Solar: a least-angle regression of stable and ultrafast variable selection for high dimensional and large-scale data

Ning Xu
Timothy C.G. Fisher
and
Jian Hong
School of Economics, University of Sydney
NSW, Australia, 2006

December 28, 2021

Abstract

We propose a new algorithm for variable selection for high-dimensional and large-scale data, called *subsample-ordered least-angle regression (solar)*, and its coordinate-descent generalization. Relying on averaging and re-ordering lasso paths via L_0 norm, solar can be straightforwardly modified to replace most lasso variants. Solar alleviates several high-dimensional issues with lasso, strong/safe rule, variable screening and subsampling variable selection. We illustrate in simulations and real-world data that, with the same computation load, solar yields substantial improvements over lasso in terms of the sparsity (37-64% reduction in the average number of selected variables), stability and accuracy of variable selection. Our examples and real-world data also show that, compared with the lasso strong/safe rule and variable screening, solar largely rectifies incorrect variable purge and sparsity loss at complicated dependence structures and harsh settings of the irrepresentable condition. Moreover, solar supplemented with the hold-out average test (a new adaptation of the data-splitting hypothesis test) drastically improves the efficiency and accuracy of post-selection inference. Most importantly, replacing lasso with solar in a subsampling variable selection algorithm (e.g., the bootstrap lasso or stability selection) features a novel and powerful *multi-layer variable ranking scheme*, reuslting in sparsity improvement and huge computation load reduction (at least 96% fewer runtime) compared to bootstrap lasso or stability selection. Using Python interface and Intel Math Kernel Library, We provide a Fortran/C++ based parallel computing package for solar (`solarpy`) at both the supplementary file and dedicated Github page.

Keywords: sparsity, dependence structure, irrepresentable condition, lasso, bolasso, stability selection, variable ranking, runtime

1 Introduction

In the last decade, lasso-related algorithms have been widely applied to high dimensional, large-scale applications (Efron et al., 2004; Friedman et al., 2007, 2010). To address the issue of selection sparisty, stability and accuracy (Weisberg, 2004; Lim and Yu, 2016), various improvements have been proposed for lasso. As one of the most effective, subsampling variable selection (e.g., Bach (2008); Meinshausen and Bühlmann (2010)) have been proposed to improve the variable-selection accuracy and stability of lasso. Given that optimization of the lasso-type tuning parameter is often combined with cross validation (CV), merging CV-lasso with subsampling hugely magnifies computation load, making the subsampling-based lasso variants particularly less attractive in large-scale applications like computer vision and natural language processing (typically with p at millions and data at GBs) (Xu et al., 2012). Thus, in a world of ever-expanding data size, it would be ideal to accompany the performance improvement of variable selection with significant computation load reduction.

To offer improvements without exponentially increasing the computation runtime, post-lasso selection rules are proposed (e.g., the 'safe rule' (Ghaoui et al., 2010) and the 'strong rule' (Tibshirani et al., 2012)). However, both rules may incorrectly purge informative variables or repeatedly propose modifications (Wang et al., 2014; Zeng et al., 2017). From the pespective of hypothesis testing, Wasserman and Roeder (2009), Meinshausen et al. (2009) and Barber and Candès (2019) propose the data split scheme to conduct the classical hypothesis tests after subsample selection. The orginal data is split, reserving one part for selection and the other for testing. However, to improve the power of testing, such data-splitting scheme need to be repeated on each boostrap sample, suffering the similar computational issue as do (Bach, 2008; Meinshausen and Bühlmann, 2010). Moreover, such method depends on the split scheme and may result in efficiency and power loss due to testing and selection on only a subset of the data. Even worse, DiCiccio et al. (2020) shows that, the traditional multiple data-splitting method may eventually worsen the power and efficiency single splitting.

1.1 Main results

To address issues above, we propose a new algorithm for variable selection for high-dimensional and large-scale data, called *subsample-ordered least-angle regression (solar)*, and its coordinate-descent generalization. Relying on averaging and re-ordering lasso paths via L_0 norm, solar can be straightforwardly modified to replace most lasso variants. With the same computation

load as lasso, solar alleviates high-dimensional issues above with lasso, strong/safe rule, variable screening and subsampling variable selection.

Using simulations and examples, we demonstrate the following attributes of solar: (1) when p/n approaches 0 or 1, with the same computation load as lasso, solar is more responsive to changes in p/n, yielding significant improvements in variable-selection performance over lasso; (2) when n and p both increase rapidly in high-dimensional space, solar significantly outperforms lasso in terms of convergence tendency, sparsity (a 37-64% reduction in the average number of selected variables), stability and accuracy of variable selection; (3) in our simulation, supplementing solar with the hold-out average test (a new adaptation of the data-splitting test) drastically improves the efficiency and accuracy of post-selection inference; (4) compared with safe/strong rule and variable screening, our examples show that solar is robust to non-standard dependence structures in regression analysis, implying better variable-selection accuracy under different settings of the IRC; (5) replacing lasso with solar in subsampling variable selection algorithms features a novel *multi-layer variable ranking scheme*, resulting in a significant computation load reduction (at least 96% fewer computation runtime) with improvements on selection sparsity and accuracy.

The paper is organized as follows. In section 2, we introduce solar and its coordinate descent generalization. In section 3 and 4.6, we demonstrate solar's improvements over lasso, strong/safe rule, variable screening, forward regression and the lasso-based, subsampling variable selection using examples and a comprehensive set of simulations. Using real-world data with strong multicollinearity and complicated dependency structures, we show in section 5 that improvements from solar are feasible while lasso and elastic net lose sparsity.

2 Solar algorithm

The solar improvements over lasso and subsampling variable selection are due to a reallocation of computation. Lasso and its variants allocate most of its computation to optimizing the shrinkage parameter t and λ using cross-validation. By contrast, with the same computation load, solar allocates computation to stabilizing the solution path, which is key to the variable selection procedure. Zhang (2009, Theorem 2) implies that the earlier a variable enters the solution path, the more likely it is informative. As a result, a stable and accurate ordering of variables in the solution path may help identify informative variables. Since we focus on variable selection accuracy, the only relevant feature of the regression coefficients in the solution path

is whether $\beta_i = 0$ at each stage. Thus, we reparametrize the lasso path via L_0 norm.

Definition 2.1 (L_0 solution path)**.** Define the L_0 **solution path** of any forward regression on (Y, X) to be the order that forward regression includes variables across all stages. For example, if the least angle regression includes $\mathbf{x}_3$ at stage 1, $\mathbf{x}_2$ at stage 2 and $\mathbf{x}_1$ at stage 3, the corresponding L_0 path is the ordered set $\{\{\mathbf{x}_3\}, \{\mathbf{x}_3, \mathbf{x}_2\}, \{\mathbf{x}_3, \mathbf{x}_2, \mathbf{x}_1\}\}$.

2.1 Solar solved by least angle regression

The L_0 solution path is the foundation of forward regression. To stabilize least-angle regression in high-dimensional spaces, we first reduce the sensitivity of its solution path when $p > n$. The *average L_0 solution path estimation* algorithm (summarized in Algorithm 1 and illustrated in Figure 1) accomplishes this task by estimating the *average stage $\mathbf{x}_i$ enters the solution path* of least-angle regression.

Algorithm 1: average L_0 path estimation via least angle regression

input : (Y, X).

1 generate K subsamples $\left\{\left(Y^k, X^k\right)\right\}_{k=1}^{K}$ by randomly remove $1/K$ of observations in (Y, X);

2 set $\widetilde{p} = \min\{n_{\text{sub}}, p\}$;

3 **for** *k := 1 to K, stepsize = 1* **do**

4 run an unrestricted least-angle regression on $\left(Y^k, X^k\right)$ and record the order of variable inclusion at each stage;

5 define $\widehat{q}^k = \mathbf{0} \in \mathbb{R}^p$;

6 for all i and l, if $\mathbf{x}_i$ is included at stage l, set $\widehat{q}_i^k = (\widetilde{p} + 1 - l)/\widetilde{p}$, where $\widehat{q}_i^k$ is the i^{th} entry of $\widehat{q}^k$;

7 **end**

8 $\widehat{q} := \frac{1}{K}\sum_{k=1}^{K} \widehat{q}^k$;

9 **return** $\widehat{q}$

After subsampling , the second part of Algorithm 1 (lines 4-6) computes $\widehat{q}^k$, which summarizes the order that least angle regression includes each $\mathbf{x}_i$ across all stages (as in the Figure 1 example). The unrestricted least-angle regression only ranks variables by the stage $\mathbf{x}_i$ enters the solution path. As shown in line 6 of Algorithm 1 and Figure 1, variables included at earlier stages have larger corresponding $\widehat{q}_i^k$ values: the first variable included is assigned 1, the last is assigned $1/\widetilde{p}$ and the dropped variables are assigned 0 (which occurs only when $p > n$).

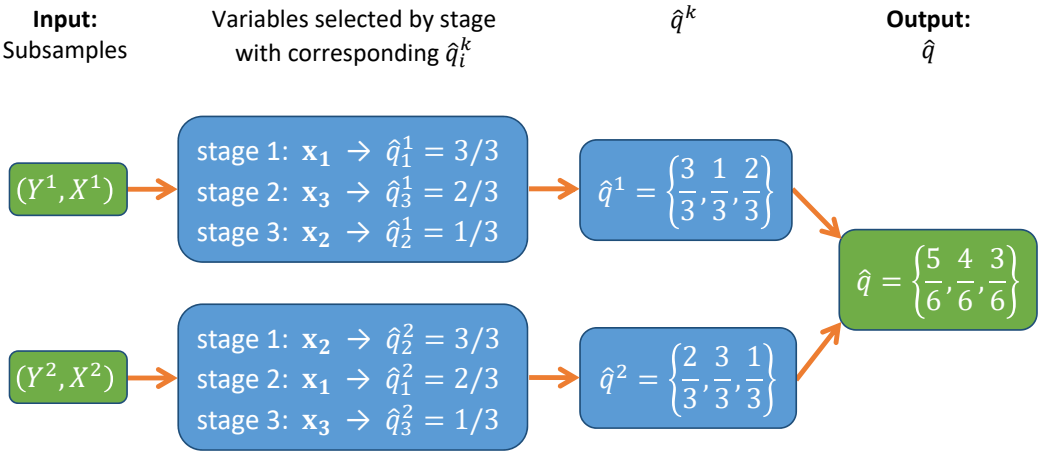


Figure 1: Computation of $\widehat{q}$ on 2 subsamples where $\{\mathbf{x}_1, \mathbf{x}_2\}$ are informative and $\mathbf{x}_3$ redundant.

