. Cooperative learning can choose the degree of agreement (or fusion) in an adaptive manner. One version of our fitting procedure is modular, where one can use different fitting mechanisms suitable for different data views. We studied the effectiveness of cooperative learning through several simulations and real data examples. The method can be easily extended to binary, count and survival data, and we leave exploration of those applications for future work.

A public-domain R language for cooperative learning is in development and will be made available.

Acknowledgments. We would like to thank Olivier Gevaert, Trevor Hastie and Ryan Tibshirani for helpful discussions. R.T. was supported by the National Institutes of Health (5R01 EB001988-16) and the National Science Foundation (19 DMS1208164).

References

Augimeri, G., Giordano, C., Gelsomino, L., Plastina, P., Barone, I., Catalano, S., Andò, S. & Bonofiglio, D. (2020), ‘The role of pparγ ligands in breast cancer: from basic research to clinical studies’, *Cancers* **12**(9), 2623.

Becker, S. & Hinton, G. E. (1992), ‘Self-organizing neural network that discovers surfaces in random-dot stereograms’, *Nature* **355**(6356), 161–163.

Chabon, J. J., Hamilton, E. G., Kurtz, D. M., Esfahani, M. S., Moding, E. J., Stehr, H., Schroers-Martin, J., Nabet, B. Y., Chen, B., Chaudhuri, A. A. et al. (2020), ‘Integrating genomic features for non-invasive early lung cancer detection’, *Nature* **580**(7802), 245–251.

Chaudhary, K., Poirion, O. B., Lu, L. & Garmire, L. X. (2018), ‘Deep learning–based multi-omics integration robustly predicts survival in liver cancer’, *Clinical Cancer Research* **24**(6), 1248–1259.

Cheerla, A. & Gevaert, O. (2019), ‘Deep learning with multimodal representation for pancancer prognosis prediction’, *Bioinformatics* **35**(14), 1446–1454.

Chen, R. J., Lu, M. Y., Wang, J., Williamson, D. F., Rodig, S. J., Lindeman, N. I. & Mahmood, F. (2020), ‘Pathomic fusion: an integrated framework for fusing histopathology and genomic features for cancer diagnosis and prognosis’, *IEEE Transactions on Medical Imaging*.

Chen, T., Kornblith, S., Norouzi, M. & Hinton, G. (2020), A simple framework for contrastive learning of visual representations, *in* ‘International Conference on Machine Learning’, PMLR, pp. 1597–1607.

Chen, X. & He, K. (2021), Exploring simple siamese representation learning, *in* ‘Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition’, pp. 15750–15758.

Cox, D. R. (1972), ‘Regression models and life-tables’, *Journal of the Royal Statistical Society: Series B (Methodological)* **34**(2), 187–202.

Friedman, J., Hastie, T. & Tibshirani, R. (2010), ‘Regularization paths for generalized linear models via coordinate descent’, *Journal of Statistical Software* **33**, 1–22.
Fu, Q., Chang, Y., An, H., Fu, H., Zhu, Y., Xu, L., Zhang, W. & Xu, J. (2015), ‘Prognostic value of interleukin-6 and interleukin-6 receptor in organ-confined clear-cell renal cell carcinoma: a 5-year conditional cancer-specific survival analysis’, *British journal of cancer* 113(11), 1581–1589.

García-Ceja, E., Galván-Tejada, C. E. & Brena, R. (2018), ‘Multi-view stacking for activity recognition with sound and accelerometer data’, *Information Fusion* 40, 45–56.

Gentles, A. J., Bratman, S. V., Lee, L. J., Harris, J. P., Feng, W., Nair, R. V., Shultz, D. B., Nair, V. S., Hoang, C. D., West, R. B. et al. (2015), ‘Integrating tumor and stromal gene expression signatures with clinical indices for survival stratification of early-stage non–small cell lung cancer’, *JNCI: Journal of the National Cancer Institute* 107(10).

Giuli, M., Giuliani, E., Screpanti, I., Bellavia, D. & Checuolo, S. (2019), ‘Notch signaling activation as a hallmark for triple-negative breast cancer subtype’, *Journal of oncology* 2019.

