Data Mining and Knowledge Discovery (2024) 38:49–78
https://doi.org/10.1007/s10618-023-00950-8

Variable screening for Lasso based on multidimensional indexing

Barbara Żogała-Siudem[1] · **Szymon Jaroszewicz**[2,3]

Received: 24 November 2022 / Accepted: 15 June 2023 / Published online: 27 August 2023

Abstract

In this paper we present a correlation based safe screening technique for building the complete Lasso path. Unlike many other Lasso screening approaches we do not consider prespecified values of the regularization parameter, but, instead, prune variables which cannot be the next best feature to be added to the model. Based on those results we present a modified homotopy algorithm for computing the regularization path. We demonstrate that, even though our algorithm provides the complete Lasso path, its performance is competitive with state of the art algorithms which, however, only provide solutions at a prespecified sample of regularization parameters. We also address problems of extremely high dimensionality, where the variables may not fit into main memory and are assumed to be stored on disk. A multidimensional index is used to quickly retrieve potentially relevant variables. We apply the approach to the important case when multiple models are built against a fixed set of variables, frequently encountered in statistical databases. We perform experiments using the complete Eurostat database as predictors and demonstrate that our approach allows for practical and efficient construction of Lasso models, which remain accurate and interpretable even when millions of highly correlated predictors are present.

Keywords Variable screening · Lasso · Linear regression · Open data

Responsible editor: Charalampos Tsourakakis.

✉ Barbara Żogała-Siudem
zogala@ibspan.waw.pl

Szymon Jaroszewicz
szymon.jaroszewicz@ipipan.waw.pl

[1] Systems Research Institute, Polish Academy of Sciences, ul. Newelska 6, 01-447 Warsaw, Poland

[2] Institute of Computer Science, Polish Academy of Sciences, ul. Jana Kazimierza 5, 01-248 Warsaw, Poland

[3] Faculty of Mathematics and Information Science, Warsaw University of Technology, ul. Koszykowa 75, 00-662 Warsaw, Poland

1 Introduction

Variable selection and screening are among the most important problems in machine learning. More and more data becomes available, not only in terms of the number of records but also in terms of the number of variables. Examples include genetic or text data where the dimensionality may be very high. Another example is the Open Data movement (McCrae et al. 2022; Heath et al. 2011), which lead to huge statistical databases from organizations such as the Eurostat, United Nations, International Monetary Fund, and others being available online for free, providing literally millions of variables which could help building accurate predictive models.

Such a huge number of available variables may, however, overwhelm rather than help researchers wanting to add useful predictors to their models. Thus, apart from building sophisticated predictive models we need efficient feature selection methods, especially in very high dimensional setups which are nowadays common.

This paper focuses on a variation of the problem where the set of available variables remains fixed and multiple models are built based on it. Moreover, we allow the number of predictors to be so large that they do not fit in main memory and have to be accessed directly from disk. This makes the approach more challenging since most current screening methods assume all variables to be kept in main memory.

As a motivating example consider a large statistical database, such as the one available from Eurostat (2022) which can be used to explain many different social phenomena. Changes to statistical databases are typically infrequent with most variables updated yearly and a few selected ones monthly, so the data stays constant for large periods of time. Another example is a collection of biological samples on which gene expression data is available. Such data can be used to explain the results of many other tests performed on the same set of samples in the future.

We adapt a well known modeling technique, the Lasso (Tibshirani 1996) to the aforementioned scenario where several models are built against a fixed ultra-high dimensional set of predictors. Our approach is based on pre-indexing all the variables in a multidimensional index, and using queries to the index to efficiently screen variables which cannot enter the model. It is worth noting that we are interested not only in finding a Lasso solution for a given value of the regularization parameter λ, but in constructing the full Lasso path with screening used at each step of the process to find new variable to enter the model. We propose a modification of the homotopy algorithm (Bach et al. 2012) for finding the Lasso path which uses a multidimensional index to speed up computations. We formally prove that our rules are *safe*, i.e. the results are identical to that obtained on the full data without pruning if an exact multidimensional index is used. Our experiments demonstrate, that the quality of the results remains high also for approximate indexing.

Since our Lasso screening criteria are different from the currently available ones, they are a general contribution to this research area. We have tested our approach also in the more typical scenario, where the data fits in main memory and found the proposed screening criteria to be competitive with current state of the art, despite the fact that we compute the full regularization path, not its approximation.

Finding the full Lasso path is beneficial since it gives the values of coefficients for *all* values of the regularization parameter in a given range, providing a complete picture of model evolution. The analyst can clearly see which variables enter/leave the model and when. Moreover, the coefficient vector for any value of λ can be recovered from the path using simple linear interpolation. In Sect. 4.1 we demonstrate that the cost of obtaining the full path using our method is comparable to approximating it with samples and analyze the quality of such approximations.

The contributions of the paper are as follows. First, we propose new safe Lasso screening rules which, unlike current state-of-the-art, allow for constructing the complete Lasso path instead of sampling it at specified points. Second, we address a scenario where several models are built against a fixed set of predictors, and use a multidimensional index to speed up Lasso path construction based on our screening rules. Third, we demonstrate that it is possible to obtain interpretable models by using complete, indexed statistical databases as predictors.

The remaining part of the paper is organized as follows. Section 2 introduces the notation, formulates the Lasso problem and discusses related work. Later, Sect. 3 introduces the multidimensional index based lasso path construction algorithm, which is the main contribution of this paper. Section 4 evaluates the method experimentally and, finally, Sect. 5 concludes the paper.

2 Notation and related literature

We now proceed to introduce the notation used throughout the paper and briefly describe the Lasso estimator.

All vectors are assumed to be column vectors, the superscript ^{T} denotes transposition, and $\|x\|_2$ the l_2 norm of a vector x. A bar above a vector will denote the mean of its components, i.e. $\bar{x} = \frac{1}{n}\sum x_i$. Further, let $\mathrm{cor}(x, y)$ denote the correlation between vectors x and y, i.e. $\mathrm{cor}(x, y) = (x - \bar{x})^T(y - \bar{y})/(\|x - \bar{x}\|_2\|y - \bar{y}\|_2)$.

Unless stated otherwise, we assume that all vectors are *normalized*, i.e. $\bar{x} = 0$ and $\|x\|_2 = 1$. Such a normalization can always be achieved by subtracting the mean and dividing by the norm. The correlation between two normalized vectors is simply equal to their dot product $\mathrm{cor}(x, y) = x^T y$.

Note, that for two normalized vectors x and y there is a monotonic relationship between their distance and correlation

$$\|x - y\|_2^2 = \sum_{i=1}^{n}(x_i - y_i)^2 = 2 - 2\sum_{i=1}^{n} x_i y_i = 2 - 2\mathrm{cor}(x, y). \tag{1}$$

This simple result means that we can use a multidimensional index to find vectors most correlated with a given query vector q, a fact which will be used frequently in the rest of the paper.

Let $y \in \mathbb{R}^n$ be a vector of quantities we are trying to predict and $X = [x_1| \ldots |x_p] \in \mathbb{R}^{n\times p}$ be a matrix consisting of all possible predictors stored as its

columns. Finding Lasso coefficients is based on solving the following optimization problem (Tibshirani 1996):

$$w^{\star}(\lambda) = \arg\min_{w \in \mathbb{R}^p} \frac{1}{2}\|y - Xw\|_2^2 + \lambda \|w\|_1, \tag{2}$$

where $\|w\|_1$ is the l_1 norm of w and $\lambda > 0$ is the regularization parameter. The variables whose coefficients are not equal 0 for a given value of λ form the current *active set*.

Typically one does not solve the problem for a specific value of λ but aims to find the so called *Lasso path*, i.e. solutions $w^{\star}(\lambda)$ for all possible values of λ. For that it suffices to find solutions for those values of the regularization parameter at which a variable is added or removed from the active set (Bach et al. 2012).

2.1 Related literature

We now review the related literature, beginning with a detailed discussion of related screening rules for the Lasso problem. Later we describe general variable screening methods and mention some existing approaches addressing a similar problem setting.

Most relevant to our research are screening methods designed specifically for the Lasso method. Their development started with the Safe Feature Elimination method (SAFE) (El Ghaoui et al. 2010), which, for a given value of the regularization parameter λ, screens out variables which cannot end up in the active solution. This basic Lasso screening technique is not very effective, but was followed by many modifications which improved and extended it in several ways.

The first direction was to increase pruning effectiveness. Examples are the Strong Rules (Tibshirani et al. 2012) used a stricter screening threshold at the expense of allowing some false rejections, or the Dual Polytope Projection (DPP) (Wang et al. 2013), a more efficient method avoiding false rejections. Further research mainly focused on more precise screening tests, usually based on the dual formulation of the problem (Xiang et al. 2011; Xiang and Ramadge 2012; Dai and Pelckmans 2012; Pan and Xu 2019; Lee et al. 2017; Fercoq et al. 2015). A fairly recent overview can be found in Xiang et al. (2016), where tests are classified based on the shape of acceptance region into spherical, dome or elliptical bounds and others.

In this work we only consider spherical tests as they can be naturally implemented using a multidimensional index and focus mainly on *safe* screening rules, i.e. those satisfying

$$\|x_i, q\| \geq R_{screening} \Rightarrow w_i^{\star}(\lambda_k) = 0, \tag{3}$$

where x_i is the vector of values of the variable to be screened, and q, $R_{screening}$ are respectively the acceptance sphere center and radius generated based on the screening rule.

