Journal of Machine Learning Research 1 (2000) 1-48 Submitted 4/00; Published 10/00

Stability-Adjusted Cross-Validation for Sparse Linear Regression

Ryan Cory-Wright R.CORY-WRIGHT@IMPERIAL.AC.UK
Department of Analytics, Marketing, and Operations
Imperial College Business School
London, SW7 2AZ, UK

Andrés Gómez GOMEZAND@USC.EDU
Department of Industrial and Systems Engineering
Viterbi School of Engineering, University of Southern California, CA

Editor: TBD

Abstract

Given a high-dimensional covariate matrix and a response vector, ridge-regularized sparse linear regression selects a subset of features that explains the relationship between covariates and the response in an interpretable manner. To select the sparsity and robustness of linear regressors, techniques like k-fold cross-validation are commonly used for hyperparameter tuning. However, cross-validation substantially increases the computational cost of sparse regression as it requires solving many mixed-integer optimization problems (MIOs) for each hyperparameter combination. Additionally, validation metrics often serve as noisy estimators of test set errors, with different hyperparameter combinations leading to models with different noise levels. Therefore, optimizing over these metrics is vulnerable to out-of-sample disappointment, especially in underdetermined settings. To improve upon this state of affairs, we make two key contributions. First, motivated by the generalization theory literature, we propose selecting hyperparameters that minimize a weighted sum of a cross-validation metric and a model's output stability, thus reducing the risk of poor out-of-sample performance. Second, we leverage ideas from the mixed-integer optimization literature to obtain computationally tractable relaxations of k-fold cross-validation metrics and the output stability of regressors, facilitating hyperparameter selection after solving fewer MIOs. These relaxations result in an efficient cyclic coordinate descent scheme, achieving lower validation errors than via traditional methods such as grid search. On synthetic datasets, our confidence adjustment procedure improves out-of-sample performance by 2%–5% compared to minimizing the k-fold error alone. On 13 real-world datasets, our procedure reduces test set error by 2%, on average.

Keywords: Cross-validation; stability; perspective formulation; sparse regression

1 Introduction

Over the past fifteen years, Moore's law has spurred an explosion of high-dimensional datasets for scientific discovery across multiple fields, fueling a revolution in big data and AI (McAfee et al., 2012; Groves et al., 2016; McAfee et al., 2023). These datasets often consist of a design matrix $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ of explanatory variables and an output vector $\boldsymbol{y} \in \mathbb{R}^n$ of response variables. Accordingly, practitioners often aim to explain the response variables

linearly via the equation $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$ for a vector of regression coefficients $\boldsymbol{\beta} \in \mathbb{R}^p$, which is to be inferred, and a vector of error $\boldsymbol{\epsilon}$, typically kept small by minimizing the least squares (LS) error of the regression.

Despite its computational efficiency, LS regression exhibits two practical limitations. First, when $p \gg n$, there is not enough data to accurately infer $\boldsymbol{\beta}$ via LS, and LS regression generates estimators which perform poorly out-of-sample due to a data curse of dimensionality (cf. Bühlmann and Van De Geer, 2011; Gamarnik and Zadik, 2022). Second, LS regression generically selects every feature, including irrelevant ones. This is a significant challenge when the regression coefficients are used for high-stakes clinical decision-making tasks. Indeed, irrelevant features could lead to suboptimal patient outcomes due to the lack of interpretability (Doshi-Velez and Kim, 2017).

To tackle the twin curses of dimensionality and false discovery, sparse learning has emerged as a popular methodology for explaining the relationship between inputs $\boldsymbol{X}$ and outputs $\boldsymbol{y}$. A popular model in this paradigm is ridge-regularized sparse linear regression, which admits the formulation (Bertsimas and Van Parys, 2020; Atamtürk and Gómez, 2020):

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^p} \quad \frac{\gamma}{2}\|\boldsymbol{\beta}\|_2^2 + \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 \quad \text{s.t.} \quad \|\boldsymbol{\beta}\|_0 \leq \tau, \tag{1}$$

where $\tau \in [p] := \{1, \ldots, p\}$ and $\gamma > 0$ are hyperparameters that respectively model the sparsity and robustness of the linear model $\boldsymbol{\beta}$ (cf. Xu et al., 2008; Bertsimas and Copenhaver, 2018), and $\boldsymbol{X}, \boldsymbol{y}$ have undergone standard preprocessing so that $\boldsymbol{y}$ is a zero-mean vector and $\boldsymbol{X}$ has zero-mean unit-variance columns, meaning γ penalizes each feature equally.

Problem (1) is computationally challenging (indeed, NP-hard Natarajan (1995)) and initial formulations could not scale to problems with thousands of features (Hastie et al., 2020). In a more positive direction, by developing and exploiting tight conic relaxations of appropriate substructures of (1), e.g., the perspective relaxation (Ceria and Soares, 1999), more recent mixed-integer optimization techniques such as branch-and-bound (Hazimeh et al., 2022) scale to larger instances with thousands of features. For brevity, we refer to Bertsimas et al. (2021); Atamtürk and Gómez (2025) for reviews of convex relaxations.

While these works solve (1) rapidly, they do not address one of the most significant difficulties in performing sparse regression. The hyperparameters (τ, γ) are not known to the decision-maker ahead of time, as is often assumed in the literature for convenience. Rather, they must be selected by the decision-maker, which is potentially much more challenging than solving (1) for a single value of (τ, γ) (Hansen et al., 1992). Indeed, selecting (τ, γ) typically involves minimizing a validation metric over a grid of values, which is computationally expensive (Larochelle et al., 2007).

Perhaps the most popular validation metric is hold-out (Hastie et al., 2009), where one omits a portion of the data when training the model and then evaluates performance on this hold-out set as a proxy for the model's test set performance. However, hold-out validation is a high-variance approach (Hastie et al., 2009), because the validation score can vary significantly depending on the hold-out set selected. To reduce the variance in this procedure, a number of authors have proposed:

The Cross-Validation Paradigm: To obtain accurate models that generalize well to unseen data, cross-validation has emerged as a popular model selection paradigm. Early iterations of this paradigm, as reviewed by Stone (1978), suggest solving (1) with the ith

data point removed for each $i \in [n]$, and estimating the out-of-sample performance of a solution to Problem (1) via the average performance of the n estimators with the ith training data point removed, on the ith data point. This approach is known as leave-one-out cross-validation (LOOCV).

A popular variant of LOOCV, known as k-fold cross-validation, comprises removing subsets of n/k data points at a time, which significantly reduces the computational burden of cross-validation (Burman, 1989; Arlot and Celisse, 2010). However, even k-fold cross-validation may be prohibitive in the case of MIOs such as (1). Indeed, as identified by Hastie et al. (2020), with a time limit of 3 minutes per MIO, using 10-fold cross-validation to choose between subset sizes $\tau = 0, \ldots, 50$ in an instance of Problem (1) with $p = 100$ and $n = 500$ requires 25 hours of computational time.

For sparse regression, given a partition $\mathcal{N}_1, \ldots, \mathcal{N}_k$ of $[n]$, performing k-fold cross-validation corresponds to selecting hyperparameters γ, τ which minimize the function:

$$h(\gamma, \tau) = \frac{1}{k} \sum_{j=1}^{k} \sum_{i \in \mathcal{N}_j} (y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta}^{(\mathcal{N}_j)}(\gamma, \tau))^2 \tag{2}$$

where $\boldsymbol{\beta}^{(\mathcal{N}_j)}(\gamma, \tau)$ denotes an optimal solution to the following lower-level problem for $\mathcal{N}_j$:

$$\boldsymbol{\beta}^{(\mathcal{N}_j)}(\gamma, \tau) \in \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\arg\min} \ \frac{\gamma}{2} \|\boldsymbol{\beta}\|_2^2 + \|\boldsymbol{y}^{(\mathcal{N}_j)} - \boldsymbol{X}^{(\mathcal{N}_j)} \boldsymbol{\beta}\|_2^2 \quad \text{s.t.} \quad \|\boldsymbol{\beta}\|_0 \leq \tau, \tag{3}$$

$\gamma > 0$ is a hyperparameter, τ is a sparsity budget, $\boldsymbol{X}^{(\mathcal{N}_j)}, \boldsymbol{y}^{(\mathcal{N}_j)}$ denotes the dataset with the data in $\mathcal{N}_j$ removed, and we take $\boldsymbol{\beta}^{(\mathcal{N}_j)}(\gamma, \tau)$ to be unique for a given τ, γ for convenience[1]. If all sets $\mathcal{N}_j$ are taken to be singletons and $k = n$, minimizing h corresponds to LOOCV. Moreover, if $k = 2$ and the term with $j = 2$ is removed from h, optimizing h reduces to minimizing the hold-out error. After selecting (γ, τ), practitioners usually train a final model on the entire dataset, by solving Problem (1) with the selected hyperparameter combination. To ensure that γ has the same impact in the final model as the cross-validated models, they sometimes first multiply γ by the bias correction term $n/(n - n/k)$ (see Liu and Dobriban, 2020, for a justification)[2].

Hyperparameter selection techniques like k-fold and leave-one-out cross-validation are popular in practice because, for given hyperparameters, the cross-validation error is typically a better estimate of the test set error than other commonly used terms like the training error (Kearns and Ron, 1997; Bousquet and Elisseeff, 2002). However, cross-validation's use is subject to some debate, because its value for a given hyperparameter combination is, in expectation, *pessimistically* biased due to generating estimators $\boldsymbol{\beta}^{(\mathcal{N}_j)}$ trained on less data than the final model $\boldsymbol{\beta}$ (Arlot and Celisse, 2010).

1. This assumption seems plausible, as the training objective is strongly convex for a fixed binary support vector, and therefore for each binary support vector there is indeed a unique solution. One could relax this assumption by defining $h(\gamma, \tau)$ to be the minimum cross-validation error over all training-optimal solutions $\boldsymbol{\beta}^{(i)}$, as is commonly done in the bilevel optimization literature, giving what is called an optimistic formulation of a bilevel problem (see Beck and Schmidt, 2021, for a review). However, this would make the cross-validation error less tractable.
2. We remark that applying this bias correction term is equivalent to normalizing the least squares error $\|\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{Y}\|_2^2$ in the training problem, by dividing this term by the number of data points n (or $n - n/k$).

Further complicating matters, a validation metric's *minimum* value is typically optimistically biased due to the optimizer's curse (Smith and Winkler, 2006; Arlot and Celisse, 2010); see also Rao et al. (2008) for an empirical study of this phenomenon in classification problems. Indeed, as observed by Rao et al. (2008); Gupta et al. (2024) (see also our motivating example in Section 1.1), this issue is particularly pronounced in underdetermined settings, where omitting even a small amount of data can dramatically worsen the performance of a regression model and increase the variance of its validation metrics, leading to an out-of-sample performance decrease of 100% or more in some ordinary least squares regression settings (Gupta et al., 2024, Figure 1).

These observations contribute to a literature that questions the machine learning paradigm of selecting hyperparameters by minimizing a validation metric without explicitly accounting for a model's stability (Breiman, 1996; Ban et al., 2018; Gupta and Rusmevichientong, 2021; Gupta and Kallus, 2022; Gupta et al., 2024) and motivate our approach.

Our Approach: We make two contributions toward hyperparameter selection.

First, motivated by the observation that minimizing the cross-validation error disappoints out-of-sample, potentially significantly in underdetermined settings, we propose a generalization bound on the out-of-sample error. This generalization bound takes the form of the cross-validation error plus a term related to a model's hypothesis stability, as discussed in Section 2. Motivated by this (often conservative) bound, we propose minimizing a weighted sum of a validation metric and the hypothesis stability, rather than the stability alone, to mitigate out-of-sample disappointment without being overly conservative. This approach facilitates cross-validation with a single hyperparameter, the weight in the weighted sum, which can be selected in a manner that satisfies probabilistic guarantees according to our generalization bound (Section 2).

Second, from an optimization perspective, we propose techniques for obtaining strong bounds on validation metrics in polynomial time and leverage these bounds to design algorithms for minimizing the cross-validation error and its confidence-adjusted variants in Sections 3-4. In particular, by performing a perturbation analysis of perspective relaxations of sparse regression problems, we construct convex relaxations of the k-fold cross-validation error, which allows us to minimize it without explicitly solving MIOs at each data fold and for each hyperparameter combination. This results in a branch-and-bound algorithm for hyperparameter selection that is substantially more efficient than state-of-the-art techniques like grid search. As an aside, we remark that as cross-validation is more general than hold-out validation, our convex relaxations apply to the hold-out case.

In numerical experiments (Section 5), we assess the impact of our two contributions numerically, and observe on synthetic and real datasets that our confidence-adjustment procedure improves the out-of-sample performance of sparse regression by 2%–7% compared to cross-validating without confidence adjustment. We also observe on synthetic datasets that confidence adjustment often improves the accuracy of the resulting regressors with respect to identifying the ground truth.

1.1 Motivating Example: Poor Performance of Cross-Validation

Suppose that we wish to recover a sparse regressor in the synthetic setting described in our numerical experiments, where the ground truth is $\tau_{\text{true}} = 5$-sparse, with autocorrelation

$\rho = 0.3$ and signal-to-noise ration $\nu = 1$ (these parameters are formally defined in Section F.1), and we have a test set of $n_{\text{test}} = 10,000$ observations drawn from the same underlying stochastic process to measure test set performance. Following the standard cross-validation paradigm, we evaluate the cross-validation error for each τ and 20 values of γ log-uniformly distributed on $[10^{-3}, 10^3]$, using the Generalized Benders Decomposition scheme developed by Bertsimas and Van Parys (2020) to solve each MIO to optimality, and selecting the hyperparameter combination with the lowest cross-validation error, for both leave-one-out and five-fold cross-validation.

Figure 1 depicts each hyperparameter combination's leave-one-out (left) and test (right) error, in an overdetermined setting where $n = 50, p = 10$ (top) and an underdetermined setting where $n = 10, p = 50$ (bottom); for conciseness, we defer the equivalent plots for five-fold cross-validation for this problem setting to Figure 9 (Appendix A). In the overdetermined setting, cross-validation performs well: a model trained by minimizing the LOO (resp. five-fold) cross-validation error attains a test error within 0.6% (resp. 1.1%) of the (unknowable) test minimum. However, in the underdetermined setting, cross-validation performs poorly: a model trained by minimizing the LOO (resp. five-fold) error attains a test set error 16.4% (resp. 31.7%) larger than the test set minimum and seven orders of magnitude larger (resp. one order of magnitude larger) than its LOOCV estimator. As we argue throughout this work, this occurs because cross-validation may generate noisy and high-variance estimators, particularly in underdetermined settings (cf. Hastie et al., 2009, chap 7.10). Therefore, its minimum may disappoint significantly on a test set.

1.2 Literature Review

Our work falls at the intersection of four areas of the optimization and machine learning literature. First, hyperparameter selection techniques for optimizing the performance of a machine learning model by selecting hyperparameters that perform well on a validation set. Second, bilevel approaches that reformulate and solve hyperparameter selection problems as bilevel problems. Third, distributionally robust optimization approaches that guard against out-of-sample disappointment when making decisions in settings with limited data. Finally, perspective reformulation techniques for mixed-integer problems with logical constraints, as discussed earlier in the introduction. To put our contributions into context, we now review the three remaining areas of the literature.

Hyperparameter Selection Techniques for Machine Learning Problems: A wide variety of hyperparameter selection techniques have been proposed for machine learning problems such as sparse regression, including grid search (Larochelle et al., 2007) as reviewed in Section 1, and random search (cf. Bergstra and Bengio, 2012). In random search, we let $\mathcal{L}$ be a random sample from a space of valid hyperparameters, e.g., a uniform distribution over $[10^{-3}, 10^3] \times [p]$ for sparse regression. Remarkably, in settings with many hyperparameters, random search usually outperforms grid search for a given budget on the number of training problems that can be solved, because validation functions often have a lower effective dimension than the number of hyperparameters present in the model (Bergstra and Bengio, 2012). However, grid search remains competitive for problems with a small number of hyperparameters, such as sparse regression.

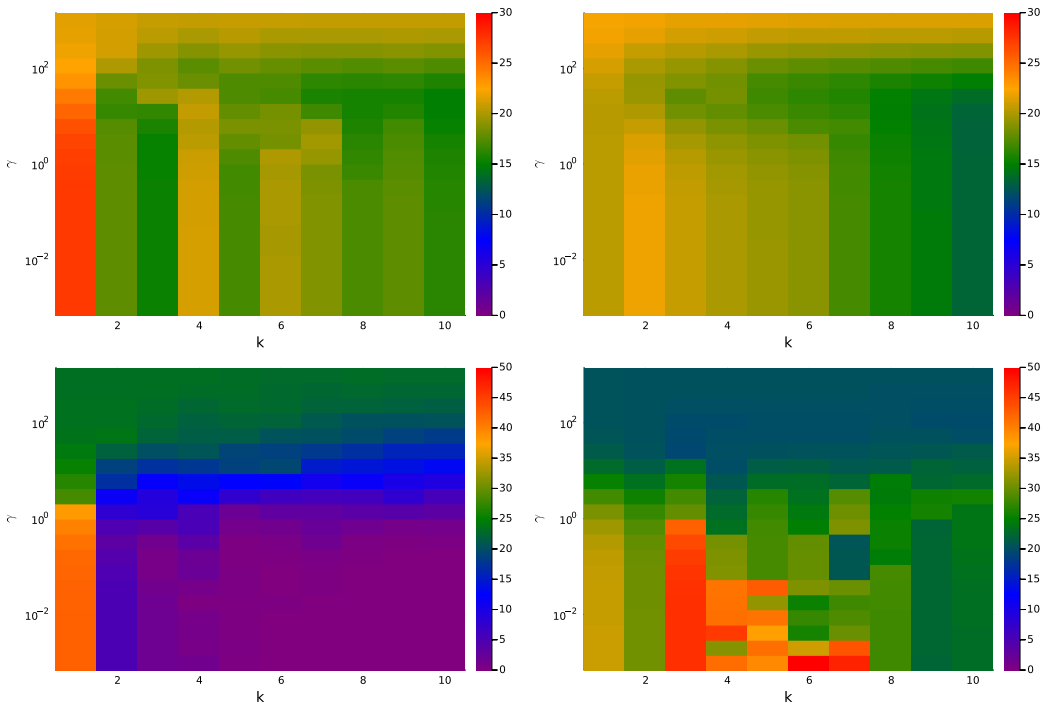

Figure 1: Leave-one-out (left) and test (right) error for varying τ and γ, for an overdetermined setting (top, $n = 50, p = 10$) and an underdetermined setting (bottom, $n = 10, p = 50$). In the overdetermined setting, the leave-one-out error is a good estimate of the test error for most values of parameters (γ, τ). In contrast, in the underdetermined setting, the leave-one-out is a poor approximation of the test error, and estimators that minimize the leave-one-out error ($\gamma \to 0$, $\tau = 10$) significantly disappoint out-of-sample. Our conclusions are identical when using leave-one-out instead of five-fold cross-validation (Appendix A).

Bilevel Optimization for Hyperparameter Selection: In a complementary direction, several authors have proposed selecting hyperparameters via bilevel optimization (see Beck and Schmidt, 2021, for a general theory), since Bennett et al. (2006) recognized that cross-validation is a special case of bilevel optimization. Therefore, in principle, we could minimize the cross-validation error in sparse regression by invoking bilevel techniques. Unfortunately, this approach seems intractable in both theory and practice (Ben-Ayed and Blair, 1990; Hansen et al., 1992). Indeed, standard bilevel approaches such as dualizing the lower-level problem are challenging to apply in our context because our lower-level problems are non-convex and cannot easily be dualized.

Although slow in its original implementations, several authors have proposed making hyperparameter optimization more tractable by combining bilevel optimization with tractable modeling paradigms to obtain locally optimal sets of hyperparameters. Among others, Sinha et al. (2020) recommends taking a gradient-based approximation of the lower-level problem and thereby reducing the bilevel problem to a single-level problem, Okuno et al. (2021) advocates selecting hyperparameters by solving the KKT conditions of a bilevel problem, and

Ye et al. (2022) proposes solving bilevel hyperparameter problems via difference-of-convex methods to obtain a stationary point.

Specializing our review to regression, two works aim to optimize the performance of regression models on a validation metric. First, Takano and Miyashiro (2020) propose optimizing the k-fold validation loss, assuming all folds share the same support. Unfortunately, although their assumption improves their method's tractability, it may lead to subpar statistical performance. Second, Stephenson et al. (2021) proposes minimizing the leave-one-out error in ridge regression problems (without sparsity constraints) and demonstrates that the upper-level objective is quasi-convex in γ. Thus, first-order methods can often optimize the leave-one-out error. Unfortunately, as we observe in our motivating example (Section 1.1), this does not hold in the presence of a sparsity constraint.

Mitigating Out-of-Sample Disappointment: The overarching goal of data-driven decision-making procedures, such as sparse regression, is to use historical data to design models that perform well on unseen data drawn from the same underlying stochastic process (King and Wets, 1991). Indeed, the original justification for selecting hyperparameters by minimizing a validation metric was that validation metrics are conceptually simple and provide more accurate estimates of out-of-sample performance than the training error (Stone, 1974). We now review the literature on cross-validation and related concepts in the context of mitigating out-of-sample disappointment.

From a statistical learning perspective, there is significant literature on quantifying the out-of-sample performance of models with respect to their training and validation error, originating with the seminal works by Vapnik (1999) on VC-dimension and Bousquet and Elisseeff (2002) on algorithmic stability theory. As noted, for instance, by Ban and Rudin (2019), algorithm stability bounds are generally preferable because they are *a posteriori* bounds with tight constants that depend on only the problem data, while VC-dimension bounds are *a priori* bounds that depend on computationally intractable constants like Rademacher averages. Irrespective, the conclusion from both streams of work is that simpler and more stable models tend to disappoint less out-of-sample.

