HHS Public Access

Author manuscript

J Comput Graph Stat. Author manuscript; available in PMC 2019 August 07.

Published in final edited form as:

J Comput Graph Stat. 2018 ; 27(4): 861–871. doi:10.1080/10618600.2018.1473777.

Algorithms for Fitting the Constrained Lasso

Brian R. Gaines

Department of Statistics, North Carolina State University

Juhyun Kim and **Hua Zhou**

Department of Biostatistics, University of California, Los Angeles (UCLA)

Abstract

We compare alternative computing strategies for solving the constrained lasso problem. As its name suggests, the constrained lasso extends the widely-used lasso to handle linear constraints, which allow the user to incorporate prior information into the model. In addition to quadratic programming, we employ the alternating direction method of multipliers (ADMM) and also derive an efficient solution path algorithm. Through both simulations and benchmark data examples, we compare the different algorithms and provide practical recommendations in terms of efficiency and accuracy for various sizes of data. We also show that, for an arbitrary penalty matrix, the generalized lasso can be transformed to a constrained lasso, while the converse is not true. Thus, our methods can also be used for estimating a generalized lasso, which has wide-ranging applications. Code for implementing the algorithms is freely available in both the MATLAB toolbox SparseReg and the JULIA package ConstrainedLasso. Supplementary materials for this article are available online.

Keywords

Alternating direction method of multipliers; Convex optimization; Generalized lasso; Linear constraints; Penalized regression; Regularization path

1 Introduction

Our focus is on estimating the constrained lasso problem (James et al., 2013)

SUPPLEMENTARY MATERIAL

Appendix: Further details on the connection between the constrained lasso and the generalized lasso are given in Appendices A.1 and A.2. Derivations of the additional conditions monitored to prevent violations of the subgradient conditions are given in Appendix A.3. Appendices A.4 and A.5 contain additional results for the first (Section 4.1) and third (Section 4.1) simulation settings. Additional results for the benchmark data examples using the microbiome data (Section 5.3) and the housing data (Section 5.4) are in Appendices A.6 and A.7 (GainesKimZhouConstrainedLassoSupplement.pdf).

SparseReg MATLAB Toolbox: MATLAB toolbox consisting of functions for sparse regression, including the algorithms derived in Section 3 (SparseReg-0.0.2.zip). The most recent version is available online at https://github.com/Hua-Zhou/SparseReg.

ConstrainedLasso Julia Package: JULIA package consisting of an open source implementation of the algorithms derived in Section 3 for fitting the constrained lasso (ConstrainedLasso-jl.zip). The most recent version is available online at https://github.com/Hua-Zhou/ConstrainedLasso.

Code and data: Code and data to reproduce the results in Sections 4 and 5 are provided. Please see the README file contained in the zip file for more details (codeData.zip)

$$\begin{aligned} &\text{minimize } \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \rho\|\boldsymbol{\beta}\|_1 \\ &\text{subject to } \boldsymbol{A}\boldsymbol{\beta} = \boldsymbol{b} \text{ and } \boldsymbol{C}\boldsymbol{\beta} \leq \boldsymbol{d}, \end{aligned} \quad (1)$$

where $\boldsymbol{y} \in \mathbb{R}^n$ is the response vector, $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ is the design matrix of predictors or covariates, $\boldsymbol{\beta} \in \mathbb{R}^p$ is the vector of unknown regression n coefficients, and $\rho \geq 0$ is a tuning parameter that controls the amount of regularization. It is assumed that the constraint matrices, $\boldsymbol{A}$ and $\boldsymbol{C}$, both have full row rank. As its name suggests, the constrained lasso augments the standard lasso (Tibshirani, 1996) with linear equality a nd inequality constraints. While the use of the ℓ_1 penalty allows a user to impose prior knowledge on the coefficient estimates in terms of sparsity, the constraints provide an additional vehicle for prior knowledge to be incorporated into the solution. For example, consider the annual data on temperature anomalies given in Figure 1. As has been previously noted in the literature on isotonic regression, in general temperature appears to increase monotonically over the time period of 1850 to 2015 (Wu et al., 2001; Tibshirani et al., 2011). This monotonicity can be imposed on the coefficient estimates using the constrained lasso with the inequality constraint matrix

$$\boldsymbol{C} = \begin{pmatrix} 1 & -1 & & & \\ & 1 & -1 & & \\ & & \ddots & \ddots & \\ & & & 1 & -1 \end{pmatrix} \quad (2)$$

and $\boldsymbol{d} = \boldsymbol{0} \in \mathbb{R}^{p-1}$. The lasso with a monotonic ordering of the coefficients was referred to by Tibshirani and Suo (2016) as the *ordered lasso*, and is a special case of the constrained lasso (1).

Another example of the constrained lasso that has appeared in the literature is the *positive lasso*. First mentioned in the seminal work of Efron et al. (2004), the positive lasso requires the lasso coefficients to be non-negative. This variant of the lasso has seen applications in areas such as vaccine design (Hu et al., 2015b), nuclear material detection (Kump et al., 2012), document classification (El-Arini et al., 2013), and portfolio management (Wu et al., 2014). The positive lasso is a special case of the constrained lasso (1) with $C = -I_p$ and $d = \boldsymbol{0}_p$. Additionally, there are several other examples throughout the literature where the original lasso is augmented with additional information in the form of linear equality or inequality constraints. Huang et al. (2013b) constrained the lasso estimates to be in the unit interval to interpret the coefficients as probabilities associated with the presence of a certain protein in a cell or tissue. The lasso with a sum-to-zero constraint on the coefficients has been used for regression (Shi et al., 2016) and variable selection (Lin et al., 2014) with compositional data as covariates. Compositional data are multivariate data that represent proportions of a whole and thus must sum to one, and arrive in applications such as consumer spending in economics, topic consumption of documents in machine learning, and the human microbiome (Lin et al., 2014). Lastly, simplex constraints were utilized by Huang et al. (2013a) when using the lasso to estimate edge weights in brain networks. Thus, the

constrained lasso is a very flexible framework for imposing additional knowledge and structure onto the lasso coefficient estimates.

During the preparation of our manuscript, we became aware of unpublished work by He (2011) that also derived a solution path algorithm for solving the constrained lasso. However, our approach to deriving the path algorithm is completely different and is more in line with the literature on solution path algorithms (Rosset and Zhu, 2007), especially in the presence of constraints (Zhou and Lange, 2013). Additionally, we address how our algorithms can be adapted to work in the high dimensional setting where $n < p$, which was not done by He (2011). Furthermore, the approach by He (2011) decomposes the parameter vector, $\boldsymbol{\beta}$, into its positive and negative parts, $\boldsymbol{\beta} = \boldsymbol{\beta}^{+} - \boldsymbol{\beta}^{-}$, thus doubling the size of the problem. On the other hand, we work directly with the original coefficient vector at the benefit of computational efficiency and notational simplicity. Lastly, another important contribution of our work is the implementation of our algorithms in the SparseReg MATLAB toolbox and the ConstrainedLasso JULIA package available on GitHub.

The constrained lasso was also studied by James et al. (2013) in an earlier version of their manuscript on penalized and constrained (PAC) regression. The current PAC regression framework extends (1) by using a negative log likelihood for the loss function to also cover generalized linear models (GLMs), and thus is more general than the problem we study. However, we believe the squared error loss function merits additional attention given its widespread use with the ℓ_1 penalty, and also since the constrained lasso is a natural approach to solving constrained least squares problems in the increasingly common high-dimensional setting. Additionally, the use of the squared error loss function yields nice properties of the coefficient solution paths which can be exploited in deriving the path algorithm (Zhou and Wu, 2014). The path algorithm developed by James et al. (2013) is not a traditional solution path algorithm as it is fit on a pre-specified grid of tuning parameters, which is fundamentally different from our path following strategy. Hu et al. (2015a) studied the constrained generalized lasso, which reduces to the constrained lasso when no penalty matrix is included ($D = I_p$). However, they do not derive a solution path algorithm but instead develop a coordinate descent algorithm for fixed values of the tuning parameter.

The rest of the article is organized as follows. In Section 2, we demonstrate a connection between the constrained lasso and the generalized lasso, which shows that the latter can always be transformed and solved as a constrained lasso, even when the penalty matrix is rank deficient. Given the flexibility of the generalized lasso, this result greatly extends the applicability of our algorithms and results. Various algorithms to solve the constrained lasso, including quadratic programming (QP), the alternating direction method of multipliers (ADMM), and a novel path following algorithm, are derived in Section 3. Simulation results that compare the performance of the algorithms are presented in Section 4. The main result from the simulations is that, in terms of run time, the solution path algorithm is more efficient than the other approaches when coefficient estimates at more than a handful of values of the tuning parameter are desired. Benchmark data examples that highlight the flexibility of the constrained lasso are given in Section 5, while Section 6 concludes.

2 Connection to the Generalized Lasso

Another flexible lasso formulation is the generalized lasso (Tibshirani and Taylor, 2011)

$$\text{minimize} \quad \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \rho\|\boldsymbol{D}\boldsymbol{\beta}\|_1, \quad (3)$$

where $D \in \mathbb{R}^{m \times p}$ is a fixed, user-specified regularization matrix. Certain choices of $\boldsymbol{D}$ correspond to different versions of the lasso, including the original lasso, various forms of the fused lasso, and trend filtering. It has been observed that (3) can be transformed to a standard lasso when D has full row rank (Tibshirani and Taylor, 2011), and it can be transformed to a constrained lasso when D has full column rank (James et al., 2013).

Here we note that it is in fact possible to solve a generalized lasso as a constrained lasso even when D is rank deficient, which is stated in Theorem 1 (see Appendix A.1 for the proof).