In this paper we rewrite the above condition in terms of correlations for reasons given in Sect. 2.2 to obtain the following, equivalent (for normalized vectors) form

Table 1 Lasso sequential screening rules used in the paper

Screening rule	Query q	$v_{screening}$
Strong Tibshirani et al. (2012)	$r_L(\lambda_{k-1})$	$\frac{2\lambda_k - \lambda_{k-1}}{\|r_L(\lambda_{k-1})\|}$
DPP Wang et al. (2013)	$r_L(\lambda_{k-1})$	$\frac{\lambda_{k-1}}{\|r_L(\lambda_{k-1})\|}\left(1 - \frac{1}{\lambda_k} + \frac{1}{\lambda_{k-1}}\right)$
GAP Fercoq et al. (2015)	$r_L(\lambda_{k-1})$	$\frac{\lambda_{k-1}}{\|r_L(\lambda_{k-1})\|}\left(1 - \frac{2G(\lambda_{k-1})}{\lambda_k \lambda_{k-1}} - \|r_L(\lambda_{k-1})\|^2\left(\frac{1}{\lambda_k} - \frac{1}{\lambda_{k-1}}\right)^2\right)$

$r_L(\lambda_{k-1})$ is the residual vector at the previous point in the path and G the corresponding duality gap (Fercoq et al. 2015)

$$|\text{cor}(x_i, q)| < v_{screening} \Rightarrow w_i^\star(\lambda_k) = 0, \tag{4}$$

where $v_{screening}$ is a minimum correlation threshold. The smaller the value of $R_{screening}$ (or equivalently the larger $v_{screening}$) the more effective the screening rule is.

A further improvement was the invention of *sequential screening rules* which can be used not on a single value of λ but for a whole sequence of values $\lambda_1, \lambda_2, \ldots$ where results computed for λ_{k-1} are used to get a much tighter bound λ_k. This way, it is possible to approximate the full regularization path much more efficiently. Most of the methods for single value of λ mentioned above have sequential versions available (Fercoq et al. 2015; Wang et al. 2013).

Yet another technique called *dynamic screening*, proposed in Bonnefoy et al. (2014) and further developed in Fercoq et al. (2015); Ndiaye et al. (2017) integrates screening directly with the optimization algorithm used to obtain the Lasso solution. Variables are screened after each iteration of the algorithm with the current solution used for improved screening before the next optimization step. While those methods are the current state-of-the-art in Lasso optimization they are not applicable to the scenario considered in this paper. Since our data is based on disk, the computation time is dominated by the time of fetching pre-screened variable into RAM (see Sect. 3, where our main algorithm is described), the actual optimization on pre-screened variables takes only a small percentage of the total.

The screening rules considered in our experiments (Sect. 4) are presented in Table 1 in the correlation based form (Eq. 4). They all have spherical acceptance regions and can thus easily be implemented using a multidimensional index. Also, since we are interested in constructing the full regularization path sequential versions of the rules are used. It can be seen that all rules use the residual vector of the previous model on the path $r_L(\lambda_{k-1})$ as the query center and expressions of growing complexity for the radius, with the GAP SAFE rule (Fercoq et al. 2015) being the most advanced and the tightest.

While many Lasso screening methods are based on tests similar to the one proposed in this paper (which is in fact a spherical test in the language of Xiang et al. (2016)), our bounds are *different from all of them*: the actual goal of screening is different in our case. Current techniques only screen variables for a single fixed value of the regularization parameter λ or for a predefined sequence $\lambda_1, \ldots, \lambda_K$ so the are only capable of constructing a discretized regularization path. The screening we

propose allows for screening away variables which cannot be *the next ones to enter the model on the regularization path*. In other words, the purpose of our screening procedure is not finding the active set for a given λ, but instead, searching for the next variable to be added to the regularization path. As a result, the complete regularization path is returned.

We now discuss other related approaches.

The general problem of screening variables in the ultra-high dimensional setup was considered in Fan and Lv (2008) where the Sure Independence Screening (SIS) criterion was proposed. The authors showed that, under certain assumptions, discarding variables on the basis of correlation with the response retains all relevant variables with high probability. Many methods based on this observation followed, modifying the approach and extending its applicability (Fan et al. 2009; Kong et al. 2017). After initial screening, any model construction method can be used, including Lasso, which provides the second stage of variable selection. There are, however, several problems with this technique. First, the number of pre-screened variables needs to be specified in advance and is difficult to choose in practice. Moreover, the approach may fail to select important features, especially in the presence of many highly correlated variables, which is typical in high dimensional settings. Additionally, theoretical guarantees are valid only asymptotically for large numbers of records, not variables. Those weaknesses have been corroborated experimentally in Żogała-Siudem and Jaroszewicz (2020), where SIS was found to perform poorly on the Eurostat database.

The idea of using multidimensional indexes for variable selection appeared in the now discontinued Google Correlate service (Mohebbi et al. 2011), which was in fact a motivation for our work. The service, given a *query time series* would find the most correlated (in the sense of changes in popularity) Google query. The service had several limitations: it was restricted to finding correlated Google queries, not general data, and only found single most correlated variables, while the approach proposed in this paper allows for building complete linear models.

Similar problems (building many models against a fixed set of predictors) have been analyzed in the context of dictionary based methods (Xiang et al. 2016). In this paper we only consider regression problems, which are frequent in social sciences and biological applications.

There are also a few systems which allow for automatically linking to publicly available open datasets. A primary example is the RapidMiner's Linked Open Data Extension (Paulheim et al. 2014; Ristoski 2015) and its predecessor the FeGeLOD system (Paulheim 2012; Ristoski and Paulheim 2013) which can be used to automatically add variables to the model. Unfortunately, those systems are based on syntactic matching of the indexing attributes (e.g., country/year) based on ontologies alone, and will thus result in huge amounts of statistically irrelevant variables being directly included in the modeling process. Consider, for example, the researcher who wants to find out which factors affect the infant mortality rate in each European country. The target variable is indexed by country/year pairs, and, since there are literally millions of variables indexed by those

attributes available in open data collections, including all of them would likely overwhelm the model construction procedure, unless additional mechanisms are used, such as the one proposed in this paper.

In Żogała-Siudem and Jaroszewicz (2020) multidimensional indices were used to speed up stepwise regression in a scenario addressed in this paper.

2.2 Multidimensional indexing

We now present a brief overview of literature on multidimensional indexing. A *multidimensional index* is a data structure, which enables efficient searches for pre-indexed vectors which are close to a given query vector q. Our approach requires two types of queries to be supported: k-nn queries, where the search returns k vectors closest to the query q, and range queries, where all vectors within a given distance r from q are returned.

A multidimensional index can be exact or approximate. Among the most popular exact indexes are *kd*-trees (Bentley 1975) and ball-trees (Omohundro 1989), however due to their poor performance for a larger number of dimensions (even as low as 20), we focused on approximate solutions.

Approximate multidimensional indexing is a very active research area which fairly recently came up with several new approaches such as Locality Sensitive Hashing (Indyk and Motwani 1998; The ANNOY library 2023), Hierarchical Navigable Small World (HNSW) graphs (Malkov and Yashunin 2020) or anisotropic nearest neighbor search (Guo et al. 2020), the last two approaches being the current state of the art. Several solutions have also been proposed for correlation indexing, where the distance measure is the correlation coefficient itself (The ANNOY library 2023; Andoni et al. 2015). (Aumüller et al. 2020a) provides a recent overview of available techniques, with up-to-date benchmarks accessible at (Aumüller et al. 2020b).

Using an appropriate multidimensional index is a key factor in the implementation of our method. Unfortunately most approaches and implementations turned out to be inadequate for our problem. First, many available software packages allow only k-nn queries, not range queries which are necessary for our algorithms (see Sect. 3). Further, most approaches assume all indexed vectors must be stored in RAM, which makes them inappropriate for our use case. We ended up using the excellent Faiss library (Jégou et al. 2020) which allows for disk based indexes and range queries.

3 Index based Lasso path construction

In this section we briefly describe the homotopy algorithm for solving the Lasso problem and introduce our algorithm for the Lasso path construction.

3.1 The basic homotopy algorithm

There exist many algorithms for solving the Lasso problem (Eq. 2) for a specific value of λ as well as for finding the whole regularization path (i.e. estimating $w^\star(\lambda)$ for all possible values of λ). Examples of path finding algorithms include LARS (Efron et al. 2004) and the homotopy method (Bach et al. 2012) used in this paper. An overview of the topic can be found in Bach et al. (2012).

The *homotopy method* (Algorithm 1) proceeds in steps, adding and removing variables one at a time. Let J be the set of indices of variables in the current active set. The algorithm starts with λ high enough for J to be empty and, during each iteration, finds (by sequentially scanning all available predictors) the largest λ value for which the next feature is going to be added or removed from the active set. Since the Lasso path is piecewise linear (Tibshirani 1996), this corresponds to finding the nearest 'kink' on the regularization path. Variables are added and removed until a specified stopping criterion is reached, for example, the maximum number of iterations k_{max}, maximum size of the active set s_{max}, or the minimum value of the regularization parameter λ_{min}. Details can be found in Bach et al. (2012).

The main bottleneck of the algorithm in the considered scenario is step 5, where all variables not in the current active set need to be checked sequentially. Since we assume $p \ggg n$ and the data not to fit in main memory, implementing this step in a brute force manner will be very slow. This motivates our multidimensional index based screening procedure described next.

Algorithm 1 BRUTE-LASSO: The basic homotopy algorithm for finding the Lasso path

Require: $y \in \mathbb{R}^{n\times 1}$, $X \in \mathbb{R}^{n\times p}$, $k_{max}, s_{max}, \lambda_{min}$

1: $\lambda_1 \leftarrow \max_{j=1,\ldots,p} |\mathrm{cor}(x_j, y)|$
2: $J \leftarrow \{j : |\mathrm{cor}(x_j, y)| = \lambda_1\}$
3: **for** $k = 2, \ldots, k_{max}$ **do**
4: Find largest $\lambda_a < \lambda_{k-1}$ and $\lambda_b < \lambda_{k-1}$ such that:
5: (a) $\exists j_a \notin J : |x_{j_a}^T (y - X w^\star(\lambda_a))| = \lambda_a$,
6: (b) $\exists j_b \in J : w^\star_{j_b}(\lambda_b) = 0$.
7: **if** $\lambda_a > \lambda_b$ **then**
8: $J \leftarrow J \cup \{j_a\}$; $\lambda_k \leftarrow \lambda_a$
9: **else**
10: $J \leftarrow J \setminus \{j_b\}$; $\lambda_k \leftarrow \lambda_b$
11: **end if**
12: Calculate $w^\star(\lambda_k)$
13: **if** $|J| \geqslant s_{max}$ or $\lambda_k < \lambda_{min}$ **then**
14: **break**
15: **end if**
16: **end for**
17: **return** Regularization path: $\{(\lambda_i, w^\star(\lambda_i))\}_{i=1,\ldots,k}$

3.2 Index based Lasso screening criterion

In this section we are going to present the main contribution of the paper: a Lasso path algorithm based on a multidimensional index. In what follows, we assume that the index is already constructed and contains all available variables. This is justified, since we assume that multiple models are built against a fixed set of predictors (see Sect. 1).