Thus, by ranking (decreasingly) the $\mathbf{x}_i$ according to their corresponding $\widehat{q}_i^k$ values, we obtain the L_0 solution path. The Zhang (2009, Theorem 2) result implies that the variables with the largest $\widehat{q}_i^k$ values, on average, are more likely to be informative variables. The $\widehat{q}_i^k$ may be sensitive in high-dimensional spaces to multicollinearity, sampling randomness and noise. A consequence is that a redundant variable may be included at an early stage in some subsample $\left(Y^k, X^k\right)$. Hence, Algorithm 1 reduces the impact of sensitivity in the $\widehat{q}_i^k$ values by computing $\widehat{q} := \frac{1}{K}\sum_{k=1}^{K} \widehat{q}^k$ and ranking the $\mathbf{x}_i$ (decreasingly) based on the corresponding value of $\widehat{q}_i$ (the i^{th} entry of $\widehat{q}$), to get the average L_0 solution path. The average L_0 solution path is formally defined as follows.

Definition 2.2 (average L_0 solution path)**.** Define the **average L_0 solution path** of forward regression on $\left\{\left(Y^k, X^k\right)\right\}_{k=1}^{K}$ to be the decreasing rank order of variables based on their corresponding $\widehat{q}_i$ values. For example, in Figure 1, the $\widehat{q}_i$ values for $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$ are, respectively, $\widehat{q}_1 = 5/6$, $\widehat{q}_2 = 4/6$ and $\widehat{q}_3 = 3/6$. As a result, the average L_0 solution path can be represented as an ordered set $\{\{\mathbf{x}_1\}, \{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}\}$.

Built on Algorithm 1, the solar algorithm is summarized in Algorithm 2. Instead of the equiangular, partial-correlation search in least-angle regression, variables are included into forward regression according to their rank order in the average L_0 solution path, represented by $\{Q(c)|c = 1, 0.98, \ldots, 0\}$ in Algorithm 2. We use $\widehat{q}$ from Algorithm 1 to generate a list of variables $Q(c) = \{\mathbf{x}_j \mid \widehat{q}_j \geqslant c, \forall j \leqslant p\}$. For any $c_1 > c_2$, $Q(c_1) \subset Q(c_2)$, implying a sequence of nested sets $\{Q(c)|c = 1, 0.98, \ldots, 0\}$. Each c denotes a stage of forward or least-angle regression. For a given value of c, $Q(c)$ denotes the set of variables with $\|\beta_i\|_0 = 1$ on average

Algorithm 2: Subsample-ordered least-angle regression

1 Randomly select 20% of the sample points as the validation set; denote the remaining points as the training set;

2 Estimate $\widehat{q}$ using Algorithm 1 on the training set and compute $Q(c) = \{\mathbf{x}_j \mid \widehat{q}_j \geqslant c, \forall j\}$ for all $c \in \{1, 0.98, \ldots, 0.02, 0\}$.

3 Run an OLS regression of each $Q(c)$ to Y on the training set and find c^*, the value of c that minimizes the validation error;

4 Compute the OLS coefficients of $Q(c^*)$ to Y on the whole sample.

and $Q(c) - Q(c - 0.02)$ is the set of variables with $\|\beta_i\|_0$ just turning to 1 at c. Therefore, $\{Q(c)|c = 1, 0.98, \ldots, 0\}$ is the average L_0 solution path of Definition 2.2. Variables that are more likely to be informative are included at $Q(c)$ with larger value of c, which will be selected first by the solar algorithm.

2.2 Solar solved by coordinate descent

The solar algorithm can be easily generalized to coordinate descent scheme. Both least-angle regression and coordinate descent generate a solution path for lasso, parametrized with β_i on vertical axis and tuning parameter t or λ on horizontal axis. Hence, to reprogram solar using coordinate descent, we just need to replace Algorithm 1 with Algorithm 3, which records the order of variable activation along the coordinate descent solution path.

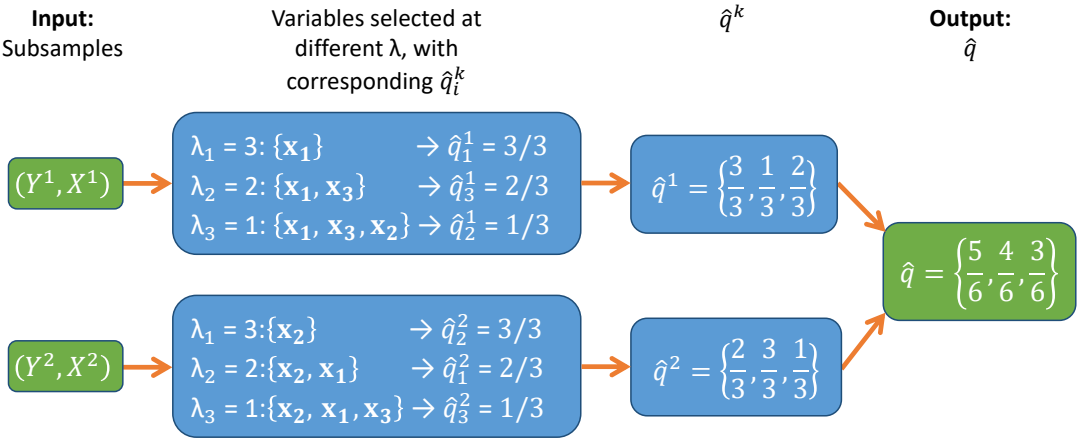


Figure 2: Computation of $\widehat{q}$ on 2 subsamples using coordinate descent.

Compared to Algorithm 1, Algorithm 3 use λ to record the order that each variale enters the

Algorithm 3: average L_0 path estimation via coordinate descent

input : (Y, X).

generate K subsamples $\left\{\left(Y^k, X^k\right)\right\}_{k=1}^K$ by randomly remove $1/K$ of observations in (Y, X);

set $\widetilde{p} = \min\{n_{\text{sub}}, p\}$;

for *k := 1 to K, stepsize = 1* **do**

  denote λ_s as the λ value that coordinate descent lasso includes s variables, $\forall s \in (0, \widetilde{p}]$;

  run a coordinate descent lasso on $\left(Y^k, X^k\right)$, $\forall \lambda \in \left\{\lambda_1, \ldots, \lambda_{\widetilde{p}}, \right\}$

  record the order of variable inclusion at each $\lambda \in \left\{\lambda_1, \ldots, \lambda_{\widetilde{p}}, \right\}$;

  define $\widehat{q}^k = \mathbf{0} \in \mathbb{R}^p$;

  for all i and s, if $\mathbf{x}_i$ is included at $\lambda = \lambda_s$, set $\widehat{q}_i^k = (\widetilde{p} + 1 - s)/\widetilde{p}$, where $\widehat{q}_i^k$ is the i^{th} entry of $\widehat{q}^k$;

end

$\widehat{q} := \frac{1}{K} \sum_{k=1}^K \widehat{q}^k$;

return $\widehat{q}$

path. Let's take Figure 2 as an example. To reparametrize the solution path, we denote λ_s as the λ value that coordinate descent lasso includes s variables, $\forall s \in (0, \min\{n/2, p\}]$. Hence, we define a sequence of λ for grid search. In each subsample $\left(Y^k, X^k\right)$, we train a standard coordinate descent lasso, allowing λ value to stepwise increase within the grid $\left\{\lambda_1, \ldots, \lambda_{\min\{n/2,p\}}\right\}$, where $\lambda_1 \geqslant \ldots \geqslant \lambda_{\min\{n/2,p\}}$. In Figure 2, when $\lambda < \lambda_3$ at subsample (Y^1, X^1), all three variables are activated in the solution path, implying that $\widehat{q}_i^1 \geqslant 1/3$ for all variables. When λ value increases to λ_2, only $\{\mathbf{x}_3, \mathbf{x}_1\}$ survive the harsher shrinkage, implying that they should be ranked higher than $\mathbf{x}_2$. As a result, $\widehat{q}_1^1, \widehat{q}_3^1 \geqslant 2/3$ and $\widehat{q}_2^1 = 1/3$. When λ value hits λ_3, only $\{\mathbf{x}_1\}$ remains activated, leaving $\widehat{q}_1^1 = 3/3$ and $\widehat{q}_3^1 = 2/3$. Afterwards, we can apply the same calculation method to each subsample, resulting in the same $\widehat{q}$ as Algorithm 1 does.

3 Solar comparison and improvements over lasso variants, lasso rules and variable screening

3.1 comparison and generaliztion to all lasso variants

As showns in previous algorithms, solar utilizes most of its computation to averages and reranks the lasso paths using the corresponding L_0 norm. Compared with lasso, solar has the following

unique features.

- in terms of optimzation technique, solar abandons the L_p shrinkage parameter t and λ, calculating a new L_0 path by discretizing the original lasso path. Compared with L_1 path, the L_0 path is more concentrated on dimensionality rather than the parameter value, hence greatly boosting the variable selection performance (see simulation 1). More importantly, to avoid the NP-hard computaion of traditional L_0 method (e.g., best subset method via AIC/BIC), the L_0 path is computed from the L_1-penalized optimization, guaranteeing the ultra-fast speed even with large-scale and repeated learning tasks (see simulation 3).