More recently, the statistical learning theory literature has been connected to the distributionally robust optimization literature by Ban et al. (2018); Ban and Rudin (2019); Gupta and Rusmevichientong (2021); Gupta and Kallus (2022); Gupta et al. (2024) among others. Ban and Rudin (2019) propose solving newsvendor problems by designing decision rules that map features to an order quantity and obtain finite-sample guarantees on the out-of-sample cost of newsvendor policies in terms of the in-sample cost. Even closer to our work, Gupta and Rusmevichientong (2021) proposes correcting solutions to high-dimensional problems by invoking Stein's lemma to obtain a Stein's Unbiased Risk Estimator (SURE) approximation of the out-of-sample disappointment and demonstrates that minimizing their bias-corrected training objective generates models that outperform sample-average approximation models out-of-sample. Moreover, they demonstrate that a naive implementation of leave-one-out cross-validation performs poorly in settings with limited data. Building upon this work, Gupta et al. (2024) proposes debiasing a model's in-sample performance by incorporating a variance gradient correction term derived via sensitivity analysis. Unfortunately, it is unclear how to extend their approach to our setting, as their approach applies to problems with linear objectives over subsets of $[0,1]^n$ (Gupta et al., 2024).

1.3 Structure

The rest of the paper is laid out as follows:

- In Section 2, we propose a generalization bound on the test set error of a sparse regressor in terms of its k-fold error and its hypothesis stability, due to Bousquet and Elisseeff (2002) for the special case of leave-one-out. Motivated by this result, we propose cross-validating by minimizing a weighted sum of output stability and cross-validation score, rather than CV score alone.
- In Section 3, we observe that the generalization bound is potentially expensive to evaluate, because computing it involves solving up to $k+1$ MIOs (in the k-fold case), and accordingly develop tractable lower and upper bounds on the generalization error that can be computed without solving any MIOs.
- In Section 4, we propose an efficient coordinate descent scheme for identifying locally optimal hyperparameters with respect to the generalization error. Specifically, in Section 4.1, we develop an efficient scheme for minimizing the confidence-adjusted cross-validation error with respect to τ, and in Section 4.2, we propose a scheme for optimizing γ.
- In Section 5, we benchmark our proposed approaches on both synthetic and real datasets. On synthetic datasets, we find that confidence adjustment significantly improves the accuracy of our regressors with respect to identifying the ground truth. Across a suite of 13 real datasets, we find that our confidence-adjusted cross-validation procedure improves the relative out-of-sample performance of our regressors by 4%, on average, compared to cross-validating without confidence adjustment. Moreover, the proposed approach leads to a 50-80% improvement in the number of MIOs solved compared to standard grid search techniques, without sacrificing solution quality.

Notation

We let non-boldface characters such as b denote scalars, lowercase bold-faced characters such as $\boldsymbol{x}$ denote vectors, uppercase bold-faced characters such as $\boldsymbol{A}$ denote matrices, and calligraphic uppercase characters such as $\mathcal{Z}$ denote sets. We let $[n]$ denote the running set of indices $\{1,\ldots,n\}$, and $\|\boldsymbol{x}\|_0 := |\{j : x_j \neq 0\}|$ denote the ℓ_0 pseudo-norm, i.e., the number of non-zero entries in $\boldsymbol{x}$. Finally, we let $\boldsymbol{e}$ denote the vector of ones, and $\boldsymbol{0}$ denote all zeros.

Further, we repeatedly use notation commonplace in the supervised learning literature. We consider a setting where we observe covariates $\boldsymbol{X} := (\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n) \in \mathbb{R}^{n\times p}$ and response data $\boldsymbol{y} := (y_1,\ldots y_n) \in \mathbb{R}^n$. We say that $(\boldsymbol{X},\boldsymbol{y})$ is a training set, and let $\boldsymbol{\beta}$ denote a regressor fitted on this training set. In cross-validation, we are also interested in the behavior of $\boldsymbol{\beta}$ after leaving out portions of the training set. We let $(\boldsymbol{X}^{(i)},\boldsymbol{y}^{(i)})$ denote the training set with the ith data point left out, and denote by $\boldsymbol{\beta}^{(i)}$ the regressor obtained after leaving out the ith point. Similarly, given a partition $\mathcal{N}_1,\ldots,\mathcal{N}_k$ of $[n]$ and $j \in [k]$, we let $(\boldsymbol{X}^{(\mathcal{N}_j)},\boldsymbol{y}^{(\mathcal{N}_j)})$ denote the training set with the jth fold left out, and $\boldsymbol{\beta}^{(\mathcal{N}_j)}$ be the associated regressor.

2 Stability Adjusted Cross-Validation

This section derives a generalization bound on the test set error of a sparse regression model and justifies the confidence-adjusted cross-validation model (9) used for hyperparameter selection throughout the paper. We first define our notation.

Recall from (2) that the k-fold cross-validation error with hyperparameters (τ, γ) is:

$$h(\gamma,\tau) = \frac{1}{n}\sum_{j=1}^{k}\sum_{i\in\mathcal{N}_j}\left(y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta}^{(\mathcal{N}_j)}(\gamma,\tau)\right)^2$$
$$\text{s.t.} \quad \boldsymbol{\beta}^{(\mathcal{N}_j)}(\gamma,\tau) \in \underset{\boldsymbol{\beta}\in\mathbb{R}^p:\ \|\boldsymbol{\beta}\|_0\le\tau}{\arg\min} \ \frac{\gamma}{2}\|\boldsymbol{\beta}\|_2^2 + \|\boldsymbol{X}^{(\mathcal{N}_j)}\boldsymbol{\beta} - \boldsymbol{y}^{(\mathcal{N}_j)}\|_2^2.$$

Here $\{\mathcal{N}_j\}_{j\in[k]}$ is a partition of $[n]$, and each $\boldsymbol{\beta}^{(\mathcal{N}_j)}$ is assumed to be the unique minimizer of the $\mathcal{N}_j$-fold training problem for convenience. For each $j \in [k]$, we let the partial k-fold error be:

$$h_j(\gamma,\tau) := \sum_{i\in\mathcal{N}_j}\left(y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta}^{(\mathcal{N}_j)}(\gamma,\tau)\right)^2. \tag{4}$$

Therefore, the average k-fold error is given by $1/n\sum_{j=1}^{k} h_j(\gamma,\tau) = h(\gamma,\tau)$. Bousquet and Elisseeff (2002) developed a generalization bound on the test set error in terms of the leave-one-out cross-validation error.

We now introduce their notation and generalize their bound to k-fold CV. Let $\mathcal{S}$ represent a random draw over the test set, $\{\mathcal{N}_j\}_{j\in[k]}$ represent a partition of a training set of size n, and $\boldsymbol{\beta}^\star$ be a regressor trained over the entire training set with fixed hyperparameters (τ,γ). Let M represent an upper bound on the loss function $\ell(\boldsymbol{x}_i^\top\boldsymbol{\beta}^\star, y_i) = (y_i - \boldsymbol{x}_i^\top\boldsymbol{\beta}^\star)^2$ for any i (e.g., if $(\boldsymbol{x}_i, y_i)$ is drawn from a bounded domain – in numerical experiments, we approximate it using $\max_{i\in[n]} y_i^2$). Define μ_h as the hypothesis stability of our learner analogously to (Bousquet and Elisseeff, 2002, Definition 3) but where $k < n$ folds are possible:

$$\mu_h := \max_{j\in[k]} \mathbb{E}_{\boldsymbol{x}_i, y_i}\left[\left|\left(y_i - \boldsymbol{x_i}^\top\boldsymbol{\beta}^\star\right)^2 - \left(y_i - \boldsymbol{x_i}^\top\boldsymbol{\beta}^{(\mathcal{N}_j)}\right)^2\right|\right], \tag{5}$$

where the expectation is taken over all $(\boldsymbol{x}_i, y_i)$ drawn i.i.d. from the underlying stochastic process that generated the training set.

Hypothesis stability measures the worst-case average absolute change in the loss after omitting a fold of data. In computations, we approximate (5) via pointwise stability analogously to (Bousquet and Elisseeff, 2002, Definition 4) but where $k < n$ folds are possible:

$$\mu_h \approx \max_{j\in[k]} \frac{1}{n}\sum_{i=1}^{n}\left|\left(y_i - \boldsymbol{x_i}^\top\boldsymbol{\beta}^\star\right)^2 - \left(y_i - \boldsymbol{x_i}^\top\boldsymbol{\beta}^{(\mathcal{N}_j)}\right)^2\right|. \tag{6}$$

The following result follows from Chebyshev's inequality (proof deferred to Section D.1):

Theorem 1 *Suppose the training data $(\boldsymbol{x}_i, y_i)_{i\in[n]}$ are drawn from an unknown distribution $\mathcal{D}$ such that M and μ_h are finite constants. Further, suppose n is exactly divisible by k and each $\mathcal{N}_j$ is of cardinality n/k. Then, the following bound on the test error holds with probability at least $1-\Omega$:*

$$\frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}}(y_i - \boldsymbol{x}_i^\top\boldsymbol{\beta}^\star)^2 \le \frac{1}{n}\sum_{j\in[k]} h_j(\gamma,\tau) + \sqrt{\frac{M^2 + 6Mk\mu_h}{2k\Omega}}. \tag{7}$$

Theorem 1 reveals that, if μ_h is small, kCV generalizes to the test set with high probability. Moreover, when models have the same cross-validation error, hypothesis stability, and loss bound M, training on more folds results in a stronger generalization bound. Addi-

tionally, Theorem 1 justifies the regularization term $\gamma\|\boldsymbol{\beta}\|_2^2$ in (1): regularization implicitly controls hypothesis stability, leading to better generalization properties when μ_h is lower.

We now formalize this idea via the following result, which generalizes Bousquet and Elisseeff (2002, Theorem 22) from ordinary least squares regression to sparse regression:

Lemma 2 *Suppose the loss is L-Lipschitz with respect to $\boldsymbol{\beta}$, and let κ be an upper bound on the ℓ_2 norm of $\boldsymbol{x}_i$. Further, suppose $\boldsymbol{\beta}^\star$, $\boldsymbol{\beta}^{(\mathcal{N}_j)}$ share the same support for any fold of the data $\mathcal{N}_j$. Then, we have the following bound on the hypothesis stability:*

$$\mu_h \leq \frac{L^2\kappa^2}{\gamma n}. \tag{8}$$

Observe that Lemma 2 holds independently of k, the number of folds in the data.

Under relatively mild assumptions on the data generation process, if n is sufficiently large and p, k, γ, τ are fixed then all $\boldsymbol{\beta}^{(\mathcal{N}_j)}$'s share the same support (Gamarnik and Zadik, 2022; Bertsimas and Van Parys, 2020). In practice, increasing γ tends to decrease μ_h, but the decrease need not be monotonic due to changes in the support of the $\boldsymbol{\beta}^{(\mathcal{N}_j)}$'s; this relationship between γ and μ_h implicitly justifies the presence of the regularization term in the original Problem (1).

It is also worth noting that if L, κ are bounded above by some finite constant and γ is fixed, then $\mu_h \to 0$ and the cross-validation error well-approximates the test set error as $n, k \to \infty$. In particular, the probabilistic upper bound decreases as $1/\sqrt{k}$. This justifies our claim in the introduction that the kCV error is particularly likely to disappoint out-of-sample in underdetermined settings.

Proof The result follows from (Bousquet and Elisseeff, 2002, Theorem 22). Note that Bousquet and Elisseeff (2002) leverages the convexity of the lower-level optimization problems in their proof technique, while our lower-level problems are non-convex due to the sparsity constraint $\|\boldsymbol{\beta}\|_0 \leq \tau$. We first restrict each lower-level problem to only consider the indices i in $\boldsymbol{\beta}$ which are non-zero in each $\boldsymbol{\beta}^{(\mathcal{N}_j)}$, and drop the sparsity constraint, thus restoring convexity. Finally, while Bousquet and Elisseeff (2002) only bound the hypothesis stability in the case of $k = n$, their result holds identically when $k < n$, after modifying the definition of the hypothesis stability to account for $k < n$ as in Equation (5). ■

In practice, while we use Theorem 1 to motivate our approach, we do not explicitly minimize their bound. Indeed, experimental evidence in Section 5 reveals that Equation (7)'s bound is excessively conservative in practice, even in the case of leave-one-out cross-validation, particularly when $n > p$. This conservatism stems from using Chebyshev's inequality in the proof of Theorem 1, which is known to be tight for discrete measures and loose over continuous measures (Bertsimas and Popescu, 2005). Rather, motivated by the robust optimization literature, where probabilistic guarantees are used to motivate uncertainty sets but less stringent guarantees are used in practice to avoid excessive conservatism (see Gorissen et al., 2015, Section 3, for a detailed discussion), we take a different approach:

Overall Approach: We aim to reduce out-of-sample disappointment without being excessively conservative. Accordingly, we propose selecting hyperparameters through:

$$(\gamma, \tau) \in \underset{\gamma\in\mathbb{R}_+, \tau\in[p]}{\arg\min}\ g(\gamma, \tau) \text{ where } \quad g(\gamma, \tau) := \frac{1}{n}\sum_{j\in[k]} h_j(\gamma, \tau) + \delta\sqrt{\frac{M^2 + 6Mk\mu_h(\gamma, \tau)}{2k}}, \tag{9}$$

represents the stability-adjusted cross-validation error for a user-specified weight $\delta > 0$, which trades off the k-fold error and the stability μ_h. In particular, this facilitates cross-validation while being aware of output stability via a single hyperparameter δ, which can either be set according to the above generalization bound, or by calibrating its performance as suggested in Section 5.4.

If $\delta > 1$ then, by Theorem 1, g is an upper bound on the test set error with probability at least $1 - 1/\sqrt{\delta}$. However, we can choose δ to be any positive value and thereby trade-off the cross-validation error and model stability. In practice, μ_h is NP-hard to compute for a fixed (γ, τ), and thus, we approximate μ_h via the perspective relaxations (Atamtürk and Gómez, 2020) of the lower level problems in our numerical results.This approach is similar to Johansson et al. (2022), who (in a different context) derive a generalization bound to motivate minimizing a weighted sum of different terms in their bound.

Remark 3 (To Train or to Validate in (9)) *A similar bound to* (7) *can be derived using the empirical risk instead of kCV (Bousquet and Elisseeff, 2002, Theorem 11). However, their bound has a larger constant (*12*, not* 6*) and still involves the hypothesis stability μ_h.*

From a multi-objective optimization perspective, our approach selects a hyperparameter combination (γ, τ) on the Pareto frontier, simultaneously minimizing the hypothesis stability score μ_h and the least kCV error $\frac{1}{n}\sum_{j=1}^{k} h_j(\gamma, \tau)$. The weight δ serves as the scalarization factor (see Ehrgott, 2005, for a general theory of multi-objective optimization).

Our approach is also justified by the "transfer theorem" in differential privacy (Jung et al., 2019). Indeed, the definition of (ϵ, δ)-differential privacy (see Dwork et al., 2006) is very similar to the definition of hypothesis stability μ_h, and the transfer theorem (Jung et al., 2019, Lemma 3.4) shows that differentially private models generalize well out-of-sample (with respect to training error).

Finally, consider the limiting behavior of our estimator (9). When $\delta = 0$ or $n \to \infty$, $g(\gamma, \tau) = h(\gamma, \tau)$ is the k-fold error. Conversely, as $\delta \to \infty$ we select the most stable regressor ($\boldsymbol{\beta} = \mathbf{0}$), rather than the regressor that minimizes the k-fold error. The former case arises naturally in overdetermined settings, as fixing δ and letting $n \to \infty$ leads to more stable sparse regression models (Gamarnik and Zadik, 2022). The latter case is analogous to the $1/N$ portfolio selection strategy (cf. DeMiguel and Nogales, 2009), which is effective in high ambiguity settings.

3 Convex Relaxations of k-fold Cross-Validation Error

Section 2 proposes selecting the hyperparameters (τ, γ) in (1) by minimizing the function g defined in Problem (9) as a weighted sum of the k-fold cross-validation error h and the output stability μ_h. From an optimization perspective, this might appear challenging, because each evaluation of h_j requires solving a MIO, thus, evaluating g involves solving $k + 1$ MIOs.

To address this challenge, this section develops tractable upper and lower approximations of g and μ_h, which can be evaluated at a given (γ, τ) without solving any MIOs. From a theoretical perspective, one of our main contributions is that, given $\boldsymbol{x} \in \mathbb{R}^p$, we show how to construct bounds $\underline{\xi}, \overline{\xi}$ such that $\underline{\xi} \leq \boldsymbol{x}^\top \boldsymbol{\beta}^{(\mathcal{N}_j)} \leq \overline{\xi}$, which we can use to infer out-of-sample

predictions. In particular, we then leverage this insight to bound from above and below the functions:

$$h(\gamma,\tau) = 1/n\sum_{j=1}^{k} h_j(\gamma,\tau) = 1/n\sum_{j=1}^{k}\sum_{i\in\mathcal{N}_j}\left(y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta}^{(\mathcal{N}_j)}(\gamma,\tau)\right)^2, \tag{10}$$

$$\text{and} \quad \upsilon_{i,j}(\gamma,\tau) = (y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta}^{(\mathcal{N}_j)})^2, \tag{11}$$

which, in turn, bounds the output stability μ_h and the function g.

3.1 Bounds on the Prediction Spread

Given any $0 < \gamma$, it is well-known that Problem (1) admits the conic quadratic relaxation:

$$\zeta_{\text{persp}} = \min_{\boldsymbol{\beta}\in\mathbb{R}^p, \boldsymbol{z}\in[0,1]^p} \quad \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \frac{\gamma}{2}\sum_{i=1}^{p}\frac{\beta_i^2}{z_i} \quad \text{s.t.} \quad \sum_{i=1}^{p} z_i \leq \tau, \tag{12}$$

which is also known as the perspective relaxation (Ceria and Soares, 1999; Xie and Deng, 2020). If integrality constraints $\boldsymbol{z} \in \{0,1\}^p$ are added to (12), then the resulting mixed-integer optimization problem (MIO) is a reformulation of (1), where the logical constraints $z_i = 0$ if $\beta_i = 0$ $\forall i \in [p]$ are implicitly imposed via the domain of the perspective function β_i^2/z_i. Moreover, the optimal objective ζ_{persp} of (12) often provides tight lower bounds on the objective value of (1) (Pilanci et al., 2015; Bertsimas and Van Parys, 2020), and the optimal solution $\boldsymbol{\beta}^*_{\text{persp}}$ is a good estimator in its own right. As we establish in our main theoretical results, the perspective relaxation can also be used to obtain accurate approximations of and lower/upper bounds on the error given in (11).

Our next result reveals that any optimal solution of (1) lies in an ellipsoid centered at its continuous (perspective) relaxation, whose radius depends on the duality gap:

Theorem 4 *Given any $0 < \gamma$ and any bound*

$$\bar{u} \geq \min_{\boldsymbol{\beta}\in\mathbb{R}^p} \ \|\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y}\|_2^2 + \frac{\gamma}{2}\|\boldsymbol{\beta}\|_2^2 \text{ s.t. } \ \|\boldsymbol{\beta}\|_0 \leq \tau, \tag{13}$$

the inequality

$$(\boldsymbol{\beta}^\star_{persp} - \boldsymbol{\beta}^\star_{MIO})^\top \left(\boldsymbol{X}^\top\boldsymbol{X} + \frac{\gamma}{2}\mathbb{I}\right)(\boldsymbol{\beta}^\star_{persp} - \boldsymbol{\beta}^\star_{MIO}) \leq (\bar{u} - \zeta_{persp}) \tag{14}$$

holds, where $\boldsymbol{\beta}^*_{MIO}$ is an optimal solution of (13) *and $\boldsymbol{\beta}^*_{persp}$ is optimal to* (12)*.*

Proof Let $\epsilon > 0$ be a small positive constant and let

$$f_\epsilon(\boldsymbol{\beta}) := \min_{\boldsymbol{z}\in[0,1]^p:\boldsymbol{e}^\top\boldsymbol{z}\leq k} \quad \|\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y}\|_2^2 + \frac{\gamma}{2}\sum_{i\in[p]}\frac{\beta_i^2+\epsilon}{z_i} - \frac{\epsilon\gamma p}{2}, \tag{15}$$

denote the objective value of the perspective relaxation at a given $\boldsymbol{\beta}$, where we apply the small perturbation ϵ so that $z_i^\star > 0$. Note that f_ϵ is non-decreasing in ϵ. The function $f(\boldsymbol{\beta})$ is twice differentiable with respect to $\boldsymbol{\beta}$, and admits the following integral Taylor series

expansion about $\boldsymbol{\beta}^*_{persp}$, an optimal solution to (15) (e.g., Sidford, 2024, Lemma 3.5.3)

$$\begin{aligned} f_\epsilon(\boldsymbol{\beta}) =& f_\epsilon(\boldsymbol{\beta}^*_{persp}) + \langle f_\epsilon(\boldsymbol{\beta}^*_{persp}), \boldsymbol{\beta} - \boldsymbol{\beta}^*_{persp}\rangle \\ &+ \int_0^1 (1-\alpha)(\boldsymbol{\beta} - \boldsymbol{\beta}^*_{persp})^\top \nabla^2 f_\epsilon(\boldsymbol{\beta}^*_{persp} + \alpha(\boldsymbol{\beta} - \boldsymbol{\beta}^*_{persp}))(\boldsymbol{\beta} - \boldsymbol{\beta}^*_{persp}) d\alpha \end{aligned}$$

Moreover, the Hessian at a given $\boldsymbol{\beta}$ is $\nabla^2 f_\epsilon(\boldsymbol{\beta}) = 2\boldsymbol{X}^\top \boldsymbol{X} + \gamma \text{Diag}(\boldsymbol{z}^*)^{-1}$, where $\boldsymbol{z}^* > \mathbf{0}$ because of the perturbation term in the objective. Since $\boldsymbol{z}^* \leq \boldsymbol{e}$, the Hessian is such that $\nabla^2 f_\epsilon(\boldsymbol{\beta}) \succeq 2\boldsymbol{X}^\top \boldsymbol{X} + \gamma \mathbb{I}$. Moreover, replacing $\nabla^2 f_\epsilon(\boldsymbol{\beta})$ with a valid lower bound with respect to the Loewener partial order gives a lower bound on $f(\boldsymbol{\beta})$. Thus, integrating with respect to α yields the bound

$$f_\epsilon(\boldsymbol{\beta}) \geq f_\epsilon(\boldsymbol{\beta}^*_{persp}) + (\boldsymbol{\beta} - \boldsymbol{\beta}^*_{persp})^\top (\boldsymbol{X}^\top \boldsymbol{X} + \frac{\gamma}{2}\mathbb{I})(\boldsymbol{\beta} - \boldsymbol{\beta}^*_{persp}),$$

where we omit the first-order term $\langle \nabla f(\boldsymbol{\beta}^*_{persp}), \boldsymbol{\beta} - \boldsymbol{\beta}_{persp}\rangle$ because it is non-negative for an optimal $\boldsymbol{\beta}^*_{persp}$ (cf. Bertsekas, 2016, Chap. 1).