Theorem 1. *For an arbitrary penalty matrix with* rank($\boldsymbol{D}$) = *r, using the following change of variables*

$$\begin{pmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\gamma} \end{pmatrix} = \tilde{\boldsymbol{D}}\boldsymbol{\beta} = \begin{pmatrix} \boldsymbol{U}_1\boldsymbol{\Sigma}_1\boldsymbol{V}_1^T \\ \boldsymbol{V}_2^T \end{pmatrix} \boldsymbol{\beta}, \quad (4)$$

where $\boldsymbol{U}_1 \in \mathbb{R}^{m \times r}$, $\boldsymbol{U}_2 \in \mathbb{R}^{m \times (m-r)}$, $\boldsymbol{\Sigma}_1 \in \mathbb{R}^{r \times r}$, $\boldsymbol{V}_1 \in \mathbb{R}^{p \times r}$, *and* $\boldsymbol{V}_2 \in \mathbb{R}^{p \times (p-r)}$ *are from the singular value decomposition (SVD) of* $\boldsymbol{D}$, $\boldsymbol{\alpha} \in \mathbb{R}^m$, *and* $\boldsymbol{\gamma} \in \mathbb{R}^{p-r}$, *the generalized lasso problem* (3) *is equivalent to a constrained lasso problem*

$$\begin{aligned} &\textit{minimize} \quad \frac{1}{2}\left\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{D}^+\boldsymbol{\alpha} - \boldsymbol{X}\boldsymbol{V}_2\boldsymbol{\gamma}\right\|_2^2 + \rho\|\boldsymbol{\alpha}\|_1 \\ &\textit{subject to} \quad \boldsymbol{U}_2^T\boldsymbol{\alpha} = \boldsymbol{0}_{m-r}, \end{aligned} \quad (5)$$

where $\boldsymbol{D}^+$ *denotes the Moore-Penrose inverse of the matrix* $\boldsymbol{D}$.

There are three special cases of interest:

1. When D has full row rank, $r = m$, the matrix $\boldsymbol{U}_2$ is null and the constraint $\boldsymbol{U}_2^T\boldsymbol{\alpha} = \boldsymbol{0}$ vanishes, reducing to a standard lasso as observed by Tibshirani and Taylor (2011).
2. When $\boldsymbol{D}$ has full column rank, $r = p$, the matrix $\boldsymbol{V}_2$ is null and the term $\boldsymbol{X}\boldsymbol{V}_2\boldsymbol{\gamma}$ drops, resulting in a constrained lasso as observed by James et al. (2013).

3. When $\boldsymbol{D}$ does not have full rank, $r < \min(m, p)$, the above problem (5) can besimplified to a constrained lasso problem only in $\boldsymbol{\alpha}$ by noticing that minimizing (5) with respect to $\boldsymbol{\gamma}$ yields

$$\boldsymbol{X}\boldsymbol{V}_2\hat{\gamma} = \boldsymbol{P}_{\boldsymbol{X}\boldsymbol{V}_2}\left(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{D}^{+}\boldsymbol{\alpha}\right)$$

or any α, where $\boldsymbol{P}_{\boldsymbol{X}\boldsymbol{V}_2}$ is the orthogonal projection onto the column space $\mathscr{C}(\boldsymbol{X}\boldsymbol{V}_2)$. Thus, the resulting constrained lasso problem is given by

$$\begin{aligned} &\text{minimize } \tfrac{1}{2}\|\tilde{\boldsymbol{y}} - \tilde{\boldsymbol{X}}\boldsymbol{\alpha}\|_2^2 + \rho\|\boldsymbol{\alpha}\|_1 \\ &\text{subject to } \boldsymbol{U}_2^T\boldsymbol{\alpha} = \boldsymbol{0}_{m-r}, \end{aligned}$$

where $\tilde{\boldsymbol{y}} = \left(\boldsymbol{I} - \boldsymbol{P}_{\boldsymbol{X}\boldsymbol{V}_2}\right)\boldsymbol{y}$ and $\tilde{\boldsymbol{X}} = \left(\boldsymbol{I} - \boldsymbol{P}_{\boldsymbol{X}\,\boldsymbol{V}_2}\right)\boldsymbol{X}\boldsymbol{D}^{+}$. The solution path $\hat{\boldsymbol{\alpha}}(\rho)$ can be translated back to that of the original generalized lasso problem via the affine transformation

$$\begin{aligned} \hat{\boldsymbol{\beta}}(\rho) &= \boldsymbol{V}_1\boldsymbol{\Sigma}_1^{-1}\boldsymbol{U}_1^T\hat{\boldsymbol{\alpha}}(\rho) + \boldsymbol{V}_2\left(\boldsymbol{V}_2^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{V}_2\right)^{-}\boldsymbol{V}_2^T\boldsymbol{X}^T\left[\boldsymbol{y} - \boldsymbol{X}\boldsymbol{D}^{+}\hat{\boldsymbol{\alpha}}(\rho)\right] \\ &= \left[\boldsymbol{I} - \boldsymbol{V}_2\left(\boldsymbol{V}_2^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{V}_2\right)^{-}\boldsymbol{V}_2^T\boldsymbol{X}^T\boldsymbol{X}\right]\boldsymbol{D}^{+}\hat{\boldsymbol{\alpha}}(\rho) \\ &+ \boldsymbol{V}_2\left(\boldsymbol{V}_2^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{V}_2\right)^{-}\boldsymbol{V}_2^T\boldsymbol{X}^T\boldsymbol{y}, \end{aligned}$$

where $\boldsymbol{X}^{-}$ denotes the generalized inverse of a matrix $\boldsymbol{X}$.

Thus, any generalized lasso problem can be reformulated as a constrained lasso, so the algorithms and results presented here are applicable to a large class of problems. However, it is not always possible to transform a constrained lasso into a generalized lasso, as detailed in Appendix A.2.

3 Algorithms

In this section, we derive three different algorithms for estimating the constrained lasso (1). Throughout this section, we assume that $\boldsymbol{X}$ has full column rank, which necessitates that $n > p$. For the increasingly prevalent high-dimensional case where $n < p$, we follow the standard approach in the related literature (Tibshirani and Taylor, 2011; Hu et al., 2015a; Arnold and Tibshirani, 2016) and add a small ridge penalty to the original objective function in (1). The problem then becomes

$$\begin{aligned} &\text{minimize } \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \rho\|\boldsymbol{\beta}\|_1 + \frac{\varepsilon}{2}\|\boldsymbol{\beta}\|_2^2 \\ &\text{subject to } \boldsymbol{A}\boldsymbol{\beta} = \boldsymbol{b} \text{ and } \boldsymbol{C}\boldsymbol{\beta} \leq \boldsymbol{d}, \end{aligned} \tag{6}$$

where ε is some small constant, such as 10^{-4}. Note that the objective (6) can be re-arranged into standard constrained lasso form (1)

$$\begin{aligned} &\text{minimize } \frac{1}{2}\|\boldsymbol{y}^* - (\boldsymbol{X}^*)\boldsymbol{\beta}\|_2^2 + \rho\|\boldsymbol{\beta}\|_1 \\ &\text{subject to } \boldsymbol{A}\boldsymbol{\beta} = \boldsymbol{b} \text{ and } \boldsymbol{C}\boldsymbol{\beta} \leq \boldsymbol{d}, \end{aligned} \tag{7}$$

using the augmented data $\boldsymbol{y}^* = \begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{0}_p \end{pmatrix}$ and $\boldsymbol{X}^* = \begin{pmatrix} \boldsymbol{X} \\ \sqrt{\varepsilon}\boldsymbol{I}_p \end{pmatrix}$. The augmented design matrix has full column rank, so the following algorithms can then be applied to the augmented form (7). As discussed by Tibshirani and Taylor (2011), this approach is attractive for more than just computational reasons as the inclusion of the ridge penalty may also improve predictive accuracy.

Before deriving the algorithms, we first define some notation. For a vector v and index set $\mathcal{S}$, let $\boldsymbol{v}_{\mathcal{S}}$ be the sub-vector of size $|\mathcal{S}|$ containing the elements of v corresponding to the indices in $\mathcal{S}$, where $|\cdot|$ denotes the cardinality or size of the index set. Similarly, for a matrix $\boldsymbol{M}$ and another index set $\mathcal{T}$, the matrix $\boldsymbol{M}_{\mathcal{S},\mathcal{T}}$ contains the rows from $\boldsymbol{M}$ corresponding to the indices in $\mathcal{S}$ and the columns of $\boldsymbol{M}$ from the indices in $\mathcal{T}$. We use a colon, :, when all indices along one of the dimensions are included. That is, $\boldsymbol{M}_{\mathcal{S},:}$ contains the rows from $\boldsymbol{M}$ corresponding to $\mathcal{S}$ but all of the columns in $\boldsymbol{M}$.

3.1 Quadratic Programming

Our first approach is to use quadratic programming to solve the constrained lasso problem (1). The key is to decompose $\boldsymbol{\beta}$ into its positive and negative parts, $\boldsymbol{\beta} = \boldsymbol{\beta}^+ - \boldsymbol{\beta}^-$, as the relation $|\boldsymbol{\beta}| = \boldsymbol{\beta}^+ + \boldsymbol{\beta}^-$ handles the ℓ_1 penalty term. By plugging these into (1) and adding the additional non-negativity constraints on $\boldsymbol{\beta}^+$ and $\boldsymbol{\beta}^-$, the constrained lasso is formulated as a standard quadratic program of $2p$ variables,

$$\begin{aligned} &\text{minimize } \frac{1}{2}\begin{pmatrix}\beta^+\\ \beta^-\end{pmatrix}^T \begin{pmatrix} X^TX & -X^TX \\ -X^TX & X^TX \end{pmatrix}\begin{pmatrix}\beta^+\\ \beta^-\end{pmatrix} + \left(\rho\mathbf{1}_{2p} - \begin{pmatrix} X^Ty \\ -X^Ty \end{pmatrix}\right)^T \begin{pmatrix}\beta^+\\ \beta^-\end{pmatrix} \\ &\text{subject to } (\boldsymbol{A} \ -\boldsymbol{A})\begin{pmatrix}\boldsymbol{\beta}^+\\ \boldsymbol{\beta}^-\end{pmatrix} = \boldsymbol{b}, \quad \boldsymbol{\beta}^+ \geq \boldsymbol{0}_p \\ &\qquad (\boldsymbol{C} \ -\boldsymbol{C})\begin{pmatrix}\boldsymbol{\beta}^+\\ \boldsymbol{\beta}^-\end{pmatrix} \leq \boldsymbol{d}, \quad \boldsymbol{\beta}^- \geq \boldsymbol{0}_p. \end{aligned}$$

MATLAB's `quadprog` function is able to scale up to $p \sim 10^2$-10^3, while the commercial GUROBI OPTIMIZER is able to scale up to $p \sim 10^3$-10^4.