- since abandon the L_p shrinkage parameter, solar no longer needs to optimize λ or t, relieving itself from the cross-validation conundrum. Even though unbiased and more generalizable than AIC/BIC, cross-validation comes at the cost of *bias-variance trade-off* (Kearns and Ron, 1999) and *stability issue* (Lim and Yu, 2016), motivating the research of lasso rules, subsampling variable selection and cross-validation variants for lasso (Xu et al., 2012). The problems above significantly affect the hyperparameter tuning, eventually affecting the lasso selection performance (sparisty, stability, etc). Without shrinkage parameter, solar only relies on a Jackknife- or Bootstrap-like algorithm — training for K rounds and removing $1/k$ of data each round — to stabilize the path. As shown in Bousquet and Elisseeff (2002), the elegant statistical properties of Bootstrap and Jackknife (such as stability, efficiecy and unbiasedness) may also be passed to solar.

Moreover, training via least-angle regression or coordinate descnet, solar can straightforwardly generalize its sparsity, efficiency and stability to almost all lasso-type problems. For example,

- 'grouped solar' could be invoked by requiring some variables to be simultaneously selected into the solution path;

- 'adaptive solar' could be obtained by weighting variable rankings in the avergae L_0 path according to OLS coefficients;

- 'solar elastic net' or 'fused solar' could be derived by replacing the loss function of coordinate descent at Algorithm 3 with the L_1-L_2 loss

$$\|Y - X\beta\|_2^2 + \lambda^{(1)} \|\beta\|_1 + \lambda^{(2)} \|\beta\|_2^2 \tag{3.1}$$

or fused loss

$$\|Y - X\beta\|_2^2 + \lambda^{(1)} \|\beta\|_1 + \lambda^{(2)} \sum_{j=2}^{p} |\beta_j - \beta_{j-1}|_1 \tag{3.2}$$

Most importantly, due to the same optimzation methods, most of the lasso improvement researchs (lasso rules, post-lasso hypothesis testing) can also be applied to the coordinate-descent/lars optimzation of solar. Instead of competing with lasso improvements, solar supplemented with them can further boost variable selection efficiency and performance in large-scale applications.

3.2 Solar advantages on post-selection hypothesis testing

Solar is amenable to post-selection testing. The lasso tests (Lockhart et al., 2014; Taylor et al., 2014) are based on least-angle regression, providing adaptions to solar. More interestingly, we can also easily modify the data-splitting tests (Wasserman and Roeder, 2009; Meinshausen et al., 2009). We illustrate this point using Example 1.

Example 1. Suppose $Y = 2\mathbf{x}_0 + 3\mathbf{x}_1 + 4\mathbf{x}_2 + 5\mathbf{x}_3 + 6\mathbf{x}_4 + e$, where all $\mathbf{x}_i$ and e are standard Gaussian with pairwise correlations of 0.5. Post-selection tests are typically applied with sufficiently large n, so $p/n = 50/100$. In this setting, forward regression (FR) selects $[\mathbf{x}_4, \mathbf{x}_3, \mathbf{x}_2, \mathbf{x}_1, \mathbf{x}_0, \mathbf{x}_{45}, \mathbf{x}_8, \mathbf{x}_{40}, \mathbf{x}_6, \mathbf{x}_{39}, \mathbf{x}_{11}, \mathbf{x}_{34}, \mathbf{x}_5]$ using BIC score minimization. By contrast, solar selects only $[\mathbf{x}_4, \mathbf{x}_3, \mathbf{x}_2, \mathbf{x}_1, \mathbf{x}_0, \mathbf{x}_{45}, \mathbf{x}_{40}]$. We then conduct post-FR and post-solar t-tests and compare the results to the OLS regression of Y on the $\mathbf{x}_i$.

After standardizing the $\mathbf{x}_i$ and $\mathbf{Y}$, the sample regression coefficient is identical to the corresponding conditional correlation. Both the t-test results and FR selection decisions are largely based on the sample regression coefficient absolute values; keeping variables with large values and purging the rest. Hence, the variables selected by FR also have low p-values on the current sample, implying that direct, post-FR t-tests overfit the sampling randomness that also affects the variable selection algorithm (referred to as 'p-value overfitting'). The key issue here is that there is no sample change between the FR sample and the post-FR test sample.

To solve this issue, we improve the data-splitting tests via Bousquet and Elisseeff (2002) method — a Jackknife-type algorithm — as follows: after solar selection on the whole sample, we randomly divide the sample into K folds, remove each fold in turn, and compute the t-values of post-solar OLS on the remaining $K-1$ folds. We calculate the average t valuess and se in K rounds and call the $K = 2$ version the *hold-out average*. The traditional Wasserman and Roeder

(2009); Meinshausen et al. (2009) methods only use a fraction of the sample for selection and the reminder for testing; by contrast, our method guarantees the whole sample for selection and the maximum sample utilization (via a Jackknife-type algorithm) for testing.

As shown in Table 1, the hold-out average t-values for $[\mathbf{x}_4, \mathbf{x}_3, \mathbf{x}_2, \mathbf{x}_1, \mathbf{x}_0]$ are very similar to their OLS counterparts while the hold-out average t-values for $[\mathbf{x}_{40}, \mathbf{x}_{45}]$ are lower, which leads to purging $\mathbf{x}_{40}$ and $\mathbf{x}_{45}$. We repeat the simulation 200 times and the results, summarized in Figure 3, confirm that improvements in the hold-out average are not simply due to chance. It is important to note that OLS uses all of the sample and has 50 residual degrees of freedom whereas the hold-out average uses only 50% of the sample in each round and consequently has 30-40 degrees of freedom. Thus, the hold-out average naturally produces slightly lower t-values for $[\mathbf{x}_4, \mathbf{x}_3, \mathbf{x}_2, \mathbf{x}_1, \mathbf{x}_0]$ and the t-value boxplots for the hold-out average are slightly less concentrated than the OLS boxplots. Simulations in Section 4.4, show that solar supplemented with the hold-out average purges almost all of the redundant variable while retaining all of the informative variables.

Table 1: Comparison of post-selection t tests between FR, hold-out average, and OLS.

	post-FR			hold-out average			classical OLS		
	se	*t*	$P > \|t\|$	*se*	*t*	$P > \|t\|$	*se*	*t*	$P > \|t\|$
$\mathbf{x}_0$	0.12	15.68	0	0.18	11.34	0	0.19	9.85	0
$\mathbf{x}_1$	0.14	20.06	0	0.22	12.76	0	0.23	12.05	0
$\mathbf{x}_2$	0.14	28.11	0	0.21	18.22	0	0.20	19.45	0
$\mathbf{x}_3$	0.13	39.19	0	0.20	24.48	0	0.20	25.08	0
$\mathbf{x}_4$	0.13	42.34	0	0.21	27.36	0	0.20	28.15	0
$\mathbf{x}_{40}$	0.12	2.59	0.01	0.19	1.90	0.18	0.18	2.36	0.02
$\mathbf{x}_{45}$	0.13	2.30	0.02	0.21	1.19	0.25	0.19	1.60	0.12

While the hold-out average procedure can be applied to other methods, solar has several intrinsic advantages. Firstly, to ensure accurate tests with only half the sample, it is important to retain residual degrees of freedom, implying that variable selection must be as sparse and accurate as possible. As shown in section 4.6 and 5, sparse and accurate selection is one of the benefits of solar. Secondly, compared to FR and lasso, the average L_0 path of solar is robust to settings of the irrepresentable condition, sampling noise, multicollinearity and other issues, which benefits post-selection tests when deciding which β_i to test. A larger K is needed in smaller samples for a larger residual degree of freedom. To conclude, this example illustrates the possibility that the hold-out average may significantly reduce the severity of 'p-value overfitting'

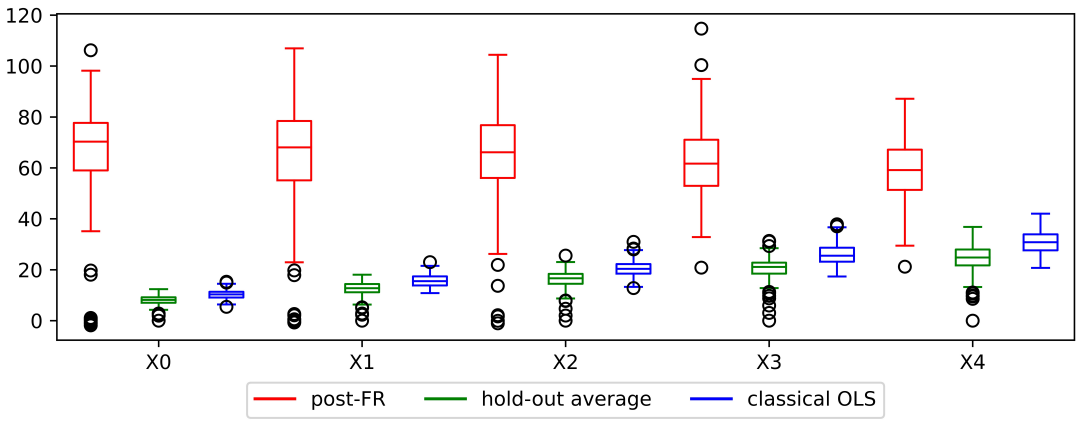


Figure 3: Boxplots of post-selection t values from FR, hold-out average, and OLS.

while utilizing as many data points as possible for selection and testing. This method helps reduce the severity of informative variable omission or redundant variable inclusion. ■

3.3 Solar advantages over lasso rules and variable screening under complicated dependence structure

Indeed, solar has another advantage: the average L_0 solution path offers additional robustness against outliers, multicollinearity, noise and outliers in high-dimensional spaces with complicated dependence structure. Thus, solar is likely to be more reliable than other variable selection methods under complicated dependence structures. We illustrate the point with the following two Bayesian network examples.