The result then follows by inserting $\boldsymbol{\beta}_{MIO}$ into this bound, taking limits as $\epsilon \to 0$ to avoid including perturbation terms within our bound, and noting that $f(\boldsymbol{\beta}_{MIO})$ does not require that $\boldsymbol{z}$ is integral, and thus is a lower bound on $\bar{u}$. We remark that taking limits is justified by, e.g., the monotone convergence theorem (Grimmett and Stirzaker, 2020). Indeed, the objective value of $f_\epsilon(\boldsymbol{\beta}^*_{persp})$ is non-increasing as we decrease ϵ, bounded from below by ζ_{persp}, and attains this bound in the limit. ■

Using Theorem 4, we can compute bounds on $h_j(\gamma, \tau)$ in (10) by solving

$$\min / \max \sum_{i \in \mathcal{N}_j} \left(y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta}\right)^2 \tag{16a}$$

$$\text{s.t. } (\boldsymbol{\beta}^{(\mathcal{N}_j)}_{\text{persp}} - \boldsymbol{\beta})^\top \left((\boldsymbol{X}^{(\mathcal{N}_j)})^\top \boldsymbol{X}^{(\mathcal{N}_j)} + \frac{\gamma}{2}\mathbb{I}\right)(\boldsymbol{\beta}^{(\mathcal{N}_j)}_{\text{persp}} - \boldsymbol{\beta}) \leq (\bar{u}^{(\mathcal{N}_j)} - \zeta^{(\mathcal{N}_j)}_{\text{persp}}), \tag{16b}$$

where $\boldsymbol{\beta}^{(\mathcal{N}_j)}_{\text{persp}}$ and $\zeta^{(\mathcal{N}_j)}_{\text{persp}}$ are in this the optimal solution and objective value of the perspective relaxation with fold $\mathcal{N}_j$ removed, and $\bar{u}^{(\mathcal{N}_j)}$ is an associated upper bound. Bounds for the function $h_(\gamma, \tau)$ then immediately follow by simply adding the bounds associated with $h_j(\gamma, \tau)$ for all $j \in [k]$. Bounds for $\upsilon_{i,j}(\gamma, \tau)$ in (11) could be obtained similarly by simply updating the objective. However, we provide a simpler method in the next section.

Remark 5 (Computability of the bounds) *Observe that a lower bound on the k-fold error can easily be computed by solving a convex quadratically constrained quadratic problem, while an upper bound can be computed by noticing that the maximization problem* (16) *is a trust region problem in $\boldsymbol{\beta}$, which can be reformulated as a semidefinite problem (Hazan and Koren, 2016). One could further tighten these bounds by imposing a sparsity constraint on $\boldsymbol{\beta}$, but this may not be practically tractable.*

3.2 Closed-form Bounds on the Prediction Spread

While solving the perspective relaxation (12) is necessary to solve the MIO (13) via branch-and-bound (in particular, the perspective relaxation is the root node in a branch-and-bound scheme (Mazumder et al., 2023)), the additional two optimization problems (16)

are not. Moreover, solving trust-region problems can be expensive in large-scale problems. Accordingly, we now present alternative bounds that may be weaker, but can be obtained in closed form. In numerical experiments (§5), these closed-form bounds already reduce the number of MIOs that need to be solved by up to 80% when compared to grid search.

Theorem 6 *Given any vector $\boldsymbol{x} \in \mathbb{R}^p$ and any bound*

$$\bar{u} \geq \min_{\boldsymbol{\beta} \in \mathbb{R}^p} \|\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y}\|_2^2 + \frac{\gamma}{2}\|\boldsymbol{\beta}\|_2^2 \text{ s.t. } \|\boldsymbol{\beta}\|_0 \leq \tau, \tag{17}$$

the inequalities

$$\begin{aligned}\boldsymbol{x}^\top \boldsymbol{\beta}^*_{persp} - \sqrt{(\bar{u} - \zeta_{persp})\, \boldsymbol{x}^\top \left(\boldsymbol{X}^\top \boldsymbol{X} + \frac{\gamma}{2}\mathbb{I}\right)^{-1} \boldsymbol{x}} &\leq \boldsymbol{x}^\top \boldsymbol{\beta}^*_{MIO} \\ &\leq \boldsymbol{x}^\top \boldsymbol{\beta}^*_{persp} + \sqrt{(\bar{u} - \zeta_{persp})\, \boldsymbol{x}^\top \left(\boldsymbol{X}^\top \boldsymbol{X} + \frac{\gamma}{2}\mathbb{I}\right)^{-1} \boldsymbol{x}}\end{aligned}$$

hold, where $\boldsymbol{\beta}^*_{MIO}$ is an optimal solution of (17) *and $\boldsymbol{\beta}^*_{persp}$ is optimal to* (12)*.*

Proof From Theorem 4, we have the inequality

$$(\boldsymbol{\beta}^\star_{\text{persp}} - \boldsymbol{\beta}^\star_{\text{MIO}})^\top \left(\boldsymbol{X}^\top \boldsymbol{X} + \frac{\gamma}{2}\mathbb{I}\right)(\boldsymbol{\beta}^\star_{\text{persp}} - \boldsymbol{\beta}^\star_{\text{MIO}}) \leq (\bar{u} - \zeta_{\text{persp}}). \tag{18}$$

By the Schur Complement Lemma (see, e.g., Boyd et al., 1994), this is equivalent to

$$(\bar{u} - \zeta_{\text{persp}}) \left(\boldsymbol{X}^\top \boldsymbol{X} + \frac{\gamma}{2}\mathbb{I}\right)^{-1} \succeq (\boldsymbol{\beta}^\star_{\text{persp}} - \boldsymbol{\beta}^\star_{\text{MIO}})(\boldsymbol{\beta}^\star_{\text{persp}} - \boldsymbol{\beta}^\star_{\text{MIO}})^\top$$

Next, we can left/right multiply this expression by an arbitrary matrix $\boldsymbol{W} \in \mathbb{R}^{m \times p}$. This gives:

$$(\bar{u} - \zeta_{\text{persp}})\boldsymbol{W} \left(\boldsymbol{X}^\top \boldsymbol{X} + \frac{\gamma}{2}\mathbb{I}\right)^{-1} \boldsymbol{W}^\top \succeq (\boldsymbol{W}\boldsymbol{\beta}^\star_{\text{persp}} - \boldsymbol{W}\boldsymbol{\beta}^\star_{\text{MIO}})(\boldsymbol{W}\boldsymbol{\beta}^\star_{\text{persp}} - \boldsymbol{W}\boldsymbol{\beta}^\star_{\text{MIO}})^\top.$$

In particular, setting $\boldsymbol{W} = \boldsymbol{x}^\top$ for a vector $\boldsymbol{x} \in \mathbb{R}^p$ gives the inequality

$$(\bar{u} - \zeta_{\text{persp}})\boldsymbol{x}^\top \left(\boldsymbol{X}^\top \boldsymbol{X} + \frac{\gamma}{2}\mathbb{I}\right)^{-1} \boldsymbol{x} \geq (\boldsymbol{x}^\top(\boldsymbol{\beta}^\star_{\text{persp}} - \boldsymbol{\beta}^\star_{\text{MIO}}))^2,$$

which we rearrange to obtain the result. ■

Corollary 7 *For any $\boldsymbol{W} \in \mathbb{R}^{m \times p}$ we have that*

$$(\bar{u} - \zeta_{persp})\text{tr}\left(\boldsymbol{W} \left(\boldsymbol{X}^\top \boldsymbol{X} + \frac{\gamma}{2}\mathbb{I}\right)^{-1} \boldsymbol{W}^\top\right) \geq \|\boldsymbol{W}(\boldsymbol{\beta}^\star_{persp} - \boldsymbol{\beta}^\star_{MIO})\|_2^2,$$

Applying Theorem 6 to the problem

$$\bar{u}^{(\mathcal{N}_j)} \geq \min_{\boldsymbol{\beta} \in \mathbb{R}^p} \|\boldsymbol{X}^{(\mathcal{N}_j)}\boldsymbol{\beta} - \boldsymbol{y}^{(\mathcal{N}_j)}\|_2^2 + \frac{\gamma}{2}\|\boldsymbol{\beta}\|_2^2 \text{ s.t. } \|\boldsymbol{\beta}\|_0 \leq \tau,$$

we have the bounds

$$\underline{\xi}_{i,j} := \boldsymbol{x_i}^\top \boldsymbol{\beta}^*_{persp} - \sqrt{\boldsymbol{x_i}^\top \left(\boldsymbol{X}^{(\mathcal{N}_j)\top}\boldsymbol{X}^{(i)} + \frac{\gamma}{2}\mathbb{I}\right)^{-1} \boldsymbol{x_i}\left(\bar{u}^{(\mathcal{N}_j)} - \zeta^{(\mathcal{N}_j)}\right)},$$
$$\bar{\xi}_{i,j} := \boldsymbol{x_i}^\top \boldsymbol{\beta}^*_{persp} + \sqrt{\boldsymbol{x_i}^\top \left(\boldsymbol{X}^{(\mathcal{N}_j)\top}\boldsymbol{X}^{(\mathcal{N}_j)} + \frac{\gamma}{2}\mathbb{I}\right)^{-1} \boldsymbol{x_i}\left(\bar{u}^{(\mathcal{N}_j)} - \zeta^{(\mathcal{N}_j)}\right)}$$

where $0 < 2\epsilon < \gamma$ and $\underline{\xi} \leq \boldsymbol{x_i}^\top \boldsymbol{\beta}^*_{MIO} \leq \bar{\xi}$. We can use these bounds to bound terms $\upsilon_{i,j}(\gamma,\tau)$ in (11).

Corollary 8 *We have the bounds on the ith prediction error associated with fold j*

$$\max\left((y_i - \underline{\xi}_{i,j})^2, (y_i - \bar{\xi}_{i,j})^2\right) \geq \nu_{i,j}(\gamma,\tau) \geq \begin{cases} (y_i - \underline{\xi}_{i,j})^2 & \text{if } y_i < \underline{\xi}_{i,j} \\ 0 & \text{if } y_i \in [\underline{\xi}_{i,j}, \bar{\xi}_{i,j}], \\ (\bar{\xi}_{i,j} - y_i)^2 & \text{if } y_i > \bar{\xi}_{i,j}. \end{cases} \tag{19}$$

Moreover, since $h(\gamma,\tau) = \frac{1}{n}\sum_{j=1}^k \sum_{i\in\mathcal{N}_j} \nu_{i,j}(\gamma,\tau)$, we can compute lower and upper bounds on the k-th fold cross-validation error by adding the individual bounds. Observe that the bounds computed by summing disaggregated bounds could be substantially worse than those obtained by letting $\boldsymbol{W}$ be a matrix with all omitted columns in the jth fold of $\boldsymbol{X}$ in the proof of Theorem 6. Nonetheless, the approach outlined here might be the only one feasible in large scale instances, as they are obtained directly from the perspective relaxation without solving additional optimization problems, while an aggregated approach would involve solving an auxiliary semidefinite optimization problem. Despite the loss in quality, we show in our computational sections that (combined with the methods discussed in §4), the disaggregated bounds are sufficient to lead to a 50%-80% reduction in the number of MIO solved with respect to grid search.

We conclude this subsection with two remarks.

Remark 9 (Relaxation Tightness) *If the perspective relaxation is tight, as occurs when n is sufficiently large under certain assumptions on the data generation process (Pilanci et al., 2015; Reeves et al., 2019) then $\underline{\xi} = \bar{\xi} = \boldsymbol{x_i}^\top \boldsymbol{\beta}^*_{persp}$, and Corollary 8's bounds on the cross-validation error are definitionally tight. Otherwise, as pointed out in Remark 10, (19)'s bound quality depends on the tightness of the relaxation and on how close the features $\boldsymbol{x}_i$ are to the rest of the data.*

Remark 10 (Intuition) *Theorem 6 states that $\boldsymbol{x}^\top \boldsymbol{\beta}^*_{MIO} \approx \boldsymbol{x}^\top \boldsymbol{\beta}^*_{persp}$, where the approximation error is determined by two components. The quantity $\sqrt{\bar{u} - \zeta_{persp}}$ is related to the strength of the perspective relaxation, with a stronger relaxation resulting in a better approximation. The quantity $\sqrt{\boldsymbol{x}^\top \left(\boldsymbol{X}^\top\boldsymbol{X} + \frac{\gamma}{2}\mathbb{I}\right)^{-1}\boldsymbol{x}}$ is related to the likelihood that $\boldsymbol{x}$ is generated from the same distribution as the rows of $\boldsymbol{X}$, with larger likelihoods resulting in better approximations. Indeed, if $n > p$, each column of $\boldsymbol{X}$ has 0 mean but has not been standardized, and each row of $\boldsymbol{X}$ is generated iid from a multivariate Gaussian distribution, then $\frac{n(n-1)}{n+1}\boldsymbol{x}^\top \left(\boldsymbol{X}^\top\boldsymbol{X}\right)^{-1}\boldsymbol{x} \sim T^2(p, n-1)$ is Hotelling's two-sample T-square test statistic (Hotelling, 1931), used to test whether $\boldsymbol{x}$ is generated from the same Gaussian distribution. Note that if $\boldsymbol{x}$ is drawn from the same distribution as the rows of $\boldsymbol{X}$ (as may be the case in cross-validation), then $\mathbb{E}\left[\boldsymbol{x}^\top \left(\boldsymbol{X}^\top\boldsymbol{X}\right)^{-1}\boldsymbol{x}\right] = \frac{p(n+1)}{n(n-p-2)}$.*

3.3 Further Improvements for Lower Bounds

Corollary 8 implies we obtain a valid upper and lower bound on h at a given hyperparameter combination γ, τ after solving k perspective relaxations and computing n terms of the form

$$\sqrt{\boldsymbol{x_i}^\top \left(\boldsymbol{X}^{(\mathcal{N}_j)^\top} \boldsymbol{X}^{(\mathcal{N}_j)} + \frac{\gamma}{2}\mathbb{I} \right)^{-1} \boldsymbol{x_i}}.$$

A drawback of Corollary 8 is that if $\boldsymbol{x_i}^\top \boldsymbol{\beta}^*_{persp} \approx y_i$ for each $i \in \mathcal{N}_j$, i.e., the prediction of the perspective relaxation (without the jth fold) is close to the response associated with point i, then Corollary 8's lower bound is 0. A similar situation can happen with the stronger bounds for $h_j(\gamma, \tau)$ obtained from Theorem 1 and Problem (16). We now propose a different bound on $h_j(\gamma, \tau)$, which is sometimes effective in this circumstance.

First, define the function $f(\gamma, \tau)$ to be the in-sample training error without removing any folds and with parameters (γ, τ),

$$f(\gamma, \tau) := \frac{1}{n} \sum_{i=1}^{n} (y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta}(\gamma, \tau))^2 \quad \text{s.t.} \quad \boldsymbol{\beta}(\gamma, \tau) \in \underset{\boldsymbol{\beta} \in \mathbb{R}^p:\ \|\boldsymbol{\beta}\|_0 \leq \tau}{\arg\min} \ \frac{\gamma}{2}\|\boldsymbol{\beta}\|_2^2 + \|\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y}\|_2^2,$$

and let $f_{\mathcal{N}_j}(\gamma, \tau) := \sum_{i \in \mathcal{N}_j} \left(y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta}(\gamma, \tau) \right)^2$ denote the training error associated with the jth fold, with $1/n \sum_{j=1}^{k} f_{\mathcal{N}_j}(\gamma, \tau) = f(\gamma, \tau)$. Observe that evaluating $h_{\mathcal{N}_j}(\gamma, \tau)$ involves solving k MIOs, while evaluating f requires solving one.

Proposition 11 *For any $\gamma \geq 0$, any $\tau \in [p]$ and any $j \in [k]$, $f_j(\gamma, \tau) \leq h_j(\gamma, \tau)$. Moreover, we have that $f(\gamma, \tau) \leq h(\gamma, \tau)$.*

Proof Given $j \in [k]$, consider the following two optimization problems

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^p : \|\boldsymbol{\beta}\|_0 \leq \tau} \sum_{i=1}^{n} (y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta})^2 + \frac{\gamma}{2}\|\boldsymbol{\beta}\|_2^2 \tag{20}$$

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^p : \|\boldsymbol{\beta}\|_0 \leq \tau} \sum_{i \notin \mathcal{N}_j} (y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta})^2 + \frac{\gamma}{2}\|\boldsymbol{\beta}\|_2^2, \tag{21}$$

let $\boldsymbol{\beta^*}$ be an optimal solution of (20), and let $\boldsymbol{\beta^j}$ be an optimal solution of (21). Since

$$\sum_{i \notin \mathcal{N}_j} (y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta^j})^2 + \frac{\gamma}{2}\|\boldsymbol{\beta^j}\|_2^2 \leq \sum_{i \notin \mathcal{N}_j} (y_i - \boldsymbol{x}^\top \boldsymbol{\beta^*})^2 + \frac{\gamma}{2}\|\boldsymbol{\beta^*}\|_2^2, \quad \text{and}$$

$$\sum_{i \notin \mathcal{N}_j} (y_i - \boldsymbol{x}^\top \boldsymbol{\beta^j})^2 + \sum_{i \in \mathcal{N}_j} (y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta^j})^2 + \frac{\gamma}{2}\|\boldsymbol{\beta^j}\|_2^2 \geq$$

$$\sum_{i \notin \mathcal{N}_j} (y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta^*})^2 + \sum_{i \in \mathcal{N}_j} (y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta^*})^2 + \frac{\gamma}{2}\|\boldsymbol{\beta^*}\|_2^2,$$

we conclude that $\sum_{i \in \mathcal{N}_j} (y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta^*})^2 \leq \sum_{i \in \mathcal{N}_j} (y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta^j})^2$. The result follows. ■

Next, we develop a stronger bound on the k-fold error, by observing that our original proof technique relies on interpreting the optimal solution when training on the entire dataset as a feasible solution when leaving out the jth fold, and that this feasible solution

can be improved to obtain a tighter lower bound. Therefore, given any $\boldsymbol{z} \in \{0,1\}^p$, let us define the function:

$$f^{(\mathcal{N}_j)}(\boldsymbol{z}) := \min_{\boldsymbol{\beta} \in \mathbb{R}^p} \quad \frac{\gamma}{2} \sum_{j \in [p]} \beta_j^2 + \|\boldsymbol{X}^{(\mathcal{N}_j)}\boldsymbol{\beta} - \boldsymbol{y}^{(\mathcal{N}_j)}\|_2^2 \text{ s.t. } \beta_j = 0 \text{ if } z_j = 0 \; \forall j \in [p],$$

to be the optimal training loss (including regularization) when we leave out the jth fold and have the binary support vector $\boldsymbol{z}$. Then, fixing γ, τ and letting u^* denote the optimal objective value of (20), i.e., the optimal training loss on the entire dataset (including regularization) and $\boldsymbol{\beta}^{(\mathcal{N}_j)}(\boldsymbol{z})$ denote an optimal choice of $\boldsymbol{\beta}$ for this $\boldsymbol{z}$, we have:

Proposition 12 *For any τ-sparse binary vector $\boldsymbol{z}$, the following inequality holds:*

$$u^* \leq f^{(\mathcal{N}_j)}(\boldsymbol{z}) + \sum_{i \in \mathcal{N}_j} \left(y_i - \boldsymbol{x}_i^\top \boldsymbol{\beta}^{(\mathcal{N}_j)}(\boldsymbol{z}) \right)^2 \tag{22}$$

Proof The right-hand side of this inequality corresponds to the objective value of a feasible solution to (20), while $u^\star$ is the optimal objective value of (20). ■

Corollary 13 *Let $\boldsymbol{z}$ denote a τ-sparse binary vector. Then, we have the following bound on the jth partial cross-validation error:*

$$h_j(\gamma, \tau) \geq u^* - f^{(\mathcal{N}_j)}(\boldsymbol{z}). \tag{23}$$

Proof The right-hand side of this bound is maximized by setting $\boldsymbol{z}$ to be a binary vector which minimizes $f^{(\mathcal{N}_j)}(\boldsymbol{z})$, and therefore this bound is valid for any $\boldsymbol{z}$. ■

We close this section with three remarks:

Remark 14 (Bound quality) *Observe that bound* (23) *is at least as strong as $f_j(\gamma, \tau)$ with $\boldsymbol{z}$ encoding an optimal choice of support in* (20)*. Indeed, if $\boldsymbol{\beta}^{(\mathcal{N}_j)}(\boldsymbol{z})$ solves* (20)*, then both bounds agree and equal $h_j(\gamma, \tau)$ but otherwise* (23) *is strictly stronger. Moreover, since $f_j(\gamma, \tau)$ is typically nonzero, then the bound* (23) *is positive as well and can improve upon the lower bound in* (19)*. Finally, it is easy to construct an example where the lower bound in* (19) *is stronger than* (23)*, thus neither lower bound dominates the other.*

Remark 15 (Computational efficiency) *Computing lower bound* (23) *for each $j \in [k]$ requires solving at least one MIO, corresponding to* (20)*, which is a substantial improvement over the k MIOs required to compute h but may still be an expensive computation. However, using any lower bound on u^*, for example, corresponding to the optimal solution of a perspective relaxation, gives valid lower bounds. Therefore, in practice, we suggest using a heuristic instead to bound h_j from below, e.g., rounding a perspective relaxation.*

4 Optimizing the Cross-Validation Loss

In this section, we present an efficient coordinate descent scheme that identifies (approximately) optimal hyperparameters (γ, τ) with respect to the metric:

$$g(\gamma, \tau) := \frac{1}{n} \sum_{j \in [k]} h_j(\gamma, \tau) + \delta \sqrt{\frac{M^2 + 6Mk\mu_h(\gamma, \tau)}{2k}}, \tag{24}$$

by iteratively minimizing τ and γ. In the tradition of coordinate descent schemes, with initialization τ_0, γ_0, we repeatedly solve the following two optimization problems:

$$\tau_t \in \arg\min_{\tau \in [p]} \quad g(\gamma_t, \tau), \tag{25}$$

$$\gamma_{t+1} \in \arg\min_{\gamma > 0} \quad g(\gamma, \tau_t), \tag{26}$$

until we either detect a cycle or converge to a locally optimal solution. To develop this scheme, in Section 4.1 we propose an efficient technique for solving Problem (25), and in Section 4.2 we propose an efficient technique for (approximately) solving Problem (26). Accordingly, our scheme could also be used to identify an optimal choice of γ if τ is already known, e.g., in a context where regulatory constraints specify the number of features that may be included in a model.