3.2 ADMM

The next algorithm we apply to the constrained lasso problem (1) is the alternating direction method of multipliers (ADMM). The ADMM algorithm has experienced renewed interest in

statistics and machine learning applications in recent years as it can solve a large class of problems, is often easy to implement, and is amenable to distributed computing; see Boyd et al. (2011) for a recent survey. In general ADMM is an algorithm to solve a problem that features a separable objective but coupling constraints,

$$\begin{aligned} &\text{minimize} \quad f(\boldsymbol{x}) + g(z) \\ &\text{subject to} \ \boldsymbol{M}\boldsymbol{x} + \boldsymbol{F}z = \boldsymbol{c}, \end{aligned}$$

where $f,g : \mathbb{R}^p \mapsto \mathbb{R} \cup \{\infty\}$ are closed proper convex functions. The idea is to employ block coordinate descent to the augmented Lagrangian function followed by an update of the dual variables $\boldsymbol{\nu}$,

$$\begin{aligned} \boldsymbol{x}^{(t+1)} &\leftarrow \arg\min_{\boldsymbol{x}} \mathscr{L}_\tau\left(\boldsymbol{x}, z^{(t)}, \boldsymbol{\nu}^{(t)}\right) \\ z^{(t+1)} &\leftarrow \arg\min_{\boldsymbol{x}} \mathscr{L}_\tau\left(\boldsymbol{x}^{(t+1)}, z, \boldsymbol{\nu}^{(t)}\right) \\ \boldsymbol{\nu}^{(t+1)} &\leftarrow \boldsymbol{\nu}^{(t)} + \tau\left(\boldsymbol{M}\boldsymbol{x}^{(t+1)} + \boldsymbol{F}z^{(t+1)} - \boldsymbol{c}\right), \end{aligned}$$

where t is the iteration counter and the augmented Lagrangian is

$$\mathscr{L}_\tau(\boldsymbol{x}, z, \boldsymbol{\nu}) = f(\boldsymbol{x}) + g(z) + \boldsymbol{\nu}^T(\boldsymbol{M}\boldsymbol{x} + \boldsymbol{F}z - \boldsymbol{c}) + \frac{\tau}{2}\|\boldsymbol{M}\boldsymbol{x} + \boldsymbol{F}z - \boldsymbol{c}\|_2^2. \quad (8)$$

Often it is more convenient to work with the equivalent scaled form of ADMM, which scales the dual variable and combines the linear and quadratic terms in the augmented Lagrangian (8). The updates become

$$\begin{aligned} \boldsymbol{x}^{(t+1)} &\leftarrow \arg\min_{\boldsymbol{x}} f(\boldsymbol{x}) + \frac{\tau}{2}\|\boldsymbol{M}\boldsymbol{x} + \boldsymbol{F}z^{(t)} - \boldsymbol{c} + \boldsymbol{u}^{(t)}\|_2^2 \\ z^{(t+1)} &\leftarrow \arg\min_{z} g(z) + \frac{\tau}{2}\|\boldsymbol{M}\boldsymbol{x}^{(t+1)} + \boldsymbol{F}z - \boldsymbol{c} + \boldsymbol{u}^{(t)}\|_2^2 \\ \boldsymbol{u}^{(t+1)} &\leftarrow \boldsymbol{u}^{(t)} + \boldsymbol{M}\boldsymbol{x}^{(t+1)} + \boldsymbol{F}z^{(t+1)} - \boldsymbol{c}, \end{aligned}$$

where $\boldsymbol{u} = \boldsymbol{\nu}/\tau$ is the scaled dual variable. The scaled form is especially useful in the case where $\boldsymbol{M} = \boldsymbol{F} = \boldsymbol{I}$, as the updates can be rewritten as

$$\begin{aligned} \boldsymbol{x}^{(t+1)} &\leftarrow \mathbf{prox}_{\frac{1}{\tau}f}\left(\boldsymbol{c} - z^{(t)} - \boldsymbol{u}^{(t)}\right) \\ z^{(t+1)} &\leftarrow \mathbf{prox}_{\frac{1}{\tau}g}\left(\boldsymbol{c} - \boldsymbol{x}^{(t+1)} - \boldsymbol{u}^{(t)}\right) \\ \boldsymbol{u}^{(t+1)} &\leftarrow \boldsymbol{u}^{(t)} + \boldsymbol{x}^{(t+1)} + z^{(t+1)} - \boldsymbol{c}, \end{aligned}$$

where $\mathbf{prox}_{\frac{1}{\tau}f}$ is the proximal mapping of a function f with parameter $\tau > 0$. Recall that the proximal mapping is defined as

$$\mathbf{prox}_{\frac{1}{\tau}f}(\boldsymbol{v}) = \underset{\boldsymbol{x}}{\operatorname{argmin}} \left(f(\boldsymbol{x}) + \frac{\tau}{2}\|\boldsymbol{x} - \boldsymbol{v}\|_2^2 \right).$$

One benefit of using the scaled form for ADMM is that, in many situations including the constrained lasso, the proximal mappings have simple, closed form solutions, resulting in straightforward ADMM updates. To apply ADMM to the constrained lasso, we identify f as the objective in (1) and g as the indicator function of the constraint set $\mathscr{C} = \{\boldsymbol{\beta} \in \mathbb{R}^p : \boldsymbol{A}\boldsymbol{\beta} = \boldsymbol{b}, \boldsymbol{C}\boldsymbol{\beta} \leq \boldsymbol{d}\}$,

$$g(\boldsymbol{\beta}) = \chi_{\mathscr{C}} = \begin{cases} \infty & \boldsymbol{\beta} \notin \mathscr{C} \\ 0 & \boldsymbol{\beta} \in \mathscr{C}. \end{cases}$$

For the updates, $\mathbf{prox}_{\frac{1}{\tau}f}$ is a regular lasso problem and $\mathbf{prox}_{\frac{1}{\tau}g}$ is a projection onto the affine space $\mathscr{C}$ (Algorithm 1). The projection onto convex sets is well-studied and, in many applications, the projection can be solved analytically (see Section 15.2 of Lange (2013) for several examples). For situations where an explicit projection operator is not available, the projection can be found by using quadratic programming to solve the dual problem, which has a smaller number of variables.

Algorithm 1:

ADMM for solving the constrained lasso (1).

1 Initialize $\boldsymbol{\beta}^{(0)} = \boldsymbol{z}^{(0)} = \boldsymbol{\beta}^0$, $\boldsymbol{u}^{(0)} = \boldsymbol{0}$, $\tau > 0$;

2 repeat

3 $\boldsymbol{\beta}^{(t+1)} \leftarrow \operatorname{argmin} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \frac{\tau}{2}\|\boldsymbol{\beta} + \boldsymbol{z}^{(t)} + \boldsymbol{u}^{(t)}\|_2^2 + \rho\|\boldsymbol{\beta}\|_1$;

4 $\boldsymbol{z}^{(t+1)} \leftarrow \operatorname{proj}_{\mathscr{C}}\left(\boldsymbol{\beta}^{(t+1)} + \boldsymbol{u}^{(t)}\right)$;

5 $\boldsymbol{u}^{(t+1)} \leftarrow \boldsymbol{u}^{(t)} + \boldsymbol{\beta}^{(t+1)} + \boldsymbol{z}^{(t+1)}$;

6 until *convergence criterion is met*;

3.3 Path Algorithm

In this section we derive a novel solution path algorithm for the constrained lasso problem (1). According to the KKT conditions, the optimal point $\boldsymbol{\beta}(\rho)$ is characterized by the stationarity condition

$$-\boldsymbol{X}^T[\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}(\rho)] + \rho\boldsymbol{s}(\rho) + \boldsymbol{A}^T\boldsymbol{\lambda}(\rho) + \boldsymbol{C}^T\boldsymbol{\mu}(\rho) = \boldsymbol{0}_p$$

coupled with the linear constraints. Here $\boldsymbol{s}(\rho)$ is the subgradient $\partial\|\boldsymbol{\beta}\|$ with elements

$$s_j(\rho) = \begin{cases} 1 & \beta_j(\rho) > 0 \\ [-1, 1] & \beta_j(\rho) = 0, \\ -1 & \beta_j(\rho) < 0 \end{cases} \quad (9)$$

and $\boldsymbol{\mu}$ satisfies the complementary slackness condition. That is, $\mu_l = 0$ if $\boldsymbol{c}_l^T \boldsymbol{\beta} < d_l$ and $\mu_l \geq 0$ if $\boldsymbol{c}_l^T \boldsymbol{\beta} = d_l$.