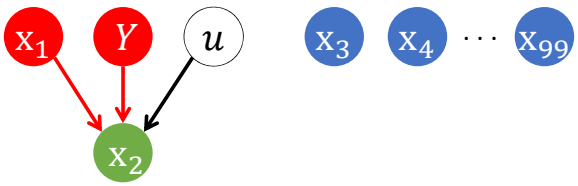


Figure 4: Y is unconditionally uncorrelated to informative $\mathbf{x}_1$.

Consider the non-standard regression case in Figure 4, where Y has two informative variables: a spouse ($\mathbf{x}_1$) and a child ($\mathbf{x}_2$). Figure 4 depicts a common scenario in empirical regression: *informative variables* may be *unconditionally uncorrelated to* Y in the DGP. With such data, Example 4 demonstrates that forward regression (including solar and lasso) is more reliable than post-lasso rules and variable screening.

Example 2. In Figure 4, there are 100 variables and $\mathbf{x}_2$ is (causally) generated by its parents $\{\mathbf{x}_1, Y\}$ as follows,

$$\mathbf{x}_2 = \alpha_1 \mathbf{x}_1 + \alpha_2 Y + u, \tag{3.3}$$

where $\mathbf{x}_1$ is unconditionally uncorrelated with Y, $\mathbf{x}_1$ and Y are both unconditionally and conditionally uncorrelated with any redundant variable, $\{\alpha_1, \alpha_2\}$ are population regression coefficients and u is a Gaussian noise term. If Y is chosen to be the response variable, we have the population regression equation

$$Y = -\frac{\alpha_1}{\alpha_2}\mathbf{x}_1 + \frac{1}{\alpha_2}\mathbf{x}_2 - \frac{1}{\alpha_2}u. \tag{3.4}$$

Note that $\mathbf{x}_1$ and $\mathbf{x}_2$ are both informative variables for Y. However, since $\mathbf{x}_1$ is unconditionally uncorrelated with Y in the population, some post-lasso rules (such as the strong rule (Tibshirani et al., 2012) and the safe rule (Ghaoui et al., 2010)) may be prone to purging $\mathbf{x}_1$ incorrectly. Given the value of shrinkage parameter λ in a grid search, the base strong rule and the safe rule for lasso purge any selected variable that satisfies, respectively, (3.5) and (3.6):

$$\left|\mathbf{x}_i^T Y\right| < \lambda - \|\mathbf{x}_i\|_2 \|Y\|_2 \frac{\lambda_{max} - \lambda}{\lambda_{max}}; \tag{3.5}$$

$$\left|\mathbf{x}_i^T Y\right| < 2\lambda - \lambda_{max}, \tag{3.6}$$

where the $\mathbf{x}_i$ are standardized and λ_{max} is the value of λ that purges all the variables. Both rules are based on the unconditional covariance between $\mathbf{x}_i$ and Y. For a given value of λ (typically selected by CV), lasso likely will select $\mathbf{x}_1$ and $\mathbf{x}_2$ along with some redundant variables from $\{\mathbf{x}_3, \ldots, \mathbf{x}_{99}\}$ (since the DGP does not violate any IRC). Since $\text{corr}(\mathbf{x}_1, Y) = \text{corr}(\mathbf{x}_3, Y) = \cdots = \text{corr}(\mathbf{x}_{99}, Y) = 0$ in the population, the sample value of $\left|\mathbf{x}_1^T Y\right|$ will be approximately as small as the $\left|\mathbf{x}_i^T Y\right|$ of any redundant variable. Put another way, $\mathbf{x}_1$ cannot be distinguished from the redundant variables by the value of $\left|\mathbf{x}_i^T Y\right|$. To ensure $\mathbf{x}_1$ is not purged by (3.5) or (3.6), both $\lambda - \|\mathbf{x}_1\|_2 \|Y\|_2 \frac{\lambda_{max} - \lambda}{\lambda_{max}}$ and $2\lambda - \lambda_{max}$ must be smaller than $\left|\mathbf{x}_1^T Y\right|$. However, this will lead to two problems. First, decreasing the right-hand side of (3.5) and (3.6) will reduce the value of λ, implying that lasso will select more redundant variables. Second, since $\left|\mathbf{x}_1^T Y\right|$ will be approximately as small as the $\left|\mathbf{x}_i^T Y\right|$ of any redundant variable selected by lasso, not purging $\mathbf{x}_1$ (by reducing both right-hand side terms) may result in (3.5) and (3.6) retaining redundant variables.

Variable screening methods (Fan and Lv, 2008) may also be prone to selecting redundant variables. Screening ranks variables decreasingly based on the absolute values of their unconditional correlations to Y, selecting the top w variables (with w selected by CV, bootstrap or

BIC). Since corr $(\mathbf{x}_2, Y) \neq 0$ in the population, screening will rank $\mathbf{x}_2$ highly. However, it may not rank $\mathbf{x}_1$ highly because corr $(\mathbf{x}_1, Y) = 0$ in the population. Thus, some redundant variables may be ranked between $\mathbf{x}_2$ and $\mathbf{x}_1$, implying that if both $\mathbf{x}_1$ and $\mathbf{x}_2$ are selected, screening will select redundant variables.

The average L_0 solution path will not suffer the same problem. For convenience, assume $-\alpha_1/\alpha_2 > 0$ and $p/n = 100/200$ or smaller. For lars, as we increase $\|\beta_2\|_1$ at stage 1 (i.e., as we 'partial' $\mathbf{x}_2$ out of Y), the unconditional correlation between $Y - \beta_2\mathbf{x}_2$ and $\mathbf{x}_1$ will increase above 0 significantly while the marginal correlation between $Y - \beta_2\mathbf{x}_2$ and any redundant variable will remain approximately 0. Thus, in the L_0 solution path and, hence, the average L_0 solution path, $\mathbf{x}_1$ will be included immediately after $\mathbf{x}_2$ is included. ■

Next example illustrates another common scenario in empirical regression: *redundant variables* may be *unconditionally correlated to* Y in the DGP. In this example, we choose specific values for β_1, β_2, ω and δ to demonstrate that, even when IRC is satisfied, the strong rule, base rule and variable screening methods may have difficulty purging the redundant $\mathbf{x}_3$, as shown in Figure 5. By contrast, solar is more reliable.

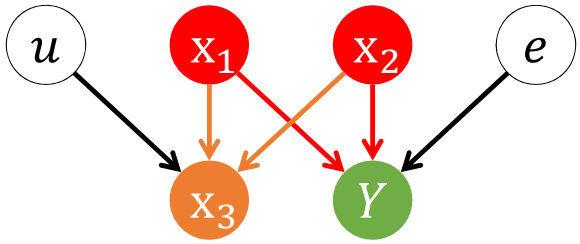


Figure 5: Y is unconditionally correlated with redundant $\mathbf{x}_3$.

Example 3. Consider the following confounding structure,

$$\begin{cases} \mathbf{x}_3 = \frac{1}{3}\mathbf{x}_1 + \frac{1}{3}\mathbf{x}_2 + \frac{\sqrt{7}}{3}u, \\ Y = \frac{7}{10}\mathbf{x}_1 + \frac{2}{10}\mathbf{x}_2 + \frac{\sqrt{47}}{10}e. \end{cases} \tag{3.7}$$

where $\mathbf{x}_1$ and $\mathbf{x}_2$ cause both Y and $\mathbf{x}_3$, implying that $\mathbf{x}_3$ is unconditionally correlated to Y; $\mathbf{x}_1$, $\mathbf{x}_2$, u and e are independent; $\mathbf{x}_3$ is independent from e; Y is independent from u; and all variables are standardized. When n is large and the sample correlations close to their population values,

the sample marginal correlations to Y can be ranked in decreasing order,

$$\begin{aligned} &\text{corr}\,(\mathbf{x}_1, Y) = 0.7 \\ &\text{corr}\,(\mathbf{x}_3, Y) = \text{corr}\left(\frac{1}{3}\mathbf{x}_1 + \frac{1}{3}\mathbf{x}_2, \frac{7}{10}\mathbf{x}_1 + \frac{2}{10}\mathbf{x}_2\right) = 0.3. \\ &\text{corr}\,(\mathbf{x}_2, Y) = 0.2 \end{aligned} \tag{3.8}$$

Because $\mathbf{x}_2$ ranks below $\mathbf{x}_1$ and $\mathbf{x}_3$ in terms of marginal correlations to Y, the variable screening method will have to select all 3 variables, including the redundant $\mathbf{x}_3$, to avoid omitting $\mathbf{x}_2$. Similarly, the base strong rule and safe rule may also have difficulty purging $\mathbf{x}_3$. Because $\text{corr}\,(\mathbf{x}_3, Y)$ is larger than $\text{corr}\,(\mathbf{x}_2, Y)$, if lasso selects both $\mathbf{x}_3$ and $\mathbf{x}_2$ and we use the base strong rule or the safe rule to purge $\mathbf{x}_3$, we will also have to purge $\mathbf{x}_2$.