Our overall approach is motivated by two key observations. First, we design a method that obtains local, rather than global, minima, because g is a highly non-convex function and even evaluating g requires solving n MIOs, which suggests that global minima of g may not be attainable in a practical amount of time at scale. Second, we use coordinate descent to seek local minima because if either τ or γ is fixed, it is possible to efficiently optimize the remaining hyperparameter with respect to g by leveraging the convex relaxations developed in the previous section.

4.1 Parametric Optimization of k-fold With Respect to Sparsity

Consider the following optimization problem, where γ is fixed here and throughout this subsection:

$$\begin{aligned} \min_{\tau \in [p]} \quad g(\gamma, \tau) := \min & \sum_{j \in [k]} \sum_{i \in \mathcal{N}_j} (y_i - \boldsymbol{x}_i^\top \boldsymbol{\beta}^{(\mathcal{N}_j)})^2 + \delta \sqrt{\frac{M^2 + 6Mk\mu_h(\gamma, \tau)}{2k}}, \\ \text{s.t.} \quad & \boldsymbol{\beta}^{(\mathcal{N}_j)} \in \arg\min_{\boldsymbol{\beta} \in \mathbb{R}^p:\ \|\boldsymbol{\beta}\|_0 \leq \tau} \ \frac{\gamma}{2}\|\boldsymbol{\beta}\|_2^2 + \|\boldsymbol{X}^{(\mathcal{N}_j)}\boldsymbol{\beta} - \boldsymbol{y}^{(\mathcal{N}_j)}\|_2^2 \quad \forall i \in [n]. \end{aligned} \tag{27}$$

This problem can be solved by complete enumeration, i.e., for each $\tau \in [p]$, we compute an optimal $\beta^{(\mathcal{N}_j)}$ for each $j \in [k]$ by solving an MIO, and we also compute $\boldsymbol{\beta}$, an optimal regressor when no data points are omitted, in order to compute the terms $(y_i - \boldsymbol{x}_i^\top \boldsymbol{\beta})^2$ which appear in the hypothesis stability μ_h. This involves solving $(k+1)p$ MIOs, which is extremely expensive at scale. We now propose a technique[3] for minimizing g without solving all these MIOs, namely Algorithm 1.

Algorithm 1 has two main phases, which both run in a loop. In the first phase, we construct valid lower and upper bounds on $h_{\mathcal{N}_j}(\tau)$ for each $\mathcal{N}_j$ and each τ without solving any MIOs. We begin by solving, for each potential sparsity budget $\tau \in [p]$, the perspective relaxation with all datapoints included. Call this relaxation's objective value $\bar{v}_\tau$. We then solve each perspective relaxation that arises after omitting one data fold $\mathcal{N}_j : j \in [k]$, with objective values $v_{\tau,\mathcal{N}_j}$ and solutions $\boldsymbol{\beta}_{\tau,\mathcal{N}_j}$. Next, we compute lower and upper bounds on the k-fold error $h_{\mathcal{N}_j}(\tau)$ using the methods derived in Section 3, which are summarized in the routine `compute_bounds` described in Algorithm 2. Finally, we compute lower and upper

3. We omit some details around bounding the hypothesis stability for the sake of brevity; these bounds can be obtained similarly to those for the LOOCV error.

bounds on the stability using similar techniques (omitted here and from Algorithm 2 for the sake of conciseness). By solving $\mathcal{O}(kp)$ relaxations (and no MIOs), we have upper and lower estimates on the k-fold error and stability that are often accurate in practice, as described by Theorem 6. This concludes the first phase of Algorithm 1.

After completing the first loop in Algorithm 1, one may already terminate the algorithm. Indeed, according to our numerical experiments in Section 5, this already provides high-quality solutions. Alternatively, one may proceed with the second phase of Algorithm 1 and solve (25) to optimality, at the expense of solving (a potentially large number of) MIOs.

In the second phase, Algorithm 1 identifies the cardinality τ^* with the best lower bound (and thus, in an optimistic scenario, the best potential value). Then, it identifies the fold $\mathcal{N}_j^*$ with the largest uncertainty around the k-fold estimate $h_{\mathcal{N}_j^*}(\tau^*)$, and solves an MIO to compute the exact partial k-fold error. This process is repeated until (27) is solved to provable optimality, or a suitable termination condition (e.g., a limit on computational time) is met.

To solve each MIO in Algorithm 1, we invoke a Generalized Benders Decomposition scheme (Geoffrion, 1972), which was specialized to sparse regression problems by Bertsimas and Van Parys (2020), enhanced with some ideas from the optimization literature. For the sake of conciseness, we defer these implementation details to Appendix E.

Algorithm 1 in Action: Figure 2 depicts visually the lower and upper bounds on g from Algorithm 2 (left) and after running Algorithm 1 to completion (right) on a synthetic sparse regression instance generated in the fashion described in our numerical experiments, with $k = n, \delta = 0$, $n = 200, p = 20$, $\gamma = 1/\sqrt{n}$, $\tau_{\text{true}} = 10$, $\rho = 0.7$, $\nu = 1$, where $\tau \in \{2, \ldots, 19\}$, and using the outer-approximation method of Bertsimas and Van Parys (2020) as our solver for each MIO with a time limit of 60s. We observe that Algorithm 1 solved 1694 MIOs to identify the optimal τ, which is a 53% improvement on complete enumeration. Interestingly, when $\tau = 19$, the perspective relaxation is tight after omitting any fold of the data and we have tight bounds on the LOOCV error without solving any MIOs. In our computational experiments, see Section 5.1, we test Algorithm 1 on real datasets and find that it reduces the number of MIOs that need to be solved by 50-80% with respect to complete enumeration. For more information on how the bounds evolve over time, we provide a GIF with one frame each time a MIO is solved at the anonymous link `https://drive.google.com/file/d/1EZdNwlV9sEEnludGGM7v2nGpB7tzZvz4/view?usp=sharing`.

4.2 Parametric Optimization of Confidence-Adjusted k-fold With Respect to γ

In this section, we propose a technique for approximately minimizing the confidence-adjusted LOOCV error with respect to the regularization hyperparameter γ.

We begin with two observations from the literature. First, as observed by Stephenson et al. (2021), the LOOCV error $h(\gamma, \tau)$ is often quasi-convex with respect to γ when $\tau = p$. Second, Bertsimas et al. (2021); Bertsimas and Cory-Wright (2022) reports that, for sparsity-constrained problems, the optimal support often does not change as we vary γ. Combining these observations suggests that, after optimizing τ with γ fixed, a good strategy for minimizing g with respect to γ is to fix the optimal support $\boldsymbol{z}^{(\mathcal{N}_j)}$ with respect to each fold i and invoke a root-finding method to find a γ which locally minimizes g.

Algorithm 1: Optimal sparsity parameter for confidence-adjusted k-fold error

Data: γ: ℓ_2^2 regularization parameter; $\epsilon > 0$: desired optimality tolerance; r: budget on number of MIOs; δ: confidence-adjustment parameter; M: upper bound on ℓ_2^2 loss

Result: Cardinality with best estimated confidence-adjusted k-fold error

for $\tau \in [p]$ **do**

  $\bar{v}_\tau \leftarrow \min_{\boldsymbol{\beta}\in\mathbb{R}^p, \boldsymbol{z}\in[0,1]^p} \|\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y}\|_2^2 + \frac{\gamma}{2}\sum_{i=1}^p \beta_i^2/z_i$ s.t. $\boldsymbol{e}^\top \boldsymbol{z} \leq \tau$

  for $j \in [k]$ **do**

    $v_{\tau,\mathcal{N}_j} \leftarrow \min_{\boldsymbol{\beta}\in\mathbb{R}^p, \boldsymbol{z}\in[0,1]^p} \|\boldsymbol{X}^{(\mathcal{N}_j)}\boldsymbol{\beta} - \boldsymbol{y}^{(\mathcal{N}_j)}\|_2^2 + \frac{\gamma}{2}\sum_{i=1}^p \beta_i^2/z_i$ s.t. $\boldsymbol{e}^\top \boldsymbol{z} \leq \tau$

    $\boldsymbol{\beta}_{\tau,\mathcal{N}_j} \in \arg\min_{\boldsymbol{\beta}\in\mathbb{R}^p, \boldsymbol{z}\in[0,1]^p} \|\boldsymbol{X}^{(\mathcal{N}_j)}\boldsymbol{\beta} - \boldsymbol{y}^{(\mathcal{N}_j)}\|_2^2 + \frac{\gamma}{2}\sum_{i=1}^p \beta_i^2/z_i$ s.t. $\boldsymbol{e}^\top \boldsymbol{z} \leq \tau$

    $h_{\mathcal{N}_j}(\tau) \leftarrow \sum_{i\in\mathcal{N}_j}(y_i - \boldsymbol{x_i}^\top \boldsymbol{\beta_{\tau,i}})^2$; `// Perspective sol. estimates k-fold for` $\mathcal{N}_j$

    $u_{\tau,\mathcal{N}_j} \leftarrow$ `round`$(\boldsymbol{\beta}_{\tau,\mathcal{N}_j})$; `// Any heuristic can be used`

    $\zeta^L_{\mathcal{N}_j}(\tau), \zeta^U_{\mathcal{N}_j}(\tau) \leftarrow$ `compute_bounds`$(\mathcal{N}_j, \boldsymbol{\beta}_{\tau,\mathcal{N}_j}, \bar{v}_\tau, v_{\tau,\mathcal{N}_j}, u_{\tau,\mathcal{N}_j})$

$LB \leftarrow \min_{\tau\in[p]} \sum_{j\in[k]} \zeta^L_{\mathcal{N}_j}(\tau) + \delta\sqrt{\frac{M^2+6Mk\underline{\mu_h}(\gamma,\tau)}{2k}}$; ; `//` $\underline{\mu}_h, \overline{\mu}_h$ `denotes lower/upper bound on` μ_h `computed analogously to Algorithm 2`

$UB \leftarrow \min_{\tau\in[p]} \sum_{j\in[k]} \zeta^U_{\mathcal{N}_j}(\tau) + \delta\sqrt{\frac{M^2+6Mk\overline{\mu_h}(\gamma,\tau)}{2k}}$; `// Bounds on the optimal` k`-fold`

$num\_mip \leftarrow 0$

repeat

  $\tau^* \leftarrow \arg\min_{\tau\in[p]} \sum_{i=1}^n \zeta^L_{\mathcal{N}_j}(\tau) + \delta\sqrt{\frac{M^2+6Mk\underline{\mu_h}(\gamma,\tau)}{2k}}$; `// Cardinality with best bound`

  $\mathcal{N}_j^* \leftarrow \arg\max_{j\in[k]}\{\zeta^U_{\mathcal{N}_j}(\tau) - \zeta^L_{\mathcal{N}_j}(\tau)\}$; `// Fold with largest` k`-fold uncertainty`

  $h_{\mathcal{N}_j^*}(\tau^*) \leftarrow \min_{\boldsymbol{\beta}\in\mathbb{R}^p, \boldsymbol{z}\in\{0,1\}^p} \|\boldsymbol{X}^{(\mathcal{N}_j^*)}\boldsymbol{\beta} - \boldsymbol{y}^{(\mathcal{N}_j^*)}\|_2^2 + \frac{\gamma}{2}\|\boldsymbol{\beta}\|_2^2$ s.t. $\boldsymbol{e}^\top \boldsymbol{z} \leq \tau^*$; `// Solve MIO`

  $\zeta^L_{\mathcal{N}_j^*}(\tau^*) \leftarrow h_{\mathcal{N}_j^*}(\tau^*),\ \zeta^U_{\mathcal{N}_j^*}(\tau^*) \leftarrow h_{\mathcal{N}_j^*}(\tau^*)$

  Update $\underline{\mu_h}(\tau), \overline{\mu_h}(\tau)$

  $LB \leftarrow \min_{\tau\in[p]} \sum_{j\in[k]} \zeta^L_i(\tau) + \delta\sqrt{\frac{M^2+6Mk\underline{\mu_h}(\gamma,\tau)}{2k}}$

  $UB \leftarrow \min_{\tau\in[p]} \sum_{j\in[k]} \zeta^U_{\mathcal{N}_j}(\tau) + \delta\sqrt{\frac{M^2+6Mk\overline{\mu_h}(\gamma,\tau)}{2k}}$

  $num\_mip \leftarrow num\_mip + 1$

until $(UB - LB)/UB \geq \epsilon$ *or* $num\_mip > r$;

return $\arg\min_{\tau\in[p]} \sum_{j\in[k]} h_{\mathcal{N}_j}(\tau) + \delta\sqrt{\frac{M^2+6Mk\mu_h(\gamma,\tau)}{2k}}$; `// Cardinality with best confidence-adjusted error`

Accordingly, we now use the fact that γ and $\boldsymbol{z}^{(\mathcal{N}_j)}$ fully determine $\boldsymbol{\beta}^{(\mathcal{N}_j)}$ to rewrite

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p} \quad \frac{\gamma}{2}\|\boldsymbol{\beta}\|_2^2 + \|\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y}\|_2^2 \text{ s.t. } \beta_i = 0 \text{ if } \hat{z}_i = 0,$$

$$\text{as} \qquad \boldsymbol{\beta}^\star = \left(\frac{\gamma}{2}\mathbb{I} + \boldsymbol{X}^\top \mathrm{Diag}(\hat{\boldsymbol{z}})\boldsymbol{X}\right)^{-1} \mathrm{Diag}(\hat{\boldsymbol{z}})\boldsymbol{X}^\top \boldsymbol{y}.$$

Algorithm 2: `compute_bounds`$(\mathcal{N}_j, \boldsymbol{\beta}, \bar{v}, v, u)$

Data: $\mathcal{N}_j$: fold left out; $\boldsymbol{\beta}$: optimal solution of perspective relaxation with $\mathcal{N}_j$ left out; $\bar{v}$: lower bound of obj val of MIO with all data; v: optimal obj value of perspective relaxation with $\mathcal{N}_j$ left out; u: upper bound of obj val of MIO with $\mathcal{N}_j$ left out

Result: Lower and upper bounds on the k-fold error attributable to datapoint i

$\underline{\xi} \leftarrow \boldsymbol{x_i}^\top \boldsymbol{\beta} - \sqrt{\boldsymbol{x_i}^\top \left(\boldsymbol{X}^{(\mathcal{N}_j)^\top} \boldsymbol{X}^{(\mathcal{N}_j)} + \frac{\gamma}{2}\mathbb{I}\right)^{-1} \boldsymbol{x_i}\,(u-v)}$

$\overline{\xi} \leftarrow \boldsymbol{x_i}^\top \boldsymbol{\beta} + \sqrt{\boldsymbol{x_i}^\top \left(\boldsymbol{X}^{(\mathcal{N}_j)^\top} \boldsymbol{X}^{(\mathcal{N}_j)} + \frac{\gamma}{2}\mathbb{I}\right)^{-1} \boldsymbol{x_i}\,(u-v)}$

$\zeta^L \leftarrow \bar{v} - u,\ \zeta^U \leftarrow \max\{(y_i - \underline{\xi})^2, (\overline{\xi} - y_i)^2\}$

if $\underline{\xi} > y_i$ **then**

  $\zeta^L \leftarrow \max\{\zeta^L, (\underline{\xi} - y_i)^2\}$

if $\overline{\xi} < y_i$ **then**

  $\zeta^L \leftarrow \max\{\zeta^L, (y_i - \overline{\xi})^2\}$

return $\left(\zeta^L, \zeta^U\right)$

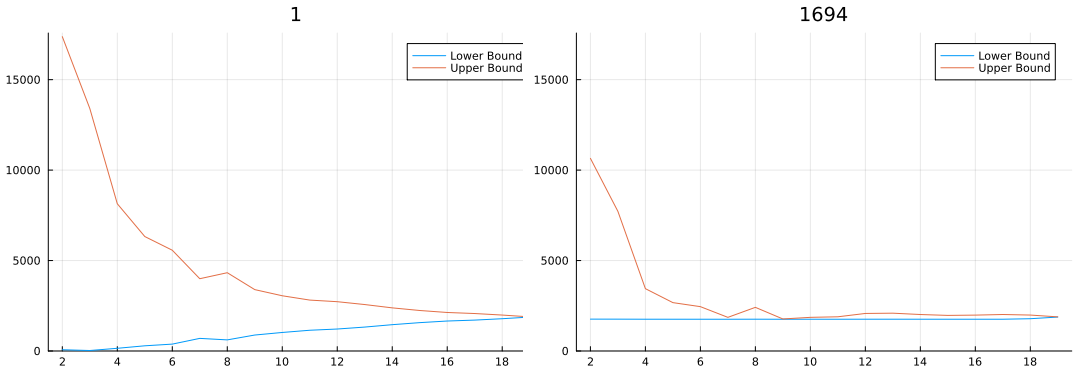

Figure 2: Comparison of initial bounds on LOOCV (k-fold with $k = n$) from Algorithm 2 (left) and bounds after running Algorithm 1 (right) for a synthetic sparse regression instance where $p = 20, n = 200, \tau_{\text{true}} = 10$, for varying τ. The black number in the top middle depicts the iteration number of the method.

Therefore, we fix each $\boldsymbol{z}^{(\mathcal{N}_j)}$ and substitute the resulting expressions for each $\boldsymbol{\beta}^{(\mathcal{N}_j)}$ into the k-fold error. This substitution yields the following univariate optimization problem, which can be solved via standard root-finding methods to approximately minimize the confidence-adjusted k-fold loss in the special case where $\delta = 0$:

$$\min_{\gamma>0} \sum_{j\in[k]} \sum_{i\in\mathcal{N}_j} \left(y_i - \boldsymbol{X}_i^\top \text{Diag}(\boldsymbol{z}^{(\mathcal{N}_j)}) \left(\frac{\gamma}{2}\mathbb{I} + \boldsymbol{X}^{(i)\top} \text{Diag}(\boldsymbol{z}^{(\mathcal{N}_j)}) \boldsymbol{X}^{(i)^\top}\right)^{-1} \text{Diag}(\boldsymbol{z}^{(\mathcal{N}_j)}) \boldsymbol{X}^{(i)^\top} \boldsymbol{y}^{(i)} \right)^2. \tag{28}$$

Moreover, if $\delta > 0$ and we are interested in minimizing the confidence-adjusted k-fold error, rather than the k-fold error itself, we assume that the index j at which the expression

$$\mu_h := \max_{j\in[k]} \frac{1}{n}\sum_{j=1}^{n} \left|(y_i - \boldsymbol{\beta}^{\star\top}\boldsymbol{x}_i)^2 - (y_i - \boldsymbol{\beta}^{(\mathcal{N}_j)}\boldsymbol{x}_i)^2\right|$$

attains its maximum[4], does not vary as we vary γ. Fixing j then allows us to derive a similar approximation for the hypothesis stability, namely:

$$\mu_h(\gamma,\tau) \approx \frac{1}{n}\sum_{i\in[n]} \left| \left(y_i - \boldsymbol{X}_i^\top \text{Diag}(\boldsymbol{z}^{(j)}) \left(\frac{\gamma}{2}\mathbb{I} + \boldsymbol{X}^{(j)\top}\text{Diag}(\boldsymbol{z}^{(j)})\boldsymbol{X}^{(j)^\top}\right)^{-1} \text{Diag}(\boldsymbol{z}^{(j)})\boldsymbol{X}^{(j)^\top}\boldsymbol{y}^{(j)}\right)^2 \right.$$
$$\left. - \left(y_i - \boldsymbol{X}_i^\top \text{Diag}(\boldsymbol{z}) \left(\frac{\gamma}{2}\mathbb{I} + \boldsymbol{X}^\top\text{Diag}(\boldsymbol{z})\boldsymbol{X}^\top\right)^{-1} \text{Diag}(\boldsymbol{z})\boldsymbol{X}^\top\boldsymbol{y}\right)^2 \right|,$$

where $\boldsymbol{z}$ denotes the optimal support when no data observations are omitted. With these expressions, it is straightforward to minimize the confidence-adjusted LOOCV error with respect to γ. Details on minimizing γ using `Julia` are provided in Appendix E.1.