We begin by presenting the theoretical results behind our index-based screening procedure. First, we show that the equation in step 5 of Algorithm 1 can be rewritten in terms of correlations of a potential new variable x with two specific vectors denoted s and u. Let J be the set of indices of variables in the current active set, let X_J denote the matrix of active variables, and let $t_J = \text{sgn}(X_J^T(y - Xw^\star(\lambda)))$. The following theorem holds (the proofs of this and all remaining theorems can be found in Appendix 1).

Theorem 1 *Let x be a variable not in the current active set J. Then the condition $|x^T(y - Xw^\star(\lambda))| = \lambda$ (step 5 of Algorithm 1)*

can be written as

$$|d_u \text{cor}(x, u) + \lambda d_s \text{cor}(x, s)| = \lambda, \tag{5}$$

where

$$u = \frac{y - X_J(X_J^T X_J)^{-1} X_J^T y}{d_u}, \qquad s = \frac{X_J(X_J^T X_J)^{-1} t_J}{d_s},$$
$$d_u = \|y - X_J(X_J^T X_J)^{-1} X_J^T y\|_2, \quad d_s = \|X_J(X_J^T X_J)^{-1} t_J\|_2.$$

Further, u and s are normalized, orthogonal, $d_u \leqslant 1$, $d_s \geqslant 1$, and $|J| = 1$ implies $d_s = 1$.

Theorem 1 states that in order to find the value of λ at which a variable x not currently in the active set would enter the model, it is sufficient to find its correlations with two orthogonal vectors u and s. Notice that u is the normalized residual vector of Ordinary Least Squares regression of y on X_J (Rao 2009).

We begin with an intuitive explanation of our criterion and later provide a formal theorem with a proof. Due to orthogonality of u and s, $\text{cor}(x, u)^2 + \text{cor}(x, s)^2 \leqslant 1$ for any vector x, and we can visualize pairs $(\text{cor}(x, u), \text{cor}(x, s))$ as points lying in a unit circle.[1] This interpretation, shown in Fig. 1, will be used to graphically illustrate our selection criterion.

Let us first note that not all points in this circle can be achieved by potential new variables:

[1] To see this note that u and s are orthonormal so the length of the projection of x onto the subspace they span is $\sqrt{(xu)^2 + (xs)^2}$. Since x is a unit vector we get $(xu)^2 + (xs)^2 \leqslant \|x\|_2^2 = 1$.

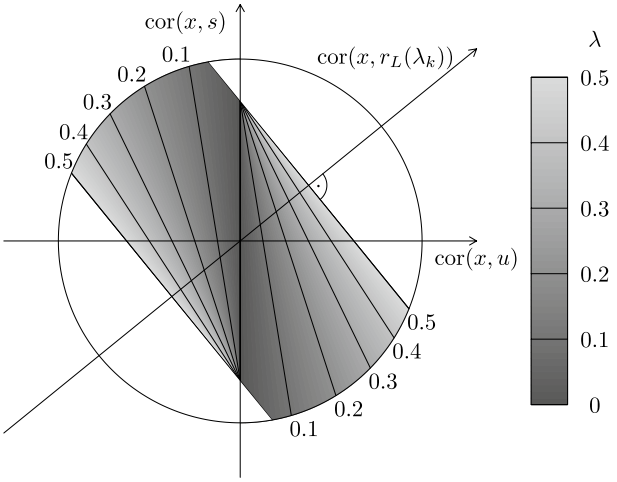

Fig. 1 The area of possible solutions to Eq. 5 with additional constraints based on Theorem 2 for $\lambda_k = 0.5$. The shade of gray represents the value of λ at which a variable corresponding to this point would be added to the model. A 'third axis' shows the correlation of potential new variables with current Lasso residual $r_L(\lambda_k)$

Theorem 2 *Let λ_k be the value of the regularization parameter at the time of the last 'kink' on the path (i.e. point where a variable is added or removed from the model), X_J the active set at this point, and u, s, d_u, d_s be defined as in Theorem 1. Then for any variable x not in the current active set*

$$|x^T r_L(\lambda_k)| = |d_u \text{cor}(x, u) + \lambda_k d_s \text{cor}(x, s)| < \lambda_k, \tag{6}$$

where $r_L(\lambda_k) = y - Xw^\star(\lambda_k)$ is the residual vector of the current Lasso solution.

Variables x which satisfy the above condition form a band in the $(\text{cor}(x, u), \text{cor}(x, s))$ coordinates, shown as a shaded area in Fig. 1. The values of λ at which a given x would enter the active set are presented with different shades of gray and cannot exceed λ_k. x's which would enter the model at given λ correspond to pairs of line segments (see Eq. 5) marked in the figure for a few possible values of λ. We are interested in finding a variable with the maximum $\lambda < \lambda_k$, i.e. lying on the segment within the gray band which forms the largest angle with the y axis.

In order to restrict the search space of such variables, we first select a candidate variable x_0 which would be added to the model at λ_0 (see Sect. 3.3). Then, we are going to search only variables which would enter the model for $\lambda_k > \lambda > \lambda_0$. In the $(\text{cor}(x, u), \text{cor}(x, s))$ coordinates such x's correspond to points on line segments angled further away from the y-axis than segments corresponding to λ_0. The areas which we need to search are shown as red wedges in Fig. 2. Those areas lie within the hatched circular caps shown in the figure. Notice that there are no variables corresponding to points in the caps which are not in the red wedges: such variables would have entered the model earlier with $\lambda > \lambda_k$ which contradicts our

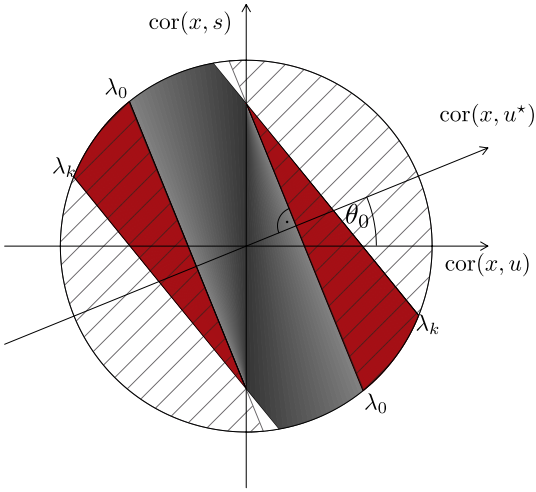

Fig. 2 The areas which must be searched in order to find all variables better than a candidate x_0 are marked with dark red color. The hatched caps show the final search areas for new variables. A 'third axis' shows the correlation of potential new variables with a vector $u^\star$ defined in Theorem 3 (Color figure online)

assumption that x is not in the active set. Therefore, searching within those caps provides a safe screening rule for the next variable to be added to the model.

As will be shown in the theorem below, points in the caps correspond to variables highly correlated with a vector $u^\star = u\cos\theta_0 + s\sin\theta_0$ for an appropriately chosen θ_0. This allows all candidate variables to be selected using a range query around $\pm u^\star$. The correlation with $u^\star$ is shown as an additional 'third axis' in Fig. 2.

Theorem 3 *Let vectors $u, s \in \mathbb{R}^n$ and values $d_u, d_s \in \mathbb{R}$ be defined as in Theorem 1. Additionally, let $x_0 \in \mathbb{R}^n$ be a variable not in the current active set, and let λ_0 be the value for which it would enter the model. Define the vector*

$$u^\star = u\cos\theta_0 + s\sin\theta_0,$$

where $\tan\theta_0 = \frac{\lambda_0 d_s}{d_u}$, $\theta_0 \in (0, \frac{\pi}{2})$. Then $|\mathrm{cor}(x_0, u^\star)| = \frac{\sin\theta_0}{d_s}$. Moreover, for any other variable $x \in \mathbb{R}^n$ not in the active set, the value λ at which x would enter the model satisfies

$$\lambda > \lambda_0 \quad \Leftrightarrow \quad |\mathrm{cor}(x, u^\star)| > |\mathrm{cor}(x_0, u^\star)|.$$

Theorem 3 states that all variables more strongly correlated with $u^\star$ than x_0 would be added to the model for larger values of λ and that candidates better than x_0 can only be found among such variables. The theorem thus provides a safe screening rule for the next variable to be added to the Lasso path. Since the test is based only on correlations it is suitable for use with multidimensional indices.

3.3 Index based homotopy algorithm

Based on Theorem 3 we can modify Algorithm 1, such that in step 5 we do not search through the entire set of all possible variables but will, instead, narrow the search to candidates selected based on range queries to the multidimensional index. Algorithm 2, which we call MI-Lasso, shows those modifications.

Algorithm 2 MI-Lasso: Multidimensional index based homotopy algorithm for finding the Lasso path

Require: $y \in \mathbb{R}^{n\times 1}$, $X \in \mathbb{R}^{n\times p}$, $k_{max}, s_{max}, \lambda_{min}$

1: $\lambda_1 \leftarrow \max_{j=1,\ldots,p} |\mathrm{cor}(x_j, y)|$; $J \leftarrow \{j : |\mathrm{cor}(x_j, y)| = \lambda_1\}$
2: **for** $k \leftarrow 2, \ldots, k_{max}$ **do**
3: Calculate u, s, d_u, d_s ▷ see Theorem 1
4: Find x_0 and λ_0 ▷ 1-nn queries to index, see text
5: $\theta_0 \leftarrow \mathrm{atan}(\frac{\lambda_0 d_s}{d_u})$ ▷ see Theorem 3
6: $u^\star \leftarrow u\cos\theta_0 + s\sin\theta_0$; $v \leftarrow \frac{\sin\theta_0}{d_s}$
7: $J_{cand} \leftarrow \{j \notin J : |\mathrm{cor}(x_j, u^\star)| > v\}$ ▷ 2 range queries
8: Find largest $\lambda_a < \lambda_{k-1}$ and $\lambda_b < \lambda_{k-1}$ such that:
9: (a) $\exists j_a \in J_{cand} : |d_u \mathrm{cor}(x_{j_a}, u) + \lambda_a d_s \mathrm{cor}(x_{j_a}, s)| = \lambda_a$
10: (b) $\exists j_b \in J : w^\star_{j_b}(\lambda_b) = 0$
11: **if** $\lambda_a > \lambda_b$ **then**
12: $J \leftarrow J \cup \{j_a\}$; $\lambda_k \leftarrow \lambda_a$
13: **else**
14: $J \leftarrow J \setminus \{j_b\}$; $\lambda_k \leftarrow \lambda_b$
15: **end if**
16: Compute $w^\star(\lambda_k)$
17: **if** $|J| \geqslant s_{max}$ or $\lambda_k < \lambda_{min}$ **then**
18: **break**
19: **end if**
20: **end for**
21: **return** Regularization path: $\{(\lambda_i, w^\star(\lambda_i))\}_{i=1,\ldots,k}$

In each iteration we need to update the vectors u and s, and then find a candidate vector x_0 which should correspond to a large value of λ_0 to make pruning effective. We select x_0 by finding predictors outside the active set which are most correlated with OLS and Lasso residual vectors (respectively u and r_L), and picking the one with a higher value of λ. The search is implemented efficiently using 1-nn queries to the index.