Forward regression and lasso will not make the same errors. Because (3.7) does not violate the IRC, variable-selection consistency of forward regression and lars is assured by the theoretical results of Zhang (2009) and Zhao and Yu (2006). Specifically in forward regression, $\mathbf{x}_1$ will be included at the first stage. After controlling for $\mathbf{x}_1$, the partial correlations to Y of both $\mathbf{x}_2$ and $\mathbf{x}_3$ can be ranked in decreasing order as follows (when n is large),

$$\begin{aligned} &\text{corr}\,(\mathbf{x}_2, Y|\mathbf{x}_1) = \text{corr}\left(\mathbf{x}_2, \frac{2}{10}\mathbf{x}_2\right) = 0.2. \\ &\text{corr}\,(\mathbf{x}_3, Y|\mathbf{x}_1) = \text{corr}\left(\frac{1}{3}\mathbf{x}_1 + \frac{1}{3}\mathbf{x}_2, \frac{2}{10}\mathbf{x}_2\right) = 0.0667. \end{aligned} \tag{3.9}$$

Thus, at the second stage, forward regression will include $\mathbf{x}_2$, not $\mathbf{x}_3$. After controlling for both $\mathbf{x}_1$ and $\mathbf{x}_2$, the remaining variation in Y comes from e, which $\mathbf{x}_3$ cannot explain. As a result, CV or BIC will terminate forward regression after the second stage and $\mathbf{x}_3$ will not be selected. Similarly, because solar relies on the average L_0 path, it will include $\mathbf{x}_1$ and $\mathbf{x}_2$ but not $\mathbf{x}_3$. ■

Essentially, the strong rule, safe rule and variable screening struggle in these two examples because they rely on unconditional correlations to Y, whereas informative variables in regression analysis are defined in terms of conditional correlations. In many scenarios, unconditional and conditional correlations are aligned. However, when they are not, variable selection based conditional correlation is better placed to select informative variables.

Fan and Lv (2008) notice the Example 2 issue for variable screening and proposed two patches for Example 2 problem (Fan and Lv, 2008; Barut et al., 2016). However, Barut et al. (2016) relies on assumptions knowing the identity of $\mathbf{x}_1$ (also called *the collider* in Bayesian network or probablistic graph modelling) in Example 2 and Figure 4. Fan and Lv (2008) requires repeatedly running the following steps in a boosting-like ensemble: running a variable screening

on Y first, selecting variables with high unconditional correaltion with Y and running a lasso of Y on the dropped variables. By contrast, relying on the conditional correlation ranking, solar can efficiently fix such issue and finish variable selection under complicated dependence structure in only one realization; hence, solar avoids the hassle of repeatedly fitting lasso and variable screening to the residual of an GLM, implying more efficient and computationally affordable in large-scale applications. Moreover, the Fan and Lv (2008) patch cannot rectify the Example 3 issue. In their first step, all variables with high unconditional correlation to Y (including the redundant $\mathbf{x}_3$) will be selected; hence, the redundant $\mathbf{x}_3$ will never be dropped, implying a false variable inclusion. Such issue will be enormous if Y has multiple siblings like $\mathbf{x}_3$ in a complicated dependence structure, causing huge multicollinearity and biasing the regression coefficient and their standard error in finite sample. As a result, solar is more attractive in both computaional efficieny and elegancy of variable selection under complicated dependence structure.

3.4 Solar robustness to different settings of the IRC

Here we illustrate that, compared to lasso, solar has the superior robustness to different settings of the IRC. We skip lasso rules and variable screening here since, as discussed above, their selection accuracy may be affected by their reliance on unconditional correlation to Y. Following Zhang (2009), we define IRC as follows,

Definition 3.1 (IRC)**.** Given $F \subset \{1, \ldots, p\}$, define X_F to be the $n \times |F|$ matrix with only the full set of informative variables. Define

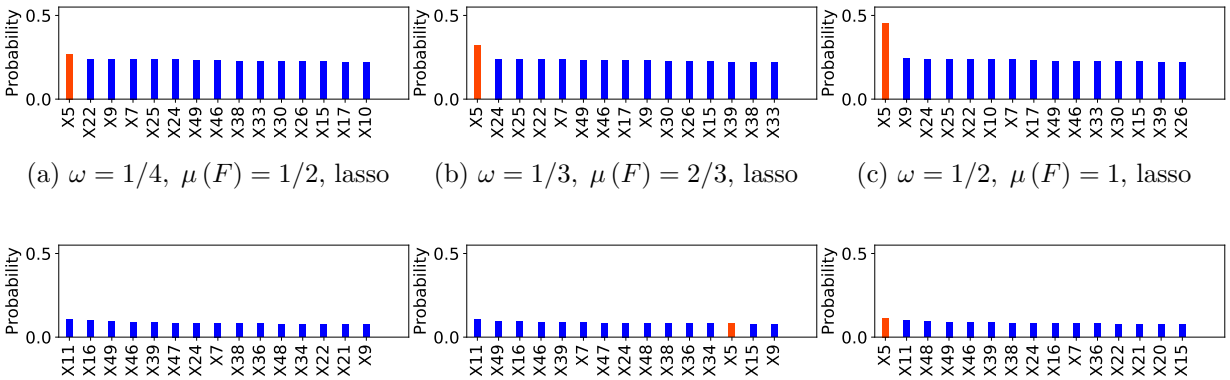
$$\mu(F) = \max \left\{ \left\| \left((X_F)^T X_F \right)^{-1} (X_F)^T \mathbf{x}_j \right\|_1 \mid \forall j \notin F \right\}.$$

Given a constant $1 \geqslant \eta > 0$, the strong irrepresentable condition is satisfied if $\mu(F) \leqslant 1 - \eta$ and the weak irrepresentable condition is satisfied if $\mu(F) < 1$.■

We follow the simulation of Zhao and Yu (2006) and slightly modify the Example 3 DGP. Here, $n = 200$, $p = 50$ and $[\mathbf{x}_0, \ldots, \mathbf{x}_4, \mathbf{x}_6, \ldots, \mathbf{x}_{50}]$ is generated from a zero-mean, unit-variance multivariate Gaussian distribution, where all the correlation coefficients are 0.5. The DGP of Y and $\mathbf{x}_5$ is

$$\begin{cases} \mathbf{x}_5 = \omega \mathbf{x}_0 + \omega \mathbf{x}_1 + \gamma \cdot \sqrt{1 - 2\omega^2} \\ Y = 2\mathbf{x}_0 + 3\mathbf{x}_1 + 4\mathbf{x}_2 + 5\mathbf{x}_3 + 6\mathbf{x}_4 + e \end{cases} \tag{3.10}$$

where $\omega \in \mathbb{R}$ and γ, e are both standard Gaussian noise terms, independent from each other and all the other variables in the simulation. Compared with Example 3, such DGP comes



(a) $\omega = 1/4,\ \mu(F) = 1/2$, lasso (b) $\omega = 1/3,\ \mu(F) = 2/3$, lasso (c) $\omega = 1/2,\ \mu(F) = 1$, lasso

(d) $\omega = 1/4,\ \mu(F) = 1/2$, solar (e) $\omega = 1/3,\ \mu(F) = 2/3$, solar (f) $\omega = 1/2,\ \mu(F) = 1$, solar

Figure 6: Probability of including redundant variables (top 15) in simulation 2 ($\mathbf{x}_5$ in orange).

with more redudant signals, increasing the challenge of purging redundant variables. With such DGP, it is straightforward to control the value of $\mu(F)$ with ω.

In Eqn (3.10), IRC only affects the redundant $\mathbf{x}_5$; hence, we focus on the probability of wrongly selecting $\mathbf{x}_5$ over 200 repetitions. By setting ω to either 1/4, 1/3 or 1/2, the population value of $\mu(F)$ changes, respectively, to either 1/2, 2/3 or 1, gradually increasing the difficulty of purging the redundant $\mathbf{x}_5$. Figure 6 displays the simulation results. When $\mu(F) = 1/2$, lasso wrongly includes $\mathbf{x}_5$ with probability 0.25. By contrast, $\mathbf{x}_5$ does not crack the top-10 list for solar, implying that the corresponding probability remains below 0.1. When $\mu(F)$ increases to 2/3, the probability that lasso includes $\mathbf{x}_5$ increases to around 0.3, standing out from the other 9 variables. When $\mu(F)$ jumps to 1 in the population and strong IRC is violated, the probability that lasso includes $\mathbf{x}_5$ rises to almost 0.5. Despite the increase in $\mu(F)$, the probability that solar includes $\mathbf{x}_5$ is always below 0.1. The results illustrate that solar is more robust to different settings of the IRC.

4 Solar improvements over lasso-related, subsampling variable selection algorithms

In this section, we show that (i) solar and hold-out average offer significant improvements over lasso-type algorithms in terms of variable selection sparsity, stability, accuracy; and (ii) replacing lasso with solar in subsampling variable slection drastically reduce the computational load (in terms of runtime and subsample repetition) while improving the sparisty and accuracy.

4.1 Simulation competitors

We do not consider all lasso-type algorithms for comparison. Firstly, some lasso modifications (e.g., fused lasso, grouped lasso) are designed to solve specific empirical problems that are not relevant to our paper. Secondly, it may be difficult to investigate by how much some lasso variants outperform lasso.[1] Most importantly, since both solar and lasso can be evaluated via least-angle regression and coordinate descent, most lasso modifications can be directly applied to solar, as discuessed in section 3.1. Hence, we focus on comparing lasso and solar.

Except runtime comparison, least-angle regression and coordinate descent yield nearly identical selection results for solar and lasso. Hence, we combine the lars result and coordinate descent result for solar and ignore the coordinate descent lasso, which is still included in the supplementary files. We report the comparison between least-angle regression and coordinate descent only for the computation speed.