Along the path we need to keep track of two sets,

$$\mathcal{A} := \left\{ j : \beta_j \neq 0 \right\}, \quad \mathcal{I}_I := \left\{ l : \boldsymbol{c}_l^T \boldsymbol{\beta} = d_l \right\}.$$

The first set indexes the non-zero (active) coefficients and the second keeps track of the set of (binding) inequality constraints on the boundary. Focusing on the active coefficients for the time being, we have the (sub)vector equation

$$\begin{aligned} \mathbf{0}_{|\mathcal{A}|} &= -\boldsymbol{X}_{:,\mathcal{A}}^T \left(\boldsymbol{y} - \boldsymbol{X}_{:,\mathcal{A}} \boldsymbol{\beta}_{\mathcal{A}} \right) + \rho \boldsymbol{s}_{\mathcal{A}} + \boldsymbol{A}_{:,\mathcal{A}}^T \boldsymbol{\lambda} + \boldsymbol{C}_{\mathcal{I}_I,\mathcal{A}}^T \boldsymbol{\mu}_{\mathcal{I}_I} \\ \begin{pmatrix} \boldsymbol{b} \\ \boldsymbol{d}_{\mathcal{I}_I} \end{pmatrix} &= \begin{pmatrix} \boldsymbol{A}_{:,\mathcal{A}} \\ \boldsymbol{C}_{\mathcal{I}_I,\mathcal{A}} \end{pmatrix} \boldsymbol{\beta}_{\mathcal{A}}, \end{aligned} \quad (10)$$

involving dependent unknowns $\boldsymbol{\beta}_{\mathcal{A}}$, $\boldsymbol{\lambda}$, and $\boldsymbol{\mu}_{\mathcal{I}_I}$, and independent variable ρ. Applying the implicit function theorem to the vector equation (10) yields the path following direction

$$\frac{d}{d\rho} \begin{pmatrix} \boldsymbol{\beta}_{\mathcal{A}} \\ \boldsymbol{\lambda} \\ \boldsymbol{\mu}_{\mathcal{I}_I} \end{pmatrix} = - \begin{pmatrix} \boldsymbol{X}_{:,\mathcal{A}}^T \boldsymbol{X}_{:,\mathcal{A}} & \boldsymbol{A}_{:,\mathcal{A}}^T & \boldsymbol{C}_{\mathcal{I}_I,\mathcal{A}}^T \\ \boldsymbol{A}_{:,\mathcal{A}} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{C}_{\mathcal{I}_I,\mathcal{A}} & \mathbf{0} & \mathbf{0} \end{pmatrix}^{-1} \begin{pmatrix} \boldsymbol{s}_{\mathcal{A}} \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix}. \quad (11)$$

The right hand side is constant on a path segment as long as the sets $\mathcal{A}$ and $\mathcal{I}_I$ and the signs of the active coefficients $\boldsymbol{s}_{\mathcal{A}}$ remain unchanged. This shows that the solution path of the constrained lasso is piecewise linear. The involved matrix is non-singular as long as $X_{:,\mathcal{A}}$ has full column rank and the constraint matrix $\begin{pmatrix} \boldsymbol{A}_{:,\mathcal{A}} \\ \boldsymbol{C}_{\mathcal{I}_I,\mathcal{A}} \end{pmatrix}$ has linearly independent rows. The stationarity condition restricted to the inactive coefficients

$$-\boldsymbol{X}_{:,\mathcal{A}^C}^T \left[\boldsymbol{y} - \boldsymbol{X}_{:,\mathcal{A}} \boldsymbol{\beta}_{\mathcal{A}}(\rho) \right] + \rho \boldsymbol{s}_{\mathcal{A}^C}(\rho) + \boldsymbol{A}_{:,\mathcal{A}^C}^T \boldsymbol{\lambda}(\rho) + \boldsymbol{C}_{\mathcal{I}_I,\mathcal{A}^C}^T \boldsymbol{\mu}_{\mathcal{I}_I}(\rho) = \mathbf{0}_{|\mathcal{A}^C|}$$

determines

$$\rho s_{\mathscr{A}^c}(\rho) = \boldsymbol{X}^T_{:,\mathscr{A}^c}\left[\boldsymbol{y} - \boldsymbol{X}_{:,\mathscr{A}}\boldsymbol{\beta}_{\mathscr{A}}(\rho)\right] - \boldsymbol{A}^T_{:,\mathscr{A}^c}\boldsymbol{\lambda}(\rho) - \boldsymbol{C}^T_{Z_I,\mathscr{A}^c}\boldsymbol{\mu}_{\mathscr{Z}_I}(\rho). \quad (12)$$

Thus $\rho \boldsymbol{s}_{\mathscr{A}^c}$ moves linearly along the path via

$$\frac{d}{d\rho}\left[\rho \boldsymbol{s}_{\mathscr{A}^c}\right] = -\begin{pmatrix} \boldsymbol{X}^T_{:,\mathscr{A}}\boldsymbol{X}_{:,\mathscr{A}^c} \\ \boldsymbol{A}_{:,\mathscr{A}^c} \\ \boldsymbol{C}_{\mathscr{Z}_I,\mathscr{A}^c} \end{pmatrix}^T \frac{d}{d\rho}\begin{pmatrix} \boldsymbol{\beta}_{\mathscr{A}} \\ \boldsymbol{\lambda} \\ \boldsymbol{\mu}_{\mathscr{Z}_I} \end{pmatrix}. \quad (13)$$

The inequality residual $\boldsymbol{r}_{\mathscr{Z}_I^c} := \boldsymbol{C}_{\mathscr{Z}_I^c,\mathscr{A}}\boldsymbol{\beta}_{\mathscr{A}} - \boldsymbol{d}_{\mathscr{Z}_I^c}$ also moves linearly with gradient

$$\frac{d}{d\rho}\boldsymbol{r}_{Z_I^c} = \boldsymbol{C}_{Z_I^c,\mathscr{A}}\frac{d}{d\rho}\boldsymbol{\beta}_{\mathscr{A}}. \quad (14)$$

Together, equations (11), (13), and (14) are used to monitor changes to $\mathscr{A}$ and $\mathscr{Z}_I$, which can potentially result in kinks in the solution path.

To recap, since the solution path is piecewise linear we only need to monitor the events discussed above that can result in kinks along the path, and then the rest of the path can be interpolated. A summary of these events is given in the column on the left in Table 1. We perform path following in the decreasing direction from $\rho_{\max}$ towards $\rho = 0$. Let $\boldsymbol{\beta}^{(t)}$ denote the solution at kink t, then the next kink $t+1$ is identified by the smallest $\Delta\rho$, where $\Delta\rho > 0$ is determined by the first four conditions listed in the right column of Table 1. In addition to monitoring these events along the path, we also need to monitor a technical condition to ensure that the subgradient conditions (9) remain satisfied along the path for the solution path to be well-defined. An issue arises when inactive coefficients on the boundary of the subgradient interval are moving too slowly along the path such that their subgradient would escape $[-1, 1]$ at the next kink $t + 1$. To handle this issue, if an inactive coefficient $\boldsymbol{\beta}_j$, $j \in \mathscr{A}^c$, with subgradient $s_j = \pm 1$ is moving too slowly, the coefficient is moved to the active set $\mathscr{A}$ and equation (11) is recalculated before continuing the path algorithm. The explicit range of $\frac{d}{d\rho}\left[\rho \boldsymbol{s}_{\mathscr{A}^c}\right]$ that needs to be monitored is given in Table 1, and the corresponding derivations are in Appendix A.3.

3.3.1 Initialization—Since we perform path following in the decreasing direction, a starting value for the tuning parameter, $\rho_{\max}$, is needed. As $\rho \to \infty$, the solution to the original problem (1) is given by

$$\begin{aligned} &\text{minimize} \quad \|\boldsymbol{\beta}\|_1 \\ &\text{subject to} \quad \boldsymbol{A}\boldsymbol{\beta} = \boldsymbol{b} \text{ and } \boldsymbol{C}\boldsymbol{\beta} \leq \boldsymbol{d}, \end{aligned} \tag{15}$$

which is a standard linear programming problem. We first solve (15) to obtain initial coefficient estimates $\boldsymbol{\beta}^0$ and the corresponding sets $\mathcal{A}$ and $\mathcal{Z}_I$, as well as initial values for the Lagrange multipliers $\boldsymbol{\lambda}^0$ and $\boldsymbol{\mu}^0$. Following Rosset and Zhu (2007), path following begins from

$$\rho_{\max} = \max \left| \boldsymbol{x}_j^T \left(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}^0 \right) - \boldsymbol{A}_{\cdot j}^T \boldsymbol{\lambda}^0 - \boldsymbol{C}_{Z_I, j}^T \boldsymbol{\mu}_{Z_I}^0 \right|, \tag{16}$$

and the subgradient is set according to (9) and (12). As noted by James et al. (2013), this approach can fail when (15) does not have a unique solution. For example, consider a constrained lasso with a sum-to-one constraint on the coefficients, $\Sigma_j \boldsymbol{\beta}_j = 1$. Any elementary vector e_j, which has a 1 for the j^{th} element and 0 otherwise, satisfies the constraint while also achieving the minimum ℓ_1 norm, resulting in multiple solutions to (15). In this case, it is still possible to use (15) and (16) to identify $\rho_{\max}$, which is then used in (1) to initialize $\boldsymbol{\beta}^0$, $\mathcal{A}$, $\mathcal{Z}_I$, $\boldsymbol{\lambda}^0$, and $\boldsymbol{\mu}^0$ via quadratic programming.

3.3.2 Termination—Another practical consideration for implementing the solution path algorithm is a principled way for terminating the algorithm. To this end, we derive a formula for the degrees of freedom of the constrained lasso. The standard approach in the lasso literature (Efron et al., 2004; Zou et al., 2007; Tibshirani and Taylor, 2011, 2012) is to rely on the expression for degrees of freedom given by Stein (1981),

$$\mathrm{df}(g) = \mathrm{E}\left[\sum_{i=1}^{n} \frac{\partial g_i}{\partial y_i} \right], \tag{17}$$

where g is a continuous and almost differentiable function, which with $g(y) = \hat{\boldsymbol{y}} = \boldsymbol{X}\hat{\boldsymbol{\beta}}$ is satisfied in our case (Hu et al., 2015a). In order to apply (17), we need to assume that the response is normally distributed, i.e. $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}, \sigma^2 \boldsymbol{I})$. As before, we also assume that both constraint matrices, $\boldsymbol{A}$ and $\boldsymbol{C}$, have full row rank, and $\boldsymbol{X}$ has full column rank. Then, using the results in Hu et al. (2015a) with $\boldsymbol{D} = \boldsymbol{I}$, for a fixed $\rho \geq 0$ the degrees of freedom are given by

$$\mathrm{df}(\boldsymbol{X}\hat{\boldsymbol{\beta}}(\rho)) = \mathrm{E}\left[\left| \mathcal{A} \right| - \left(q + \left| \mathcal{Z}_I \right| \right) \right], \tag{18}$$

where $|\mathcal{A}|$ is the number of active predictors, q is the number of equality constraints, and $\left|\mathcal{Z}_I\right|$ is the number of binding inequality constraints. The unbiased estimator for the degrees of

freedom is then $|\mathcal{A}| - (q + |\mathcal{Z}_I|)$. This result is intuitive as the degrees of freedom start out as the number of active predictors, and then one degree of freedom is lost for each equality constraint and each binding inequality constraint. Additionally, when there are no constraints, (1) becomes a standard lasso problem with degrees of freedom equal to $|\mathcal{A}|$, consistent with the result in Zou et al. (2007). The formula (18) is also consistent with results for constrained estimation presented in Zhou and Lange (2013) and Zhou and Wu (2014). We use the degrees of freedom when implementing the solution path algorithm to terminate the path once the degrees of freedom equal n. The number of degrees of freedom is also an important measure that is an input for several metrics used for model assessment and selection, such as Mallows' C_p, AIC, and BIC. Specifically, these criteria can be plotted along the path as a function of ρ as a technique for selecting the optimal value for the tuning parameter, as alternatives to cross-validation.