Gligorijević, V., Malod-Dognin, N. & Pržulj, N. (2016), ‘Integrative methods for analyzing big data in precision medicine’, *Proteomics* 16(5), 741–758.

Gross, S. M. & Tibshirani, R. (2015), ‘Collaborative regression’, *Biostatistics* 16(2), 326–338.

Gudbrandsdottir, G., Aarstad, H. H., Bostad, L., Hjelle, K. M., Aarstad, H. J., Bruserud, Ø., Tvedt, T. H. A. & Beisland, C. (2021), ‘Serum levels of the il-6 family of cytokines predict prognosis in renal cell carcinoma (rcc)’, *Cancer Immunology, Immunotherapy* 70(1), 19–30.

Hastie, T. J. & Tibshirani, R. J. (1990), *Generalized additive models*, CRC Press.

Jaiswal, A., Babu, A. R., Zadeh, M. Z., Banerjee, D. & Makedon, F. (2021), ‘A survey on contrastive self-supervised learning’, *Technologies* 9(1), 2.

Kalscheuer, S., Khanna, V., Kim, H., Li, S., Sachdev, D., DeCarlo, A., Yang, D. & Panyam, J. (2019), ‘Discovery of hspg2 (perlecan) as a therapeutic target in triple negative breast cancer’, *Scientific reports* 9(1), 1–11.

Kamińska, K., Czarnecka, A. M., Escudier, B., Lian, F. & Szczylik, C. (2015), Interleukin-6 as an emerging regulator of renal cell cancer, in ‘Urologic oncology: seminars and original investigations’, Vol. 33, Elsevier, pp. 476–485.

Karczewski, K. J. & Snyder, M. P. (2018), ‘Integrative omics for health and disease’, *Nature Reviews Genetics* 19(5), 299.

Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C. & Krishnan, D. (2020), Supervised contrastive learning, in ‘Proceedings of the 34th Conference on Neural Information Processing Systems’.

Kristensen, V. N., Lingjærde, O. C., Russnes, H. G., Vollan, H. K. M., Frigessi, A. & Børresen-Dale, A.-L. (2014), ‘Principles and methods of integrative genomic analyses in cancer’, *Nature Reviews Cancer* 14(5), 299–313.

Liu, F.-M., Yost, S. E., Wen, W., Frankel, P. H., Schmolze, D., Chu, P.-G., Yuan, Y.-C., Liu, Z., Yim, J., Chen, Z. et al. (2019), ‘Differential gene expression and akt targeting in triple negative breast cancer’, *Oncotarget* 10(43), 4356.

Liu, G., Dong, C. & Liu, L. (2016), ‘Integrated multiple “-omics” data reveal subtypes of hepatocellular carcinoma’, *PLoS One* 11(11), e0165457.

Ma, A., McDermaid, A., Xu, J., Chang, Y. & Ma, Q. (2020), ‘Integrative methods and practical challenges for single-cell multi-omics’, *Trends in Biotechnology* .

Martorana, F., Motta, G., Pavone, G., Motta, L., Stella, S., Vitale, S. R., Manzella, L. & Vigneri, P. (2021), ‘Akt inhibitors: New weapons in the fight against breast cancer?’, *Frontiers in Pharmacology* 12, 546.
Miao, K., Lei, J. H., Valecha, M. V., Zhang, A., Xu, J., Wang, L., Lyu, X., Chen, S., Miao, Z., Zhang, X. et al. (2020), ‘Notch1 activation compensates brca1 deficiency and promotes triple-negative breast cancer formation’, *Nature communications* **11**(1), 1–15.

Nelder, J. A. & Wedderburn, R. W. (1972), ‘Generalized linear models’, *Journal of the Royal Statistical Society: Series A (General)* **135**(3), 370–384.

Perkins, B. A., Caskey, C. T., Brar, P., Dec, E., Karow, D. S., Kahn, A. M., Hou, Y.-C. C., Shah, N., Boeddt, D., Coughlin, E. et al. (2018), ‘Precision medicine screening using whole-genome sequencing and advanced imaging to identify disease risk in adults’, *Proceedings of the National Academy of Sciences* **115**(14), 3686–3691.