Based on Theorem 3, we compute in step 6 the query vector $u^\star$ and the search radius $\sqrt{2-2v}$ (see Eq. 1) for the range query used to obtain a subset J_{cand} of variables which need to be checked explicitly as in the standard homotopy algorithm. We show in Sect. 4 that very few variables are typically selected.

Finding λ_b (step 10 of Algorithm 2) in which a variable may be removed from the active set proceeds as in the standard homotopy algorithm since it is not computationally demanding.

3.4 Approximate indices and Lasso path consistency

Theorem 3 guarantees that the Lasso paths obtained using Algorithms 1 and 2 are identical as long as the set J_{cand} is selected correctly. However, if an approximate multidimensional index is used, it may omit the actual variable which should be added to the path at a given iteration. Such a variable is going to be incorrectly present in the inactive set and may be selected by the index in consecutive steps, albeit with an incorrect value of λ. This will lead to the regularization path being inconsistent and/or discontinuous.

Fortunately, there is a simple test to check whether the currently added variable should have in fact been added earlier: when a variable is added in step 9 of Algorithm 2 its coefficient should be equal to zero; a significantly larger value indicates an error. In order to correct such situations we can either discard such a variable and select the next best one, or find the true value of λ at which it should have been added and restart the algorithm at that value.

In our experiments this situation did not happen often, only in about 0.2% of all iterations in all runs of our simulation experiments (described in Sect. 4). We thus decided to simply discard such variables.

4 Experimental evaluation

In this section we will evaluate MI-Lasso and compare it with another screening techniques.

4.1 Experiments on genetic data

First, let us test MI-Lasso on a relatively small data set which fits in RAM in order to test the effectiveness of the proposed screening rules in a general context. The dataset contains gene expression levels in the rat eye (Scheetz et al. 2006; Huang et al. 2006). The task is to predict the expression level of gene TRIM32 (which is known to be one of the causes of a genetic disorder called the Bardet-Biedl syndrome) based on expression levels of other genes. The data set consists of 120 observations and 31, 098 variables, and due to its size can be processed entirely in RAM. Thus, in this case we used the so called brute force multidimensional index which simply scans all variables to answer a query. This enables us to compare with state of the art method for memory based data, namely the dynamic version of GAP screening criterion with active warm starts (Ndiaye et al. 2017) implemented in the gsroptim package (Ndiaye 2023) with the 'active GS' option.

Table 2 Computation times in seconds for MI-Lasso and GAP safe screening (gsroptim) for different values of λ_{min} and varying number m of points on gsroptim path approximation

	$m = k$			$m = 2k$			$m = 10k$		
λ_{min}	0.1	0.01	0.001	0.1	0.01	0.001	0.1	0.01	0.001
k	67	217	254	67	217	254	67	217	254
MI-Lasso	0.50	1.79	2.18	**0.50**	**1.79**	**2.18**	**0.50**	**1.79**	**2.18**
gsroptim	**0.38**	**1.17**	**1.56**	0.70	2.22	2.66	3.20	10.50	12.68

Shortest computation times are marked in bold

k is the true number of change points on the regularization path between λ_{max} and λ_{min}

In our comparisons we will use the same value of λ_{min} (see Sect. 3.3) for all methods. Also, $\lambda_{max} = \max_j |\text{cor}(x_j, y)|$ is the same for all methods. Let k denote the number of 'kinks' on the regularization path returned by the MI-Lasso algorithm.

Since our method explicitly finds the values of the regularization parameter λ at which a variable is added or removed from the path, and the GAP screening method finds coefficient vectors for pre-specified values of λ, the comparison becomes difficult. Let us first discuss the choice of the sequence $\lambda_{max} = \lambda_1 \geqslant \lambda_2 \geqslant \ldots \geqslant \lambda_m = \lambda_{min}$ for gsroptim, where m is the length of the sequence. There are two ways of constructing such a sequence used most frequently in literature. The first assumes equal spacing between consecutive lambdas, the second assumes lambdas to be equally spaced on the logarithmic scale, i.e. λ_{i-1}/λ_i to be constant for all i. We chose the second strategy, since it corresponds the real life scenarios where the path changes more frequently for smaller λ's.

Let us first compare computation times for various values of λ_{min} and m. Let k denote the true number of 'kinks' on the regularization path returned by the MI-Lasso algorithm. To make the comparisons meaningful we express m as multiples of k such that the computational effort of both methods is similar. Table 2 gives the results. Since MI-Lasso does not use the parameter m, its execution times are the same in all three subtables.

It can be seen that runtimes of both methods are comparable, with MI-Lasso gaining an upper hand when the number of points on the path approximated by gsroptim is larger; for $m = 10k$ the differences become significant. This is despite the fact that MI-Lasso provides a complete regularization path, not an approximation. We also note, that our implementation is written pure Python (using the `numpy` library), while gsroptim uses compiled code for faster execution.

We will now analyze how well does the GAP screening based method approximate the full regularization path returned by MI-Lasso. To this end we will visualize how many variables changed (were added or removed from the model) between two consecutive approximation points λ_{i-1}, λ_i. For MI-Lasso we will use true 'kinks' on the path, and the difference in model size will always be $+1$ or -1.

The results are shown in Fig. 3. The three plots correspond to increasing path approximation accuracy with, respectively, $m = k$ (left), $m = 2k$ (middle), $m = 10k$ (right), where m is the number of points in the approximated path and k is the true number of change points on the path. For MI-Lasso each step results in either

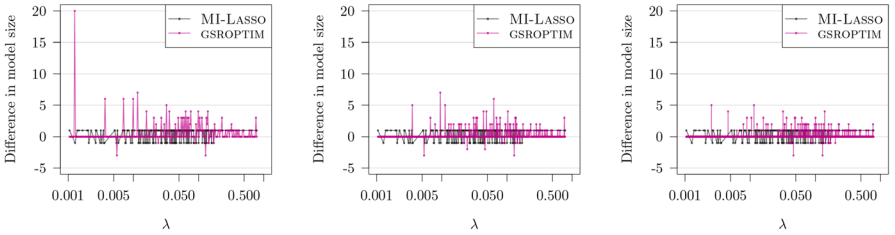

Fig. 3 Change in model size between consecutive steps on the approximated path (GSROPTIM) and the exact path (MI-LASSO). In each step on the exact path exactly one variable is added or discarded. The plots correspond to $m = k$ (left), $m = 2k$ (middle), $m = 10k$ (right), where m is the number of points in approximated path and k is the true number of change points on the path

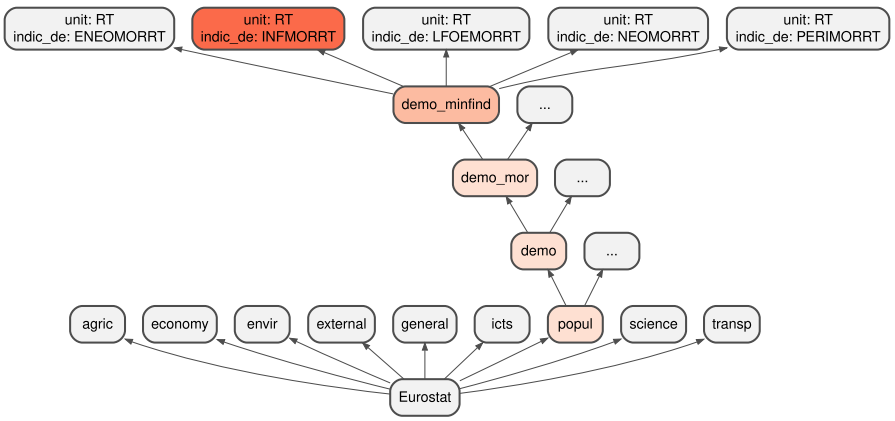

Fig. 4 The structure of the Eurostat database

adding or removing exactly one variable (as expected), but for the GAP screening based method we often end up with more than 5 variables being added between consecutive path points. This means that we are missing parts of the regularization path where significant changes to the model happen. With increasing number of points on the path the approximation improves, but the full path is never completely reconstructed.

We believe that the above experiments clearly demonstrate the advantages of our method since the approximation remains coarse, even when computation times significantly exceed those of MI-LASSO.

4.2 The Eurostat database

This section covers an experimental evaluation of Algorithm 2 on the huge Eurostat database (Eurostat 2022). The Eurostat database consists of thousands of data sets which are organized in a hierarchical structure, starting with the general categories, such as economy and finance, transport, etc. Below each main category there is

an extensive subcategory tree, and specific datasets are the leaves of the structure. Each of the datasets contains variables corresponding to different values of certain measures.

Figure 4 illustrates the location in the database of the variable describing infant mortality rate, used later in Sect. 4.3. The variable is located in the `demo_minfind` dataset, which is located in the `popul` category, `demo` subcategory which contains demographic data, and in the more specific `demo_mor` subcategory which contains mortality related data.