4 Simulated Examples

To investigate the performance of the various algorithms outlined in Section 3 for solving a constrained lasso problem, we consider three simulated examples. For the simulations, we used the three different algorithms discussed in Section 3 to solve (1). As noted in Section 3.2, the ADMM algorithm includes an additional tuning parameter τ, which we fix at $1/n$ based on initial experiments. Additionally, as pointed out in Boyd et al. (2011), the performance of the ADMM method can be greatly impacted by the choice of the algorithm's stopping criteria, which we set to be 10^{-4} for both the absolute and relative error tolerances. When possible (simulations 1 and 2), we also use a user-defined function handle to solve the subproblem of projecting onto the constraint set $\mathcal{C}$ for ADMM as this improves efficiency. Two factors of interest in the simulations are the size of the problem, (n, p), and the value of the regularization tuning parameter, ρ. Four different levels were used for the size factor, (n, p): (50, 100), (100, 500), (500, 1000), and (1000, 2000). For the latter factor, the values of ρ were calculated as a fraction of the maximum ρ. The fractions, or ρ_{scale} values (i.e., $\rho = \rho_{\text{scale}} \cdot \rho_{\text{max}}$) used in the simulations were 0.2, 0.4, 0.6, and 0.8, to investigate how the degree of regularization impacts algorithm performance. To make the results more comparable, the total runtimes for the solution path algorithm are averaged across the total number of kinks in the path. To generate the data for the first two simulations, the covariates in the design matrix, $\boldsymbol{X}$, were generated as independent and identical (iid) standard normal variables, and the response was generated as $y = \boldsymbol{X}\boldsymbol{\beta} + \varepsilon$ where $\varepsilon \sim N(\mathbf{0}_n, I_n)$. For the third simulation setting, the covariates are again generated from a normal distribution with mean zero but using a covariance matrix with (i,j) elements given by $0.5^{|i-j|}$. As discussed later, the third simulation setting is inspired by an example in the literature (Hu et al., 2015a), so we follow their data generating process. All simulations used 20 replicates and were conducted in MATLAB using the SparseReg toolbox on a computer with an Intel i7–6700 3.4 GHz processor and 32 GB memory. Quadratic programming uses the GUROBI OPTIMIZER via the MATLAB interface, while ADMM and the solution path algorithm are pure MATLAB implementations.

4.1 Sum-to-zero Constraints

The first simulation involves a sum-to-zero constraint on the true parameter vector, $\Sigma_j \boldsymbol{\beta}_j = 0$. Recently, this type of constraint on the lasso has seen increased interest as it has been used in the analysis of compositional data as well as analyses involving any biological measurement analyzed relative to a reference point (Lin et al., 2014; Shi et al., 2016; Altenbuchinger et al., 2017). Written in the constrained lasso formulation (1), this corresponds to $\boldsymbol{A} = \mathbf{1}_p^T$ and $\boldsymbol{b} = 0$. For this simulation, the true parameter vector, $\boldsymbol{\beta}$, was defined such that the first 25% of the entries are 1, the next 25% of the entries are −1, and the rest of the elements are 0. Thus the true parameter satisfies the sum-to-zero constraint, which we can impose on the estimation using the constraints.

The main results of the simulation are given in Figure 2(a), which plots the average algorithm runtime results across different problem sizes, (n, p). The results using quadratic programing (QP) and ADMM are each graphed at two values of ρ_{scale}, 0.2 and 0.6. In the graph we can see that the solution path algorithm was faster than the other methods, and its relative performance is even more impressive as the problem size grows. The graph also shows the impact of the tuning parameter, ρ, on both QP and ADMM. QP performed similarly across both values of ρ_{scale}, but that was not the case for ADMM. At $\rho_{\text{scale}} = 0.6$, ADMM performed very similarly to QP, but ADMM's performance was much worse at smaller values of ρ. Smaller values of ρ correspond to less weight on the ℓ_1 penalty which results in solutions that are less sparse. The ADMM runtimes are also more variable than the other algorithms. estimate at one value of the tuning parameter, $\rho_{\text{scale}}\, \rho_{max}$

While algorithm runtime is the metric of primary interest, a fast algorithm is not of much use if it is woefully inaccurate. When we adopt the objective value error relative to QP as a measure of accuracy, the solution path algorithm is not only efficient but also accurate. On the other hand, the accuracy of ADMM decreases as ρ increases. Part of this is to be expected given that the convergence tolerance used for ADMM is less stringent than the one used for QP. Regardless, the magnitude of these errors, which is generally less than 0.005%, is probably not of practical importance. The plot of the objective value error, both on the original scale and a log scale, for the solution path and ADMM for (n,p) = (500,1000) is given in Figure A.2 in Appendix A.4. The results from the other problem sizes are qualitatively similar and are thus omitted.

4.2 Non-negativity Constraints

The second simulation involves the positive lasso mentioned in Section 1, as to our knowledge it is the most common version of the constrained lasso that has appeared in the literature. Also referred to as the non-negative lasso, as its name suggests it constrains the lasso coefficient estimates to be non-negative. In the constrained lasso formulation (1), the positive lasso corresponds to the constraints $\boldsymbol{C} = -\boldsymbol{I}_p$ and $\boldsymbol{d} = \mathbf{0}_p$. For each problem size, the true parameter vector was defined as

$$\beta_j = \begin{cases} j, & j = 1, \ldots, 10 \\ 0, & j = 11, \ldots, p \end{cases},$$

so the true coefficients obey the constraints and the constrained lasso allows us to incorporate this prior knowledge into the estimation. Figure 2(b) is a graph of the average runtime for each algorithm for the different problem sizes considered. As with simulation 1, the results for quadratic programming (QP) and ADMM are graphed at two different values of ρ, corresponding to $\rho_{\text{scale}} = \rho/\rho_{max} \in \{0.2, 0.6\}$ to also demonstrate the impact of ρ on the estimation time. One noteworthy result is that ADMM fared better relative to QP as the problem size grew and was faster than QP for the two larger sizes under investigation, whereas ADMM had runtimes that were comparable or slower than QP in the first simulation. Since ADMM generally scales more efficiently than QP, we expected ADMM to outperform QP for larger problems and this happened more quickly in the second simulated setting since the inclusion of p inequality constraints notably increased the complexity of the problem. As with the first simulation, another thing that stands out in the results is the strong performance of the solution path algorithm, which generally outperformed the other two methods. However, for $(n, p) = (1000, 2000)$, ADMM and the solution path performed similarly, partly due to the initialization of the path algorithm hampering its performance as the problem size grows. In terms of accuracy, the objective value errors relative to QP for the solution path and ADMM were negligible and are thus omitted.

4.3 Complex Constraints

While the first two simulation settings were motivated by the popularity of the constraints in the literature, it is also of interest to see how the algorithms perform when the constraints are more complex and involve multiple parameters at a time. To this end, we borrow the constraints used in one of the simulations studied by Hu et al. (2015a). The true parameter vector is defined as $\boldsymbol{\beta} = (1, 0.5, -1, 0, \ldots, 0, 1, 0.5, -1, 0, \ldots, 0)^T$, so only its 1st, 2nd, 3rd, 11th, 12th, and 13th elements are nonzero. The constrained lasso is estimated subject to the constraints

$$\begin{aligned} \beta_1 + \beta_2 + \beta_3 \geq 0, \quad & \beta_1 + \beta_3 + \beta_{11} + \beta_{13} = 0 \\ \beta_2 + \beta_5 + \beta_{11} \geq 1, \quad & \beta_2 + \beta_8 + \beta_{12} = 1. \end{aligned}$$

The results for this setting are given in Figure 2(c). For this setting, the performance gap between QP and ADMM is even larger than what was observed in simulation 2 as QP is much slower for the largest setting. As with the other two simulations, ADMM is much more sensitive to the extent of the regularization and does worse as the solutions become less sparse. QP, on the other hand, is again invariant to the value of ρ. The solution path algorithm is able to handle the more complex constraints in stride and again performs noticeably faster than the other methods as the size grows. For even larger problem sizes, the pattern in the relative performance of the algorithms is similar but the gap between QP and the other methods is even more striking as QP does not scale as well, even when using a very efficient commercial solver. Supplemental simulation results that include a larger problem size to highlight this pattern are given in Appendix A.5.

5 Benchmark Data Applications

To highlight the flexibility of the general formulation of the constrained lasso that we have studied, we explore four examples from the literature. For the results in this section, the constrained lasso is estimated using the solution path algorithm (Section 3.3) since that is our main contribution.

5.1 Global Warming Data

For our first application of the constrained lasso on a benchmark dataset, we revisit the global temperature data presented in Section 1, which was provided by Jones et al. (2016). The dataset consists of annual temperature anomalies from 1850 to 2015, relative to the average for 1961–90. As mentioned, there appears to be a monotone trend to the data over time, so it is natural to want to incorporate this information when estimating the trend. Wu et al. (2001) achieved this on a previous version of the dataset by using isotonic regression, which is given by

$$\begin{aligned} &\text{minimize} \ \ \frac{1}{2}\|\boldsymbol{y}-\boldsymbol{\beta}\|_2^2 \\ &\text{subject to} \ \ \beta_1 \leq \cdots \leq \beta_n, \end{aligned} \tag{19}$$

where $\boldsymbol{y} \in \mathbb{R}^n$ is the monotonic data series of interest and $\boldsymbol{\beta} \in \mathbb{R}^n$ is a monotonic sequence of coefficients. The lasso analog of isotonic regression, which adds an ℓ_1 penalty term to (19), can be estimated by the constrained lasso (1) using the constraint matrix $\boldsymbol{C}$ as in (2) and $d = \boldsymbol{0} \in \mathbb{R}^{p-1}$. In this formulation, the constrained lasso provides an entire family of solutions with isotonic regression as a special case when $\rho = 0$. Figure 1 verifies this result by comparing the estimates from the solution path algorithm at $\rho = 0$ with the isotonic regression fit.