Ritchie, M. D., Holzinger, E. R., Li, R., Pendergrass, S. A. & Kim, D. (2015), ‘Methods of integrating data to uncover genotype-phenotype interactions’, *Nature Reviews Genetics* **16**(2), 85–97.

Safo, S. E., Min, E. J. & Haine, L. (2021), ‘Sparse linear discriminant analysis for multiview structured data’, *Biometrics* .

Simon, N. & Tibshirani, R. (2012), ‘Standardization and the group lasso penalty’, *Statistica Sinica* **22**(3), 983.

Sindhwani, V., Niyogi, P. & Belkin, M. (2005). A co-regularization approach to semi-supervised learning with multiple views, in *Proceedings of ICML workshop on learning with multiple views*, Vol. 2005, Citeseer, pp. 74–79.

Telli, M. L., Jensen, K. C., Vinayak, S., Kurian, A. W., Lipson, J. A., Flaherty, P. J., Timms, K., Abkevich, V., Schackmann, E. A., Wapnr, I. L. et al. (2015), ‘Phase 2 study of gemcitabine, carboplatin, and iniparib as neoadjuvant therapy for triple-negative and brca1/2 mutation–associated breast cancer with assessment of a tumor-based measure of genomic instability: Precog 0105’, *Journal of Clinical Oncology* **33**(17), 1895.

The Cancer Genome Atlas Research Network (2013), ‘Comprehensive molecular characterization of clear cell renal cell carcinoma’, *Nature* **499**(7456), 43.

Tibshirani, R. J. (2017), ‘Dykstra’s algorithm, admm, and coordinate descent: Connections, insights, and extensions’, *arXiv preprint arXiv:1705.04768* .

Tibshirani, R. & Taylor, J. (2011), ‘On the degrees of freedom of the lasso’, *Annals of Statistics* (40), 1198–1232.

Tomczak, K., Czerwiska, P. & Wiznerowicz, M. (2015), ‘The cancer genome atlas (tcga): an immeasurable source of knowledge’, *Contemporary Oncology* **19**(1A), A68.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.-A. & Bottou, L. (2010), ‘Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion.’, *Journal of Machine Learning Research* **11**(12).

Wold, S., Esbensen, K. & Geladi, P. (1987), ‘Principal component analysis’, *Chemometrics and Intelligent Laboratory Systems* **2**(1–3), 37–52.

Wu, L., Yang, Y., Guo, X., Shu, X.-O., Cai, Q., Shu, X., Li, B., Tao, R., Wu, C., Nikas, J. B. et al. (2020), ‘An integrative multi-omics analysis to identify candidate dna methylation biomarkers related to prostate cancer risk’, *Nature communications* **11**(1), 1–11.

Yang, P., Hwa Yang, Y., B Zhou, B. & Y Zomaya, A. (2010), ‘A review of ensemble methods in bioinformatics’, *Current Bioinformatics* **5**(4), 296–308.

Yuan, M. & Lin, Y. (2006), ‘Model selection and estimation in regression with grouped variables’, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **68**(1), 49–67.
Yuan, X., Zhang, M., Wu, H., Xu, H., Han, N., Chu, Q., Yu, S., Chen, Y. & Wu, K. (2015), ‘Expression of notch1 correlates with breast cancer progression and prognosis’, *PloS one* **10**(6), e0131689.

Yuan, Y., Van Allen, E. M., Omberg, L., Wagle, N., Amin-Mansour, A., Sokolov, A., Byers, L. A., Xu, Y., Hess, K. R., Diao, L. et al. (2014), ‘Assessing the clinical utility of cancer genomic and proteomic data across tumor types’, *Nature Biotechnology* **32**(7), 644–652.