The `demo_minfind` dataset has two dimensions. The first, `indic_de`, describes the statistic under consideration and takes 5 different values: INFMORRT (infant mortality rate, used as a response variable later in our experiments), NEOMORRT, ENEOMORRT, LFOEMORRT and PERIMORRT. The second dimension is the unit in which the values were expressed, which in this dataset can take only one value: `RT` – the number of cases per 1000 births. The dataset therefore contains a total of $5 \times 1 = 5$ variables. Detailed descriptions of all datasets and variables can be found on the Eurostat website (Eurostat 2022). Variables can be indexed by various attributes, but in this work we only use variables indexed by `(country,year)` pairs. There are about 8 million such variables, which were all included in our experiments.

We limit our consideration to 32 European countries (all current EU members, the UK, and four EFTA members: Iceland, Norway, Liechtenstein, Switzerland) over 25 years (1990–2014). There are thus $n = 800$ observations for each variable. All included variables amounted to the total of 25 GB of uncompressed binary data. For experiments involving predictive accuracy we split the data into a training set (years 1990–2011) and a test set (years 2012–2014).

In the database, many values are missing. The actual amount varies depending on year and type of data. For example, for newer data in the environment section there are about 33% missing values, while for earlier years the value can be as high as 90% in some categories. We used the following simple imputation scheme: missing values between two known time points were imputed using linear interpolation, and missing values at the beginning and end of a time series were imputed using, respectively, the first or last available value for a given country.

An important problem was the presence of highly correlated or even identical variables. Many of such correlations followed from statistical dependencies between features which must exist in such a broad database (for example, many factors are correlated with the country's general level of economic development). Other reasons are of a more technical nature, e.g. the same variable may be present in more than one dataset, or expressed in different units.

We begin with performance evaluation of MI-Lasso on semi-synthetic data, later we present an illustrative example demonstrating that our approach is capable of efficiently discovering useful and interpretable models even in the presence of millions of highly correlated variables.

4.2.1 Evaluation on semi-synthetic data

In order to test the performance of Algorithm 2 in a controlled setting we resort to simulation experiments. To make them as close to real data as possible, we use original Eurostat variables as predictors, but the response vectors y are generated according to a known ground truth model with 3 randomly selected predictors. More specifically, we simulate the data according to the following steps:

1. Select three Eurostat variables x_1, x_2 and x_3 at random and normalize them.
2. Generate $y = 10x_1 + 3x_2 + x_3 + \varepsilon$ where ε is an i.i.d. normally distributed noise with standard deviation of 0.02.
3. Split y and all available variables into train and test sets (see above).
4. Run Algorithm 2 with $s_{\max} = 50$, $\lambda_{\min} = 0.01$ on the training part of y with all indexed Eurostat variables as features.

The above simulation was repeated 100 times. The parameters in step 4 were chosen such that the algorithm selects 50 variables but stops early if the regularization parameter λ decreases below 0.01.

The computations were performed on an Intel(R) Core(TM) i7-8550U CPU running at 1.80GHz with 32GB RAM. To compare models we used the RMSE metric which is most common in regression tasks. In our experiments we used an approximate index from the Faiss library (Johnson et al. 2017). More specifically, we used the `IVF4096,Flat` index which allows for disk based operations. The index is based on a fairly simple but practically very effective algorithm (Babenko and Lempitsky 2014). The method applies k-means clustering to the original data to obtain 4096 clusters. To perform queries a brute force search is performed to find a given number (controlled by the `nprobe` parameter, which we set to 64) of clusters with centers closest to the query vector. The final solution is then found within only those clusters, see (Johnson et al. 2017) for details. While the approach is much simpler than newer algorithms, we found it to work surprisingly well for our use case.

4.2.2 Comparison with other Lasso screening methods

We compare our approach (MI-Lasso) with three other Lasso screening approaches: Strong Rules (Strong), DPP, and GAP (see Sect. 2.1 for details). The first method is approximate, the second and third (as well as our method) are safe, i.e. they should return the same Lasso path that would be obtained if no screening was used. We consider the GAP method, based on the duality gap of the Lasso optimization problem, to be the current state of the art. As can be seen in Table 1, the three competing methods are spherical tests and can thus also be implemented using a multidimensional index. For a fair comparison, we modified the Strong, DPP, and GAP algorithms to use the same index as the method proposed in this paper. This was possible since all those methods are based on spherical tests.

Moreover, we use iterative versions of Strong, DPP, and GAP methods which allow for efficient screening for a whole sequence of values of λ (see Sect. 2.1),

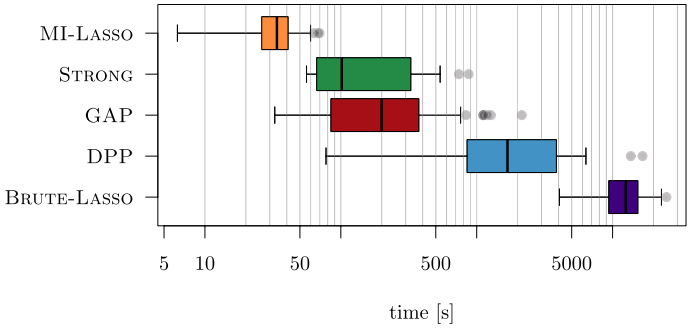

Fig. 5 Computation time needed to obtain a Lasso model with 50 variables for various screening methods

thus providing a sample from the regularization path. In our experiments we used a sequence of values between $\lambda_{min} = 0.01$ and $\lambda_{max} = \max_j |\text{cor}(x_j, y)|$, equally spaced at 0.002 intervals. This configuration gave the best results in our experiments.

Additionally, we include the Brute-Lasso method (Algorithm 1) which is a classic Lasso implementation without any additional screening. Due to long computation times (recall that the variables were stored on disk) only 50 simulations were run for this method.

As mentioned in the previous section, it is difficult to directly compare earlier methods with MI-Lasso because they compute sets of active variables for a fixed sequence of values of the λ parameter, while our method directly finds points on the path where variables are added/removed. We overcome this problem by aligning the regularization paths returned by all methods by the moment the given size of the active set is reached. The size of the active set is used as the x-axis in charts comparing the methods' performance. This way, meaningful comparisons of cost needed to obtain a model of a given size become possible.

Additionally, we impose a limit of 200, 000 (about 2.5% of available predictors) on the number of variables returned by the multidimensional index due to performance issues with Strong, DPP, and GAP methods; the limit was never reached by MI-Lasso.

4.2.3 Computation time

We begin with a high level comparison of computation times. Since we assume that many models will be built against the same set of predictors, index construction time is not included in reported computation times. For completeness we note that index construction took 33 min and 27 s and was much faster than downloading the full Eurostat database and converting it into tabular form which took several hours.

Figure 5 shows the computation time needed to construct a Lasso model with 50 variables averaged over 100 simulation (20 simulations were used for Brute-Lasso due to very long computation time).

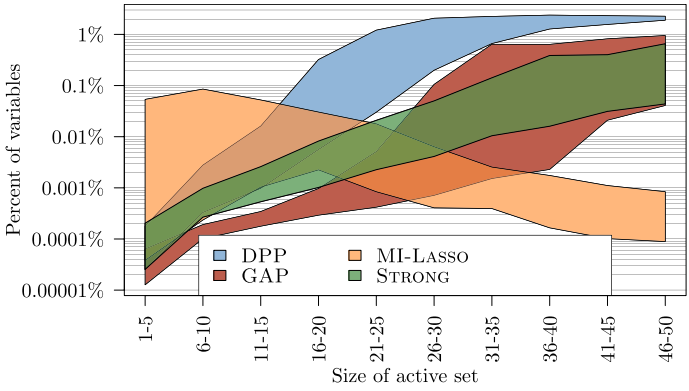

Fig. 6 Percentage of candidate variables left after screening as a function of the size of the active set. The bands show values between first and third quartiles over all simulations

It can be seen that MI-Lasso is an order of magnitude faster than Strong (an approximate method) and GAP. It is two orders of magnitude faster than DPP, which in turn is much faster than Brute-Lasso.

4.2.4 Screening effectiveness

To better pinpoint the differences in performance of the methods we investigate the effectiveness of screening, i.e. the percentage of variables which pass the test and need to be checked explicitly (as in step 5 of Algorithm 1). The results are shown in Fig. 6 as a function of model size. Sizes are averaged in groups of 5 for clarity.

We can see that the percentage of variables preselected by MI-Lasso remains relatively constant, but increases rapidly with the size of active set for Strong, DPP, and GAP. For DPP it quickly reaches the upper bound of 200, 000 variables. Screening effectiveness of the proposed method is thus orders of magnitude higher for larger active sets. For models with a small number of variables DPP and Strong are more effective but the overall number of candidates is small for such models, so there is little effect on the overall performance, as seen in Fig. 5. Moreover, it is important to remember that MI-Lasso constructs a complete regularization path which the other methods only sample at predefined points.

4.2.5 Predictive accuracy

Now, let us take a look at predictive accuracy of the models. While all methods (except for the approximate Strong) should in principle return the same regularization path, the use of an approximate multidimensional index may result in significant differences. Moreover, after screening, the actual Lasso coefficients need to be found through optimization which may not be reliable for larger sets of candidates

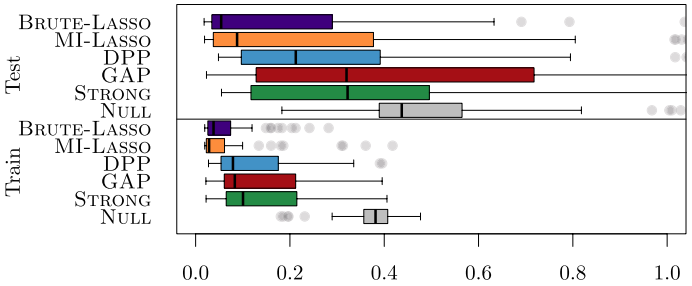

Fig. 7 Predictive accuracy for Lasso models constructed using different variable screening methods

due to numerical issues[2]. Larger sets of candidates which pass screening make it more likely that a wrong variable will enter the model due to numerical problems. Numerical issues may also adversely affect the estimated coefficients.