5.2 Brain Tumor Data

Our second application of the constrained lasso uses a version of the comparative genomic hybridization (CGH) data from Bredel et al. (2005) that was modified and studied by Tibshirani and Wang (2008), which can be seen in Figure 3. The dataset contains CGH measurements from 2 glioblastoma multiforme (GBM) brain tumors. CGH array experiments are used to estimate each gene's DNA copy number by obtaining the $\log_2$ ratio of the number of DNA copies of the gene in the tumor cells relative to the number of DNA copies in the reference cells. Mutations to cancerous cells result in amplifications or deletions of a gene from the chromosome, so the goal of the analysis is to identify these gains or losses in the DNA copies of that gene (Michels et al., 2007). Tibshirani and Wang (2008) proposed using the sparse fused lasso to approximate the CGH signal by a sparse, piecewise constant function in order to determine the areas with non-zero values, as positive (negative) CGH values correspond to possible gains (losses). The sparse fused lasso (Tibshirani et al., 2005) is given by

$$\text{minimize} \quad \frac{1}{2}\|\boldsymbol{y}-\boldsymbol{\beta}\|_2^2+\rho_1\|\boldsymbol{\beta}\|_1+\rho_2\sum_{j=2}^{p}\left|\beta_j-\beta_{j-1}\right|, \quad (20)$$

where the additional penalty term encourages the estimates of neighboring coefficients to be similar, resulting in a piecewise constant function. This modification of the lasso was originally termed the fused lasso, but in line with Tibshirani and Taylor (2011) we refer to (20) as the sparse fused lasso to distinguish it from the related problem that does not include the sparsity-inducing ℓ_1 norm on the coefficients. Regardless, the sparse fused lasso is a special case of the generalized lasso (3) with the penalty matrix

$$\boldsymbol{D}=\begin{pmatrix}-\boldsymbol{C}\\ \boldsymbol{I}_p\end{pmatrix}\in\mathbb{R}^{(2p-1)\times p},$$

where C is as in (2) and $\boldsymbol{I}_p$ is the $p\times p$ identity matrix. As discussed in Section 2, the sparse fused lasso can be reformulated and solved as a constrained lasso problem. Estimates of the underlying CGH signal from solving the sparse fused lasso as both a generalized lasso (using the genlasso R package (Arnold and Tibshirani, 2014)) and a constrained lasso are given in Figure 3. As can be seen, the estimates from the two different methods match, providing empirical verification of the transformation derived in Section 2.

5.3 Microbiome Data

Our third data application with the constrained lasso uses microbiome data. The analysis of the human microbiome, which consists of the genes from all of the microorganisms in the human body, has been made possible by the emergence of next-generation sequencing technologies. Microbiome research has garnered much interest as these cells play an important role in human health, including energy levels and diseases; see Li (2015) and the references therein. Since the number of sequencing reads varies greatly from sample to sample, often the counts are normalized to represent the relative abundance of each bacterium, resulting in compositional data, which are proportions that sum to one. Motivated by this, regression (Shi et al., 2016) and variable selection (Lin et al., 2014) tools for compositional covariates have been developed, which amount to imposing sum-to-zero constraints on the lasso.

Altenbuchinger et al. (2017) built on this work by demonstrating that a sum-to-zero constraint is useful anytime the normalization of data relative to some reference point results in proportional data, as is often the case in biological applications, since the analysis using the constraint is insensitive to the choice of the reference. Altenbuchinger et al. (2017) derived a coordinate descent algorithm for the elastic net with a zero-sum constraint,

$$\begin{aligned} &\text{minimize } \frac{1}{2}\|\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}\|_2^2+\rho\left(\alpha\|\boldsymbol{\beta}\|_1+\frac{1-\alpha}{2}\|\boldsymbol{\beta}\|_2^2\right) \\ &\text{subject to } \sum_j \beta_j = 0, \end{aligned} \tag{21}$$

but the focus of their analysis, which they refer to as *zero-sum regression*, corresponds to $\alpha = 1$, for which (21) reduces to the constrained lasso (1). Altenbuchinger et al. (2017) applied zero-sum regression to a microbiome dataset from Weber et al. (2015) to demonstrate zero-sum regression's insensitivity to the reference point, which was not the case for the regular lasso. The data contains the microbiome composition of patients undergoing allogeneic stem cell transplants (ASCT) as well as their urinary levels of 3indoxyl sulfate (3-IS), a metabolite of the organic compound indole that is produced in the colon and liver. ASCT patients are at high risk for acute graft-versus-host disease and other infectious complications, which have been associated with the composition of the microbiome and the absence of protective microbiota-born metabolites in the gut (Taur et al., 2012; Holler et al., 2014; Murphy and Nguyen, 2011). One such protective substance is indole, which is a byproduct when gut bacteria breaks down the amino acid tryptophan (Weber et al., 2015).

Of interest, then, is to identify a small subset of the microbiome composition associated with 3-IS levels, as the presence of relatively more indole-producing bacteria in the intestines is expected to result in higher levels of 3-IS in urine. ASCT patients receive antibiotics that kill gut bacteria, but with a better understanding of which bacteria produce indole, antibiotics that spare those bacteria could be used instead (Altenbuchinger et al., 2017). The dataset itself contains information on 160 bacteria genera from 37 patients. Also included in the dataset are their urinary 3-IS levels that are normalized against urinary creatinine concentration to correct for variations in urine flow rate (Waikar et al., 2010). In order to benchmark the performance of the constrained lasso against zero-sum regression, we followed the data preprocessing procedure used by Altenbuchinger et al. (2017). After one pseudo count was added, the bacteria counts were $\log_2$-transformed and then centered. The response variable of interest, normalized 3-IS levels, was $\log_2$-transformed as well. Figure A.4 plots the coefficient estimate solution paths, using both zero-sum regression and the constrained lasso. As can be seen in the graphs, the coefficient estimates are nearly indistinguishable except for some very minor differences, which are a result of the slightly different formulations of the two problems. Since this is a case where $n < p$, a small ridge penalty is added to the constrained lasso objective function (6) as discussed in Section 3, but unlike (21), the weight on the ℓ_2 penalty does not vary across ρ. The observed versus fitted values plot at the optimal ρ is given in Figure A.5 in Appendix A.6. The optimal ρ was chosen via the extended Bayesian Information Criterion (EBIC) proposed by Chen and Chen (2008, 2012) since the classic BIC generally does not perform well when the number of parameters is large relative to the number of observations.

5.4 House Price Data

For the fourth and final data application, we apply the constrained lasso to a housing dataset from Ames, Iowa from 2006 to 2010. The Ames Housing data set (De Cock, 2011) contains

2,930 residential properties as observations and 80 explanatory variables, which consist of 23 nominal, 23 ordinal, 14 count, and 20 continuous variables. The predictor variables include neighborhood, building type, garage size, lot size, and type of road access to property, among others. Although originally used for tax assessment purposes, this data set is suitable for predicting or modeling the sale prices of homes using the extensive property information available.

Traditionally, factor variables are incorporated into a model through a coding scheme that requires the choice of a reference level for each factor. Such a choice is not always obvious and also complicates the interpretation of the coefficients in regression analysis. Here we take an alternative approach by constructing an indicator variable for each level of a factor while imposing sum-to-zero constraints within each factor, which alleviates the need to choose a reference level. That is, factor variable i with k_i levels is treated as a collection of k_i indicator variables, and the k_i coefficients for each factor are constrained to sum to 0. After preprocessing the data and encoding factor variables as dummy variables, the design matrix $\boldsymbol{X}$ is a 2925 × 324 matrix with a constraint matrix $\boldsymbol{A}$ of size 48 × 324 and $\boldsymbol{b} = \mathbf{0}_{48}$. Note that the ith row of $\boldsymbol{A}$ contains k_i 1's:

$$\boldsymbol{A}_{\boldsymbol{i},:} = \left(\mathbf{0}_{k_1}^T \ \mathbf{0}_{k_2}^T \ \cdots \ \mathbf{1}_{k_i}^T \ \mathbf{0}_{k_{i+1}}^T \ \cdots \ \mathbf{1}_{k_{48}}^T\right), \quad (22)$$

where $\mathbf{1}_{k_i} \in \mathbb{R}^{k_i}$ and $\mathbf{0}_{k_j} \in \mathbb{R}^{k_i}$ is a vector of ones and zeros, respectively.

The response variable of interest is the log-transformed sale price, which is then standardized to have mean 0 and standard deviation 1. Figure 4 plots the resulting solution path coefficient estimates against ρ, as well as the observed vs. fitted values at the optimal ρ chosen using the classic Bayesian Information Criterion. The selected features with the largest estimated coefficients in absolute value were overall quality score, living area, and year built. The predicted R^2, based on the PRESS statistic, was equal to 0.893. More details are provided in Appendix A.7.

6 Conclusion

We have studied the constrained lasso problem, in which the original lasso problem is expanded to include linear equality and inequality constraints. As we have discussed and demonstrated through benchmark data applications, as well as other examples cited from the literature, the constraints allow users to impose prior knowledge on the coefficient estimates. Additionally, we have shown that another flexible lasso variant, the generalized lasso, can always be reformulated and solved as a constrained lasso, which greatly enlarges the trove of problems the constrained lasso can solve.

We derived and compared three different algorithms for computing the constrained lasso solutions as a function of the tuning parameter ρ: quadratic programing (QP), the alternating direction method of multipliers (ADMM), and a novel derivation of a solution path algorithm. When the entire solution path is desired, the path algorithm outperforms the other

methods in terms of estimation time without sacrificing accuracy. Although the initialization of the path may obstruct its performance as the problem size grows, the path algorithm is at worst comparable to QP and ADMM. For fixed values of ρ in problems of modest size, QP is a good candidate since it is competitive with ADMM and invariant to the weight of ℓ_1 penalty. For large and complex problems, however, ADMM is preferred due to its scalability. The main caveat to ADMM is its sensitivity to the extent of regularization; ADMM's performance tends to suffer with less sparse true parameters. MATLAB code to implement these algorithms is available in the SparseReg toolbox, and an open source implementation is available in the Julia package ConstrainedLasso.