Zhao, J., Feng, Q., Wu, P., Lupu, R. A., Wilke, R. A., Wells, Q. S., Denny, J. C. & Wei, W.-Q. (2019), ‘Learning from longitudinal data in electronic health record and genetic data to improve cardiovascular event prediction’, *Scientific Reports* **9**(1), 1–10.
9 Appendix

9.1 Adaptive cooperative learning

In this section, we outline an adaptive strategy for optimizing over $\lambda_x$ and $\lambda_z$ for different data views. We call this adaptive cooperative learning. The method incorporates the values of $\lambda_x$ and $\lambda_z$ that have been adaptively optimized by the one-at-a-time algorithm (Algorithm 3) as a penalty factor in the direct algorithm (Algorithm 4). In the two-dimensional grid of $\lambda_x$ and $\lambda_z$, our proposed strategy works by iteratively searching along one axis of $\lambda$ while fixing the other constant.

Algorithm 3 One-at-a-time algorithm for adaptive cooperative learning (regression), which we denote as adap-coop-one-at-a-time($X$, $Z$).

**Input:** $X \in \mathbb{R}^{n \times p_x}$ and $Z \in \mathbb{R}^{n \times p_z}$, the response $y \in \mathbb{R}^n$, and a fixed hyperparameter $\rho \in \mathbb{R}$.

1. Initialize $\theta_x^{(0)} \in \mathbb{R}^{p_x}$ and $\theta_z^{(0)} \in \mathbb{R}^{p_z}$.

2. For $k = 0, 1, 2, \ldots$ until convergence:

   (a) Set $y_x^* = \frac{y}{1+\rho} - \frac{(1-\rho)Z\theta_z}{(1+\rho)}$. Solve Lasso($X$, $y_x^*$, $\lambda$) over a decreasing grid of $\lambda$ values. Update $\theta_x^{(k+1)}$ to be the solution and record the hyperparameter $\lambda_x^*$ that minimizes the CV error.

   (b) Set $y_z^* = \frac{y}{1+\rho} - \frac{(1-\rho)X\theta_x}{(1+\rho)}$. Solve Lasso($Z$, $y_z^*$, $\lambda$) over a decreasing grid of $\lambda$ values. Update $\theta_z^{(k+1)}$ to be the solution and record the hyperparameter $\lambda_z^*$ that minimizes the CV error.

**Output:** $\hat{\theta}_x$ and $\hat{\theta}_z$ from the last iteration, along with the hyperparameters $\lambda_x^*$ and $\lambda_z^*$ and the corresponding CV errors.

Algorithm 4 Direct algorithm for adaptive cooperative learning (regression).

**Input:** $X \in \mathbb{R}^{n \times p_x}$ and $Z \in \mathbb{R}^{n \times p_z}$, the response $y \in \mathbb{R}^n$, and a grid of hyperparameter values $(\rho_{\min}, \ldots, \rho_{\max})$.

for $\rho \leftarrow \rho_{\min}, \ldots, \rho_{\max}$ do

1. Run adap-coop-one-at-a-time($X$, $Z$) and adap-coop-one-at-a-time($Z$, $X$) with the same folds for CV. Select the one with the lower sum of the two CV errors. Get the corresponding $\lambda_x^*$ and $\lambda_z^*$.

2. Set

   $$\tilde{X} = \begin{pmatrix} X & Z \\ -\sqrt{\rho}X & \sqrt{\rho}Z \end{pmatrix}, \tilde{y} = \begin{pmatrix} y \\ 0 \end{pmatrix}.$$

   Solve Lasso($\tilde{X}$, $\tilde{y}$, $\lambda$) over a decreasing grid of $\lambda$ values, with a penalty factor of $(1, \ldots, 1, \frac{\lambda_x^*}{\rho_x}, \ldots, \frac{\lambda_z^*}{\rho_z})$.
   Note that we form folds from the rows of $X$ and $Z$ and then construct the corresponding $\tilde{X}$.

end

Select the optimal value of $\rho$ based on the CV error and get the corresponding final fit.

22
9.2 Procedure for generating the imaging and “omics” data

Here we outline the detailed procedure for generating the imaging and “omics” data for the simulation study in Section 4.2. The “omics” data ($X$), imaging data ($Z$), and the response $y$ are generated such that there are correlations between $X$, $Z$, and $y$.

Algorithm 5 Simulation procedure for generating the imaging and “omics” data.