In order to test predictive accuracy we need to select a single model based on the Lasso path. Lasso is known to often find wrong variables quite early on the regularization path (Su et al. 2017), so we decided use the Lasso itself as a screening procedure as suggested in Bühlmann and Van De Geer (2011). To this end we used the first 50 variables selected by Lasso and built, based on them, a new linear model using a second stage of variable selection. We chose the SOS algorithm proposed in Pokarowski and Mielniczuk (2015), which combines the Lasso with the Zheng-Loh model selection proceedure (Zheng and Loh 1995), i.e. orders variables preselected by Lasso based on their t-statistics and considers models obtained by adding variables in that order. The final model is selected based on the EBIC criterion (Chen and Chen 2008).

Figure 7 presents root mean squared errors for all screening methods on training and test data. Additionally a Null, intercept-only model is included for comparison.

As we can see, MI-Lasso offers the best predictive performance of all screening methods, comparable with the orders of magnitude slower Brute-Lasso. All methods clearly outperform the Null model, proving that Lasso can work reliably even in the presence of millions of highly correlated potential predictors.

4.2.6 Variable selection

The final question we are going to ask is whether the true predictors used to build the ground truth model are included in the variables selected by the Lasso estimator. Unfortunately, since the Eurostat data contains many highly correlated (or even identical) variables, one cannot expect to select the exact same variables which were used to construct the base models. To overcome this issue we check how many of the variables selected by Lasso have high correlation with any of the original variables used in the ground truth model. More precisely, we consider a variable to be correctly selected if its correlation with one of the original variables is above 0.95.

[2] We used the optimization procedure implemented in the `scikit-learn` package.

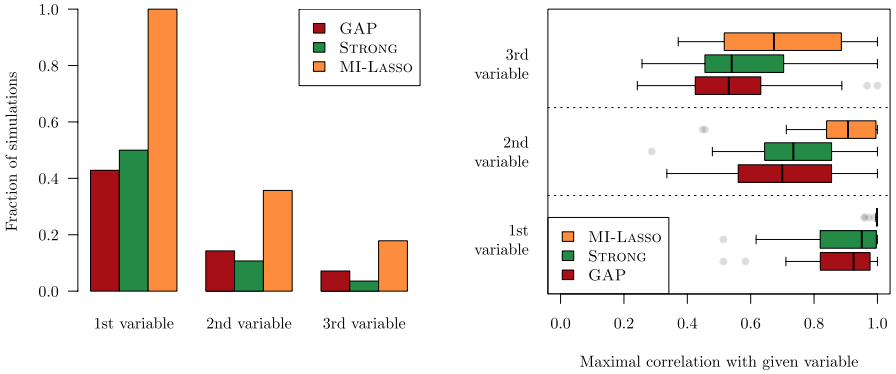

Fig. 8 Variable selection effectiveness. Left chart: fraction of simulations in which variables highly correlated with each variable in the ground truth model were found. Right chart: maximal correlation across simulations of any selected variable with each ground truth variable

As in the previous experiments we continued the Lasso path construction until 50 variables were selected or $\lambda_{min} = 0.01$ was reached; the set of variables active at that point was used as the final selection.

Figure 8 shows the results. The left part of the figure shows the fraction of simulations in which at least one variable highly correlated with, respectively, first (x_1), second (x_2), and third (x_3) ground truth variable was found. It can be seen that the proposed algorithm always selected at least one variable highly correlated with the first predictor (denoted x_1 in Sect. 4.2.1) which had the largest coefficient equal to 10. In about 40% of simulations MI-Lasso found at least two correct variables. The recall is thus not perfect but the results are certainly useful.

The performance of GAP and Strong screening methods was much worse and even the first variable was recovered in less than 50% of cases. We conjecture that the reason was their lower screening effectiveness which resulted in the index returning too many variables in response to range queries.

For illustration, the right chart of Fig. 8 shows maximal correlation across simulations of any selected variable with each ground truth variable. This chart confirms that MI-Lasso found variables which tended to be more correlated with those in the ground truth model.

4.3 An illustrative example

In this section we show an example of using Algorithm 2 on the Eurostat database, and demonstrate that interpretable results may be obtained with standard Lasso even on such a big and highly correlated dataset. As the response variable we chose infant mortality rate per country per year (`INFMORRT` variable in the `demo_minfind` dataset[3]). The reason was the availability of data for this variable for all countries over long periods of time, as well as its popularity in social sciences, which makes

[3] https://ec.europa.eu/eurostat/databrowser/bookmark/65bc64f2-a011-40e7-9880-754134facd65.

Table 3 Variables added to the model predicting infant mortality rate during the first five iterations

Eurostat id	Description
1. First variable (x_1) added to the model	
Section: economy, dataset:nama_10_co3_p3, variable:coicop:CP01, unit:PC_GDP	Final consumption expenditure on food and non-alcoholic beverages of households as a percentage of gross domestic product
link: https://ec.europa.eu/eurostat/databrowser/bookmark/319240c5-5f39-4d81-906e-af950a9fc3b1	
First variable, direct index search, $\lambda = 0.820$	
2. Second variable (x_2) added to the model	
Section: general, dataset:nama_10_gdp, variable:na_item:D21, unit:PD10_NAC	GDP component (in national currency) concerning taxes, in reference to 2010 prices
Link: https://ec.europa.eu/eurostat/databrowser/bookmark/1920b00f-6de8-4ed1-ad1e-2f6ed4b7dfee	
Candidates selected: 123 (0,0015% total), $\lambda = 0.753$	
3. Third variable (x_3) added to the model	
Section: popul, dataset:demo_find, variable:AGEMOTH	Mean age of women at childbirth
Link: https://ec.europa.eu/eurostat/databrowser/bookmark/b6d7eb96-e4d0-4f56-9535-2be38316a0a5	
Candidates selected: 27728 (0.37 % total), $\lambda = 0.715$	
4. Fourth variable (x_4) added to the model	
Section: economy, dataset:nama_10_co3_p3, variable:coicop:CP011, unit:PC_GDP	Final consumption expenditure on food of households as a percentage of gross domestic product
Link: https://ec.europa.eu/eurostat/databrowser/bookmark/d9e631b5-a43c-4241-b749-051b3f8c45e1	
Candidates selected: 43801 (0,58% total), $\lambda = 0.690$	
5. Fifth variable (x_5) added to the model	
Section: popul, dataset:demo_find, variable:AGEMOTH3	Mean age of women at birth of third child
Link: https://ec.europa.eu/eurostat/databrowser/bookmark/c4c52360-afda-4c0f-a640-916bbf033f92	
Candidates selected: 41499 (0.55% total), $\lambda = 0.575$	

Full descriptions of the variables can be found on the Eurostat website. The total time for finding those variables was 2.89 s

it easier to verify our findings. Since in this example we focus on the interpretability of the model, we decided to stop the construction of the path after selecting five variables.

Before we started the modeling process, we discarded variables leaking information about the target, specifically the data sets demo_r_minfind, demo_mlifetable, demo_r_minfind, demo_minfind, hlth_cd_asdr, demo_mlexpec, hlth_cd_asdr2, hlth_cd_aperrto, demo_r_mlife, hlth_cd_acdr as well as all variables with designation INFMORRT from the hlth_cd_info dataset. Those datasets either contain copies of the target variable itself or variables highly influenced by it, such as various measures of life expectancy. During each iteration we ignored variables having too many missing values (more than 50%).

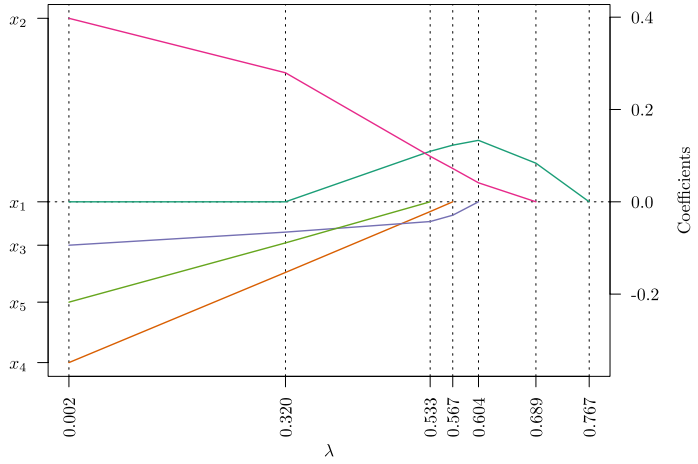

Fig. 9 The regularization path for the Lasso estimator of infant mortality rate. Numbers on the left axis correspond to variable numbers in Table 3. The figure shows the part of the path until the fifth variable is added to the model at $\lambda = 0.575$. For illustrative purposes the path is continued for those five variables until $\lambda = 0$

The model construction process is summarized in Table 3, where the first five selected variables are shown. For each variable we provide Eurostat identifiers, a bookmark to the variable on the Eurostat website, and a short description. The actual Lasso path for those variables is shown in Fig. 9.

The first variable, which is simply the one most correlated with the target, describes consumption expenditure on food as percentage of GDP. This value is likely to be high in poorer countries, where satisfying the basic needs is more challenging. The relation between poverty and infant mortality rates is well known (Sims et al. 2007), and the higher (relatively to e.g. total GDP) the expenditure on basic needs, the worse the economic situation of a country is. Figure 9 shows that the coefficient of the variable is positive, as expected.

The second variable added to the model is one of the components of a country's GDP, which in turn is related to infant health and development perspectives (Finlay et al. 2011). The variable's coefficient is negative (Fig. 9): higher GDP correlates with lower infant mortality. The next feature concerns mean age of women at childbirth, which is known to be related to infant health; the coefficient is negative (Fig. 9) because children born by underage mothers tend to have higher mortality (Finlay et al. 2011). The next variable is very similar to the first one and is in fact highly correlated with it. As the Lasso path evolves (Fig. 9), the fourth variable's coefficient keeps growing and it eventually completely replaces the first variable added to the model. This illustrates the fact that Lasso may mistakenly add variables early on in the regularization path (Su et al. 2017). A similar situation occurs when the fifth feature, concerning the age of women at birth of a third child is added, as it is closely related to the third variable.