5 Numerical Experiments

We now present numerical experiments testing our proposed methods. First, in Section 5.1, we study the computational savings of using Algorithm 1 over a complete grid search when optimizing the k-fold error as a function of the sparsity parameter τ. Then, in Sections 5.2 and 5.3, we use synthetic data to benchmark the statistical performance of the proposed methods (without and with confidence adjustment) against alternatives in the literature. Finally, in Section 5.4, we benchmark the proposed approach on real datasets. For conciseness, we describe the datasets used throughout the section in Appendix F.

Evaluation Metrics: We now remind the reader of evaluation metrics that we use throughout this section and are standard in subset selection (e.g., Bertsimas et al., 2020; Hastie et al., 2020).

Suppose that our data observations $(\boldsymbol{x}_i, y_i)$ are generated according to some stochastic process via $y_i = \boldsymbol{x}_i^\top\boldsymbol{\beta} + \epsilon_i$, where ϵ_i is zero-mean noise and $\boldsymbol{\beta}_{\text{true}}$ is some fixed but unknown ground truth regressor. Then, we assess the statistical performance of various methods in terms of their accuracy:

$$A(\boldsymbol{\beta}) := \frac{\|\boldsymbol{\beta}_{\text{true}} \circ \boldsymbol{\beta}\|_0}{\|\boldsymbol{\beta}_{\text{true}}\|_0},$$

i.e., the proportion of true features that are selected, and their false discovery rate

$$FDR(\boldsymbol{\beta}) := \frac{|\{j : \beta_j \neq 0, \beta_{\text{true},j} = 0\}|}{\|\boldsymbol{\beta}_{\text{true}}\|_0},$$

i.e., the proportion of selected features not included in the true support.

It is worth noting that these metrics rely on knowledge of the ground truth $\boldsymbol{\beta}_{\text{true}}$, and thus cannot be applied when the ground truth is unknown. Accordingly, we also compare

4. We pick the first index j which attains this maximum in the rate case of ties.

performance in terms of the Mean Square Error, namely

$$MSE(\boldsymbol{\beta}) := \frac{1}{n}\sum_{i=1}^{n}(y_i - \boldsymbol{x}_i^\top \boldsymbol{\beta})^2,$$

which can either be taken over a training set to obtain the in-sample Mean Square Error, or over a test set to obtain the out-of-sample Mean Square Error.

The methods developed here require computing M, an upper bound on the loss. Accordingly, we approximate M by $\max_{i\in[n]} y_i^2$, the largest y_i^2 over the training set, throughout this section.

5.1 Exact K-fold Optimization

We first assess whether Algorithm 1 significantly reduces the number of MIOs that need to be solved to minimize the kCV error with respect to τ, compared to grid search. For simplicity, we consider the special case where $\delta = 0$ and set either $k = n$ or $k = 10$, corresponding to leave-one-out and ten-fold cross-validation problems (27) respectively.

We compare the performance of two approaches. First, a standard grid search approach (`Grid`), where we solve the inner MIO in (27) for all combinations of cardinality $\tau \in [p]$ and all folds of the data $j \in [k]$, and select the hyperparameter combination which minimizes the objective. To ensure the quality of the resulting solution, we solve all MIOs to optimality (without any time limit). Second, we consider using Algorithm 1 with parameter $r = \infty$ (thus solving MIOs to optimality until the desired optimality gap ϵ for problem (27) is proven). We test regularization parameter $\gamma \in \{0.01, 0.02, 0.05, 0.10, 0.20, 0.50, 1.00\}$ in Algorithm 1, and solve all MIOs via their perspective reformulations, namely

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p, \boldsymbol{z}\in\{0,1\}^p} \|\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y}\|_2^2 + \frac{\gamma}{2}\sum_{j=1}^{p}\frac{\beta_j^2}{z_j} \text{ s.t. } \sum_{j=1}^{p} z_j \leq \tau,$$

using Mosek 10.0. Since the approach `Grid` involves solving $\mathcal{O}(kp)$ MIOs (without a time limit), we are limited to testing these approaches on small datasets, and accordingly use the Diabetes, Housing, Servo, and AutoMPG datasets for this experiment. Moreover, we remark that the specific solution times and the number of nodes expanded by each method are not crucial, as those could vary substantially if relaxations other than the perspective are used, different solvers or solution approaches are used, or if advanced techniques are implemented (but both methods would be affected in the same way). Thus, we focus our analysis on relative performance.

We now summarize our experimental results and defer the details to Tables 8 and 9 of Appendix G. Figures 3 and 4 summarize the percentage reduction of the number of MIOs and the number of branch-and-bound nodes achieved by Algorithm 1 over `Grid`:

$$\text{Reduction in MIOs} = \frac{\#\text{ MIO}_{\texttt{Grid}} - \#\text{ MIO}_{\text{Alg.1}}}{\#\text{ MIO}_{\texttt{Grid}}}, \text{ Reduction in nodes} = \frac{\#\text{ nodes}_{\texttt{Grid}} - \#\text{ nodes}_{\text{Alg.1}}}{\#\text{ nodes}_{\texttt{Grid}}},$$

where $\#\text{ MIO}_{\text{Y}}$ and $\#\text{ nodes}_{\text{Y}}$ indicate the number of MIOs or branch-and-bound nodes used by method Y.

We observe that across these four datasets, Algorithm 1 reduces the number of MIO that need to be solved by an average of 70% for leave-one-out cross-validation and by 52% for 10-fold cross-validation. The overall number of branch-and-bound nodes is reduced by

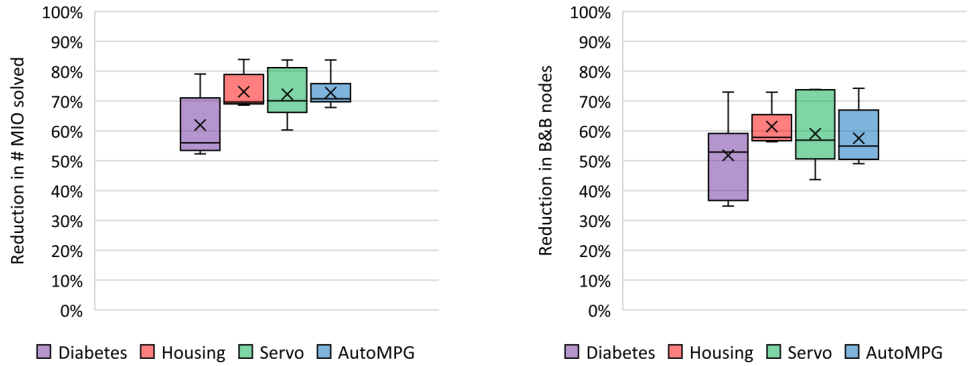

Figure 3: Reduction in the number of MIO solved (left) and the total number of branch-and-bound nodes (right) when using Algorithm 1 for leave-one-out cross-validation, when compared with `Grid` (i.e., independently solving $\mathcal{O}(pn)$ MIOs) in four real datasets. The distributions shown in the figure correspond to solving the same instance with different values of γ. All MIOs are solved to optimality, without imposing any time limits.

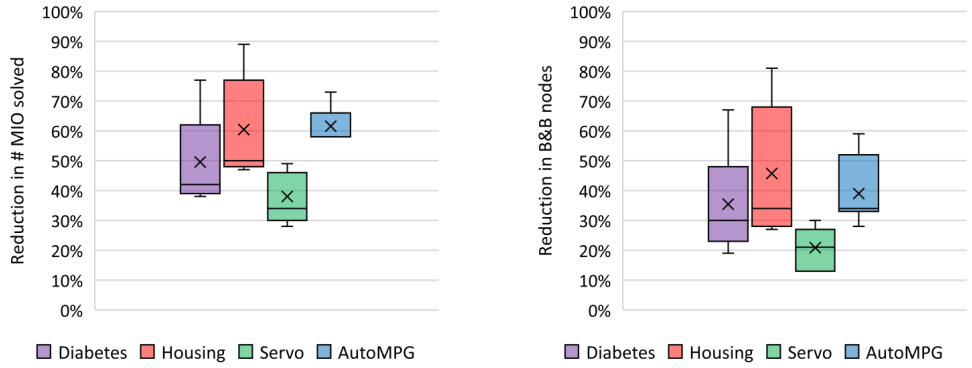

Figure 4: Reduction in the number of MIO solved (left) and the total number of branch-and-bound nodes (right) when using Algorithm 1 for 10-fold cross-validation, when compared with `Grid` (i.e., independently solving $\mathcal{O}(pk)$ MIOs) in four real datasets. The distributions shown in the figure correspond to solving the same instance with different values of γ. All MIOs are solved to optimality, without imposing any time limits.

an average of 57% for leave-one-out cross-validation and 35% for 10-fold cross-validation (the reduction in computational times is similar to the reduction of nodes). These results indicate that the relaxations of the bilevel optimization (27) derived in §3 are sufficiently strong to avoid solving most of the MIOs that traditional methods such as `Grid` would solve, without sacrificing solution quality. The proposed methods are especially beneficial for settings where k is large, that is, in the settings that would require more MIOs and are more computationally expensive using standard approaches. The resulting approach still requires solving several MIOs, but, as we show throughout the rest of this section, approximating each MIO with its perspective relaxation yields similarly high-quality statistical estimators at a fraction of the computational cost.

5.2 Sparse Regression on Synthetic Data

We now benchmark our coordinate descent approach on synthetic sparse regression problems where the ground truth is known to be sparse, but the number of non-zeros is unknown. For simplicity, we set $\delta = 0, k = n$ in this subsection (we study the same problem setting with $\delta > 0$ in the next subsection). The goal is to highlight the dangers of cross-validating without confidence adjustment.

We consider two problem settings. First, a smaller-scale setting ($p = 50, \tau_{\text{true}} = 10$) that allows us to benchmark two implementations of our coordinate descent approach:

1. An exact implementation of our approach, where we optimize τ according to Algorithm 1, using `Gurobi` version 9.5.1 to solve all MIOs with a time limit of 120s, and warm-start `Gurobi` with a greedily rounded solution to each MIO's perspective relaxation (computed using `Mosek` version 10.0) before running `Gurobi`. We denote this approach by "EX" (stands for EXact).

2. An approximate implementation of our approach, where we optimize τ by greedily rounding the perspective relaxation of each MIO we encounter (computed using `Mosek` version 10.0), and using these greedily rounded solutions, rather than optimal solutions to MIOs, to optimize the leave-one-out error with respect to τ. We denote this approach by "GD" (stands for GreeDy).

For both approaches, we optimize γ as described in Section 4.2, and set $\tau_{\min} = 4, \tau_{\max} = 20$.

We also consider a large-scale setting ($p = 1000, \tau_{\text{true}} = 20$) where grid search is not sufficient to identify a globally optimal solution with respect to the kCV loss. In this setting, the subproblems are too numerically expensive to solve exactly, and accordingly, we optimize τ using an approach very similar to "GD", except to optimize τ we solve each subproblem using the saddle-point method of Bertsimas et al. (2020) with default parameters, rather than greedily rounding the perspective relaxations of MIOs. This approach generates solutions that are almost identical to those generated by GD, but is more scalable. We term this implementation of our coordinate descent approach "SP" (stands for Saddle Point), and set $\tau_{\min} = 10, \tau_{\max} = 40$ when optimizing τ in this experiment.

We compare against the following state-of-the-art methods, using in-built functions to approximately minimizes the cross-validation loss with respect to the method's hyperparameters via grid search, and subsequently fit a regression model on the entire dataset with these cross-validated parameters (see also Bertsimas et al. (2020) for a detailed discussion of these approaches):

- The `ElasticNet` method in the ubiquitous `GLMNet` package, with grid search on their parameter $\alpha \in \{0, 0.1, 0.2, \ldots, 1\}$, using 100-fold cross-validation as in (Bertsimas et al., 2020).
- The Minimax Concave Penalty (MCP) and Smoothly Clipped Absolute Deviation Penalty (SCAD) as implemented in the `R` package `ncvreg`, using the `cv.ncvreg` function with 100 folds and default parameters to (approximately) minimize the cross-validation error.
- The `L0Learn.cvfit` method implemented in the `L0Learn` `R` package (cf. Hazimeh and Mazumder, 2020), with n folds, a grid of 10 different values of γ and default parameters otherwise.

We remark that, in preliminary experiments, we found that using the `cv.GLMNet` and `cv.ncvreg` to minimize the kCV error when $k = n$ was orders-of-magnitude more expensive than other approaches. Accordingly, we settled with minimizing the 100-fold cross-validation error as a surrogate.

Experimental Methodology: We measure each method's ability to recover the ground truth (true positive rate) while avoiding detecting irrelevant features (false discovery rate). We consider two sets of synthetic data, following Bertsimas et al. (2020): a small (medium noise, high correlation) dataset: $\tau_{\text{true}} = 10$, $p = 50$, $\rho = 0.7$ and $\nu = 1$; and a large (medium noise, low correlation) dataset: $\tau_{\text{true}} = 20$, $p = 1000$, $\rho = 0.2$ and $\nu = 1$. Figure 5 reports results in small instances with varying number of samples $n \in \{10, 20, \ldots, 200\}$, and Figure 6 reports results for large datasets with $n \in \{100, 200, \ldots, 3000\}$. We report the average leave-one-out error and average MSE on a different (out-of-sample) set of 10000 observations of $\boldsymbol{X}, \boldsymbol{y}$ drawn from the same distribution. Further, we report the average cross-validated support size and runtime for both experiments in §B.

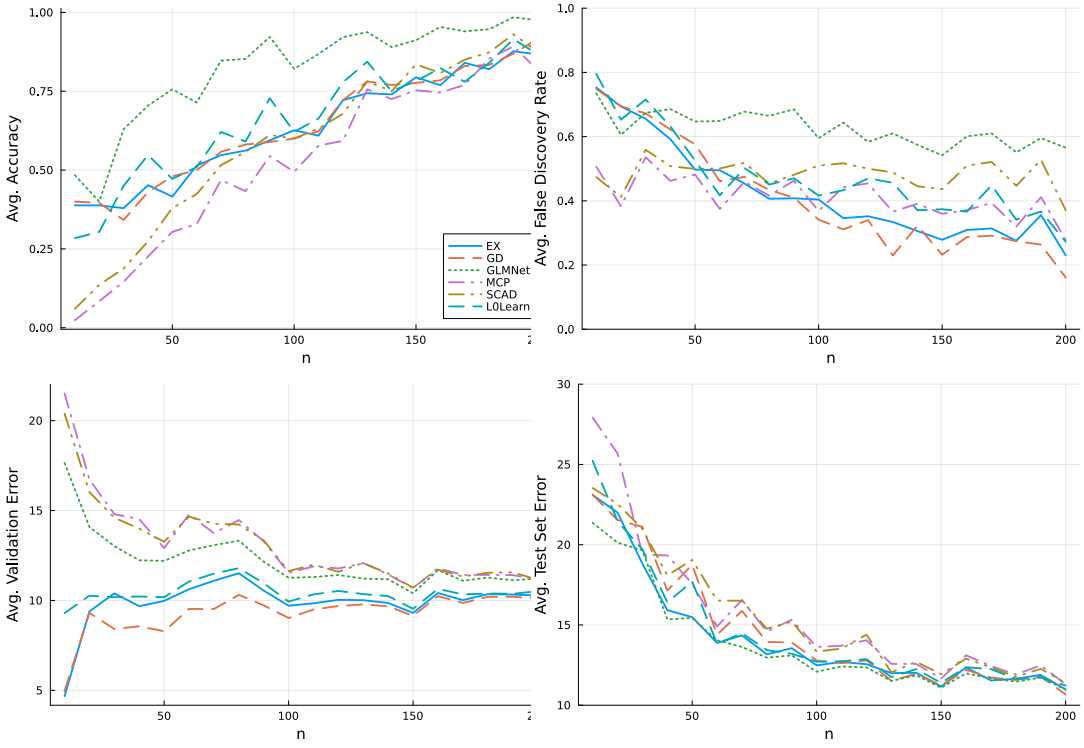

Figure 5: Average accuracy (top left), false discovery rate (top right), normalized validation error (bottom left), normalized MSE on test set (bottom right), as n increases with $p = 50, \tau_{\text{true}} = 10$, for coordinate descent with τ optimized using Algorithm 1 (EX), coordinate descent with τ optimized by greedily rounding perspective relaxations (GD), GLMNet, MCP, SCAD, and L0Learn. We average results over 25 datasets.

Accuracy and Performance of Methods: We observe that our coordinate descent schemes and `L0Learn` consistently provide the best performance in large-sample settings,

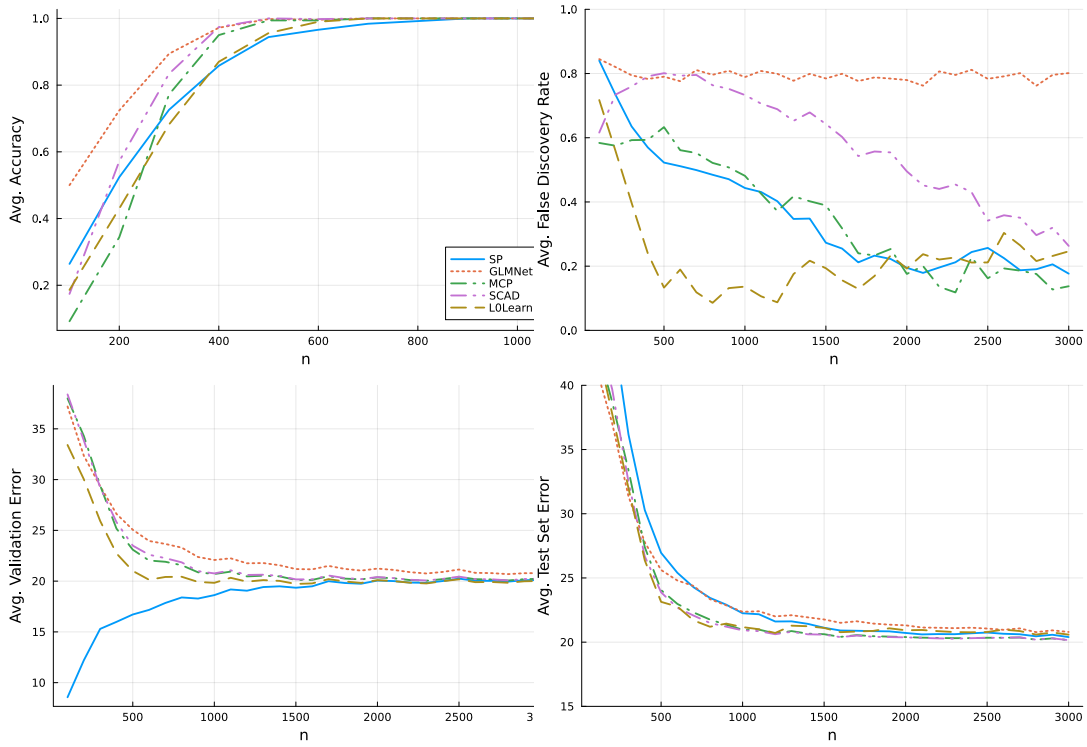

Figure 6: Average accuracy (top left), false discovery rate (top right), normalized validation error (bottom left), and normalized MSE on test set (bottom right) as n increases with $p = 1000, \tau_{\text{true}} = 20$, for coordinate descent with a saddle-point method to solve each training problem when optimizing τ (SP), GLMNet, MCP, SCAD, and L0Learn. We average results over 25 datasets.

by returning sparser solutions with a lower false discovery rate and a similar out-of-sample MSE to all other methods when $n > p$. On the other hand, `GLMNet` appears to perform best when $n \ll p$, where it consistently returns solutions with a lower out-of-sample MSE and less out-of-sample disappointment than any other method. Thus, the best-performing methods vary depending on the number of samples, as recently suggested by a number of authors (Hastie et al., 2020; Bertsimas and Van Parys, 2020). In particular, coordinate descent performs worse than `GLMNet` when $p = 1000$ and $n < 2000$ despite having a much lower kCV error, precisely because of the dangers of minimizing the kCV error without confidence adjustment, as discussed in the introduction.

Out-of-Sample Disappointment: We observe that all methods suffer from the optimizer's curse (cf. Smith and Winkler, 2006; Van Parys et al., 2021), with the average MSE on the test set being consistently larger than the average leave-one-out error on the validation set, especially when n is smaller. However, out-of-sample disappointment is most pronounced for our coordinate descent schemes and `L0Learn` (bottom two panels of Figures 5–6), which consistently exhibit the lowest LOOCV error at all sample sizes but the highest test set error in small-data settings. As reflected in the introduction, this phenomenon

can be explained by the fact that *optimizing* the kCV error without confidence adjustment generates highly optimistic estimates of the test set error. This reinforces the need for confidence-adjusted alternatives to the kCV error, particularly in small-sample settings, and motivates the confidence-adjusted variants of our coordinate scheme we study next.

5.3 Confidence-Adjusted Sparse Regression on Synthetic Data

We now benchmark our coordinate descent schemes with confidence adjustment. In particular, we revisit the problem settings studied in the previous section and consider setting $\delta \in \{1, \sqrt{10}\}$ in our GD and SP implementations of coordinate descent. Specifically, we solve each subproblem using greedy rounding when $p = 50, \tau_{\text{true}} = 10$ and via a saddle-point method when $p = 1000, \tau_{\text{true}} = 20$. For ease of comparison, we also report the performance of these inexact methods without any confidence adjustment, as reported in the previous section (here denoted by $\delta = 0$).

Experimental Methodology: We implement the same methodology as in the previous section, and vary $n \in \{10, 20, \ldots, 100\}$ for small instances (Figure 7) and $n \in \{100, 200, \ldots, 1500\}$ for large instances (Figure 8). We report the average accuracy, average false positive rate, average cross-validated support size and average MSE on a different (out-of-sample) set of 10000 observations of $\boldsymbol{X}, \boldsymbol{y}$ drawn from the same distribution.