There are several possible extensions that have been left for future research. It may be possible to improve the efficiency of the solution path algorithm by using the sweep operator (Goodnight, 1979) to update (11) along the path, as was done in related work by Zhou and Lange (2013). Distributed implementations of the algorithms developed here is another direction of research that would improve runtimes. As noted by Boyd et al. (2011), the ADMM algorithm is especially well-suited for distributed computing. It also may be of interest to extend the algorithms to more general formulations of the constrained lasso. All of the algorithms can be extended to handle general convex loss functions, such as a negative log likelihood function for a generalized linear model extension, which was already studied by James et al. (2013) using a modified coordinate descent algorithm. In this case, an extension of the solution path algorithm could be tracked by solving a system of ordinary differential equations (ODE) as in Zhou and Wu (2014).

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgments

The authors thank Colleen McKendry, the editor, the associate editor, and two anonymous referees for the many helpful comments. This research is partially supported by National Science Foundation grants DMS-1055210 and DMS-1127914, and National Institutes of Health grants R01 HG006139, R01 GM53275, and R01 GM105785. All plots were created using the ggplot2 (Wickham, 2009) package in R (R Core Team, 2018).

References

Altenbuchinger M, Rehberg T, Zacharias H, Stämmler F, Dettmer K, Weber D, Hiergeist A, Gessner A, Holler E, Oefner PJ, et al. (2017), "Reference point insensitive molecular data analysis," Bioinformatics, 33, 219–226. [PubMed: 27634945]

Arnold TB and Tibshirani RJ (2014), genlasso: Path Algorithm for Generalized Lasso Problems, R package version 1.3.

Arnold TB (2016), "Efficient Implementations of the Generalized Lasso Dual Path Algorithm," Journal of Computational and Graphical Statistics, 25, 1–27.

Boyd S, Parikh N, Chu E, Peleato B, and Eckstein J (2011), "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers," Foundations and Trends in Machine Learning, 3, 1–122.

Bredel M, Bredel C, Juric D, Harsh GR, Vogel H, Recht LD, and Sikic BI (2005), "High-Resolution Genome-Wide Mapping of Genetic Alterations in Human Glial Brain Tumors," Cancer Research, 65, 4088–4096. [PubMed: 15899798]

Chen J and Chen Z (2008), "Extended Bayesian information criteria for model selection with large model spaces," Biometrika, 95, 759–771.

Chen J (2012), "Extended BIC for small-n-large-P sparse GLM," Statistica Sinica, 555–574.

De Cock D (2011), "Ames, Iowa: Alternative to the Boston housing data as an end of semester regression project," Journal of Statistics Education, 19.

Efron B, Hastie T, Johnstone I, and Tibshirani R (2004), "Least Angle Regression," The Annals of Statistics, 32, 407–499.

El-Arini K, Xu M, Fox EB, and Guestrin C (2013), "Representing Documents Through their Readers," in Proceedings of the 19th Association for Computing Machinery International Conference on Knowledge Discovery and Data Mining, pp. 14–22.

Goodnight JH (1979), "A Tutorial on the SWEEP Operator," The American Statistician, 33, 149–158.

He T (2011), "Lasso and General L1-Regularized Regression Under Linear Equality and Inequality Constraints," unpublished Ph.D. thesis, Purdue University, Dept. of Statistics.

Holler E, Butzhammer P, Schmid K, Hundsrucker C, Koestler J, Peter K, Zhu W, Sporrer D, Hehlgans T, Kreutz M, Holler B, Wolff D, Edinger M, Andreesen R, Levine JE, Ferrara JL, Gessner A, Spang R, and Oefner PJ (2014), "Metagenomic Analysis of the Stool Microbiome in Patients Receiving Allogeneic Stem Cell Transplantation: Loss of Diversity is Associated with use of Systemic Antibiotics and More Pronounced in Gastrointestinal Graft-versus-Host Disease," Biology of Blood and Marrow Transplantation, 20, 640–645. [PubMed: 24492144]

Hu Q, Zeng P, and Lin L (2015a), "The Dual and Degrees of Freedom of Linearly Constrained Generalized Lasso," Computational Statistics & Data Analysis, 86, 13–26.

Hu Z, Follmann DA, and Miura K (2015b), "Vaccine Design via Nonnegative Lassobased Variable Selection," Statistics in Medicine, 34, 1791–1798. [PubMed: 25643693]

Huang H, Yan J, Nie F, Huang J, Cai W, Saykin AJ, and Shen L (2013a), "A New Sparse Simplex Model for Brain Anatomical and Genetic Network Analysis," in International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 625–632.

Huang T, Gong H, Yang C, and He Z (2013b), "ProteinLasso: A Lasso Regression Approach to Protein Inference Problem in Shotgun Proteomics," Computational Biology and Chemistry, 43, 46–54. [PubMed: 23385215]

James GM, Paulson C, and Rusmevichientong P (2013), "Penalized and Constrained Regression," unpublished manuscript, University of Southern California.

Jones P, Parker D, Osborn T, and Briffa K (2016), "Global and Hemispheric Temperature Anomalies - Land and Marine Instrumental Records," Trends: A Compendium of Data on Global Change.

Kump P, Bai E-W, Chan K-S, Eichinger B, and Li K (2012), "Variable Selection via RIVAL (Removing Irrelevant Variables Amidst Lasso Iterations) and its Application to Nuclear Material Detection," Automatica, 48, 2107–2115.

Lange K (2013), Optimization, New York, NY: Springer-Verlag, 2nd ed.

Li H (2015), "Microbiome, Metagenomics, and High-Dimensional Compositional Data Analysis," Annual Review of Statistics and Its Application, 2, 73–94.

Lin W, Shi P, Feng R, and Li H (2014), "Variable Selection in Regression with Compositional Covariates," Biometrika, 101, 785–797.

Michels E, De Preter K, Van Roy N, and Speleman F (2007), "Detection of DNA Copy Number Alterations in Cancer by Array Comparativ-e Genomic Hybridization," Genetics in Medicine, 9, 574–584. [PubMed: 17873645]

Murphy S and Nguyen VH (2011), "Role of Gut Microbiota in Graft-versus-Host Disease," Leukemia & Lymphoma, 52, 1844–1856. [PubMed: 21663498]

R Core Team (2018), *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria.

Rosset S and Zhu J (2007), "Piecewise Linear Regularized Solution Paths," The Annals of Statistics, 35, 1012–1030.

Shi P, Zhang A, and Li H (2016), "Regression Analysis for Microbiome Compositional Data," Annals of Applied Statistics, 10, 1019–1040.

Stein CM (1981), "Estimation of the Mean of a Multivariate Normal Distribution," The Annals of Statistics, 9, 1135–1151.

Taur Y, Xavier JB, Lipuma L, Ubeda C, Goldberg J, Gobourne A, Lee YJ, Dubin KA, Socci ND, Viale A, Perales M-A, Jenq RR, van den Brink MRM, and Pamer1 EG. (2012), "Intestinal Domination and the Risk of Bacteremia in Patients Undergoing Allogeneic Hematopoietic Stem Cell Transplantation," Clinical Infectious Diseases, 55, 905–914. [PubMed: 22718773]

Tibshirani R (1996), "Regression Shrinkage and Selection via the Lasso," *Journal of the Royal Statistical* Society: Series B (Methodological), 58, 267–288.

Tibshirani R, Saunders M, Rosset S, Zhu J, and Knight K (2005), "Sparsity and Smoothness via the Fused Lasso," *Journal of* the Royal Statistical Society: Series B (Statistical Methodology), 67, 91–108.

Tibshirani R and Suo X (2016), "An Ordered Lasso and Sparse Time-lagged Regression," Technometrics, 58, 415–423.

Tibshirani R and Wang P (2008), "Spatial Smoothing and Hot Spot Detection for CGH Data using the Fused Lasso," Biostatistics, 9, 18–29. [PubMed: 17513312]

Tibshirani RJ, Hoefling H, and Tibshirani R (2011), "Nearly-Isotonic Regression," Technometrics, 53, 54–61.

Tibshirani RJ and Taylor J (2011), "The Solution Path of the Generalized Lasso," The Annals of Statistics, 39, 1335–1371.

Tibshirani RJ (2012), "Degrees of Freedom in Lasso Problems," The Annals of Statistics, 40, 1198–1232.

Waikar SS, Sabbisetti VS, and Bonventre JV (2010), "Normalization of urinary biomarkers to creatinine during changes in glomerular filtration rate," Kidney international, 78, 486–494. [PubMed: 20555318]

Weber D, Oefner PJ, Hiergeist A, Koestler J, Gessner A, Weber M, Hahn J, Wolff D, Sta¨mmler F, Spang R, Herr W, Dettmer K, and Holler E (2015), "Low Urinary Undoxyl Sulfate Levels Early After Transplantation Reflect a Disrupted Microbiome and are Associated with Poor Outcome," Blood, 126, 1723–1728. [PubMed: 26209659]

Wickham H (2009), ggplot2: Elegant Graphics for Data Analysis, Springer-Verlag New York.

Wu L, Yang Y, and Liu H (2014), "Nonnegative-Lasso and Application in Index Tracking," Computational Statistics & Data *Analysis*, 70, 116–126.

Wu WB, Woodroofe M, and Mentz G (2001), "Isotonic Regression: Another Look at the Changepoint Problem," Biometrika, 88, 793–804.

Zhou H and Lange K (2013), "A Path Algorithm for Constrained Estimation," Journal of Computational and Graphical Statistics, 22, 261–283. [PubMed: 24039382]

Zhou H and Wu Y (2014), "A Generic Path Algorithm for Regularized Statistical Estimation," Journal of the American Statistical Association, 109, 686–699. [PubMed: 25242834]

Zou H, Hastie T, and Tibshirani R (2007), "On the Degrees of Freedom of the Lasso," The Annals of Statistics, 35, 2173–2192.