**Input:** Parameters $n, p_x, p_u, s_u, t, \sigma, \beta_u, I_{\text{max}}, \text{ndim}, \text{threshold}.$

**Output:** $X \in \mathbb{R}^{n \times p_x}$ (omics), $Z \in \mathbb{R}^{n \times \text{ndim} \times \text{ndim} \times 1}$ (images assuming one color channel), $y \in \mathbb{R}^n$.

1. $x_j \in \mathbb{R}^n$ distributed i.i.d. $\text{MVN}(0, I_n)$ for $j = 1, 2, \ldots, p_x$

2. For $i = 1, 2, \ldots, p_u$ ($p_u < p_x$, where $p_u$ corresponds to the number of factors):
   (a) $u_i \in \mathbb{R}^n$ distributed i.i.d. $\text{MVN}(0, s_u^2 I_n)$
   (b) $x_i = x_1 + t * u_i$  

3. $U = [u_1, u_2, \ldots, u_{p_u}], X = [x_1, x_2, \ldots, x_{p_x}]$

4. $y_u = U\beta_u + \epsilon$ where $\epsilon \in \mathbb{R}^n$ distributed i.i.d. $\text{MVN}(0, \sigma^2 I_n)$

5. For $i = 1, 2, \ldots, n$:
   (a) $P_i = \frac{1}{1 + \exp(y_{ui})}, y_i \sim \text{Bernoulli}(P_i)$
   (b) Generate a 2D pixel matrix of image $Z_i \in \mathbb{R}^{\text{ndim} \times \text{ndim} \times 1}$
   (c) Generate a polygon $\text{PG}_i$ inside $Z_i$, defined by 4 vertices $[v_1, v_2, v_3, v_4]$ on the axes, i.e. $v_1 = [0, a_1], v_2 = [0, a_2], v_3 = [a_3, 0], v_4 = [a_4, 0]$, where $a_1 \sim \text{Unif}(\text{ndim}/2, \text{ndim}), a_2 \sim \text{Unif}(-\text{ndim} - \text{ndim}/2, \text{ndim}), a_3 \sim \text{Unif}(\text{ndim}/2, \text{ndim}), a_4 \sim \text{Unif}(-\text{ndim} - \text{ndim}/2, \text{ndim})$
   (d) Randomly sample points from $Z_i$: if the point $[x', y']$ falls inside the polygon $\text{PG}_i$, i.e. $[x', y'] \in \text{PG}_i$, then $Z_i[x', y'] \sim \text{Unif}(0, 1)$
   (e) If $y_i = 1$, $I_{\text{disease}} = I_{\text{max}} \times y_{ui}$, where $I_{\text{max}}$ is the maximum intensity of pixel values for images,
      - For $x' = 1, 2, \ldots, \text{ndim}$:
        - For $y' = 1, 2, \ldots, \text{ndim}$:
          * $P(x', y') \sim \text{Unif}(0, 1)$
          * If $[x', y'] \in \text{PG}_i$ and $P(x', y') < \text{threshold}$, $Z_i[x', y'] = I_{\text{disease}}$
9.3 More comprehensive simulation studies on cooperative regularized linear regression

9.3.1 More simulation results of the high-dimensional settings \((p = 1000, n = 200)\)

**Figure 6**: Simulation results when \(X\) and \(Z\) have a medium level of correlation \((t = 2, s_u = 1)\); both \(X\) and \(Z\) contain signal \((b_x = b_z = 2)\), \(n = 200, p = 1000, SNR = 3.5\). The setup is the same as in Figure 5.

**Figure 7**: Simulation results when \(X\) and \(Z\) have no correlation \((t = 0, s_u = 1)\); both \(X\) and \(Z\) contain signal \((b_x = b_z = 2)\), \(n = 200, p = 1000, SNR = 1.0\). The setup is the same as in Figure 5.