Impact of Confidence Adjustment on Test Set Performance: We observe that for both problem settings, accounting for out-of-sample disappointment via confidence adjustment significantly improves the test set performance of sparse regression methods with hyperparameters selected via cross-validation. On average, we observe a 3.04% (resp. 2.55%) average MSE improvement for $\delta = 1$ (resp. $\delta = \sqrt{10}$) when $p = 50$, and a 5.48% (resp. 2.75%) average MSE improvement for $\delta = 1$ (resp. $\delta = \sqrt{10}$) when $p = 1000$, with the most significant out-of-sample performance gains occurring when n is smallest. This performance improvement highlights the benefits of accounting for out-of-sample disappointment by selecting more stable sparse regression models, and suggests that accounting for model stability when cross-validating is a viable alternative to selecting hyperparameters that minimize the leave-one-out error that often yields better test set performance.

Additionally, when $p = 1000$, accounting for stability via confidence adjustment yields sparser regressors with a significantly lower false discovery rate (5% vs. 40% when $n = 1000$), which suggests that models selected via confidence adjustment may sometimes be less likely to select irrelevant features. However, we caution that when $p = 50$, the models selected via a confidence-adjusted procedure exhibit a similar false discovery rate to models selected by minimizing the LOOCV error, so this effect does not appear to occur uniformly.

The best value of δ to select for confidence adjustment appears to depend on the amount of data available, with larger values like $\delta = \sqrt{10}$ performing better when n is small, but being overly conservative when n is large, and smaller values like $\delta = 1$ providing a less significant benefit when n is very small but performing more consistently when n is large.

5.4 Benchmarking on Real-World Datasets

We now benchmark our proposed cross-validation approaches against the methods studied in the previous section on a suite of real-world datasets previously studied in the literature.

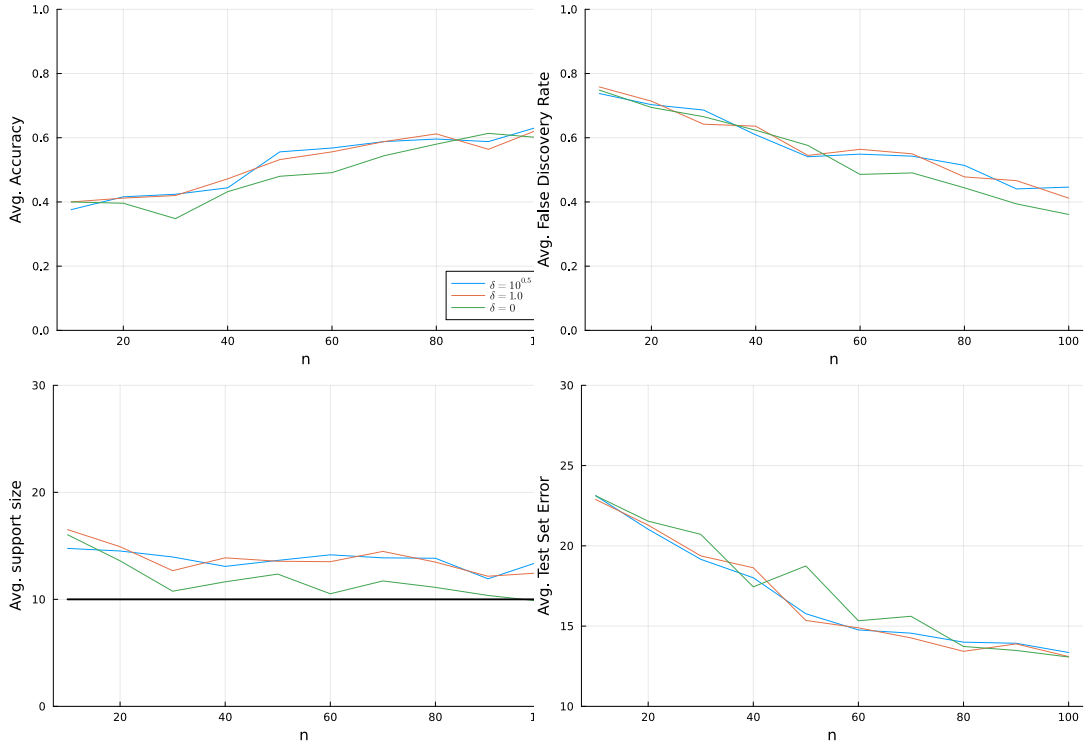

Figure 7: Average accuracy (top left), false discovery rate (top right), cross-validated support (bottom left), and normalized MSE on test set (bottom right) as n increases with $p = 50, \tau_{\text{true}} = 10$, with $\delta \in \{1, \sqrt{10}\}$, and without any confidence adjustment ($\delta = 0$). We average results over 25 datasets.

For each dataset, we repeat the following procedure five times: we randomly split the data into 80% training/validation data and 20% testing data, and report the average sparsity of the cross-validated model and the average test set MSE.

Table 1 depicts the dimensionality of each dataset, the average k-fold cross-validation error ("CV"), the average test set error ("MSE"), and the sparsity attained by our cyclic coordinate descent without any hypothesis stability adjustment ("$\delta = 0$), our cyclic coordinate descent with $\delta = 1.0$ ("$\delta = 1.0$"), MCP, and GLMNet on each dataset. We use leave-one-out (n-fold) cross-validation for all methods except where stated otherwise, and repeat all experiments in this section using five-fold cross-validation for all methods in Section C; the sparsity and out-of-sample MSEs for n-fold and 5-fold cross-validation are almost identical.

We used the same number of folds for MCP and GLMNet as in the previous section, i.e., a cap of 100 folds, for the sake of numerical tractability. Note that for our coordinate descent methods, after identifying the final hyperparameter combination (γ, τ) we solve a MISOCP with a time limit of 3600s to fit a final model to the training dataset. Moreover, for our cyclic coordinate descent schemes, we set the largest permissible value of τ such that $\tau \log \tau \leq n$ via the `Lambert.jl Julia` package, because Gamarnik and Zadik (2022, Theorem 2.5) demonstrated that, up to constant terms and under certain assumptions on the data generation process, on the order of $\tau \log \tau$ observations are necessary to recover a sparse

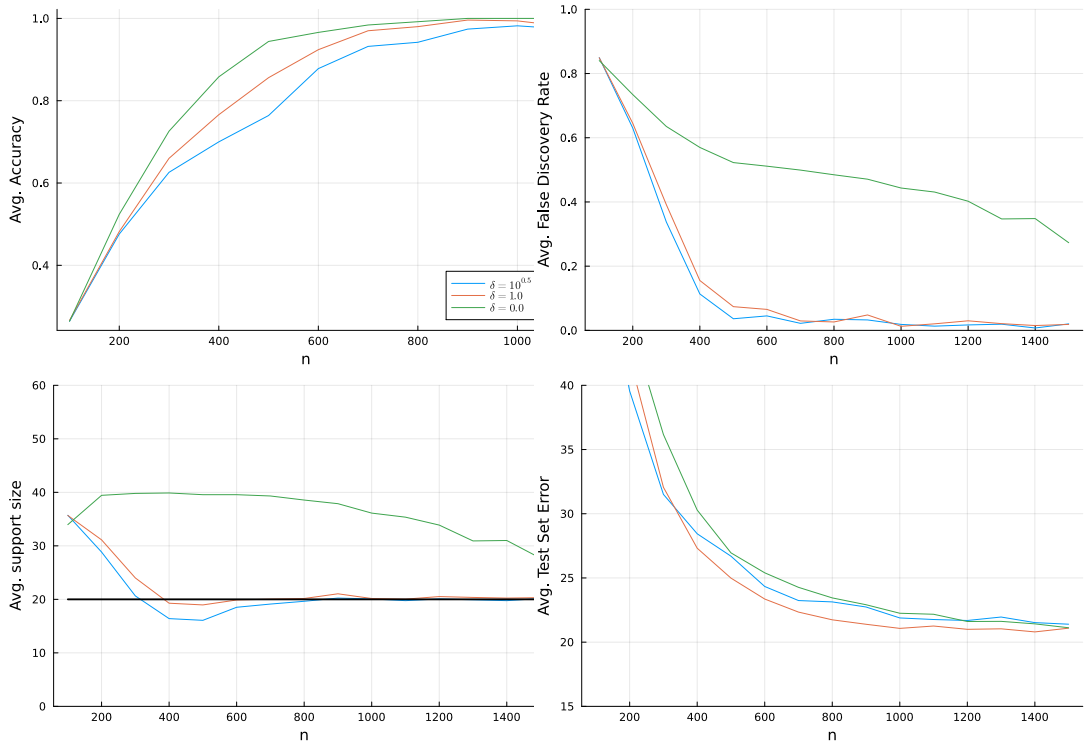

Figure 8: Average accuracy (top left), false discovery rate (top right), cross-validated support (bottom left), and normalized MSE on test set (bottom right) as n increases with $p = 1000, \tau_{\text{true}} = 20$, for coordinate descent with confidence adjustment and $\delta \in \{1.0, \sqrt{10}\}$, and without confidence adjustment ($\delta = 0$). We average results over 25 datasets.

model with binary coefficients. In preliminary experiments, we relaxed this requirement to $\tau \leq p$ and found that this did not change the optimal value of τ.

We observe that $\delta = 0.0$ returns solutions with a significantly lower cross-validation error than all other methods. Specifically, our kCV error is 34.2% lower than GLMNet's and 48.5% lower than MCP's on average. Moreover, our methods obtain significantly sparser solutions than GLMNet ($\delta = 0.0$ is 37.9% sparser than GLMNet, $\delta = 1.0$ is 49.6% sparser than GLMNet, on average).

However, this does not result in a lower test set error on most datasets ($\delta = 0.0$ is 6.35% higher, $\delta = 1.0$ is 30.49% higher, MCP is 6.82% higher, on average), because optimizing the cross-validation error increases the cyclic coordinate descent scheme's vulnerability to out-of-sample disappointment, due to the optimizer's curse (Smith and Winkler, 2006). In the case of confidence-adjusted coordinate descent, this can be explained by the fact that $\delta = 1.0$ causes the method to be excessively risk-averse, and a smaller value of δ may actually be more appropriate. In particular, calibrating $\delta \in \mathbb{R}_{++}$ to match the cross-validation error of GLMNet or MCP may be a better strategy for obtaining high-quality solutions that do not disappoint significantly out-of-sample.

Motivated by these observations, we now rerun our cyclic coordinate descent scheme with $\delta \in \{10^{-3.5}, 10^{-3}, 10^{-2.5}, \ldots, 10^{-0.5}\}$. Tables 2-3 depicts the average validation and

test set error from these variants of our cyclic coordinate descent scheme, and verifies that, in circumstances where $\delta = 1$ led to an excessively conservative validation error, a smaller value of δ performs better on the test set. We also report the sparsity and MSE for the values of δ such that the confidence-adjusted LOOCV error most closely matches the cross-validation error reported by GLMNet.

We observe that, after normalizing all metrics against the metric obtained by GLMNet on the same dataset to weigh all datasets equally, the average relative MSE from cyclic coordinate descent with confidence adjustment (calibrated) is 2.62% higher than GLMNet, and the average regressor is 33.6% sparser than GLMNet. This compares favorably with our previous results with $\delta = 1, \delta = 0$ and MCP, because it corresponds to an MSE improvement of 4% out-of-sample without compromising the sparsity of our regressors. In particular, these experiments suggest that accounting for confidence adjustment with a small multiplier ($\delta = 10^{-3.5}$) provides more stable models that reliably perform better out-of-sample. Upon repeating this experiment with five folds for all methods, our findings are very similar (deferred to Section C). Namely, we find regressors around 30% sparser than GLMNet, albeit with a (5%) worse out-of-sample MSE.

The better MSE performance and worse sparsity performance of GLMNet can be explained by the fact that we use $\ell_0 - \ell_2^2$ regularization, while GLMNet employs $\ell_1 - \ell_2^2$ regularization, which is known to perform better in low signal-to-noise ration regimes (like the datasets studied in this section), and worse in high signal-to-noise ratio regimes (like the synthetic datasets studied previously) (Hastie et al., 2020). It is worth noting that the techniques proposed here could also be applied to an $\ell_1 - \ell_2^2$ training problem, although exploring this further would be beyond the scope of this work.

6 Conclusion

We proposed a new approach for selecting hyperparameters in ridge-regularized sparse regression: minimizing a generalization bound on the test set performance. We leverage perspective relaxations and branch-and-bound techniques from mixed-integer optimization. Using these techniques, we minimize the generalization bound by performing alternating minimization on a sparsity hyperparameter and a regularization hyperparameter. Our approach obtains locally optimal hyperparameter combinations with $p = 1000$ features in a few hours, and thus is a viable hyperparameter selection technique in offline settings where sparse and stable regressors are desirable. We observe that, in underdetermined settings, our approach improves the out-of-sample MSE by 2%–7% compared to approximately minimizing the leave-one-out error, which suggests that model stability and performance on a validation metric should both be accounted for when selecting hyperparameters.

Future work could explore the benefits of minimizing a weighted sum of output stability and a validation metric, rather than a validation metric alone, when hyperparameter tuning in other problem settings with limited data. It would also be interesting to investigate whether tighter convex relaxations of sparse regression than the perspective relaxation could be used to develop tighter bounds on the prediction spread and the hypothesis stability, and whether perturbation analysis of convex relaxations could facilitate more efficient hyperparameter optimization in contexts other than sparse regression.

Acknowledgments:

Andrés Gómez is supported in part by grant 2152777 from the National Science Foundation and grant FA9550-22-1-0369 from the Air Force Office of Scientific Research. Ryan Cory-Wright acknowledges the MIT-IBM Research Lab for hosting him while part of this work was conducted.

References

Sylvain Arlot and Alain Celisse. A survey of cross-validation procedures for model selection. *Statistics surveys*, 4:40–79, 2010.

Alper Atamtürk and Andrés Gómez. Safe screening rules for l0-regression from perspective relaxations. In *ICML*, pages 421–430, 2020.

Alper Atamtürk and Andres Gómez. Rank-one convexification for sparse regression. *Journal of Machine Learning Research, conditionally accepted*, 2025.

Gah-Yi Ban and Cynthia Rudin. The big data newsvendor: Practical insights from machine learning. *Operations Research*, 67(1):90–108, 2019.

Gah-Yi Ban, Noureddine El Karoui, and Andrew EB Lim. Machine learning and portfolio optimization. *Management Science*, 64(3):1136–1154, 2018.

Yasmine Beck and Martin Schmidt. A gentle and incomplete introduction to bilevel optimization. 2021.

Omar Ben-Ayed and Charles E Blair. Computational difficulties of bilevel linear programming. *Operations Research*, 38(3):556–560, 1990.

Kristin P Bennett, Jing Hu, Xiaoyun Ji, Gautam Kunapuli, and Jong-Shi Pang. Model selection via bilevel optimization. In *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, pages 1922–1929. IEEE, 2006.

James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(2), 2012.

Dimitri Bertsekas. *Nonlinear Programming*. Athena Scientific, 3rd edition, 2016.

Dimitris Bertsimas and Martin S Copenhaver. Characterization of the equivalence of robustification and regularization in linear and matrix regression. *European Journal of Operational Research*, 270(3):931–942, 2018.

Dimitris Bertsimas and Ryan Cory-Wright. A scalable algorithm for sparse portfolio selection. *INFORMS Journal on Computing*, 34(3):1489–1511, 2022.

Dimitris Bertsimas and Ioana Popescu. Optimal inequalities in probability theory: A convex optimization approach. *SIAM Journal on Optimization*, 15(3):780–804, 2005.

Dimitris Bertsimas and Bart Van Parys. Sparse high-dimensional regression: Exact scalable algorithms and phase transitions. *The Annals of Statistics*, 48(1):300–323, 2020.

Dimitris Bertsimas, Jean Pauphilet, and Bart Van Parys. Sparse regression: Scalable algorithms and empirical performance. *Statistical Science*, 35(4):555–578, 2020.

Dimitris Bertsimas, Ryan Cory-Wright, and Jean Pauphilet. A unified approach to mixed-integer optimization problems with logical constraints. *SIAM Journal on Optimization*, 31(3):2340–2367, 2021.

Lea Bottmer, Christophe Croux, and Ines Wilms. Sparse regression for large data sets with outliers. *European Journal of Operational Research*, 297(2):782–794, 2022.

Olivier Bousquet and André Elisseeff. Stability and generalization. *The Journal of Machine Learning Research*, 2:499–526, 2002.

Stephen Boyd, Laurent El Ghaoui, Eric Feron, and Venkataramanan Balakrishnan. *Linear matrix inequalities in system and control theory*. SIAM, 1994.

Leo Breiman. Heuristics of instability and stabilization in model selection. *The annals of statistics*, 24(6): 2350–2383, 1996.

Peter Bühlmann and Sara Van De Geer. *Statistics for high-dimensional data: methods, theory and applications*. Springer Science & Business Media, 2011.

Prabir Burman. A comparative study of ordinary cross-validation, v-fold cross-validation and the repeated learning-testing methods. *Biometrika*, 76(3):503–514, 1989.

Sebastián Ceria and João Soares. Convex programming for disjunctive convex optimization. *Math. Prog.*, 86:595–614, 1999.

Anthony-Alexander Christidis, Laks Lakshmanan, Ezequiel Smucler, and Ruben Zamar. Split regularized regression. *Technometrics*, 62(3):330–338, 2020.

P. Cortez, A. Cerdeira, F. Almeida, and J. Matos, T.and Reis. Wine Quality. UCI Machine Learning Repository, 2009. DOI: https://doi.org/10.24432/C56S3T.

Victor DeMiguel and Francisco J Nogales. Portfolio selection with robust estimation. *Operations Research*, 57(3):560–577, 2009.

Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3*, pages 265–284. Springer, 2006.

Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. *The Annals of Statistics*, 32(2):407–499, 2004.

Matthias Ehrgott. *Multicriteria optimization*, volume 491. Springer Science & Business Media, 2005.

David Gamarnik and Ilias Zadik. Sparse high-dimensional linear regression. estimating squared error and a phase transition. *The Annals of Statistics*, 50(2):880–903, 2022.

Arthur M Geoffrion. Generalized Benders decomposition. *Journal of Optimization Theory and Applications*, 10(4):237–260, 1972.

Andrés Gómez and Oleg A Prokopyev. A mixed-integer fractional optimization approach to best subset selection. *INFORMS Journal on Computing*, 33(2):551–565, 2021.

Bram L Gorissen, İhsan Yanıkoğlu, and Dick Den Hertog. A practical guide to robust optimization. *Omega*, 53:124–137, 2015.

Geoffrey Grimmett and David Stirzaker. *Probability and random processes*. Oxford university press, 2020.

Peter Groves, Basel Kayyali, David Knott, and Steve Van Kuiken. The big data revolution in healthcare: Accelerating value and innovation. 2016.

Vishal Gupta and Nathan Kallus. Data pooling in stochastic optimization. *Management Science*, 68(3): 1595–1615, 2022.

Vishal Gupta and Paat Rusmevichientong. Small-data, large-scale linear optimization with uncertain objectives. *Management Science*, 67(1):220–241, 2021.

Vishal Gupta, Michael Huang, and Paat Rusmevichientong. Debiasing in-sample policy performance for small-data, large-scale optimization. *Operations Research*, 72(2):848–870, 2024.

Pierre Hansen, Brigitte Jaumard, and Gilles Savard. New branch-and-bound rules for linear bilevel programming. *SIAM Journal on scientific and Statistical Computing*, 13(5):1194–1217, 1992.

Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

Trevor Hastie, Robert Tibshirani, and Ryan Tibshirani. Best subset, forward stepwise or Lasso? analysis and recommendations based on extensive comparisons. *Statistical Science*, 35(4):579–592, 2020.

Elad Hazan and Tomer Koren. A linear-time algorithm for trust region problems. *Mathematical Programming*, 158(1-2):363–381, 2016.

Hussein Hazimeh and Rahul Mazumder. Fast best subset selection: Coordinate descent and local combinatorial optimization algorithms. *Operations Research*, 68(5):1517–1537, 2020.

Hussein Hazimeh, Rahul Mazumder, and Ali Saab. Sparse regression at scale: Branch-and-bound rooted in first-order optimization. *Mathematical Programming*, 196(1):347–388, 2022.

Harold Hotelling. The generalization of student's ratio. *The Annals of Mathematical Statistics*, 2(3):360–378, 1931.

Fredrik D Johansson, Uri Shalit, Nathan Kallus, and David Sontag. Generalization bounds and representation learning for estimation of potential outcomes and causal effects. *Journal of Machine Learning Research*, 23(166):1–50, 2022.

Christopher Jung, Katrina Ligett, Seth Neel, Aaron Roth, Saeed Sharifi-Malvajerdi, and Moshe Shenfeld. A new analysis of differential privacy's generalization guarantees. *arXiv preprint arXiv:1909.03577*, 2019.

Michael Kearns and Dana Ron. Algorithmic stability and sanity-check bounds for leave-one-out cross-validation. In *Proceedings of the tenth annual conference on Computational learning theory*, pages 152–162, 1997.

Alan J King and Roger JB Wets. Epi-consistency of convex stochastic programs. *Stochastics and Stochastic Reports*, 34(1-2):83–92, 1991.

Hugo Larochelle, Dumitru Erhan, Aaron Courville, James Bergstra, and Yoshua Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *Proc. 24th Int. Conf. Mach. Learn.*, pages 473–480, 2007.

Sifan Liu and Edgar Dobriban. Ridge regression: Structure, cross-validation, and sketching. *Proc. Int. Conf. Learn. Repres.*, 2020.

Rahul Mazumder, Peter Radchenko, and Antoine Dedieu. Subset selection with shrinkage: Sparse linear modeling when the snr is low. *Operations Research*, 71(1):129–147, 2023.

Andrew McAfee, Erik Brynjolfsson, Thomas H Davenport, DJ Patil, and Dominic Barton. Big data: The management revolution. *Harvard Business Review*, 90(10):60–68, 2012.

Andrew McAfee, Daniel Rock, and Erik Brynjolfsson. How to capitalize on generative AI. *Harvard Business Review*, 101(6):42–48, 2023.