To summarize, the example demonstrates that the proposed method is able to quickly construct interpretable models. We believe, that we correctly identified the most important factors influencing infant mortality rate such as economic development

level and age of mothers at childbirth. We were able to find plausible explanations in literature for all the first five variables included in the model. Moreover, the signs of their coefficients agree with intuitive interpretations. For this example, model construction took just under 3 seconds.

5 Conclusion

We introduced a new screening technique for the homotopy based Lasso path construction. Our criterion is different from other currently available approaches, as it allows for explicit selection of the next variable on the regularization path without the need for specifying the value of the regularization parameter λ. We also assumed that the data used to build the model contains a huge number of fixed predictors which can be reused for multiple analyses. As a result we proposed to use a multidimensional index to speed up model construction. We have rigorously proven the correctness of our screening criterion, when an exact multidimensional index is used.

Experiments on genetic data which fits in main memory confirmed that our criterion is competitive with current state of the art, even though we provide a full and exact regularization path, unlike other methods which only approximate the path at selected values of the regularization parameter. Our analyses have shown that even very dense sampling will not fully recover the path. The approach also proved effective for large datasets not fitting in main memory, where an approximate multidimensional index was used to speed up model construction. The proposed algorithm again outperformed other available screening methods in terms of computation time, screening efficiency and predictive accuracy.

Finally, we have shown that the Lasso method is capable of efficiently constructing interpretable models even in the presence of millions of highly correlated variables. In the paper, we have created a model predicting infant mortality rate in various countries and confirmed the selected variables based on relevant literature.

Future work will involve extending the method to classification problems such as logistic regression. The challenge is that in this case the Lasso path is no longer piecewise linear and several formulas on which we relied for linear regression no longer hold.

Appendix 1: Proofs of theorems

Proof of Theorem 1 Let us first prove that u and s are normalized. It is easy to see that is suffices to show that the sums of their coordinates are zero. Since u is a vector of OLS residuals of y on X_J, and the sum of OLS residuals is equal to zero (Rao 2009), u is normalized. Since the columns of X_J are assumed to be normalized, we have $(1, \ldots, 1)s = \frac{1}{d_s}(1, \ldots, 1)X_J(X_J^T X_J)^{-1}t_J = 0$ implying s is also normalized.

For a vector w, let $w_J = (w_j : j \in J)$ be its projection on J. First, let us notice that we can substitute $Xw^\star(\lambda)$ with $X_J w_J^\star(\lambda)$, because the remaining coordinates of $w^\star$ are equal to 0. The expression $|x^T(y - Xw^\star(\lambda))|$ can then be rewritten as

$$\begin{aligned} &|x^T(y - Xw^\star(\lambda))| \\ &= |x^T(y - X_J(X_J^T X_J)^{-1}(X_J^T y - \lambda t_J))| \\ &= |x^T(y - X_J(X_J^T X_J)^{-1} X_J^T y + \lambda X_J(X_J^T X_J)^{-1} t_J)| \\ &= |x^T(d_u u + \lambda d_s s)| = |d_u \mathrm{cor}(x,u) + \lambda d_s \mathrm{cor}(x,s)|. \end{aligned} \tag{A1}$$

The first equality follows from an explicit expression for $w^\star(\lambda)$ given in (Bach et al. 2012, Chapter 6.2) and the last from the fact that the vectors x, u, s are normalized. To see that s and u are orthogonal, notice that

$$\begin{aligned} (d_u d_s) s^T u &= t_J^T (X_J^T X_J)^{-1} X_J^T (I - X_J (X_J^T X_J)^{-1} X_J^T) y \\ &= t_J^T (X_J^T X_J)^{-1} X_J^T y - t_J^T (X_J^T X_J)^{-1} X_J^T X_J (X_J^T X_J)^{-1} X_J^T y = 0. \end{aligned}$$

Notice further that d_u is the l_2 norm of a residual vector of an OLS model of y on X_J, and since $\|y\|_2 = 1$ and the norm of the residual vector cannot be larger than that of y, $d_u \leqslant 1$. Further

$$\begin{aligned} d_s^2 &= (X_J (X_J^T X_J)^{-1} t_J)^T X_J (X_J^T X_J)^{-1} t_J \\ &= t_J^T (X_J^T X_J)^{-1} t_J. \end{aligned}$$

Due to the normalization of the columns of X_J, we can observe that $X_J^T X_J$ has 1's on its diagonal

$$(X_J^T X_J)_{ii} = \sum_{k=1}^{n} (X_J)_{ki}^2 = 1,$$

and since the elements of the matrix $X_J^T X_J$ are correlations between normalized variables, all its elements have absolute value not greater than 1.

The Gershgorin theorem (see e.g. Horn and Johnson (2012)) states that each eigenvalue of a symmetric matrix $A = (a_{ij})$ belongs to at least one Gershgorin disc centered at a_{ii} with radius equal to $R_i = \sum_{i \neq j} |a_{ij}|$. If we take $A = X_J^T X_J$ then $R_i \leqslant |J| - 1$, and the largest eigenvalue of $X_J^T X_J$ cannot be greater than $|J|$. Consequently, the smallest eigenvalue of $(X_J^T X_J)^{-1}$ is not smaller than $\frac{1}{|J|}$.

Denote with μ_i, v_i the eigenvalues and eigenvectors of $(X_J^T X_J)^{-1}$, so $(X_J^T X_J)^{-1} v_i = \mu_i v_i$. Since v_i's can be chosen to form an orthonormal basis, we can write $t_J = \sum_i \alpha_i v_i$ for some constants α_i yielding

$$
\begin{aligned}
t_J^T (X_J^T X_J)^{-1} t_J &= t_J^T (X_J^T X_J)^{-1} \sum_i \alpha_i v_i \\
&= t_J^T \sum_i \alpha_i \mu_i v_i = (\sum_i \alpha_i v_i)^T (\sum_j \alpha_j \mu_j v_j) \\
&= \sum_i \alpha_i^2 \mu_i \geqslant (\min_j \mu_j) \sum_i \alpha_i^2 \\
&= (\min_j \mu_j) t_J^T t_J \geqslant \frac{1}{|J|} |J| = 1,
\end{aligned}
$$

due to $v_i v_j = \delta_{ij}$ (the Kronecker symbol) and $\sum_i \alpha_i^2 = t_J^T t_J = |J|$.

Finally, for $|J| = 1$: $d_s = \|X_J (X_J^T X_J)^{-1} t_J\| = \|X_J t_J\| = \|X_J\| = 1$, since X_J consists of a single normalized column. □

Proof of Theorem 2 The fact is a simple consequence of the optimality conditions for the Lasso (see e.g. subsec. 1.4 in Bach et al. (2012)), which state that for any variable x not in the active set

$$|x^T (y - X w^\star(\lambda))| < \lambda.$$

□

Proof of Theorem 3 It is easy to see that the vector $u^\star$ is normalized. Also, since $\theta_0 \in (0, \frac{\pi}{2})$, we have $\sin\theta_0, \cos\theta_0 > 0$.

Let us first compute the correlation of $u^\star$ with the candidate variable x_0 based on which it was computed. Assume first that $\text{cor}(x_0, s) \geqslant -\frac{d_u}{\lambda_0 d_s} \text{cor}(x_0, u)$ such that we can drop the absolute value in (5). We have

$$
\begin{aligned}
\text{cor}(x_0, u^\star) &= \cos\theta_0 \text{cor}(x_0, u) + \sin\theta_0 \text{cor}(x_0, s) \\
&= \cos\theta_0 \text{cor}(x_0, u) - \sin\theta_0 \frac{d_u}{\lambda_0 d_s} \text{cor}(x_0, u) + \frac{\sin\theta_0}{d_s} \\
&= \cos\theta_0 \text{cor}(x_0, u) - \cos\theta_0 \text{cor}(x_0, u) + \frac{\sin\theta_0}{d_s} = \frac{\sin\theta_0}{d_s}.
\end{aligned}
\tag{A2}
$$

The first equality follows from the fact that x_0, u, s and $u^\star$ are all normalized (correlation becomes identical to dot product), the second from (5), and the third from the fact that, by definition, $\tan\theta_0 = \frac{\lambda_0 d_s}{d_u}$, so $\cos\theta_0 = \frac{d_u}{\lambda_0 d_s} \sin\theta_0$. When $\text{cor}(x_0, s) < -\frac{d_u}{\lambda_0 d_s} \text{cor}(x_0, u)$ an analogous reasoning leads to $\text{cor}(x_0, u^\star) = -\frac{\sin\theta_0}{d_s}$ and thus to

$$|\text{cor}(x_0, u^\star)| = \frac{\sin\theta_0}{d_s}. \tag{A3}$$

Let us now consider the case when $\text{cor}(x, s) > -\frac{d_u}{\lambda d_s} \text{cor}(x, u)$ (calculations are analogous for $\text{cor}(x, s) < -\frac{d_u}{\lambda_0 d_s} \text{cor}(x, u)$). In this case (5) implies

$$\operatorname{cor}(x,s) = -\frac{d_u}{\lambda d_s}\operatorname{cor}(x,u) + \frac{1}{d_s}. \tag{A4}$$

From (A4) and Theorem 2 it follows that

$$\begin{aligned} d_u\operatorname{cor}(x,u) + \lambda_k d_s \operatorname{cor}(x,s) &= d_u\operatorname{cor}(x,u) + \lambda_k d_s\left[-\frac{d_u}{\lambda d_s}\operatorname{cor}(x,u) + \frac{1}{d_s}\right] \\ &= d_u\operatorname{cor}(x,u)\left(1 - \frac{\lambda_k}{\lambda}\right) + \lambda_k < \lambda_k. \end{aligned}$$