Alongside with lasso, we also include subsampling variable selection algorithms, also referred to as a subsampling ensemble. A lasso subsampling ensemble repeats lasso multiple times across subsamples, producing the averaged (or accumulated) selection result of all the repetitions. In simulation 3, we carefully compare lasso subsampling ensemble with solar subsampling ensemble, showing that replacing lasso with solar in an subsampling ensemble (such as Bach (2008) and Meinshausen and Bühlmann (2010)) improves the variable selection performance and substantially reduces the computation load. Due to large similarity across lasso subsampling ensembles, we take bootstrap lasso (Bach, 2008) as the competitor. Bach (2008) proposes two bolasso algorithms: bolasso-H and bolasso-S. Bolasso-H selects only variables that are selected in all subsamples, i.e., the subsample selection frequency threshold, $f = 1$. Bolasso-S selects variables that are selected in 90% of the subsamples ($f = 0.9$). In simulation 3, we use both as the competitors. Bach (2008) finds that it is always beneficial, in terms of selection and prediction, to generate more subsamples in bolasso. Hence, to compare solar with state-of-art bolasso performance, we set the number of subsamples in bolasso to 256, the maximum in the Bach (2008) simulations.

In the similar vein, we introduce the bootstrap solar ensemble *(bsolar)*, which runs solar on

[1] For example, while Jia and Yu (2010) show numerically that elastic net has slightly better variable-selection accuracy than lasso, they also find that "when the lasso does not select the true model, it is more likely that the elastic net does not select the true model either" (a point we verify in Section 5). While simulations in Zou (2006) show that adaptive lasso outperforms lasso when $p/n < 1$, it requires first computing the OLS estimates of all $\mathbf{x}_i$ coefficients, which is difficult when $p/n > 1$.

each subsample and computes the selection frequency for each variable across subsamples. To ensure that any performance improvement is due to replacing lasso with solar in the ensemble system, bsolar and bolasso use the same f. Thus, we evaluate 2 versions of bsolar: bsolar-H ($f = 1$) and bsolar-S ($f = 0.9$). [2] To denote the number of subsamples clearly, we use the notation bsolar-mH and bsolar-mS, where m is the number of subsamples used to compute the selection frequency.

4.2 Simulation settings

The DGP in this simulation is as follows. The response variable $Y \in \mathbb{R}^{n\times 1}$ is generated by

$$Y = X\beta + e = 2\mathbf{x}_0 + 3\mathbf{x}_1 + 4\mathbf{x}_2 + 5\mathbf{x}_3 + 6\mathbf{x}_4 + e \tag{4.1}$$

where $X \in \mathbb{R}^{n\times p}$ is zero-mean, multivariate Gaussian, with covariance matrix having 1 on the main diagonal and 0.5 for the off-diagonal elements. All data points are independently and identically distributed. Each $\mathbf{x}_j$ is independent from the noise term e, which is standard Gaussian. Solar competes with K-fold, cross-validated lasso (denoted 'lasso' for short). We choose the number of CV folds and the number of subsamples generated in Algorithm 1 to be 10, following the Friedman et al. (2001) simulations that show $K = 10$ balances the bias-variance trade-off in CV error minimization.

We use three criteria to compare the variable-selection performance of solar and lasso: sparsity, accuracy and runtime. Sparsity is measured by the mean of the number of variables selected. Accuracy is measured by the average number of selected informative variables. Runtime is the average CPU time of one realization.

In this simulation, we generate data with various p/n (each with 200 repeats and fixed Python random generators) as follows. In scenario 1, p/n approaches 0 from above, corresponding to the classical $p < n$ setting. In scenario 2, p/n approaches 1 from above, corresponding to high-dimensional settings. In scenario 3, $p/n = 2$ as $\log(p)/n$ slowly approaches 0, corresponding to ultrahigh-dimensional settings, i.e., where $(p - n) \to \infty$. The raw simulation results are also in the supplementary file.

[2] If the sample comes with small n or large noise level, we may reserve part of the data as a validation set for the bootstrap ensemble, choosing the optimal f by minimizing the validation error.

4.3 Programming language, platforms and hardware for simulations

To (i) make the result comparison fair, representative and generalizable to real-world application; and (ii) compare with the state-of-art performance of lasso, we carefully design the simulation setting and environment in the following ways.

Firstly, to speed up the computation to the maximum extend, we use `Numpy`, `Scipy` and `Cython` — numerical packages well-known for its performance and speed — to implement all numerical/matrix operations in Fortran/C++ library `Intel MKL`. Assuming without GPU, `Intel MKL` is currently the fastest and most accurate C++/Fortran library for numerical operations; hence, the choice of different languages virtually have no effect on the computation speed and accuracy.[3]

Secondly, to reduce the possibility that CPU and RAM bottleneck the parallel computing performance of lasso and lasso-related subsampling algorithms, we skip R and choose Python as the programming interface. Research (for example, Donoho (2017)) shows that R has the well-known reputation of being less scalable than Python to large problem sizes, due to its memory-CPU management. Our simulations repeats solar, lasso and lasso-realted subsampling algorithms in a large number for average performance; hence, choosing Python over R greatly reduce the possible hardware limitation. Moreover, all computations are accomplished on i9-9900K CPU with 32GB RAM, further reducing the possibility of CPU-RAM bottlenecking.

Thirdly, to guarantee the programming quality of the lasso package, we implemented lasso and lasso-related subsampling algorithms from the Sci-kit learn library (Pedregosa et al., 2011) — one of the most efficient machine-learning package widely used in research and industry.[4] Sci-kit learn and solar follow the same programming scheme, using `Numpy`, `Scipy` and `Cython` to implement all numerical/matrix operations in the Fortran/C++. Such setting pushes the lasso computation speed to its possible maximum.

Lastly, to guarantee consistent performance of solar across different platforms, our Python package for solar (`solarpy`) is coded in Python 3.7.3 (under Anaconda3 version 2019-03) in a Linux environment (Ubuntu 18.04.2). `solarpy` is extensively documented with detailed comments, thorough demostrations and stepwise explanations in the supplementary file.

Table 2: Simulation 1 result.

		$p/n \to 0$			$p/n \to 1$			$\log(p)/n \to 0$		
		$\frac{100}{100}$	$\frac{100}{150}$	$\frac{100}{200}$	$\frac{150}{100}$	$\frac{200}{150}$	$\frac{250}{200}$	$\frac{400}{200}$	$\frac{800}{400}$	$\frac{1200}{600}$
average number of selected variables	lasso	19.73	19.84	19.54	22.30	23.57	26.57	28.92	33.88	37.96
	solar	9.94	8.34	8.48	11.34	9.80	8.20	10.54	13.28	15.53
	solar + hold out	5.02	5.12	5.17	4.99	5.16	5.13	5.12	5.24	5.26
	bsolar-3S/3H	5.45	5.16	5.20	5.47	5.17	5.07	5.16	5.25	5.46
	bsolar-5S/5H	5.14	5.04	5.10	5.15	5.04	5.04	5.06	5.12	5.12
	bsolar-10S/10H	5.06	5.01	5	5.03	5	5	5.01	5.01	5.03
	bolasso-S	5.46	6.09	6.60	5.39	5.72	6.06	5.66	6.72	7.69
	bolasso-H	5	5.02	5.01	5	5.01	5	5	5	5.02
average number of selected informative variables	lasso	5	5	5	5	5	5	5	5	5
	solar	5	5	5	5	5	5	5	5	5
	solar + hold out	4.95	5	5	4.91	5	5	5	5	5
	bsolar-3S/3H	5	5	5	5	5	5	5	5	5
	bsolar-5S/5H	5	5	5	5	5	5	5	5	5
	bsolar-10S/10H	5	5	5	5	5	5	5	5	5
	bolasso-S	5	5	5	5	5	5	5	5	5
	bolasso-H	5	5	5	5	5	5	5	5	5
average runtime	solar + hold out	0.09	0.10	0.11	0.09	0.12	0.16	0.19	0.92	1.62
	bsolar-3	0.33	0.22	0.24	0.19	0.27	0.35	0.46	1.07	2.40
	bolasso (lars, 256 SR)	6.78	11.94	18.67	7.50	16.25	25.75	46.41	636.29	1731.38
	bolasso (cd, 256 SR)	16.99	61.34	61.10	33.60	166.99	196.50	240.36	632.26	1412.01

4.4 Accuracy and sparsity comparison with varying p/n

Table 2 summarizes the average selection performance.[5] While solar and lasso always include all 5 informative variables, solar clearly outperforms lasso in terms of sparsity in every p/n scenario, implying a much better ability to control redunant variable inclusion. More interestingly, the lasso sparsity deteriorates as p/n approaching 1, while the solar sparsity continues improving. As $\log(p)/n \to 0$, while the sparsity of each competitor deteriorates, solar maintain a clear advantage over lasso.

Table 2 also confirms the solar advantage over lasso in subsampling ensembles. All subsampling ensembles select all 5 informative variable with probability 1. Among all subsampling ensembls, bolasso-S sparsity always remains the worst while others perform nearly identical. However, it is important to note that, bolasso takes requires 256 subsampe lasso repetitions to achieve such performance; by contrast, bsolar-3, -5 and -10 only take 3, 5 and 10 subsampe solar repetitions. Hence, solar saves more than 96% of all subsample repetitions. As we will show in section 4.6, solar and lasso have the identical computation load. As a result, 96% of subsample repetition reduction implies that at least 96% of computation time reduction, assuming time

[3]See https://jwalton.info/Python-MKL-openBLAS/ for more detail.

[4]More detail is available at https://scikit-learn.org/stable/.

[5]Detailed histograms are in the supplementary file.

complexity increases linearly.

Most importantly, the performance of *solar + hold out* (the combination of solar and the hold-out average) at Table 2 illustrates that, in a large scale, solar performance can be further boosted by hold-out average test. As long as n is not extremely small (a necessary requirement for hypothesis testing), *solar + hold out* performance is totally on par with any subsampling variable selection method. Such result further confirms the Example 1 findings.