Balas Kausik Natarajan. Sparse approximate solutions to linear systems. *SIAM journal on computing*, 24 (2):227–234, 1995.

Takayuki Okuno, Akiko Takeda, Akihiro Kawana, and Motokazu Watanabe. On lp-hyperparameter learning via bilevel nonsmooth optimization. *J. Mach. Learn. Res.*, 22:245–1, 2021.

Mert Pilanci, Martin J Wainwright, and Laurent El Ghaoui. Sparse learning via boolean relaxations. *Mathematical Programming*, 151(1):63–87, 2015.

R. Quinlan. Auto MPG. UCI Machine Learning Repository, 1993. DOI: https://doi.org/10.24432/C5859H.

R Bharat Rao, Glenn Fung, and Romer Rosales. On the dangers of cross-validation. an experimental evaluation. In *Proceedings of the 2008 SIAM international conference on data mining*, pages 588–596. SIAM, 2008.

Galen Reeves, Jiaming Xu, and Ilias Zadik. The all-or-nothing phenomenon in sparse linear regression. In *Conference on Learning Theory*, pages 2652–2663. PMLR, 2019.

Peter Rousseeuw, Christophe Croux, Valentin Todorov, Andreas Ruckstuhl, Matias Salibian-Barrera, Tobias Verbeke, Manuel Koller, and Martin Maechler. Robustbase: basic robust statistics. *R package version 0.4-5*, 2009.

Aaron Sidford. Optimization algorithms. *Lecture notes for Introduction to Optimization Theory and Optimization Algorithms*, 2024. Stanford University.

Ankur Sinha, Tanmay Khandait, and Raja Mohanty. A gradient-based bilevel optimization approach for tuning hyperparameters in machine learning. *arXiv preprint arXiv:2007.11022*, 2020.

James E Smith and Robert L Winkler. The optimizer's curse: Skepticism and postdecision surprise in decision analysis. *Management Science*, 52(3):311–322, 2006.

Will Stephenson, Zachary Frangella, Madeleine Udell, and Tamara Broderick. Can we globally optimize cross-validation loss? quasiconvexity in ridge regression. *Advances in Neural Information Processing Systems*, 34, 2021.

Mervyn Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the royal statistical society: Series B (Methodological)*, 36(2):111–133, 1974.

Mervyn Stone. Cross-validation: A review. *Statistics: A Journal of Theoretical and Applied Statistics*, 9(1): 127–139, 1978.

Yuichi Takano and Ryuhei Miyashiro. Best subset selection via cross-validation criterion. *Top*, 28(2):475–488, 2020.

Karl Ulrich. Servo. UCI Machine Learning Repository, 1993. DOI: https://doi.org/10.24432/C5Q30F.

Bart PG Van Parys, Peyman Mohajerin Esfahani, and Daniel Kuhn. From data to decisions: Distributionally robust optimization is optimal. *Management Science*, 67(6):3387–3402, 2021.

Vladimir N Vapnik. An overview of statistical learning theory. *IEEE transactions on neural networks*, 10 (5):988–999, 1999.

Weijun Xie and Xinwei Deng. Scalable algorithms for the sparse ridge regression. *SIAM Journal on Optimization*, 30(4):3359–3386, 2020.

Huan Xu, Constantine Caramanis, and Shie Mannor. Robust regression and lasso. *Advances in neural information processing systems*, 21, 2008.

Chenglong Ye, Yi Yang, and Yuhong Yang. Sparsity oriented importance learning for high-dimensional linear regression. *Journal of the American Statistical Association*, 113(524):1797–1812, 2018.

Jane J Ye, Xiaoming Yuan, Shangzhi Zeng, and Jin Zhang. Difference of convex algorithms for bilevel programs with applications in hyperparameter selection. *Mathematical Programming*, pages 1–34, 2022.

Dataset	n	p	$\delta = 0.0$			$\delta = 1.0$			MCP			GLMNet		
			τ	CV	MSE	τ	CV	MSE	τ	CV	MSE	τ	CV	MSE
Wine	6497	11	10	0.434	0.543	2	0.809	0.709	10.6	0.434	0.543	11	0.434	0.543
Auto-MPG	392	25	16.4	6.731	8.952	11.4	45.44	13.10	15	7.066	8.983	18.8	7.072	8.839
Hitters	263	19	7.4	0.059	0.080	4.6	0.169	0.095	12.4	0.062	0.080	13	0.062	0.080
Prostate	97	8	4.4	0.411	0.590	2.6	1.825	0.632	6.8	0.439	0.566	6.8	0.435	0.574
Servo	167	19	9.2	0.537	0.812	3.6	4.094	1.095	11.6	0.565	0.729	15.4	0.568	0.717
Housing2	506	91	56.4	10.32	13.99	65.6	79.33	16.37	31.2	12.93	15.54	86.4	9.677	11.52
Toxicity	38	9	3.2	0.031	0.057	2.8	0.249	0.064	3	0.033	0.061	4.2	0.035	0.061
SteamUse	25	8	3	0.346	0.471	2.6	2.948	0.769	2.4	0.441	0.506	4.2	0.458	0.507
Alcohol2	44	21	3	0.186	0.254	5.6	13.79	0.304	2	0.185	0.232	4.4	0.212	0.260
TopGear	242	373	18	0.032	0.053	11.6	0.482	0.072	8.2	0.040	0.069	24.6	0.036	0.053
BarDet	120	200	19.6	0.005	0.011	18	0.107	0.011	6	0.006	0.010	61	0.006	0.009
Vessel	180	486	21	0.014	0.031	28.6	2.272	0.025	2.6	0.028	0.033	53.2	0.015	0.022
Riboflavin	71	4088	18	0.055	0.304	14.6	1.316	0.443	9.2	0.277	0.232	82.8	0.164	0.279

Table 1: Average performance of methods across a suite of real-world datasets where the ground truth is unknown (and may not be sparse), sorted by how overdetermined the dataset is (n/p), and separated into the underdetermined and overdetermined cases. In overdetermined settings, cyclic coordinate descent (without confidence) returns sparser solutions than MCP or GLMNet and maintains a comparable average MSE. In underdetermined settings, cyclic coordinate descent with confidence returns significantly sparse solutions than GLMNet with a comparable MSE, and more accurate (although denser) solutions than MCP.

Dataset	n	p	$\delta = 10^{-0.5}$			$\delta = 10^{-1}$			$\delta = 10^{-1.5}$			$\delta = 10^{-2}$		
			τ	CV	MSE	τ	CV	MSE	τ	CV	MSE	τ	CV	MSE
Wine	6497	11	2	0.642	0.565	3.6	0.587	0.682	10	0.560	0.543	10	0.548	0.565
Auto-MPG	392	25	18.6	20.43	9.206	18	12.23	8.867	17.8	9.638	8.854	17.8	8.857	8.881
Hitters	263	19	7.2	0.108	0.085	7.2	0.085	0.081	7.2	0.078	0.080	7.2	0.075	0.080
Prostate	97	8	2.8	0.941	0.600	3	0.653	0.598	3.6	0.560	0.563	4.4	0.529	0.590
Servo	167	19	10	1.817	0.761	10.2	1.049	0.771	10.2	0.798	0.775	9.8	0.715	0.729
Housing2	506	91	78.4	32.91	11.65	77.2	18.01	11.31	62.2	15.887	14.218	57.8	13.795	16.021
Toxicity	38	9	3	0.1	0.061	3	0.057	0.060	3.2	0.045	0.057	3.2	0.040	0.057
SteamUse	25	8	4.6	1.268	0.597	3.4	0.729	0.589	3.4	0.536	0.653	3.4	0.484	0.662
Alcohol2	44	21	4.6	4.521	0.289	4.6	1.594	0.296	2	0.674	0.213	2	0.360	0.218
TopGear	242	373	10.4	0.211	0.073	28.4	0.115	0.062	40.4	0.066	0.057	26.6	0.050	0.053
Bardet	120	200	23	0.033	0.011	22	0.015	0.011	20.4	0.010	0.009	19	0.007	0.013
Vessel	180	486	32.2	0.731	0.026	25.8	0.317	0.028	23.8	0.114	0.028	16.8	0.048	0.030
Riboflavin	71	4088	18.2	0.469	0.272	18.2	0.194	0.259	18.6	0.105	0.303	18.6	0.076	0.379

Table 2: Performance of methods across real-world datasets where the ground truth is unknown (continued).

Dataset	n	p	$\delta = 10^{-2.5}$			$\delta = 10^{-3}$			$\delta = 10^{-3.5}$			δ calibrated		
			τ	CV	MSE	τ	CV	MSE	τ	CV	MSE	δ	τ	MSE
Wine	6497	11	10	0.544	0.543	10	0.543	0.543	10	0.542	0.565	$10^{-3.5}$	10	0.565
Auto-MPG	392	25	17.2	8.561	8.880	16.6	8.473	8.859	16.8	8.441	8.893	$10^{-3.5}$	16.8	8.893
Hitters	263	19	5.8	0.075	0.080	8.8	0.074	0.080	8.6	0.074	0.080	$10^{-3.5}$	8.6	0.080
Prostate	97	8	4.4	0.518	0.590	4.4	0.515	0.590	4.4	0.514	0.590	$10^{-3.5}$	4.4	0.590
Servo	167	19	10	0.690	0.725	9.4	0.678	0.816	9.8	0.672	0.725	$10^{-3.5}$	9.8	0.725
Housing2	506	91	60	12.496	13.310	64.4	11.029	11.337	55.4	12.547	13.154	$10^{-3.0}$	64.4	11.337
Toxicity	38	9	3.2	0.039	0.057	3.2	0.038	0.057	3.2	0.038	0.057	$10^{-3.5}$	3.2	0.057
SteamUse	25	8	3.4	0.466	0.652	3.4	0.460	0.652	3.4	0.458	0.662	$10^{-3.5}$	3.4	0.662
Alcohol2	44	21	2	0.256	0.230	2	0.227	0.230	2	0.217	0.230	$10^{-3.5}$	2	0.230
TopGear	242	373	24.6	0.043	0.053	29.2	0.041	0.053	26.2	0.040	0.053	$10^{-3.5}$	26.2	0.053
Bardet	120	200	14.4	0.007	0.011	19.6	0.007	0.010	21.4	0.006	0.010	$10^{-3.5}$	21.4	0.010
Vessel	180	486	16.4	0.030	0.027	15	0.023	0.030	16	0.019	0.026	$10^{-3.5}$	16	0.026
Riboflavin	71	4088	18.8	0.071	0.288	17.2	0.072	0.282	18.4	0.065	0.316	$10^{-1.0}$	18.2	0.259

Table 3: Performance of methods across real-world datasets where the ground truth is unknown (continued).

Appendix A. Globally Minimizing Five-Fold Cross-Validation Error

We now revisit the setting considered in Section 1 and Figure 1, using five-fold rather than leave-one-out cross-validation (Figure 9). Our conclusions remain consistent with Figure 1.

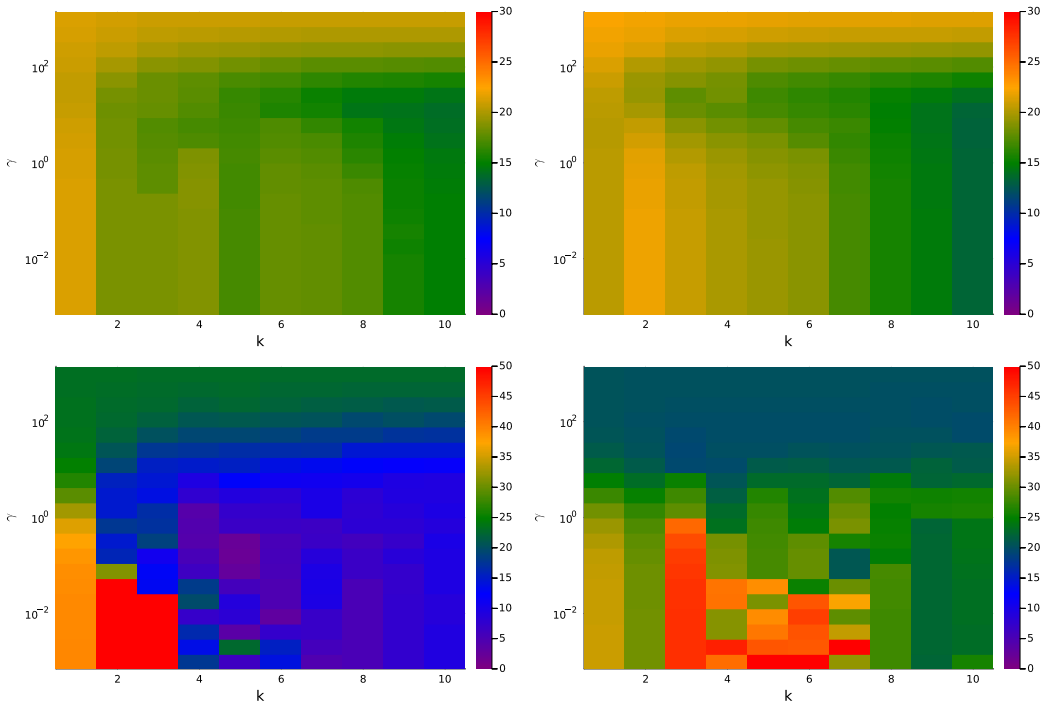

Figure 9: Five-fold (left) and test (right) error for varying τ and γ, for the overdetermined setting (top, $n = 50, p = 10$) and an underdetermined setting (bottom, $n = 10, p = 50$) considered in Figure 1. In the overdetermined setting, the five-fold error is a good estimate of the test error for most values of parameters (γ, τ). In contrast, in the underdetermined setting, the five-fold error is a poor approximation of the test error, and the estimator that minimizes the five-fold error ($\gamma = 6.15$, $\tau = 5$) significantly disappoint out-of-sample.

Appendix B. Supplementary Experimental Results for Section 5.2

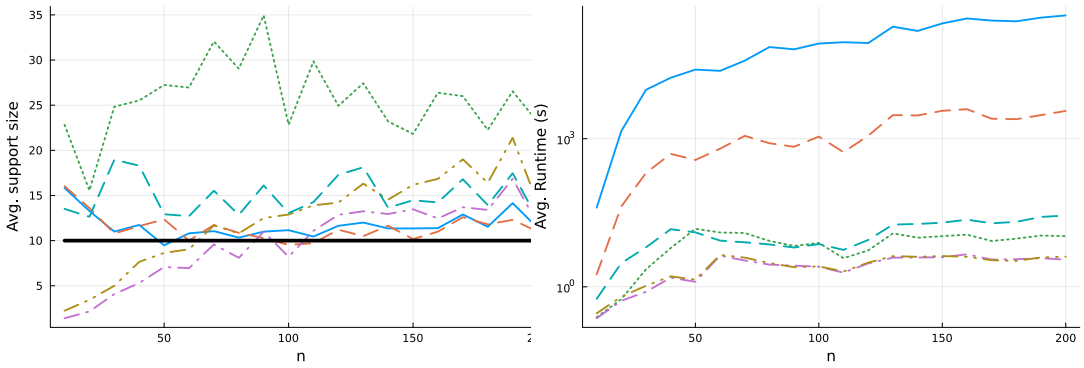

Figure 10: Average cross-validated support (left) and runtime (right) as n increases with $p = 50, \tau_{\text{true}} = 10$, for coordinate descent with τ optimized using Algorithm 1 (EX), coordinate descent with τ optimized by greedily rounding perspective relaxations (GD), GLMNet, MCP, SCAD, and L0Learn. We average results over 25 datasets.

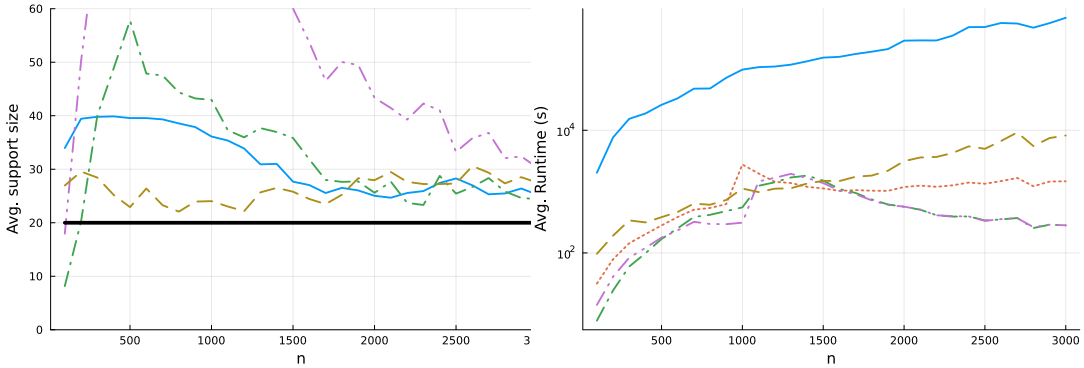

Figure 11: Average cross-validated support (left) and runtime (right) as n increases with $p = 1000, \tau_{\text{true}} = 20$, for coordinate descent with a saddle-point method to solve each training problem when optimizing τ (SP), GLMNet, MCP, SCAD, and L0Learn. We average results over 25 datasets. Note that the average cross-validated support size of GLMNet exceeds 100 for all values of p considered, and GLMNet does not appear in the left plot.

Appendix C. Supplementary Experimental Results for Section 5.4

We observe that similarly to Section 5.4, without any confidence adjustment, we obtain a five-fold cross-validation error 9.4% lower than GLMNet and 37% lower than MCP, respectively. Similarly, this does not translate to a lower test set error, due to the optimizer's curse. In particular, the average MSE without (with calibrated) confidence adjustment is 5.7% (6.2%) higher than GLMNet, and the average regressor is 32.8% (29.8%) sparser than the regressor generated by GLMNet. The MSE without (with calibrated) confidence adjustment is also 3% (2.5%) lower than MCP.

Interestingly, the impact of the confidence adjustment procedure on the MSE appears to be negligible in the five-fold case (the MSEs with and without confidence adjustment are statistically indistinguishable), likely because the five-fold error is a more biased and

lower variance estimator of the test set error than the leave-one-out error in underdetermined settings (Hastie et al., 2009). Indeed, the optimized five-fold error when $\delta = 0$ for Riboflavin, the most underdetermined dataset, is nearly three times the optimized leave-one-out error when $\delta = 0$. This suggests that the benefits of confidence adjustment may be most pronounced in the leave-one-out case where there is more variance, particularly in underdetermined settings with many folds. Indeed, over the four underdetermined datasets, the average MSE of models trained via a calibrated leave-one-out approach is 20% lower than the average MSE of models trained via a calibrated five-fold approach.

Dataset	n	p	$\delta = 0.0$			$\delta = 1.0$			MCP			GLMNet		
			τ	CV	MSE	τ	CV	MSE	τ	CV	MSE	τ	CV	MSE
Wine	6497	11	9.8	0.435	0.543	1.2	0.886	0.750	10.8	0.435	0.542	11	0.433	0.543
Auto-MPG	392	25	17.4	6.840	8.871	9	47.855	13.095	16	7.323	8.993	20.8	6.880	8.979
Hitters	263	19	10.2	0.060	0.077	14.4	0.178	0.081	13	0.065	0.081	16	0.061	0.079
Prostate	97	8	4.4	0.418	0.567	1.2	1.842	0.659	5.8	0.457	0.574	6.4	0.420	0.561
Servo	167	19	12	0.585	0.725	6.2	4.457	0.810	13.8	0.601	0.722	16.4	0.548	0.715
Housing2	506	91	66	9.415	11.356	60.4	84.76	17.216	36.8	12.823	14.895	66	10.142	13.158
Toxicity	38	9	3.2	0.029	0.060	3.6	0.234	0.061	2.6	0.039	0.057	5	0.030	0.061
Steam	25	8	2.2	0.321	0.479	4	2.629	0.814	2.2	0.466	0.559	4.6	0.401	0.495
Alcohol2	44	21	2.6	0.183	0.256	5.6	11.17	0.267	2	0.182	0.232	3.8	0.189	0.255
TopGear	242	373	26.2	0.030	0.061	17.8	0.444	0.062	7.4	0.044	0.073	29.8	0.035	0.056
Bardet	120	200	21.8	0.005	0.011	20	0.083	0.010	6	0.007	0.010	30.2	0.006	0.009
Vessel	180	486	23.2	0.011	0.031	24.4	1.798	0.030	2.8	0.026	0.033	49.6	0.014	0.023
Riboflavin	71	4088	9.6	0.120	0.364	16	1.331	0.408	8	0.280	0.352	105.6	0.170	0.285

Table 4: Average performance of five-fold version of methods across a suite of real-world datasets where the ground truth is unknown (and may not be sparse), sorted by how overdetermined the dataset is (n/p), and separated into the underdetermined and overdetermined cases. In overdetermined settings, cyclic coordinate descent (without confidence) returns sparser solutions than MCP or GLMNet and maintains a comparable average MSE. In underdetermined settings, cyclic coordinate descent with confidence returns significantly sparse solutions than GLMNet with a comparable MSE, and more accurate (although denser) solutions than MCP.