The condition $\lambda < \lambda_k$ now implies $1 - \frac{\lambda_k}{\lambda} < 0$ and

$$\operatorname{cor}(x,u) > 0. \tag{A5}$$

Theorem 2 also implies

$$\operatorname{cor}(x,s) \geqslant -\frac{d_u}{\lambda_k d_s}\operatorname{cor}(x,u) - \frac{1}{d_s}, \tag{A6}$$

and (by an argument similar to the proof for $\operatorname{cor}(x_0,u^\star)$ above)

$$\begin{aligned} \operatorname{cor}(x,u^\star) &= \cos\theta_0\operatorname{cor}(x,u) + \sin\theta_0\operatorname{cor}(x,s) \\ &\geqslant \cos\theta_0\operatorname{cor}(x,u) + \sin\theta_0\left[-\frac{d_u}{\lambda_k d_s}\operatorname{cor}(x,u) - \frac{1}{d_s}\right] \\ &= \cos\theta_0\operatorname{cor}(x,u)\left(1 - \frac{\lambda_0}{\lambda_k}\right) - |\operatorname{cor}(x_0,u^\star)|, \end{aligned}$$

where the last equality follows from (A3). Now $\lambda_0 < \lambda_k$ implies $1 - \frac{\lambda_0}{\lambda_k} > 0$, which, together with (A5) implies

$$\operatorname{cor}(x,u^\star) \geqslant -|\operatorname{cor}(x_0,u^\star)|. \tag{A7}$$

Another bound on the correlation of $u^\star$ with x can be obtained as follows:

$$\begin{aligned} \operatorname{cor}(x,u^\star) &= \cos\theta_0\operatorname{cor}(x,u) + \sin\theta_0\operatorname{cor}(x,s) \\ &= \cos\theta_0\operatorname{cor}(x,u) - \sin\theta_0\frac{d_u}{\lambda d_s}\operatorname{cor}(x,u) + \frac{\sin\theta_0}{d_s} \\ &= \operatorname{cor}(x,u)\left(\cos\theta_0 - \sin\theta_0\frac{d_u}{\lambda d_s}\right) + \frac{\sin\theta_0}{d_s} \\ &= \operatorname{cor}(x,u)\sin\theta_0\frac{d_u}{d_s}\left(\frac{1}{\lambda_0} - \frac{1}{\lambda}\right) + |\operatorname{cor}(x_0,u^\star)|, \end{aligned}$$

where the second equality follows from (5). Now, (A5) implies the equivalence

$$\lambda > \lambda_0 \quad \Leftrightarrow \quad \operatorname{cor}(x,u^\star) > |\operatorname{cor}(x_0,u^\star)|,$$

which combined with (A7) yields the desired result. □

Acknowledgements The authors did not receive support from any organization for the submitted work.

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose.

Human or animal rights The research only used publicly available data, no human participants and/or animals were involved.

References

Andoni A, Indyk P, Laarhoven T, Razenshteyn I, Schmidt L (2015) Practical and optimal lsh for angular distance. In: NIPS

Aumüller M, Bernhardsson E, Faithfull A (2020a) ANN-benchmarks: a benchmarking tool for approximate nearest neighbor algorithms. Inf Syst 87:101374

Aumüller M, Bernhardsson E, Faithfull A (2020b) ANN-Benchmarks. http://ann-benchmarks.com. Accessed 12 Feb 2020

Babenko A, Lempitsky V (2014) The inverted multi-index. IEEE Trans Pattern Anal Mach Intell 37(6):1247–1260

Bach F, Jenatton R, Mairal J, Obozinski G (2012) Optimization with sparsity-inducing penalties. Found Trends Machine Learn 4(1):1–106

Bentley JL (1975) Multidimensional binary search trees used for associative searching. Commun ACM 18(9):509–517

Bonnefoy A, Emiya V, Ralaivola L, Gribonval R (2014) A dynamic screening principle for the lasso. In: 2014 22nd European signal processing conference (EUSIPCO), pp 6–10

Bühlmann P, Van De Geer S (2011) Statistics for high-dimensional data: methods, theory and applications. Springer, New York

Chen J, Chen Z (2008) Extended Bayesian information criteria for model selection with large model spaces. Biometrika 95(3):759–771

Dai L, Pelckmans K (2012) An ellipsoid based, two-stage screening test for BPDN. In: 2012 proceedings of the 20th European signal processing conference (EUSIPCO), pp 654–658

Efron B, Hastie T, Johnstone I, Tibshirani R (2004) Least angle regression. Ann Stat 32(2):407–451

El Ghaoui L, Viallon V, Rabbani T (2010) Safe feature elimination in sparse supervised learning. Technical Report UC/EECS-2010-126, EECS Dept., UC Berkeley

Eurostat: Eurostat Database (2022). http://ec.europa.eu/eurostat. Accessed 28 Jan 2022

Fan J, Lv J (2008) Sure independence screening for ultrahigh dimensional feature space. J R Stat Soc Series B Stat Methodol 70:849–883

Fan J, Samworth R, Wu Y (2009) Ultrahigh dimensional feature selection: beyond the linear model. J Mach Learn Res 10:2013–2038

Fercoq O, Gramfort A, Salmon J (2015) Mind the duality gap: safer rules for the lasso. In: International conference on machine learning, pp 333–342

Finlay JE, Özaltin E, Canning D (2011) The association of maternal age with infant mortality, child anthropometric failure, diarrhoea and anaemia for first births: evidence from 55 low- and middle-income countries. BMJ Open 1(2):e000226

Guo R, Sun P, Lindgren E, Geng Q, Simcha D, Chern F, Kumar S (2020) Accelerating large-scale inference with anisotropic vector quantization. In: Proceedings of the 37th international conference on machine learning, ICML'20, pp 3887–3896
Heath T, Bizer C (2011) Linked data: evolving the web into a global data space. Synthesis lectures on the semantic web: theory and technology. Morgan & Claypool, San Rafael, CA. http://linkeddatabook.com/editions/1.0/
Horn RA, Johnson CR (2012) Matrix Analysis. Cambridge University Press, Cambridge, UK
Huang J, Ma S, Zhang C-H (2006) Adaptive lasso for sparse high-dimensional regression. Stat Sin 18:1603–1618
Indyk P, Motwani R (1998) Approximate nearest neighbors: Towards removing the curse of dimensionality. In: Proceedings of the thirtieth annual ACM symposium on theory of computing. STOC '98, pp 604–613, New York, NY, USA
Jégou H, Douze M, Johnson J, Hosseini L (2020) Faiss–a library for efficient similarity search and clustering of dense vectors. https://github.com/facebookresearch/faiss
Johnson J, Douze M, Jégou H (2017) Billion-scale similarity search with GPUs. arXiv preprint arXiv:1702.08734
Kong X-B, Liu Z, Yao Y, Zhou W (2017) Sure screening by ranking the canonical correlations. TEST 26(1):46–70
Lee S, Görnitz N, Xing EP, Heckerman D, Lippert C (2017) Ensembles of lasso screening rules. IEEE Trans Pattern Anal Mach Intell 40(12):2841–2852
Malkov YA, Yashunin DA (2020) Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. IEEE Trans Pattern Anal Mach Intell 42(4):824–836
McCrae JP, Abele A, Paul B, Cyganiak R, Jentzsch A, Andryushechkin V, Debattista J (2020) The Linked Open Data Cloud. https://cas.lod-cloud.net/. Accessed 8 Aug 2023
Mohebbi M, Vanderkam D, Kodysh J, Schonberger R, Choi H, Kumar S (2011) Google Correlate Whitepaper
Ndiaye E (2023) The Gsroptim package. https://github.com/EugeneNdiaye/Gap_Safe_Rules. Accessed 27 Apr 2023
Ndiaye E, Fercoq O, Gramfort A, Salmon J (2017) Gap safe screening rules for sparsity enforcing penalties. J Machine Learn Res 18(1):4671–4703
Omohundro SM (1989) Five balltree construction algorithms. Technical report, International Computer Science Institute, Berkeley
Pan X, Xu Y (2019) A safe reinforced feature screening strategy for lasso based on feasible solutions. Inf Sci 477:132–147
Paulheim H (2012) Generating possible interpretations for statistics from linked open data. In: 9th extended semantic web conference, pp 560–574
Paulheim H, Ristoski P, Mitichkin E, Bizer C (2014) Data mining with background knowledge from the web. In: 5th RapidMiner World Conference, pp 1–14
Pokarowski P, Mielniczuk J (2015) Combined l_1 and greedy l_0 penalized least squares for linear model selection. J Mach Learn Res 16(5):961–992
Rao CR (2009) Linear statistical inference and its applications. Wiley, Hoboken, New Jersey
Ristoski P (2015) Towards linked open data enabled data mining. In: European semantic web conference, pp 772–782
Ristoski P, Paulheim H (2013) Analyzing statistics with background knowledge from linked open data. In: Workshop on semantic statistics
Scheetz TE, Kim K-YA, Swiderski RE, Philp AR, Braun TA, Knudtson KL, Dorrance AM, DiBona GF, Huang J, Casavant TL et al (2006) Regulation of gene expression in the mammalian eye and its relevance to eye disease. Proc Natl Acad Sci 103(39):14429–14434
Sims M, Sims TL, Bruce MA (2007) Urban poverty and infant mortality rate disparities. J Natl Med Assoc 99:349–56
Su W, Bogdan M, Candes E (2017) False discoveries occur early on the lasso path. Ann Stat 45(5):2133–2150
The ANNOY library (2023) Retrieved on 8 August 2023. https://github.com/spotify/annoy
Tibshirani R (1996) Regression shrinkage and selection via the Lasso. J R Stat Soc Series B Methodol 58(1):267–288
Tibshirani R, Bien J, Friedman J, Hastie T, Simon N, Taylor J, Tibshirani RJ (2012) Strong rules for discarding predictors in lasso-type problems. J R Stat Soc Series B (Stat Methodol) 74(2):245–266

Wang J, Zhou J, Wonka P, Ye J (2013) Lasso screening rules via dual polytope projection. In: Advances in neural information processing systems, pp 1070–1078

Xiang ZJ, Ramadge PJ (2012) Fast lasso screening tests based on correlations. In: 2012 IEEE international conference on acoustics, speech and signal processing (ICASSP), pp 2137–2140

Xiang ZJ, Xu H, Ramadge PJ (2011) Learning sparse representations of high dimensional data on large scale dictionaries. In: Advances in neural information processing systems, pp 900–908

Xiang ZJ, Wang Y, Ramadge PJ (2016) Screening tests for lasso problems. IEEE Trans Pattern Anal Mach Intell 39(5):1008–1027

Zheng X, Loh W-Y (1995) Consistent variable selection in linear models. J Am Stat Assoc 90(429):151–156

Żogała-Siudem B, Jaroszewicz S (2020) Fast stepwise regression based on multidimensional indexes. Inf Sci 549:288–309