4.5 Explanation to the bolasso-bsolar efficiency discrepency

With only 4% of subsample repetitions, bsolar performance comes on par with bolasso, which requires 256 repetitions. The bsolar efficency are quite expected due to its unique *multi-layer variable ranking scheme*. As shown in Algorithm 1 and 3, solar itself internalizes a variable ranking scheme, the *internal ranking*. Bsolar further embeds a variable ranking algorithm into another bootstrap ranking framework, resulting in the *external ranking*. Within each subsample realization, solar ranks all variables efficiently based on the average conditional correlation, purging low-ranked variables at each subsample. After all subsample solar realizations, bsolar collects the selected variables across subsamples and conduct the external ranking based on a new test data, selecting the stablest signals across all strong variables. With researches confirming the efficiency of one-layer ranking (Fan and Lv (2008); Hall et al. (2009); Hall and Miller (2009); Li et al. (2012a,b)), the improvements of multi-layer ranking is largely explainable.

Table 3 shows numerically how the mutl-layer ranking make a difference. Table 3a shows the subsample selection frequency list from 256 subsamples for one bolasso realization with $p/n = 100/200$. Equipped with only one layer of ranking, bolasso is not able clearly to separate informative from redundant variables even with 256 subsample repetitions. The frequency discrepancy between the highest-ranking redundant ($\mathbf{x}_{28}$) and the lowest-ranking informative variable ($\mathbf{x}_0$) is only 0.12. By contrast, bsolar-10 returns a much shorter list, the frequency discrepancy between the highest-ranking redundant ($\mathbf{x}_{91}$) and the lowest-ranking informative variable ($\mathbf{x}_0$) is 0.9.

As a possible fix, Bach (2008) sugguests that increasing the number of subsamples in bolasso may increase the discrepancy between the lowest ranked informative and the highest ranked redundant variables. However, increasing the subsample repetitions substantially raises the computation load of bolasso in high-dimensional spaces, making bsolar even more computationally preferable.

Table 3: Subsample selection frequency for bolasso and bsolar-10.

(a) bolasso (subsample selection frequency $\geqslant 0.67$)

frequency	variables
$\geqslant 1.00$	$\mathbf{x}_4, \mathbf{x}_3, \mathbf{x}_2, \mathbf{x}_1, \mathbf{x}_0$
$\geqslant 0.88$	$\mathbf{x}_4, \mathbf{x}_3, \mathbf{x}_2, \mathbf{x}_1, \mathbf{x}_0, \mathbf{x}_{28}$
$\geqslant 0.84$	$\mathbf{x}_4, \mathbf{x}_3, \mathbf{x}_2, \mathbf{x}_1, \mathbf{x}_0, \mathbf{x}_{28}, \mathbf{x}_{71}$
$\geqslant 0.76$	$\mathbf{x}_4, \mathbf{x}_3, \mathbf{x}_2, \mathbf{x}_1, \mathbf{x}_0, \mathbf{x}_{28}, \mathbf{x}_{71}, \mathbf{x}_{91}$
$\geqslant 0.70$	$\mathbf{x}_4, \mathbf{x}_3, \mathbf{x}_2, \mathbf{x}_1, \mathbf{x}_0, \mathbf{x}_{28}, \mathbf{x}_{71}, \mathbf{x}_{91}, \mathbf{x}_{94}$
$\geqslant 0.69$	$\mathbf{x}_4, \mathbf{x}_3, \mathbf{x}_2, \mathbf{x}_1, \mathbf{x}_0, \mathbf{x}_{28}, \mathbf{x}_{71}, \mathbf{x}_{91}, \mathbf{x}_{94}, \mathbf{x}_{70}, \mathbf{x}_{40}$
$\geqslant 0.67$	$\mathbf{x}_4, \mathbf{x}_3, \mathbf{x}_2, \mathbf{x}_1, \mathbf{x}_0, \mathbf{x}_{28}, \mathbf{x}_{71}, \mathbf{x}_{91}, \mathbf{x}_{94}, \mathbf{x}_{70}, \mathbf{x}_{40}, \mathbf{x}_{90}$

(b) bsolar-10 (subsample selection frequency > 0)

frequency	variables
$\geqslant 1.00$	$\mathbf{x}_4, \mathbf{x}_3, \mathbf{x}_2, \mathbf{x}_1, \mathbf{x}_0$
$\geqslant 0.10$	$\mathbf{x}_4, \mathbf{x}_3, \mathbf{x}_2, \mathbf{x}_1, \mathbf{x}_0, \mathbf{x}_{91}, \mathbf{x}_{71}$

4.6 Computation load comparison

Since the computation load of lars or coordinate descent on a given sample is fixed, we may use the number of lars or coordinate descent to approximate the computation loads of solar and lasso. For comparison purposes, we compute solar with K subsamples and lasso with K-fold cross-validation. As shown in Algorithm 1 and 3, we need to compute one lars or coordinate descent for solar on each subsample (X^k, Y^k), implying we need to estimate K lars or coordinate descent to compute $\widehat{q}$ and one to compute c^* for variable selection. Lasso also requires computing K lars or coordinate descent to optimize the tuning parameter and, given the optimal tuning parameter, one extra on the full sample to select variables. Thus, with the same optimizer, solar and lasso have the same computation load.

Due to the equality of the computataion load between solar and lasso, differences in computation load between bolasso and bsolar are due primarily to subsample repetition. So we measure computation load by *the number of subsample repetitions (SR) required by the competitors*. Thus, solar and lasso have a computation load of 1 SR, bolasso has a load of 256 SR, and bsolar-3/5/10 a load of 3/5/10 SR.

Since bsolar and bolasso are very similar in terms of their computation procedures, they are coded similarly to make the comparison fair. The only code difference lies in the computation

of subsample selection frequency.[6] Thus, runtime differences below are due to the reduction in subsample repetitions from the solar ensemble. To push bolasso to its maximum computation speed, we compute bolasso using the built-in `Sci-kit learn` parallel scheme (denoted as *built-in parallel*), which is fully optimized for lasso. Besides, we follow the recommendations of the `Sci-kit learn` package and design a high-performance parallel architecture for *solar + hold out* and bsolar, denoted as *Joblib parallel*.

Table 2 summarizes average runtime in 200 repetitions on Intel i7-10700 CPU with 16GB RAM. Both *solar + hold out* and bsolar-3 have much shorter runtimes than bolasso. The runtime differences become even more pronounced as p and n increase. Our finding is actually very consistent with Friedman et al. (2007) and other previous research on lasso. Taking bolasso runtime at $p/n = 1200/600$ as an example, the average computation time for coordinate descent bolasso is 1412 seconds, accounting for 256 realizations of 10-fold, cross-validated lasso. Hence, the total 1412 seconds is for 2816 coordinate descent realizations, resulting in 0.5 seconds per realization. In a similar vein, the built-in bolasso (solved by lars) on average take 1731 seconds for 256 realizations of 10-fold, cross-validated lasso (e.g., 2816 lars realizations), resulting in 0.61 seconds per lars realization. Such result is very similar to the Fortran result of Friedman et al. (2007) at Table 1, where $p/n = 1000/100, corr = 0.5$. The huge repetition number drastically render the bootstrap variable selection algorithm computational infeasible even with moderate p and n. By contrast, bsolar3 only require 30 realization of lars or coordinate descent. Due to very light computational load and CPU usage, the CPU can work at a much higher frequency than it does for bolasso, making runtime for each lars realization much shorter.

5 Real-world data: Sydney house price prediction

To demonstrate that the improvements from solar are empirically feasible, we apply solar to real-world data. In the context of the simulations above, the real-world data reflect both the $p/n \to 0$ scenarios of simulation 1 and the high multicollinearity and complicated dependence structures of simulation 2. As such, the data reflect the challenging IRC settings, complicated dependence structures, and grouping effects typical of data in the social sciences.

The sample is assembled from more than 10 different databases: Sydney, Australia, real estate market transaction data for 11,974 houses sold in 2010, including price and house attribute information (property address, bedrooms, bathrooms, car spaces, etc.), matched to 2011 census

[6] See Read_Me_First.html in the supplementary file for more detail.

data by Statistical Area Level 1 (the smallest census area in Australia, comprising at most 200 people or 60 households), 2010 and 2011 crime data by suburb, 2010 GIS data (extracted from a geo-spatial topological database, a climate database, a pollution database, and the Google Maps database), 2009 primary and secondary school data, and 2010 Sydney traffic and public transport data (bus routes, train stations, and ferry wharfs). We predict house price with a linear model.

Using an ensemble of Bayes network learning algorithms for data clearning, we purge variables with both very low conditional and unconditional correlations to house price. The remaining variables are listed in the first column of Table 4.[7] The 57 variables fall into 5 categories: house attributes, distance to key locations (public transport, shopping, etc.), neighbourhood socio-economic data, localized administrative and crime data, and local school quality. Pairwise correlations among all 57 covariates indicate, not surprisingly, severe multicollinearity and grouping effects, implying a harsh IRC setting.[8] Thus, heuristically increasing the value of the tuning parameter in lasso-type estimators (e.g., using the one-sd or the 'elbow' rule) is unlikely to be useful since it may trigger further grouping effects and the random dropping of variables.

Table 4 shows the selection comparison across the elastic net, lasso, and solar. With all variables in linear form, both lasso and elastic net lose sparsity due to the complicated dependence structures and severe multicollinearity in the data, consistent with Jia and Yu (2010). By contrast, solar returns a much sparser model, with only 9 variables selected from 57. A very similar result can be found with the variables in log form, hinting that solar possesses superior selection sparsity and robustness to a change in functional form. More importantly, solar variable selection outperforms the lasso-type estimators in terms of the balance between sparsity and prediction power. While pruning most of the variables from the elastic net and lasso selections, solar reduces the post-selection regression R^2 by only 3-5%. In the supplementary file, we also perform a hold-out average method to avoid the possibility that the 'p-value overfitting' issue also affects the direct post-selection OLS R^2. However, there is no concern in this sample because $p/n = 57/11,974$. This confirms that, from the perspective of prediction, solar successfully identifies the most important variables in the database.

[7] Due to 200GB size of the database, we only include the data for these variables in the supplementary file.