**Figure 8**: Simulation results when \(X\) and \(Z\) have a high level of correlation \((t = 6, s_u = 1)\); both \(X\) and \(Z\) contain signal \((b_x = b_z = 2)\), \(n = 200, p = 1000, SNR = 0.6\). The setup is the same as in Figure 5.
Figure 9: Simulation results when $X$ and $Z$ have a medium level of correlation ($t = 3, s_u = 1$); $Z$ contains more signal than $X$ ($b_x = 1.5, b_z = 3$), $n = 200, p = 1000, SNR = 1.2$. The setup is the same as in Figure 3.

Figure 10: Simulation results when $X$ and $Z$ have no correlation ($t = 0, s_u = 1$); $X$ contains more signal than $Z$ ($b_x = 3, b_z = 1$), $n = 200, p = 1000, SNR = 1.1$. The setup is the same as in Figure 3.

Figure 11: Simulation results when $X$ and $Z$ have a high level of correlation ($t = 6, s_u = 1$); $Z$ contains more signal than $X$ ($b_x = 1.5, b_z = 3$), $n = 200, p = 1000, SNR = 5.0$. The setup is the same as in Figure 3.
Figure 12: Simulation results when $X$ and $Z$ have a high level of correlation ($t = 6, s_u = 1$); $X$ contains more signal than $Z$ ($b_x = 3, b_z = 1.5$), $n = 200, p = 1000$, $SNR = 0.9$. The setup is the same as in Figure 3.

Figure 13: Simulation results when $X$ and $Z$ have a medium level of correlation ($t = 3, s_u = 1$); only $X$ contains signal ($b_x = 2, b_z = 0$), $n = 200, p = 1000$, $SNR = 0.8$. The setup is the same as in Figure 3.

Figure 14: Simulation results when $X$ and $Z$ have a high level of correlation ($t = 6, s_u = 1$); only $X$ contains signal ($b_x = 2, b_z = 0$), $n = 200, p = 1000$, $SNR = 1.2$. The setup is the same as in Figure 3.
9.3.2 Simulation results of the lower-dimensional settings \((p = 200, n = 500)\)

Figure 15: Simulation results when \(X\) and \(Z\) have a medium level of correlation \((t = 2, s_u = 1)\); both \(X\) and \(Z\) contain signal \((b_x = b_z = 2)\), \(n = 500, p = 200, SNR = 1.8\). The setup is the same as in Figure 3.

Figure 16: Simulation results when \(X\) and \(Z\) have a medium level of correlation \((t = 2, s_u = 1)\); both \(X\) and \(Z\) contain signal \((b_x = b_z = 2)\), \(n = 500, p = 200, SNR = 0.6\). The setup is the same as in Figure 3.

Figure 17: Simulation results when \(X\) and \(Z\) have a medium level of correlation \((t = 2, s_u = 1)\); both \(X\) and \(Z\) contain signal \((b_x = b_z = 2)\), \(n = 500, p = 200, SNR = 3.5\). The setup is the same as in Figure 3.
Figure 18: Simulation results when $X$ and $Z$ have no correlation ($t = 0, s_u = 1$); only $X$ contains signal ($b_x = 2, b_z = 0$), $n = 500, p = 200$, SNR = 0.3. The setup is the same as in Figure 3.

Figure 19: Simulation results when $X$ and $Z$ have a high level of correlation ($t = 6, s_u = 1$); only $X$ contains signal ($b_x = 2, b_z = 0$), $n = 500, p = 200$, SNR = 1.2. The setup is the same as in Figure 3.

Figure 20: Simulation results when $X$ and $Z$ have a high level of correlation ($t = 6, s_u = 1$); both $X$ and $Z$ contain signal ($b_x = b_z = 2$), $n = 500, p = 200$, SNR = 1.0. The setup is the same as in Figure 3.
Figure 21: Simulation results when $X$ and $Z$ have no correlation ($t = 0, s_u = 1$); both $X$ and $Z$ contain signal ($b_x = b_z = 2$), $n = 500, p = 200$, SNR = 0.3. The setup is the same as in Figure 3.

Figure 22: Simulation results when $X$ and $Z$ have a medium level of correlation ($t = 2, s_u = 1$); $Z$ contains more signal than $X$ ($b_x = 1.5, b_z = 3$), $n = 500, p = 200$, SNR = 1.1. The setup is the same as in Figure 3.