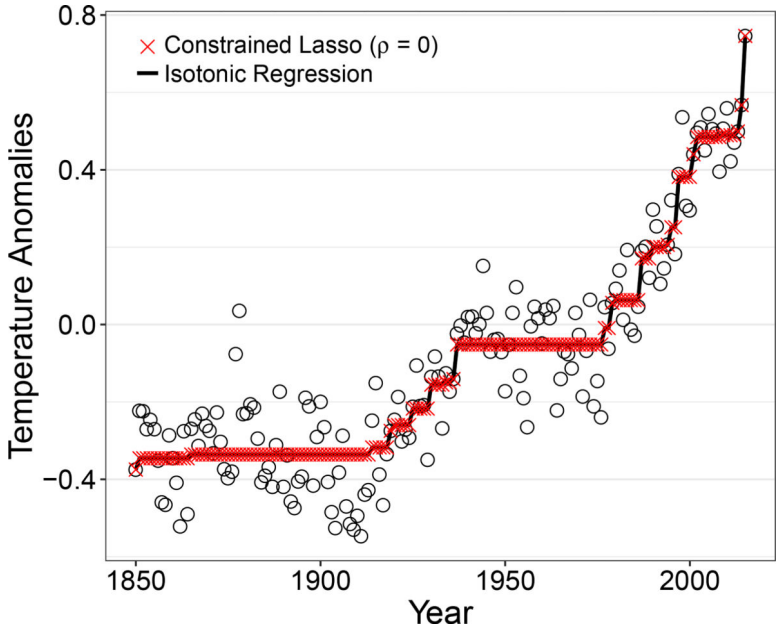
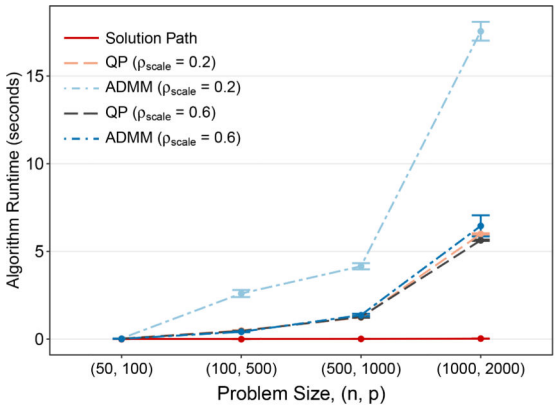
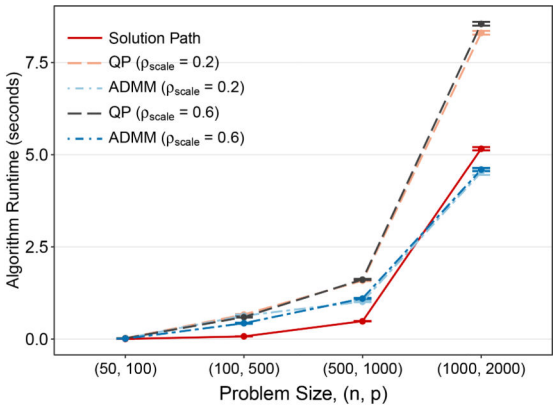


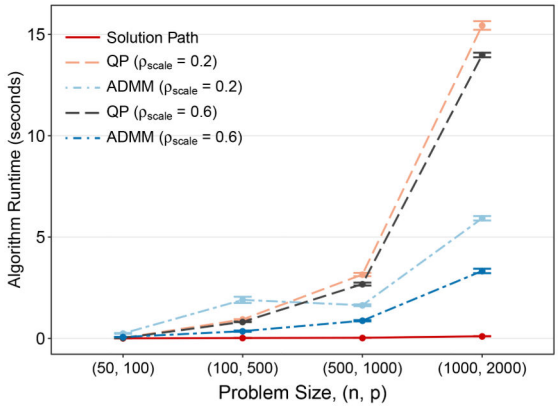
Figure 1:
Isotonic regression fit shows a monotone trend in temperature abnormalities. The constrained lasso solution at $\rho = 0$ is identical to isotonic regression.



(a) Simulation 1: sum-to-zero constraints



(b) Simulation 2: non-negativity constraints



(c) Simulation 3: complex constraints

Figure 2:

Different simulation settings show consistently comparable or superior performance of the path algorithm whereas performances of ADMM and QP vary depending on the size and complexity of the problem. The runtimes for the solution path algorithm are averaged across the number of kinks in the path to make the runtimes more comparable to the other algorithms estimated at one value of the tuning parameter, $\rho = \rho_{\text{scale}} \cdot \rho_{\text{max}}$.

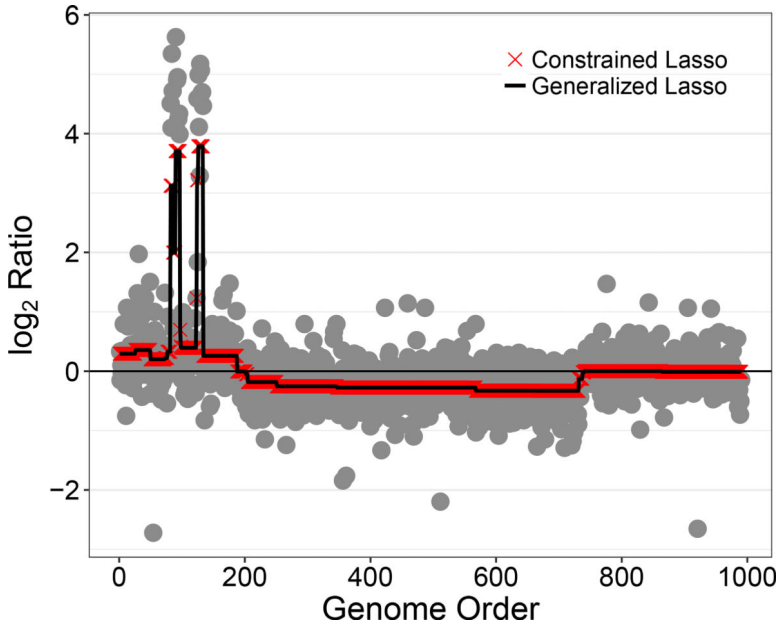


Figure 3:
Generalized lasso and constrained lasso produce identical sparse fused lasso estimates on the brain tumor data

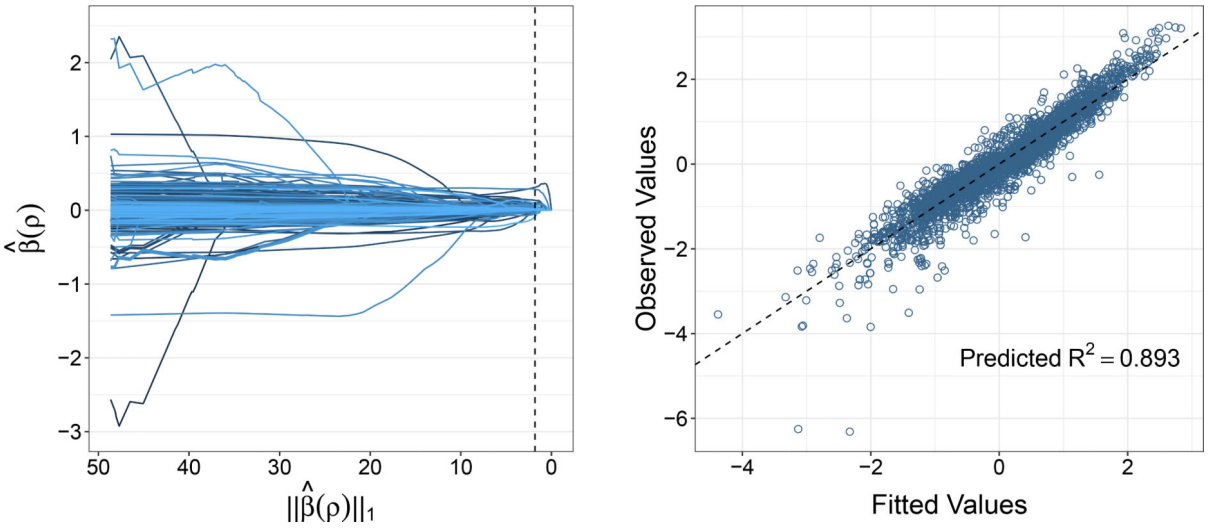


Figure 4:

Path algorithm yields the constrained lasso solution path for the Ames housing data with 48 factors. The dashed line marks the model with the lowest BIC (left panel) as well as the identity line (right panel). Here the response is the log-transformed sale price that is standardized to have mean 0 and standard deviation 1.

Table 1:

Solution Path Events

Event	Monitor
An active coefficient hits 0	$\boldsymbol{\beta}_{\mathscr{A}}^{(t)} - \Delta\rho \frac{d}{d\rho}\boldsymbol{\beta}_{\mathscr{A}}^{(t)} = \mathbf{0}_{\lvert\mathscr{A}\rvert}$
An inactive coefficient becomes active	$\left[\rho^{(t)}\boldsymbol{s}_{\mathscr{A}^c}^{(t)}\right] - \Delta\rho \frac{d}{d\rho}\left[\rho \boldsymbol{s}_{\mathscr{A}^c}\right] = \pm\left(\rho^{(t)} - \Delta\rho\right)\mathbf{1}_{\lvert\mathscr{A}^c\rvert}$
A strict inequality constraint hits the boundary	$\boldsymbol{r}_{\mathscr{Z}_I^c}^{(t)} - \Delta\rho \frac{d}{d\rho} r_{\mathscr{Z}_I^c} = \mathbf{0}_{\lvert\mathscr{Z}_I^c\rvert}$
An inequality constraint escapes the boundary	$\boldsymbol{\mu}_{Z_I}^{(t)} - \Delta\rho \frac{d}{d\rho}\boldsymbol{\mu}_{Z_I} = \mathbf{0}_{\lvert Z_I\rvert}$
Subgradient violations	$s_j \frac{d}{d\rho}\left[\rho s_j\right] < 1$ for $j \in \mathscr{A}^c$ with $s_j = \pm 1$