[8] Correlations and IRC are also reported in supplementary files.

6 Conclusion

In this paper we offer a new least-angle regression algorithm for high-dimensional data called solar (for subsample-ordered least-angle regression). We show that solar yields substantial improvements over lasso in terms of the sparsity, stability, accuracy, and robustness of variable selection. We also illustrate the improvements relative to lasso ensembles from solar ensembles in terms of variable selection accuracy, sparsity and computation load.

Detection of weak signals is a small weakness evident in solar, although relative to the lasso competitor the difference is very slight. Nonetheless, we are working on an extension to solar, the double-bootstrap solar (DBsolar), which, if early results are any indication, promises to enable solar accurately to detect variables with weak signals.

References

Bach, F.R., 2008. Bolasso: model consistent lasso estimation through the bootstrap, in: Proceedings of the 25th international conference on Machine learning, ACM. pp. 33–40.

Barber, R.F., Candès, E.J., 2019. A knockoff filter for high-dimensional selective inference. The Annals of Statistics 47, 2504–2537.

Barut, E., Fan, J., Verhasselt, A., 2016. Conditional sure independence screening. Journal of the American Statistical Association 111, 1266–1277.

Bousquet, O., Elisseeff, A., 2002. Stability and generalization. Journal of Machine Learning Research 2, 499–526.

DiCiccio, C.J., DiCiccio, T.J., Romano, J.P., 2020. Exact tests via multiple data splitting. Statistics & Probability Letters 166, 108865.

Donoho, D., 2017. 50 years of data science. Journal of Computational and Graphical Statistics 26, 745–766.

Efron, B., Hastie, T., Johnstone, I., Tibshirani, R., 2004. Least angle regression. Annals of Statistics 32, 407–499.

Fan, J., Lv, J., 2008. Sure independence screening for ultrahigh dimensional feature space. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 70, 849–911.

Friedman, J., Hastie, T., Höfling, H., Tibshirani, R., et al., 2007. Pathwise coordinate optimization. Annals of applied statistics 1, 302–332.

Friedman, J., Hastie, T., Tibshirani, R., 2001. The elements of statistical learning. volume 1 of *Springer Series in Statistics*. Springer-Verlag New York.

Friedman, J., Hastie, T., Tibshirani, R., 2010. Regularization paths for generalized linear models via coordinate descent. Journal of statistical software 33, 1.

Ghaoui, L.E., Viallon, V., Rabbani, T., 2010. Safe feature elimination for the lasso and sparse supervised learning problems. arXiv preprint arXiv:1009.4219 .

Hall, P., Miller, H., 2009. Using generalized correlation to effect variable selection in very high dimensional problems. Journal of Computational and Graphical Statistics 18, 533–550.

Hall, P., Miller, H., et al., 2009. Using the bootstrap to quantify the authority of an empirical ranking. The Annals of Statistics 37, 3929–3959.

Jia, J., Yu, B., 2010. On model selection consistency of the elastic net when $p >> n$. Statistica Sinica , 595–611.

Kearns, M., Ron, D., 1999. Algorithmic stability and sanity-check bounds for leave-one-out cross-validation. Neural Computation 11, 1427–1453.

Li, G., Peng, H., Zhang, J., Zhu, L., et al., 2012a. Robust rank correlation based screening. The Annals of Statistics 40, 1846–1877.

Li, R., Zhong, W., Zhu, L., 2012b. Feature screening via distance correlation learning. Journal of the American Statistical Association 107, 1129–1139.

Lim, C., Yu, B., 2016. Estimation stability with cross-validation (ESCV). Journal of Computational and Graphical Statistics 25, 464–492.

Lockhart, R., Taylor, J., Tibshirani, R.J., Tibshirani, R., 2014. A significance test for the lasso. Annals of Statistics 42, 413–468.

Meinshausen, N., Bühlmann, P., 2010. Stability selection. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 72, 417–473.

Meinshausen, N., Meier, L., Bühlmann, P., 2009. P-values for high-dimensional regression. Journal of the American Statistical Association 104, 1671–1681.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, 2825–2830.

Taylor, J., Lockhart, R., Tibshirani, R.J., Tibshirani, R., 2014. Exact post-selection inference for forward stepwise and least angle regression. arXiv preprint arXiv:1401.3889 7, 2.

Tibshirani, R., Bien, J., Friedman, J., Hastie, T., Simon, N., Taylor, J., Tibshirani, R.J., 2012. Strong rules for discarding predictors in lasso-type problems. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 74, 245–266.

Wang, J., Zhou, J., Liu, J., Wonka, P., Ye, J., 2014. A safe screening rule for sparse logistic regression, in: Advances in neural information processing systems, pp. 1053–1061.

Wasserman, L., Roeder, K., 2009. High dimensional variable selection. Annals of statistics 37, 2178.

Weisberg, S., 2004. Discussion following "Least angle regression," by B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Annals of Statistics 32, 490–494.

Xu, G., Huang, J.Z., et al., 2012. Asymptotic optimality and efficient computation of the leave-subject-out cross-validation. The Annals of Statistics 40, 3003–3030.

Zeng, Y., Yang, T., Breheny, P., 2017. Efficient feature screening for lasso-type problems via hybrid safe-strong rules. arXiv preprint arXiv:1704.08742 .

Zhang, T., 2009. On the consistency of feature selection using greedy least squares regression. Journal of Machine Learning Research 10, 555–568.

Zhao, P., Yu, B., 2006. On model selection consistency of Lasso. Journal of Machine Learning Research 7, 2541–2563.

Zou, H., 2006. The adaptive lasso and its oracle properties. Journal of the American statistical association 101, 1418–1429.

Table 4: Variable selection results for linear and log house price models.

		elastic net		lasso		solar	
Variable	Description	linear	log	linear	log	linear	log
Bedrooms	property, number of bedrooms	✓	✓	✓	✓	✓	✓
Baths	property, number of bathrooms	✓	✓	✓	✓	✓	✓
Parking	property, number of parking spaces	✓	✓	✓	✓	✓	✓
AreaSize	property, land size	✓	✓	✓	✓		
Airport	distance, nearest airport	✓	✓	✓	✓		
Beach	distance, nearest beach	✓	✓	✓	✓	✓	✓
Boundary	distance, nearest suburb boundary	✓	✓	✓	✓		
Cemetery	distance, nearest cemetery	✓		✓			
Child care	distance, nearest child-care centre	✓	✓	✓	✓		✓
Club	distance, nearest club	✓	✓	✓	✓		
Community facility	distance, nearest community facility	✓	✓				
Gaol	distance, nearest gaol	✓	✓			✓	✓
Golf course	distance, nearest golf course	✓	✓	✓	✓		
High	distance, nearest high school	✓	✓	✓	✓		
Hospital	distance, nearest general hospital	✓	✓		✓		
Library	distance, nearest library	✓		✓			
Medical	distance, nearest medical centre	✓	✓		✓		
Museum	distance, nearest museum	✓	✓	✓	✓		
Park	distance, nearest park	✓	✓	✓			
PO	distance, nearest post office	✓	✓		✓		
Police	distance, nearest police station	✓	✓	✓	✓		
Pre-school	distance, nearest preschool	✓	✓	✓	✓		
Primary	distance, nearest primary school	✓	✓	✓	✓		
Primary High	distance, nearest primary-high school	✓	✓	✓	✓		
Rubbish	distance, nearest rubbish incinerator	✓	✓	✓			
Sewage	distance, nearest sewage treatment	✓					
SportsCenter	distance, nearest sports centre	✓	✓	✓	✓		
SportsCourtField	distance, nearest sports court/field	✓		✓	✓		
Station	distance, nearest train station	✓		✓			
Swimming	distance, nearest swimming pool	✓	✓	✓	✓		
Tertiary	distance, nearest tertiary school	✓	✓	✓	✓		
Mortgage	SA1, mean mortgage repayment (log)	✓	✓	✓	✓	✓	✓
Rent	SA1, mean rent (log)	✓	✓	✓	✓	✓	✓
Income	SA1, mean family income (log)	✓	✓	✓	✓	✓	✓
Income (personal)	SA1, mean personal income (log)	✓					
Household size	SA1, mean household size	✓	✓	✓	✓		
Household density	SA1, mean persons to bedroom ratio	✓	✓	✓	✓		
Age	SA1, mean age	✓	✓	✓	✓		✓
English spoken	SA1, percent English at home	✓		✓			
Australian born	SA1, percent Australian-born	✓		✓			
Suburb area	suburb area	✓		✓	✓		
Population	suburb population	✓	✓		✓		
TVO2010	suburb total violent offences, 2010	✓					
TPO2010	suburb total property offences, 2010	✓	✓		✓		
TVO2009	suburb total violent offences, 2009	✓	✓	✓			
TPO2009	suburb total property offences, 2009	✓	✓				
ICSEA	local school, socio-educational advantage	✓	✓	✓	✓	✓	✓
ReadingY3	local school, year 3 mean reading score	✓	✓	✓	✓		
WritingY3	local school, year 3 mean writing score	✓	✓	✓	✓		
SpellingY3	local school, year 3 mean spelling score	✓	✓	✓			
GrammarY3	local school, year 3 mean grammar score	✓		✓			
NumeracyY3	local school, year 3 mean numeracy score	✓	✓	✓	✓		
ReadingY5	local school, year 5 mean reading score	✓					
WritingY5	local school, year 5 mean writing score	✓	✓	✓			
SpellingY5	local school, year 5 mean spelling score	✓	✓	✓			
GrammarY5	local school, year 5 mean grammar score	✓	✓	✓			
NumeracyY5	local school, year 5 mean numeracy score	✓					
	Number of variables selected	57	45	44	36	9	11
	post-selection OLS R^2	0.55	0.76	0.55	0.76	0.50	0.73
	Sample size			11,974			