Dataset	n	p	$\delta = 10^{-0.5}$			$\delta = 10^{-1}$			$\delta = 10^{-1.5}$			$\delta = 10^{-2}$		
			τ	CV	MSE	τ	CV	MSE	τ	CV	MSE	τ	CV	MSE
Wine	6497	11	1.8	0.603	0.724	3.4	0.5	0.687	9.6	0.46	0.573	9.6	0.443	0.573
Auto-MPG	392	25	13.4	20.471	11.228	17.4	11.228	9.083	17.8	8.231	8.926	17.8	7.259	8.926
Hitters	263	19	15.6	0.098	0.072	14.8	0.072	0.079	14.2	0.064	0.079	12	0.061	0.079
Prostate	97	8	2.4	0.878	0.572	4.6	0.572	0.615	4.8	0.469	0.613	4.8	0.434	0.613
Servo	167	19	11.2	1.833	0.995	15	0.995	0.719	13.6	0.723	0.725	13.6	0.634	0.725
Housing2	506	91	72.4	33.742	17.312	65.2	17.312	11.9	66.4	11.901	11.705	62.8	10.164	11.772
Toxicity	38	9	3.4	0.094	0.051	3.4	0.051	0.061	3.4	0.037	0.061	3.4	0.032	0.061
Steam	25	8	3.4	1.021	0.554	3.4	0.554	0.591	2.2	0.386	0.487	2.2	0.341	0.487
Alcohol2	44	21	2.6	3.637	1.271	2.4	1.271	0.236	2.6	0.524	0.241	2.6	0.285	0.232
TopGear	242	373	26.2	0.161	0.068	33.4	0.068	0.053	35.6	0.036	0.060	39.6	0.026	0.062
Bardet	120	200	20.6	0.030	0.014	21.2	0.014	0.010	23	0.008	0.010	20	0.006	0.011
Vessel	180	486	27.6	0.575	0.191	23.2	0.191	0.030	20.6	0.070	0.030	24.2	0.029	0.031
Riboflavin	71	4088	9.4	0.538	0.253	14	0.253	0.341	12.4	0.172	0.368	12.4	0.144	0.368

Table 5: Performance of five-fold version of methods across real-world datasets where the ground truth is unknown (continued).

Dataset	n	p	$\delta = 10^{-2.5}$			$\delta = 10^{-3}$			$\delta = 10^{-3.5}$			δ calibrated		
			τ	CV	MSE	τ	CV	MSE	τ	CV	MSE	δ	τ	MSE
Wine	6497	11	9.6	0.438	0.573	9.8	0.436	0.568	9.8	0.436	0.568	$10^{-3.5}$	9.8	0.568
Auto-MPG	392	25	17.4	6.961	8.910	17.4	6.854	8.939	17.4	6.822	8.939	$10^{-2.5}$	17.4	8.910
Hitters	263	19	12	0.061	0.079	12	0.060	0.079	11.4	0.061	0.080	$10^{-2.0}$	12	0.079
Prostate	97	8	3.8	0.418	0.607	3.8	0.414	0.607	3.8	0.413	0.607	$10^{-2.5}$	3.8	0.607
Servo	167	19	13	0.606	0.724	13	0.606	0.724	13	0.597	0.724	$10^{-3.5}$	13	0.724
Housing2	506	91	64.4	9.670	11.821	64.4	9.496	11.821	65.8	9.414	11.687	$10^{-2.0}$	62.8	11.772
Toxicity	38	9	3.4	0.031	0.061	3.4	0.030	0.061	3.4	0.030	0.061	$10^{-3.5}$	3.4	0.061
Steam	25	8	2.2	0.327	0.487	2.2	0.323	0.487	2.2	0.321	0.487	$10^{-1.5}$	2.2	0.487
Alcohol2	44	21	2.6	0.211	0.232	2.6	0.187	0.232	2.6	0.179	0.232	$10^{-3.0}$	2.6	0.232
TopGear	242	373	43.6	0.021	0.070	41.2	0.021	0.064	41.2	0.021	0.069	$10^{-1.5}$	35.6	0.060
Bardet	120	200	20	0.006	0.011	20	0.006	0.011	20	0.006	0.011	$10^{-2.0}$	20	0.011
Vessel	180	486	24.6	0.017	0.029	24.6	0.013	0.030	24.6	0.012	0.030	$10^{-3.0}$	24.6	0.030
Riboflavin	71	4088	11.6	0.134	0.383	11.6	0.131	0.383	13.4	0.125	0.370	$10^{-1.5}$	12.4	0.368

Table 6: Performance of five-fold version of methods across real-world datasets where the ground truth is unknown (continued).

Appendix D. Omitted Proofs

D.1 Proof of Theorem 1

Proof The result follows analogously to (Bousquet and Elisseeff, 2002, Theorem 11); the main novelty in this proof compared to (Bousquet and Elisseeff, 2002, Theorem 11) is the use of a more general notion of output stability. In particular, Bousquet and Elisseeff (2002)'s definition is sufficient to derive the result for leave-one-out, but not for k-fold).

Let $\mathcal{R} := \frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}}(y_i - \boldsymbol{x}_i^\top \boldsymbol{\beta}^\star)^2$ denote the average test set error on an unseen observation, and $\mathcal{R}_{CV} := \frac{1}{n}\sum_{j\in[k]} h_j(\gamma,\tau)$ denote the average k-fold cross-validation error. Further, let $\mathbb{E}[\ell(A_{\mathcal{S}}, z)]$ denote the expected generalization error of a regressor trained on a training set $\mathcal{S}$ and evaluated on an example $z = (\boldsymbol{x}_i, y_i)$ drawn from the same distribution but not included in the test set. Let z' denote an independent draw to z, and $\mathcal{S}^{(\mathcal{N}_j)}$ denote a training set with the jth fold of the data omitted.

Then, analogously to (Bousquet and Elisseeff, 2002, Lemma 9), letting $i \neq j$, we have

$$\begin{aligned}
\mathbb{E}[(\mathcal{R} - \mathcal{R}_{CV})^2] &\leq \mathbb{E}_{\mathcal{S},z,z'}[\ell(A_{\mathcal{S}}, z)\ell(A_{\mathcal{S}}, z')] - 2\mathbb{E}_{\mathcal{S},z}[\ell(A_{\mathcal{S}}, z)\ell(A_{\mathcal{S}^{(\mathcal{N}_i)}}, z_i)] \qquad (29)\\
&\quad + \frac{n - n/k}{n}\mathbb{E}_{\mathcal{S}}[\ell(A_{\mathcal{S}^{(\mathcal{N}_i)}}, z_i)\ell(A_{\mathcal{S}^{(\mathcal{N}_j)}}, z_j)] + \frac{M}{k}\mathbb{E}_{\mathcal{S}}[\ell(A_{\mathcal{S}^{(\mathcal{N}_i)}}, z_i)]\\
&= \frac{1}{k}\mathbb{E}_{\mathcal{S}}[\ell(A_{\mathcal{S}^{(\mathcal{N}_i)}}, z_i)(M - \ell(A_{\mathcal{S}^{(\mathcal{N}_j)}}, z_j)] \qquad (30)\\
&\quad + \mathbb{E}_{\mathcal{S},z,z'}[\ell(A_{\mathcal{S}}, z)\ell(A_{\mathcal{S}}, z') - \ell(A_{\mathcal{S}}, z)\ell(A_{\mathcal{S}^{(\mathcal{N}_i)}}, z_i)]\\
&\quad + \mathbb{E}_{\mathcal{S},z,z'}[\ell(A_{\mathcal{S}^{(\mathcal{N}_i)}}, z_i)\ell(A_{\mathcal{S}^{(\mathcal{N}_j)}}, z_j) - \ell(A_{\mathcal{S}}, z)\ell(A_{\mathcal{S}^{(\mathcal{N}_i)}}, z_i)]\\
&= I_1 + I_2 + I_3,
\end{aligned}$$

where we let I_1, I_2, I_3 stand for the terms in the first, second, and third lines.

Further, it follows directly from Schwarz's inequality (see also Bousquet and Elisseeff, 2002, pp. 522) that $I_1 \leq \frac{M^2}{2k}$ and it follows analogously to (Bousquet and Elisseeff, 2002, pp. 522) that $I_2 + I_3 \leq 3M\mu_h$. Therefore, we have that

$$\mathbb{E}[(\mathcal{R} - \mathcal{R}_{cv})^2] \leq \frac{M^2}{2k} + 3M\mu_h.$$

Finally, the result follows from Chebyshev's inequality. ■

Appendix E. Implementation Details

To solve each MIO in Algorithm 1, we invoke a Generalized Benders Decomposition scheme (Geoffrion, 1972), which was specialized to sparse regression problems by Bertsimas and Van Parys (2020). For any fixed γ, τ, the method proceeds by minimizing a piecewise linear approximation of

$$f(\boldsymbol{z}, \gamma) := \min_{\boldsymbol{\beta}\in\mathbb{R}^p:\ \|\boldsymbol{\beta}\|_0\leq\tau} \frac{\gamma}{2}\sum_{j\in[p]}\frac{\beta_j^2}{z_j} + \|\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{y}\|_2^2, \qquad (31)$$

until it either converges to an optimal solution or encounters a time limit.

We now discuss two enhancements that improve this method's performance in practice.

Warm-Starts: as noted by Bertsimas et al. (2021), a greedily rounded solution to the Boolean relaxation constitutes an excellent warm-start for a Generalized Benders Decomposition scheme. Therefore, when computing the lower and upper bounds on $h_{\mathcal{N}_j}(\gamma, \tau)$ for each τ by solving a perspective relaxation, we save the greedily rounded solution to the relaxation in memory, and provide the relevant rounding as a high-quality warm-start before solving the corresponding MIO.

Screening Rules: as observed by Atamtürk and Gómez (2020), if we have an upper bound on the optimal value of $f(\boldsymbol{z}, \gamma)$, say $\bar{f}$, an optimal solution to the Boolean relaxation of minimizing (31) over $\boldsymbol{z} \in [0,1]^p$, say $(\boldsymbol{\beta}, \boldsymbol{z})$, and a lower bound on the optimal value of $h(\boldsymbol{z}, \gamma)$ from the Boolean relaxation, say $\underline{f}$ then, letting $\beta_{[\tau]}$ be the τth largest value of β in absolute magnitude, we have the following screening rules:

- If $\beta_i^2 \leq \beta_{[\tau+1]}^2$ and $\underline{f} - \frac{1}{2\gamma}(\beta_i^2 - \beta_{[\tau]}^2) > \bar{f}$ then $z_i = 0$.
- If $\beta_i^2 \geq \beta_{[\tau]}^2$ and $\underline{f} + \frac{1}{2\gamma}(\beta_i^2 - \beta_{[\tau+1]}^2) > \bar{f}$ then $z_i = 1$.

Accordingly, to reduce the dimensionality of our problems, we solve a perspective relaxation for each fold of the data with $\tau = \tau_{\max}$ as a preprocessing step, and screen out the features where $z_i = 0$ at $\tau = \tau_{\max}$ (for this fold of the data) before running Benders decomposition.

E.1 Implementation Details for Section 4.2

In our numerical experiments, we find local minimizers of our approximation of g by invoking the `ForwardDiff` function in `Julia` to automatically differentiate our approximation of g, and subsequently identify local minima via the `Order0` method in the `Roots.jl` package, which is designed to be a robust root-finding method. To avoid convergence to a low-quality local minimum, we run the search algorithm initialized at the previous iterate γ_{t-1} and seven points log-uniformly distributed in $[10^{-3}, 10^{1}]$, and set γ_t to be the local minima with the smallest estimated error. Moreover, to ensure numerical robustness, we require that γ_t remains within the bounds $[10^{-3}, 10^{1}]$ and project γ_t onto this interval if it exceeds these bounds (this almost never occurs in practice, because the data is preprocessed to be standardized). This approach tends to be very efficient in practice, particularly when the optimal support does not vary significantly as we vary γ.

Appendix F. Datasets

We now describe the datasets we use to test the methods proposed in this paper, and competing alternatives in the literature. We use both synthetically generated data and real data in our experiments. This is because synthetic data allows us to control the ground truth and measure the accuracy of our methods in statistical settings, while real data allows us to measure the performance of our methods on datasets that arise in practice, and ensure that any gains with respect to out-of-sample MSE are not due to the data generation process.

F.1 Synthetic datasets

We follow the experimental setup in Bertsimas et al. (2020). Given a fixed number of features p, number of datapoints n, true sparsity $1 \leq \tau_{\text{true}} \leq p$, autocorrelation parameter $0 \leq \rho \leq 1$ and signal to noise parameter ν:

1. The rows of the model matrix are generated iid from a p-dimensional multivariate Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$, where $\Sigma_{ij} = \rho^{|i-j|}$ for all $i, j \in [p]$.

2. A "ground-truth" vector $\boldsymbol{\beta}_{\text{true}}$ is sampled with exactly τ_{true} non-zero coefficients. The position of the non-zero entries is randomly chosen from a uniform distribution, and the value of the non-zero entries is either 1 or -1 with equal probability.

3. The response vector is generated as $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta}_{\text{true}} + \boldsymbol{\varepsilon}$, where each ε_i is generated iid from a scaled normal distribution such that $\sqrt{\nu} = \|\boldsymbol{X}\boldsymbol{\beta}_{\text{true}}\|_2 / \|\boldsymbol{\varepsilon}\|_2$.

4. We standardize $\boldsymbol{X}, \boldsymbol{y}$ to normalize and center them.

F.2 Real datasets

We use a variety of real datasets from the literature in our computational experiments. The information of each dataset is summarized in Table 7. Note that we increased the number of features on selected datasets by including second-order interactions.

Dataset	n	p	Notes	Reference
Diabetes	442	11		Efron et al. (2004)
Housing	506	13		Gómez and Prokopyev (2021)
Housing2	506	91	2nd order interactions added	Gómez and Prokopyev (2021)
Wine	6497	11		Cortez et al. (2009)
AutoMPG	392	25		Quinlan (1993)
Hitters	263	19	Removed rows with missing data $\boldsymbol{y} = \log(\text{salary})$	Kaggle
Prostate	97	8		R Package `ncvreg`
Servo	167	19	One-hot encoding of features	Ulrich (1993)
Toxicity	38	9		Rousseeuw et al. (2009)
SteamUse	25	8		Rousseeuw et al. (2009)
Alcohol2	44	21	2nd order interactions added	Rousseeuw et al. (2009)
TopGear	242	373		Bottmer et al. (2022)
BarDet	120	200		Ye et al. (2018)
Vessel	180	486		Christidis et al. (2020)
Riboflavin	71	4088		R package `hdi`

Table 7: Real datasets used.

Our sources for these datasets are as follows:

- Four UCI datasets: Auto-MPG, Housing, Servo, and Wine. We obtained these datasets from the online supplement to Gómez and Prokopyev (2021). Note that we increased the number of features for the housing dataset by including second-order interactions (as was already done by Gómez and Prokopyev (2021) for the Auto-MPG dataset). We did not consider second-order interactions for the Servo dataset, as its independent variables are binary, or for the wine dataset, because n is large and considering second-order interactions would not be tractable.
- The `alcohol` dataset distributed via the `R` package `robustbase`. Note that we increased the number of features for this dataset by including second-order interactions.
- The `bardet` dataset provided by Ye et al. (2018).

- The `hitters` Kaggle dataset, after preprocessing the dataset to remove rows with any missing entries, and transforming the response by taking log(Salary), as is standard when predicting salaries via regression.
- The `Prostate` dataset distributed via the R package `ncvreg`.
- The `Riboflavin` dataset distributed by the `R` package `hdi`.
- The `steamUse` dataset provided by Rousseeuw et al. (2009).
- The `topgear` dataset provided by Bottmer et al. (2022).
- The `toxicity` dataset provided by Rousseeuw et al. (2009).
- The `vessel` dataset made publicly available by Christidis et al. (2020).

Appendix G. Detailed computational experiments

We present computational results in Tables 8 and 9 of the results reported in Section 5.1. We observe that times for both methods decrease on a given dataset as γ increases (as expected, since the perspective reformulation is stronger). Interestingly, while the improvements of Algorithm 1 over `Grid` (in terms of time, MIOs solved and nodes) are more pronounced in regimes with large regularization γ, this effect on γ is slight: Algorithm 1 consistently results in improvements over 40% even for the smallest values of γ tested.

Table 8: Comparison between using Algorithm 1 and solving $\mathcal{O}(pn)$ MIOs independently (`Grid`) for leave-one-out cross-validation in four real datasets, for different values of regularization γ. Times reported are in minutes, and correspond to the time to solve all required mixed-integer optimization problems to optimality. No time limits are imposed on the MIOs. Algorithm 1 consistently reduces to number of calls to the MIO solver by 50-85%.

Dataset	p	n	γ	`Grid`			Algorithm 1			**Improvement**		
				Time	# MIO	Nodes	Time	# MIO	Nodes	**Time**	**# MIO**	**Nodes**
			0.01	65	3,978	126,085	37	1,714	58,332	**45%**	**56%**	**53%**
			0.02	52	3,978	82,523	36	1,699	50,333	**30%**	**56%**	**37%**
			0.05	42	3,978	42,411	26	1,868	27,342	**29%**	**52%**	**35%**
Diabetes	11	442	0.10	39	3,978	31,116	25	1,652	15,456	**34%**	**53%**	**48%**
			0.20	35	3,978	22,165	20	1,316	9,111	**42%**	**67%**	**58%**
			0.50	32	3,978	11,889	15	1,147	4,444	**50%**	**71%**	**59%**
			1.00	34	3,978	9,278	14	820	2,416	**58%**	**79%**	**73%**
			0.01	247	6,072	512,723	91	1,867	216,411	**59%**	**69%**	**57%**
			0.02	187	6,072	324,238	64	1,711	139,293	**65%**	**70%**	**56%**
			0.05	166	6,072	216,116	87	1,679	91,822	**45%**	**69%**	**57%**
Housing	13	506	0.10	40	6,072	96,387	18	1,814	40,112	**51%**	**69%**	**58%**
			0.20	82	6,072	68,581	34	1,599	24,899	**55%**	**73%**	**63%**
			0.50	90	6,072	60,067	34	1,233	20,231	**62%**	**79%**	**65%**
			1.00	107	6,072	49,770	22	947	13,111	**77%**	**84%**	**73%**
			0.01	466	3,006	1,669,537	259	1,099	938,012	**41%**	**60%**	**44%**
			0.02	110	3,006	811,432	51	989	399,980	**52%**	**66%**	**51%**
			0.05	44	3,006	324,877	25	965	160,112	**77%**	**84%**	**73%**
Servo	19	167	0.10	23	3,006	162,223	9	679	58,136	**59%**	**77%**	**64%**
			0.20	15	3,006	76,739	8	898	33,030	**48%**	**70%**	**57%**
			0.50	10	3,006	40,197	4	561	10,299	**56%**	**81%**	**74%**
			1.00	8	3,006	25,683	4	479	6,639	**52%**	**84%**	**74%**
			0.01	1,100	9,408	6,772,986	584	2,999	3,221,031	**46%**	**67%**	**48%**
			0.02	1,356	9,408	3,900,417	412	2,433	1,698,234	**67%**	**70%**	**52%**
			0.05	519	9,408	2,286,681	212	2,659	1,012,099	**56%**	**70%**	**50%**
AutoMPG	25	392	0.10	355	9,408	1,548,369	139	2,675	681,344	**59%**	**71%**	**56%**
			0.20	143	9,408	629,020	64	2,387	281,001	**54%**	**71%**	**55%**
			0.50	66	9,408	176,950	28	2,101	56,165	**58%**	**76%**	**67%**
			1.00	68	9,408	116,982	36	1,477	28,112	**43%**	**84%**	**74%**

Table 9: Comparison between using Algorithm 1 and solving $\mathcal{O}(kp)$ MIOs independently (`Grid`) for 10-fold cross validation in four real datasets, for different values of regularization γ. Times reported are in minutes, and correspond to the time to solve all required mixed-integer optimization problems to optimality. No time limits are imposed on the MIOs.

Dataset	p	n	γ	`Grid`			Algorithm 1			**Improvement**		
				Time	# MIO	Nodes	Time	# MIO	Nodes	**Time**	**# MIO**	**Nodes**
			0.01	3	396	11,666	2	242	8,224	**14%**	**39%**	**30%**
			0.02	2	396	8,371	2	235	6,785	**12%**	**41%**	**19%**
			0.05	2	396	4,436	2	228	3,430	**10%**	**42%**	**23%**
Diabetes	11	442	0.10	2	396	3,185	2	247	2,277	**10%**	**38%**	**29%**
			0.20	1	396	2,268	1	206	1,536	**8%**	**48%**	**32%**
			0.50	1	396	1,233	1	149	643	**26%**	**62%**	**48%**
			1.00	1	396	872	1	93	287	**42%**	**77%**	**67%**
			0.01	25	600	48,069	19	321	35,227	**25%**	**47%**	**27%**
			0.02	19	600	34,915	14	310	25,090	**28%**	**48%**	**28%**
			0.05	14	600	21,350	10	303	14,933	**29%**	**50%**	**30%**
Housing	13	506	0.10	10	600	11,012	7	300	7,308	**31%**	**50%**	**34%**
			0.20	9	600	7,406	5	230	3,524	**46%**	**62%**	**52%**
			0.50	9	600	6,168	3	141	1,977	**62%**	**77%**	**68%**
			1.00	8	600	4,993	2	66	930	**77%**	**89%**	**81%**
			0.01	15	288	148,168	12	191	128,592	**16%**	**34%**	**13%**
			0.02	8	288	77,457	7	190	67,416	**10%**	**34%**	**13%**
			0.05	3	288	29,056	3	157	23,653	**16%**	**45%**	**19%**
Servo	19	167	0.10	2	288	15,951	2	146	12,562	**16%**	**49%**	**21%**
			0.20	1	288	8,117	1	155	6,275	**12%**	**46%**	**23%**
			0.50	1	288	4,028	1	201	2,922	**3%**	**30%**	**27%**
			1.00	1	288	2,541	1	206	1,768	**1%**	**28%**	**30%**
			0.01	111	936	691,816	76	389	460,187	**31%**	**58%**	**33%**
			0.02	68	936	401,905	44	374	264,179	**35%**	**60%**	**34%**
			0.05	42	936	225,318	30	396	161,639	**28%**	**58%**	**28%**
AutoMPG	25	392	0.10	30	936	149,243	20	389	98,261	**35%**	**58%**	**34%**
			0.20	14	936	61,534	10	389	41,323	**32%**	**58%**	**33%**
			0.50	7	936	17,865	4	318	8,550	**43%**	**66%**	**52%**
			1.00	6	936	10,848	3	251	4,480	**48%**	**73%**	**59%**